SQL Lab-5

1. **Create a table WorkCenters with the following data**

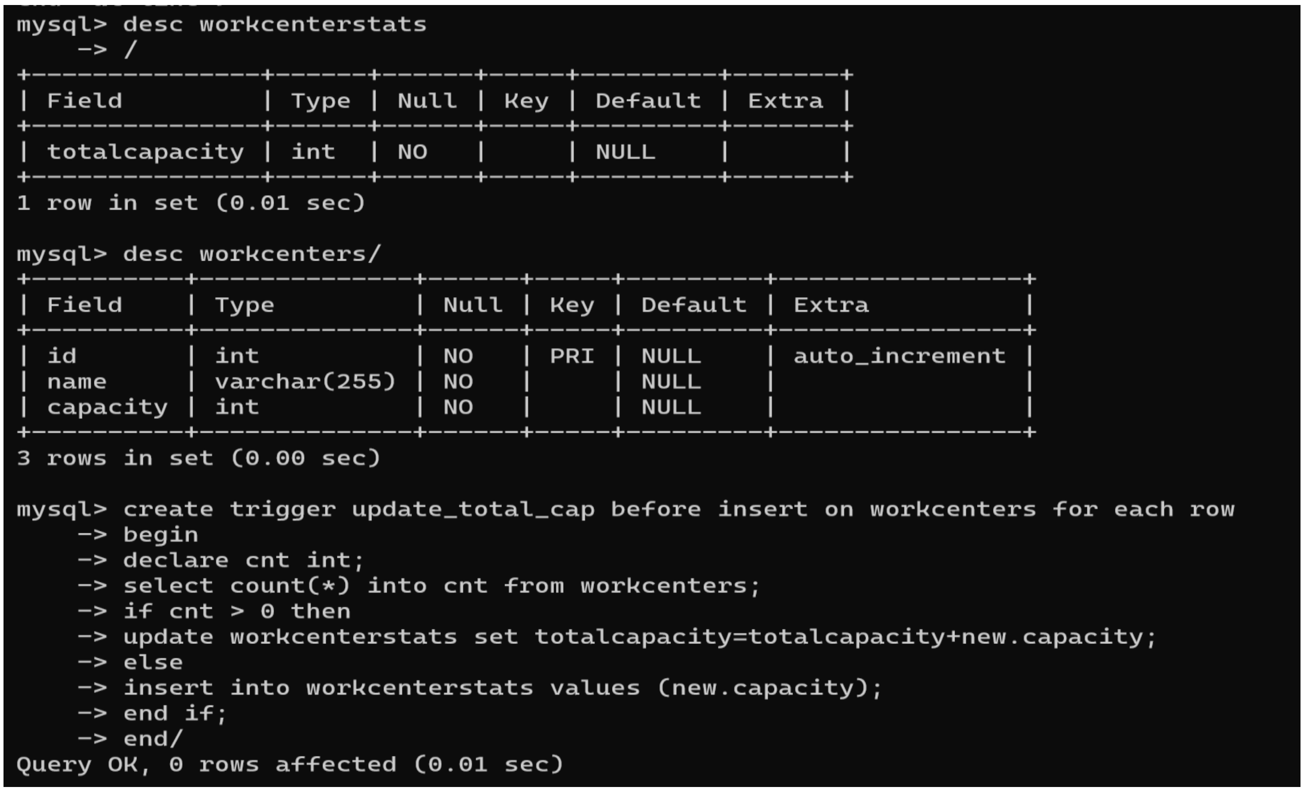
Column	Datatype	Constraint
id	Int	Primary key, autoincrement
Name	Varchar(255)	Not null
Capacity	Int	Not null

Create a table WorkcenterStats with the following data

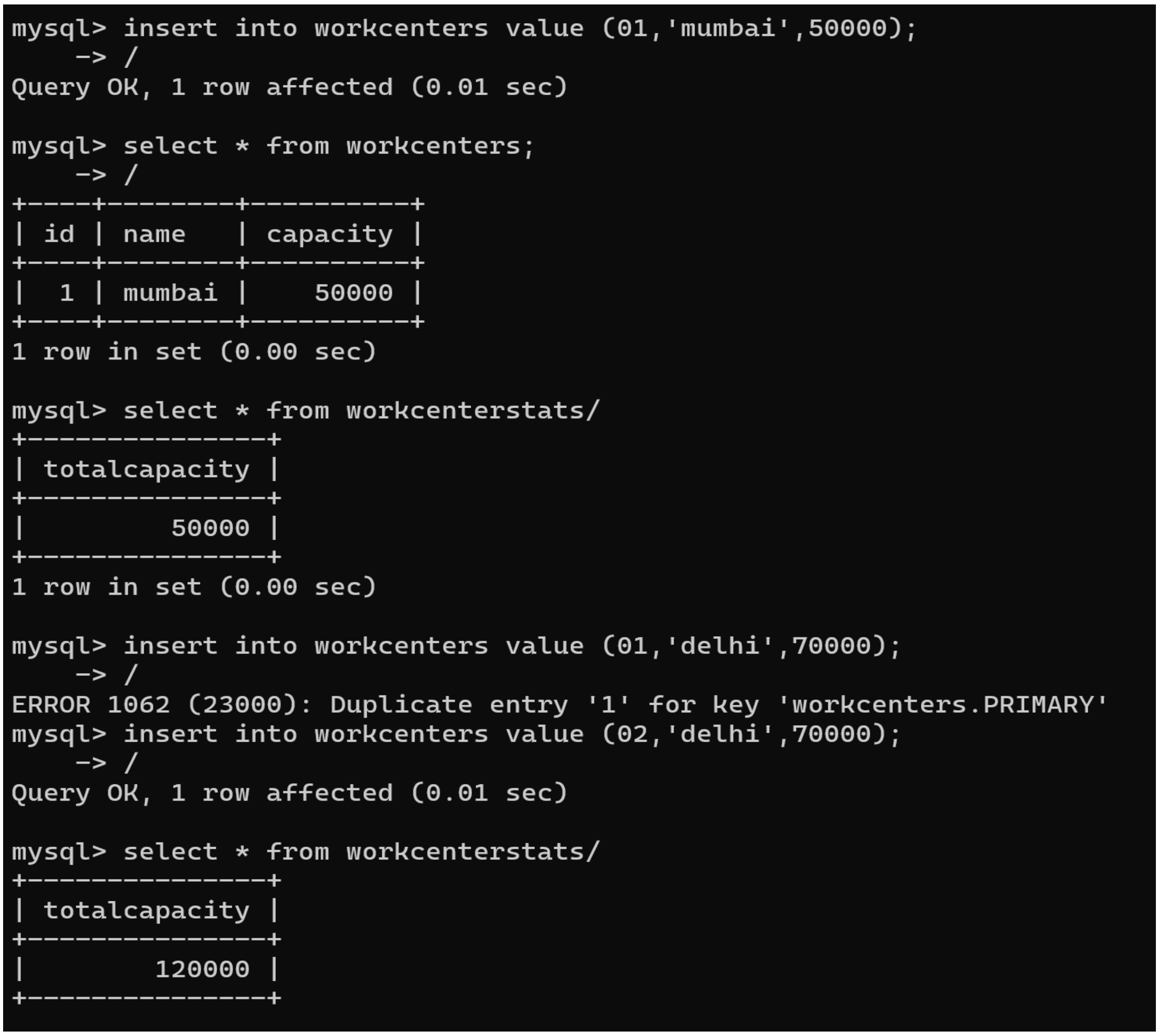
Column	Datatype	Constraint
Totalcapacity	Int	not null

Write a trigger which updates the total capacity in the WorkCenterStats table before a new work center is inserted into the WorkCenters 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 trigger update_total_cap before insert on workcenters for each
row
    -> begin
    -> declare cnt int;
    -> select count(*) into cnt from workcenters;
    -> if cnt > 0 then
    -> update workcenterstats set totalcapacity=totalcapacity+new.capacity;
    -> else
    -> insert into workcenterstats values (new.capacity);
    -> end if;
    -> end/
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 (0.01 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 trigger update_total_cap before insert on workcenters for each row
    -> begin
    -> declare cnt int;
    -> select count(*) into cnt from workcenters;
    -> if cnt > 0 then
    -> update workcenterstats set totalcapacity=totalcapacity+new.capacity;
    -> else
    -> insert into workcenterstats values (new.capacity);
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)
```



```
mysql> insert into workcenters value (01,'mumbai',50000);
    -> /
Query OK, 1 row affected (0.01 sec)

mysql> select * from workcenters;
    -> /
+----+--------+----------+
| id | name   | capacity |
+----+--------+----------+
|  1 | mumbai |    50000 |
+----+--------+----------+
1 row in set (0.00 sec)

mysql> select * from workcenterstats/
+---------------+
| totalcapacity |
+---------------+
|         50000 |
+---------------+
1 row in set (0.00 sec)

mysql> insert into workcenters value (01,'delhi',70000);
    -> /
ERROR 1062 (23000): Duplicate entry '1' for key 'workcenters.PRIMARY'
mysql> insert into workcenters value (02,'delhi',70000);
    -> /
Query OK, 1 row affected (0.01 sec)

mysql> select * from workcenterstats/
+---------------+
| totalcapacity |
+---------------+
|        120000 |
+---------------+
```

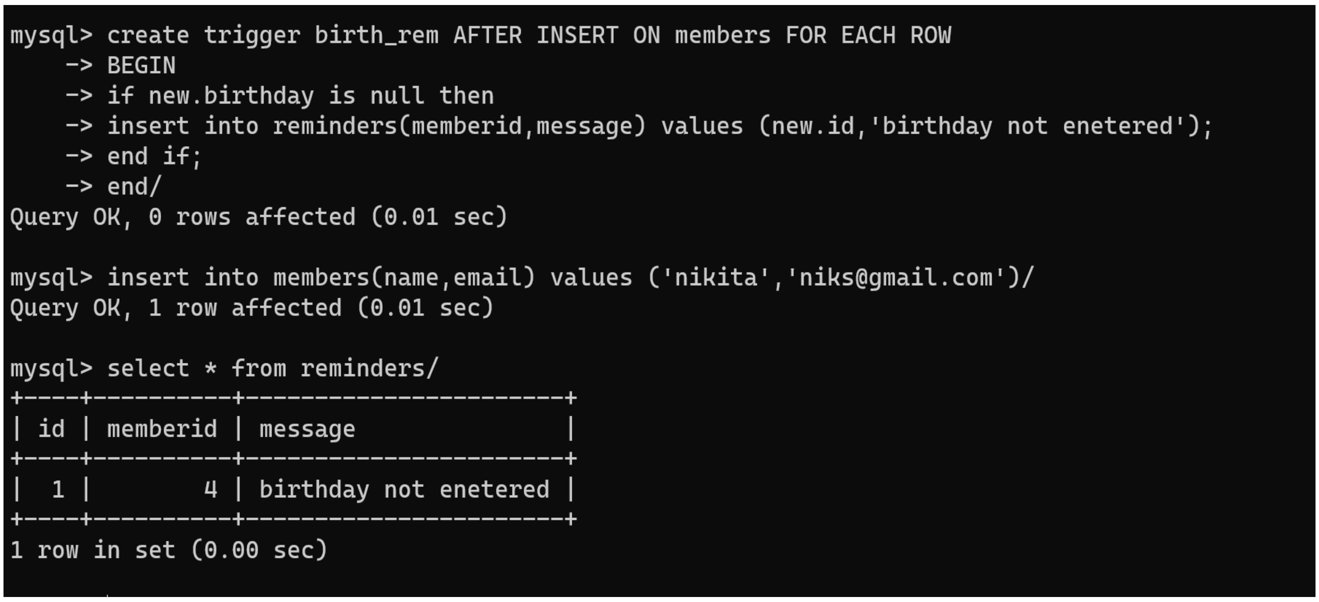
2. **Create a table Members with the following data**

Column	Datatype	Constraint
id	Int	Primary key, autoincrement
Name	Varchar(50)	Not null
email	Varchar(255)	
birthday	Date	

Create a table Reminders with the following data

Column	Datatype	Constraint
id	Int	Primary key, autoincrement
memberId	int	Primary key
message	Varchar(255)	Not null

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



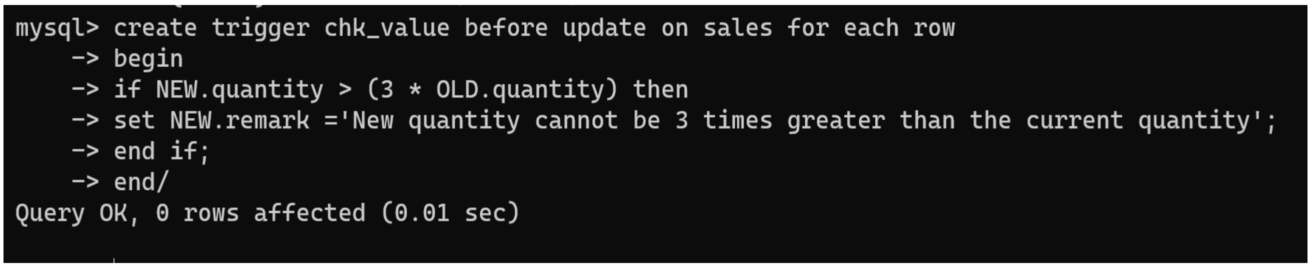
```
mysql> create trigger birth_rem AFTER INSERT ON members FOR EACH ROW
    -> BEGIN
    -> if new.birthday is null then
    -> insert into reminders(memberid,message) values (new.id,'birthday not enetered');
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)

mysql> insert into members(name,email) values ('nikita','niks@gmail.com')/
Query OK, 1 row affected (0.01 sec)

mysql> select * from reminders/
+----+----------+-----------------------+
| id | memberid | message               |
+----+----------+-----------------------+
|  1 |        4 | birthday not enetered |
+----+----------+-----------------------+
1 row in set (0.00 sec)
```

3. **Create a table Sales with the following data**

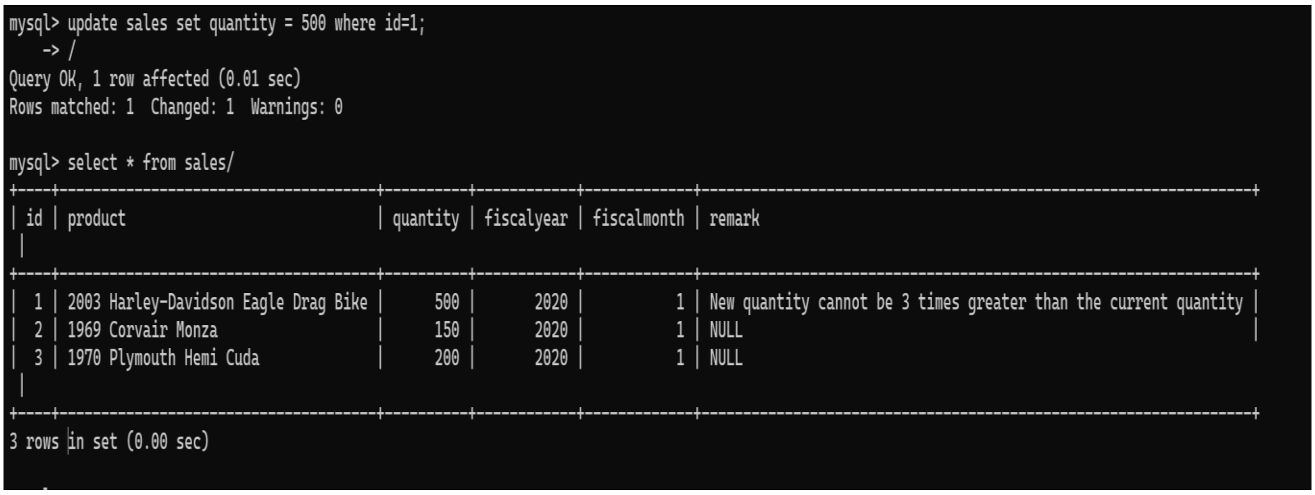
Column	Datatype	Constraint
id	Int	Primary key, autoincrement
Product	Varchar(50)	Not null
Quantity	Int	Not null
fiscalYear	Smallint	Not null
fiscalmonth	Tinyint	Not null
Remarks	Varchar(255)	

INSERT 3 rows in the columns product, quantity, fiscalYear, fiscalMonth the following
VALUES
1. '2003 Harley-Davidson Eagle Drag Bike',120, 2020,1
2. '1969 Corvair Monza', 150,2020,1
3. '1970 Plymouth Hemi Cuda', 200,2020,1

Create a before update trigger which does the following
If the value in the quantity column is updated to a new value that is 3 times greater than
the current value, the remarks column of that row should be updated with a message
"New quantity cannot be 3 times greater than the current quantity"
Update the row and check with different values.



```
mysql> create trigger chk_value before update on sales for each row
    -> begin
    -> if NEW.quantity > (3 * OLD.quantity) then
    -> set NEW.remark ='New quantity cannot be 3 times greater than the current quantity';
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)
```

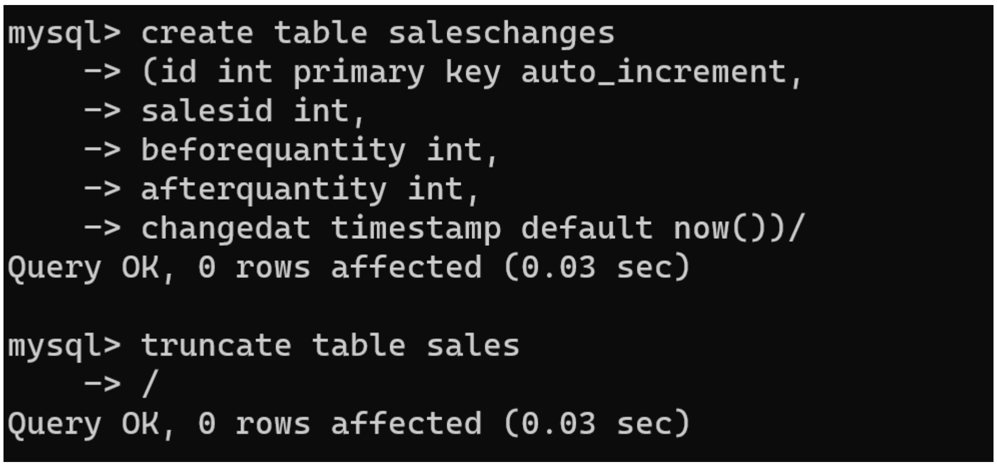


```
mysql> update sales set quantity = 500 where id=1;
    -> /
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from sales/
+----+--------------------------------------+----------+------------+-------------+-------------------------------------------------------------------+
| id | product                              | quantity | fiscalyear | fiscalmonth | remark                                                            |
+----+--------------------------------------+----------+------------+-------------+-------------------------------------------------------------------+
|  1 | 2003 Harley-Davidson Eagle Drag Bike |      500 |       2020 |           1 | New quantity cannot be 3 times greater than the current quantity |
|  2 | 1969 Corvair Monza                   |      150 |       2020 |           1 | NULL                                                              |
|  3 | 1970 Plymouth Hemi Cuda              |      200 |       2020 |           1 | NULL                                                              |
+----+--------------------------------------+----------+------------+-------------+-------------------------------------------------------------------+
3 rows in set (0.00 sec)
```

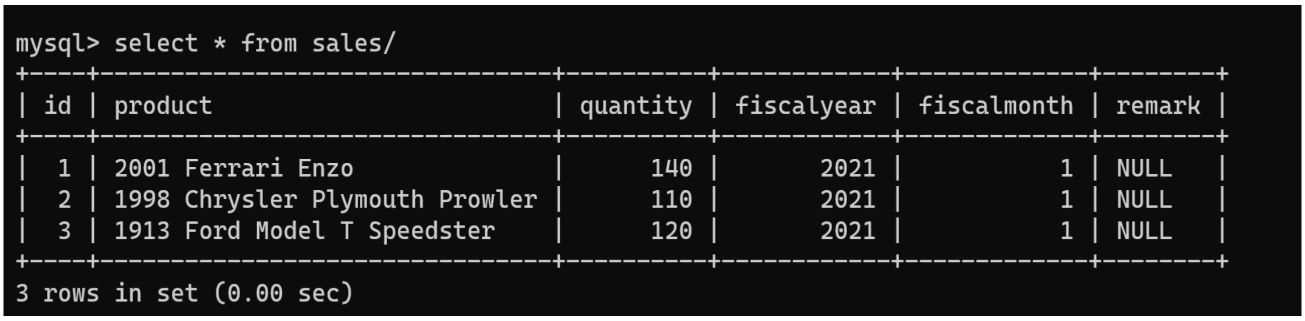
4. **Create a table SalesChanges with the following data**

Column	Datatype	Constraint
id	Int	Primary key, autoincrement
salesid	int	
beforequantity	Int	
afterquantity	int	
changedat	timestamp	Default current_timestamp

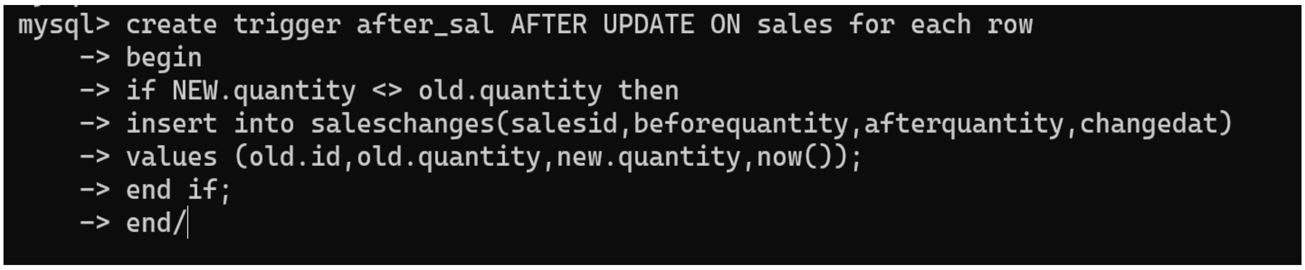
Delete the existing rows in the Sales table
INSERT 3 rows in the columns product, quantity, fiscalYear, fiscalMonth the following
VALUES
1. '2001 Ferrari Enzo',140, 2021,1
2. '1998 Chrysler Plymouth Prowler', 110,2021,1
3. '1913 Ford Model T Speedster', 120,2021,1
Create an after update trigger which 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.



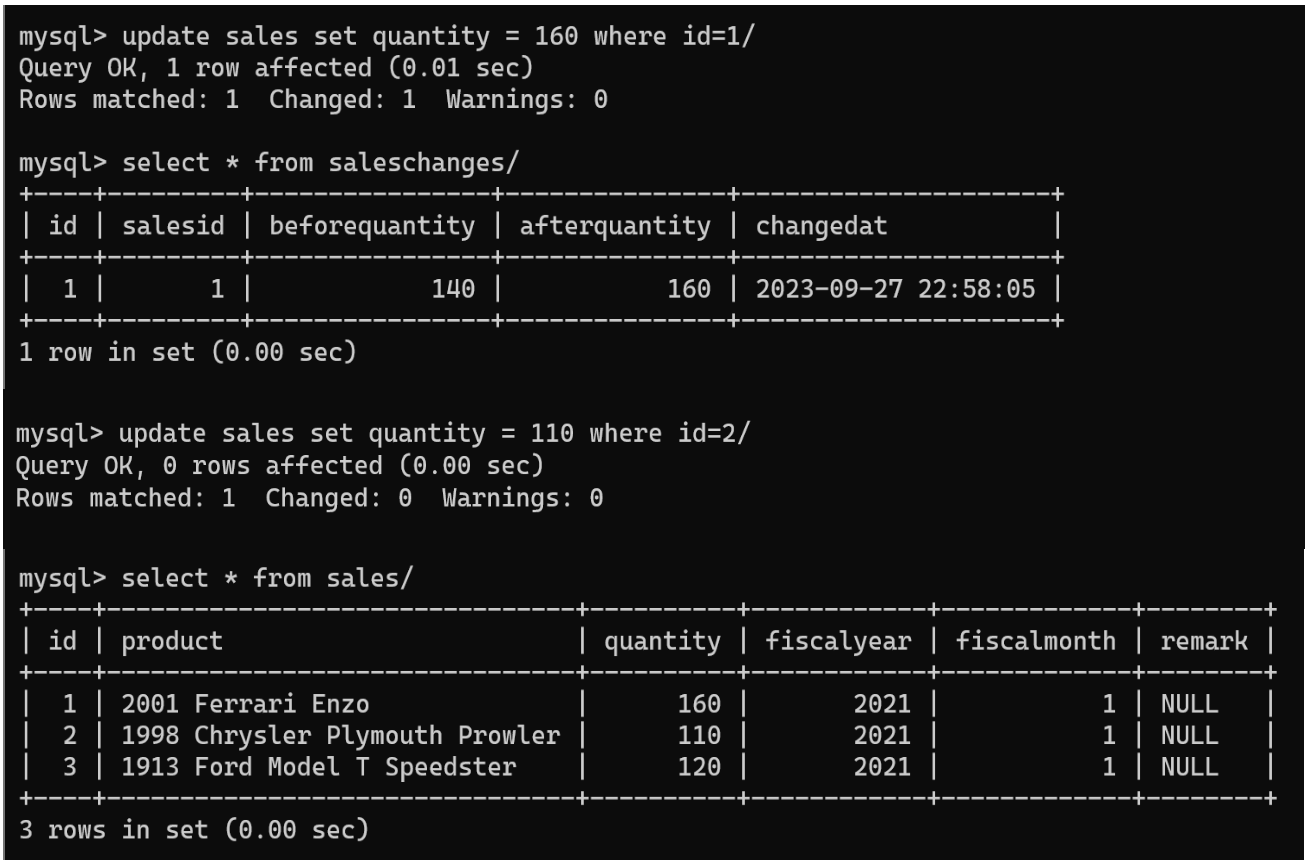
```
mysql> create table saleschanges
    -> (id int primary key auto_increment,
    -> salesid int,
    -> beforequantity int,
    -> afterquantity int,
    -> changedat timestamp default now())/
Query OK, 0 rows affected (0.03 sec)

mysql> truncate table sales
    -> /
Query OK, 0 rows affected (0.03 sec)
```



```
mysql> select * from sales/
+----+-------------------------------+----------+------------+-------------+--------+
| id | product                       | quantity | fiscalyear | fiscalmonth | remark |
+----+-------------------------------+----------+------------+-------------+--------+
|  1 | 2001 Ferrari Enzo             |      140 |       2021 |           1 | NULL   |
|  2 | 1998 Chrysler Plymouth Prowler |     110 |       2021 |           1 | NULL   |
|  3 | 1913 Ford Model T Speedster   |      120 |       2021 |           1 | NULL   |
+----+-------------------------------+----------+------------+-------------+--------+
3 rows in set (0.00 sec)
```



```
mysql> create trigger after_sal AFTER UPDATE ON sales for each row
    -> begin
    -> if NEW.quantity <> old.quantity then
    -> insert into saleschanges(salesid,beforequantity,afterquantity,changedat)
    -> values (old.id,old.quantity,new.quantity,now());
    -> end if;
    -> end/
```



```
mysql> update sales set quantity = 160 where id=1/
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from saleschanges/
+----+---------+----------------+---------------+---------------------+
| id | salesid | beforequantity | afterquantity | changedat           |
+----+---------+----------------+---------------+---------------------+
|  1 |       1 |            140 |           160 | 2023-09-27 22:58:05 |
+----+---------+----------------+---------------+---------------------+
1 row in set (0.00 sec)

mysql> update sales set quantity = 110 where id=2/
Query OK, 0 rows affected (0.00 sec)
Rows matched: 1  Changed: 0  Warnings: 0

mysql> select * from sales/
+----+--------------------------------+----------+------------+-------------+--------+
| id | product                        | quantity | fiscalyear | fiscalmonth | remark |
+----+--------------------------------+----------+------------+-------------+--------+
|  1 | 2001 Ferrari Enzo              |      160 |       2021 |           1 | NULL   |
|  2 | 1998 Chrysler Plymouth Prowler |      110 |       2021 |           1 | NULL   |
|  3 | 1913 Ford Model T Speedster    |      120 |       2021 |           1 | NULL   |
+----+--------------------------------+----------+------------+-------------+--------+
3 rows in set (0.00 sec)
```

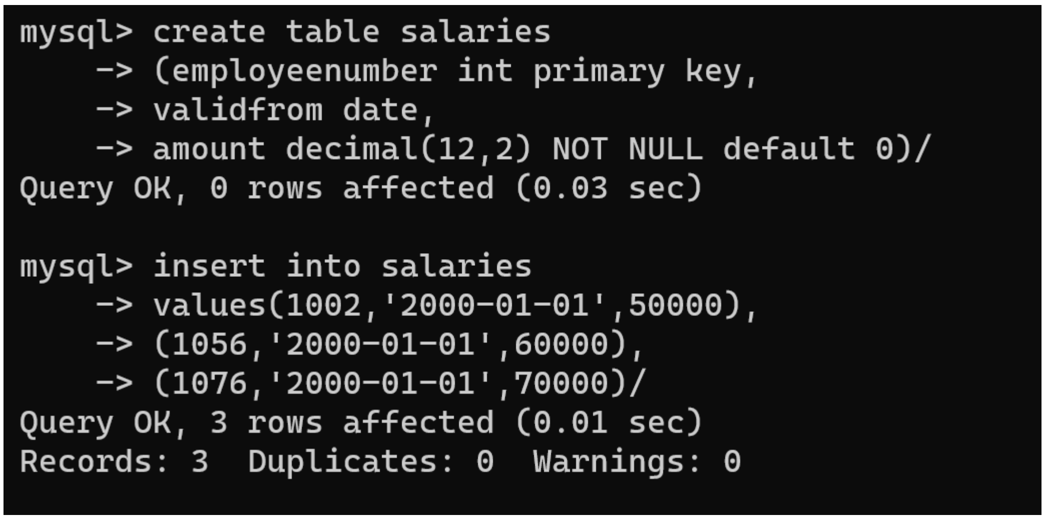
5. **Create a table Salaries with the following data**

Column	**Datatype**	**Constraint**
employeenumber	**Int**	**Primary key**
validFrom	**date**	
amount	**Decimal(12,2)**	Not Null Default 0

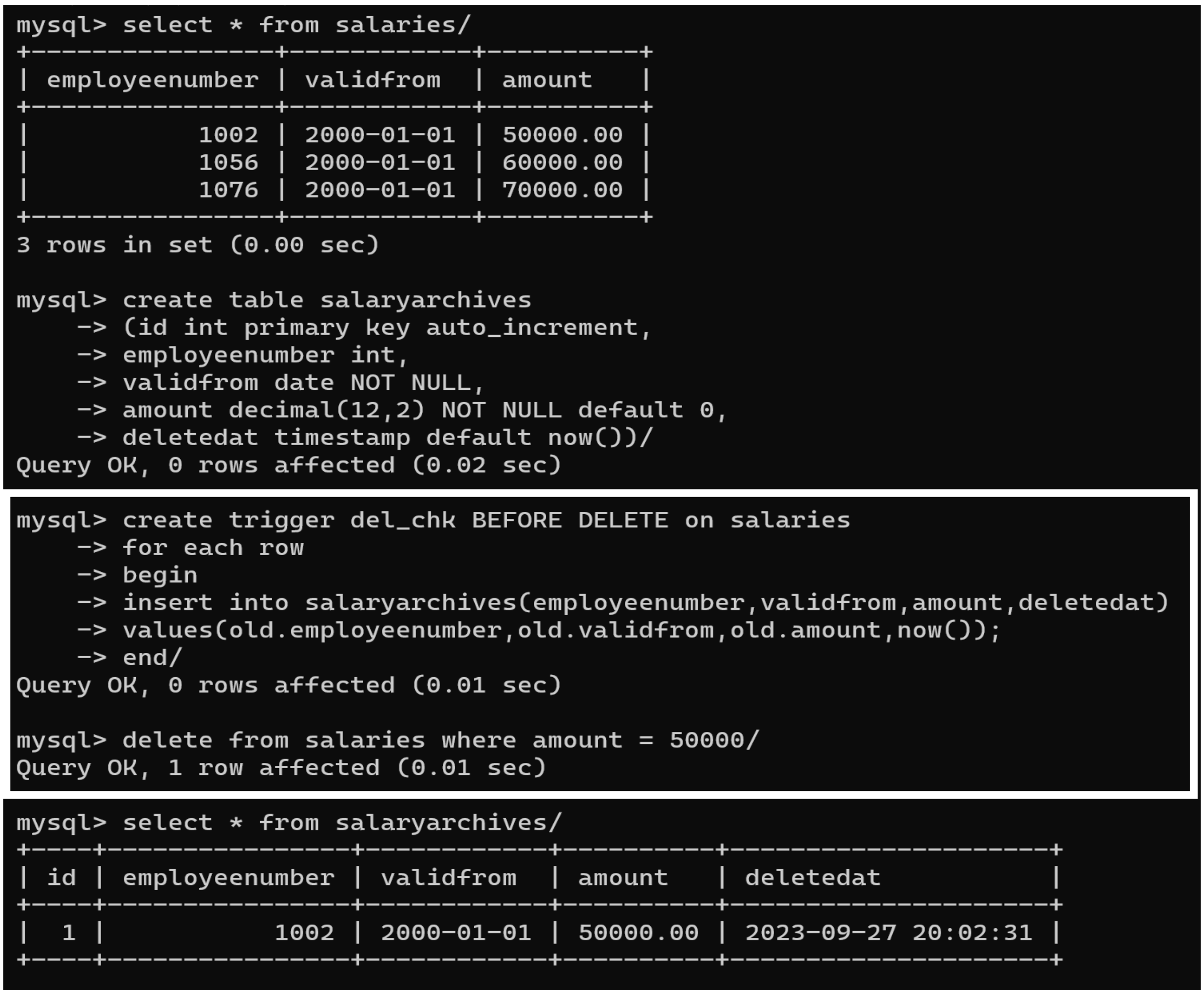
INSERT 3 rows in the table the following VALUES
1. 1002,'2000-01-01',50000
2. 1056,'2000-01-01',60000
3. 1076,'2000-01-01',70000
Create a table SalaryArchives with the following data

Column	**Datatype**	**Constraint**
id	**Int**	**Primary key autoincremen**
employeenumber	**Int**	
validFrom	**date**	**Not Null**
amount	**Decimal(12,2)**	**Not Null Default 0**
Deletedat	**Timestamp**	**Default now()**

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,
    -> amount decimal(12,2) NOT NULL default 0)/
Query OK, 0 rows affected (0.03 sec)

mysql> insert into salaries
    -> 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> select * from salaries/
+----------------+------------+----------+
| employeenumber | validfrom  | amount   |
+----------------+------------+----------+
|           1002 | 2000-01-01 | 50000.00 |
|           1056 | 2000-01-01 | 60000.00 |
|           1076 | 2000-01-01 | 70000.00 |
+----------------+------------+----------+
3 rows in set (0.00 sec)

mysql> create table salaryarchives
    -> (id int primary key auto_increment,
    -> employeenumber int,
    -> validfrom date NOT NULL,
    -> amount decimal(12,2) NOT NULL default 0,
    -> deletedat timestamp default now())/
Query OK, 0 rows affected (0.02 sec)
mysql> create trigger del_chk BEFORE DELETE on salaries
    -> for each row
    -> begin
    -> insert into salaryarchives(employeenumber,validfrom,amount,deletedat)
    -> values(old.employeenumber,old.validfrom,old.amount,now());
    -> end/
Query OK, 0 rows affected (0.01 sec)

mysql> delete from salaries where amount = 50000/
Query OK, 1 row affected (0.01 sec)
mysql> select * from salaryarchives/
+----+----------------+------------+----------+---------------------+
| id | employeenumber | validfrom  | amount   | deletedat           |
+----+----------------+------------+----------+---------------------+
|  1 |           1002 | 2000-01-01 | 50000.00 | 2023-09-27 20:02:31 |
+----+----------------+------------+----------+---------------------+
```

6. **Drop the table salaries**
   Create a table Salaries with the following data

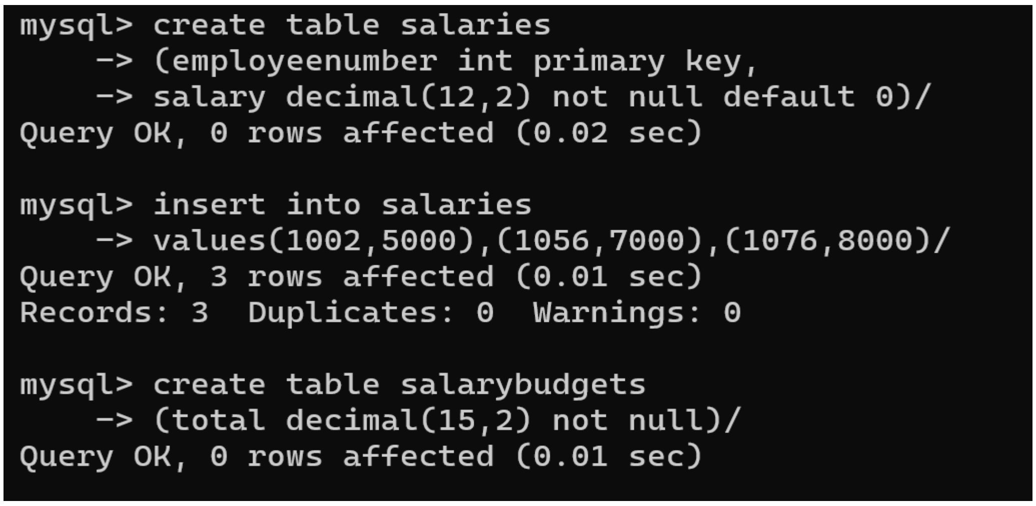
Column	Datatype	Constraint
employeenumber	Int	Primary key
salary	Decimal(12,2)	Not Null Default 0

INSERT 3 rows in the table the following VALUES
1. 1002,5000
2. 1056,,7000
3. 1076,8000
Create a table SalaryBudgets with the following data

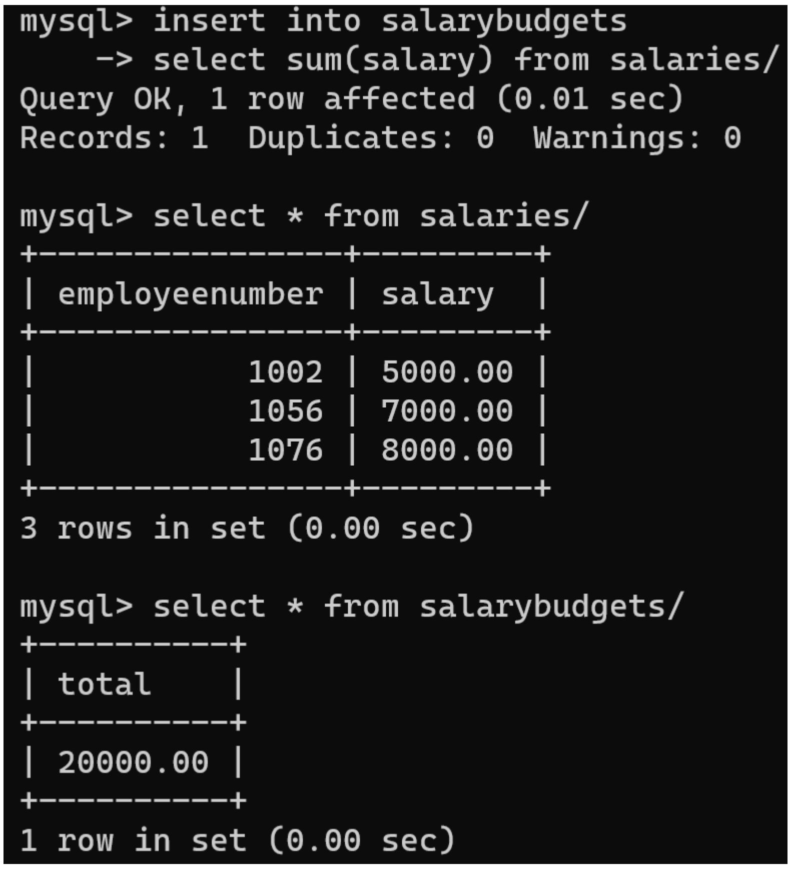
Column	Datatype	Constraint
total	Decimal(15,2)	Not Null

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

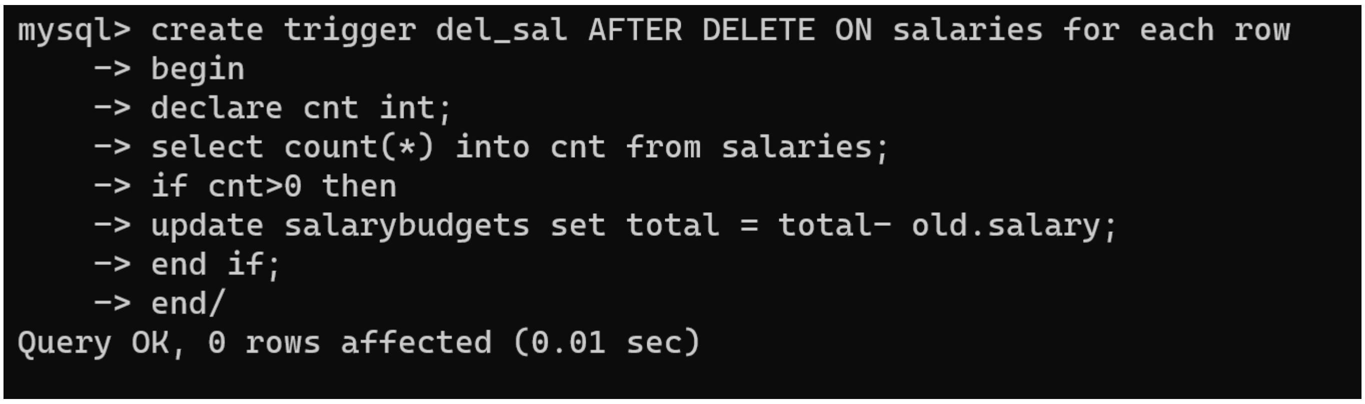
```
mysql> create trigger del_sal AFTER DELETE ON salaries for each row
   -> begin
   -> declare cnt int;
   -> select count(*) into cnt from salaries;
   -> if cnt>0 then
   -> update salarybudgets set total = total- old.salary;
   -> end if;
   -> end/
Query OK, 0 rows affected (0.01 sec)
```



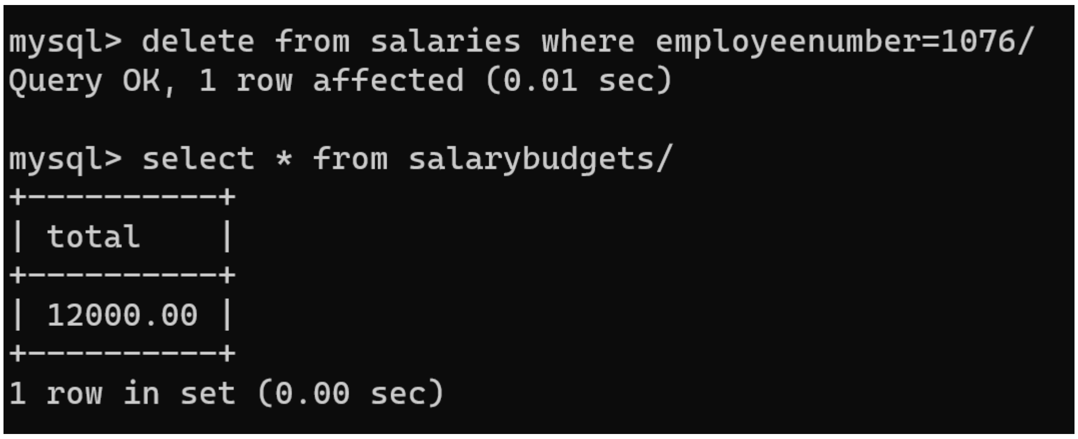
```
mysql> create table salaries
    -> (employeenumber int primary key,
    -> salary decimal(12,2) not null default 0)/
Query OK, 0 rows affected (0.02 sec)

mysql> insert into salaries
    -> values(1002,5000),(1056,7000),(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
    -> select sum(salary) from salaries/
Query OK, 1 row affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> select * from salaries/
+----------------+---------+
| employeenumber | salary  |
+----------------+---------+
|           1002 | 5000.00 |
|           1056 | 7000.00 |
|           1076 | 8000.00 |
+----------------+---------+
3 rows in set (0.00 sec)

mysql> select * from salarybudgets/
+----------+
| total    |
+----------+
| 20000.00 |
+----------+
1 row in set (0.00 sec)
```



```
mysql> create trigger del_sal AFTER DELETE ON salaries for each row
    -> begin
    -> declare cnt int;
    -> select count(*) into cnt from salaries;
    -> if cnt>0 then
    -> update salarybudgets set total = total- old.salary;
    -> end if;
    -> end/
Query OK, 0 rows affected (0.01 sec)
```



```
mysql> delete from salaries where employeenumber=1076/
Query OK, 1 row affected (0.01 sec)

mysql> select * from salarybudgets/
+----------+
| total    |
+----------+
| 12000.00 |
+----------+
1 row in set (0.00 sec)
```