Name : Chandradhar Rao
SRN : PES1UG19CS123
SQL – Creating Triggers and Functions

1. Create an employee table which contains employee details and the department he works for. Create another table department consisting of dname and number of employees. Write triggers to increment or decrement the number of employees in a department table when the record in the employee table is inserted or deleted respectively.

```
drop table if exists department;
drop table if exists employee;
drop function if exists change_emp_count;
drop function if exists inc_emp_count;
drop function if exists dec_emp_count;

CREATE TABLE employee(
   id serial not null,
   name text not null,
   ssn char(3) not null,

   dept_id integer not null
);

CREATE TABLE department(
   id serial not null,
   name text not null,
   num_emps INTEGER not NULL
);

--Trigger function
CREATE FUNCTION inc_emp_count()
   RETURNS TRIGGER
   LANGUAGE  PLPGSQL
AS $$
BEGIN
   update department set num_emps = num_emps+1
   where id = new.dept_id; --new refers to the record isnerted
   return NULL;
END;
$$;
```

```
create function dec_emp_count()
   returns trigger
   language plpgsql
as $$
BEGIN
   update department set num_emps = num_emps -1
   where id=old.dept_id; --old : refers to the value deleted
   return old; --return NOT NULL
END;
$$;

--creating the trigger itself
create trigger tr_tblEmployee_ForInsert
After insert
on employee
for each row
execute procedure inc_emp_count();

create trigger tr_tblEmployee_ForDelete
after delete
on employee
for each row
execute procedure dec_emp_count();

--insert statements
insert into department (name,num_emps) values
('IT',0),('ME',0),('PSY',0),('EEE',0);
insert into employee (name,ssn,dept_id) values
('emma','123',1),('mia','124',1),('dwayne','125',1),('ethan','126',3),('rahane','12
7',3),('virat','128',4),('zazai','129',2),('mujeeb','130',4);

--display updated tables
select * from department;
select * from employee;

--delete commands
delete from employee where id=3;
delete from employee where id=4;
delete from employee where id=2;

--display the tables
select * from department;
```

```
select * from employee;
```

Output tables of the above command:

1)After Inserting records:

```
test1db=# --display updated tables
test1db=# select * from department;
 id | name | num_emps
----+------+----------
  1 | IT   |        3
  3 | PSY  |        2
  2 | ME   |        1
  4 | EEE  |        2
(4 rows)

test1db=# select * from employee;
 id |  name  | ssn | dept_id
----+--------+-----+---------
  1 | emma   | 123 |       1
  2 | mia    | 124 |       1
  3 | dwayne | 125 |       1
  4 | ethan  | 126 |       3
  5 | rahane | 127 |       3
  6 | virat  | 128 |       4
  7 | zazai  | 129 |       2
  8 | mujeeb | 130 |       4
(8 rows)
```

➔ **As we can see that for every entry in the employee table,the numumber of employee count has been increased due to the trigger function getting executed on the event of insert.**

After Deleting Records:

```
test1db=# --display the tables
test1db=# select * from department;
 id | name | num_emps
----+------+----------
  2 | ME   |        1
  4 | EEE  |        2
  3 | PSY  |        1
  1 | IT   |        1
(4 rows)

test1db=# select * from employee;
 id |  name  | ssn | dept_id
----+--------+-----+---------
  1 | emma   | 123 |       1
  5 | rahane | 127 |       3
  6 | virat  | 128 |       4
  7 | zazai  | 129 |       2
  8 | mujeeb | 130 |       4
(5 rows)
```

- ➔ **As we can see,when we deleted records,the trigger function updated the values from the department table too.**
- ➔ **Import ant point here is that we need to use "old" keyword to point t othe value after we delete it in order to update the correct record.**
- ➔ **Also we should return the "old" value back as a nomenclature.**

2. Create an order_item table which contains details like name, quantity and unit price of every item purchased. Create an order summary table that contains number of items and total price. Create triggers to update entry in order summary whenever an item is inserted or deleted in the order item table.

```
drop table if exists order_item;
drop table if exists order_summary;
drop function if exists inc_order_summ;
drop function if exists dec_order_summ;

--create shopping cart table
create table order_item(
   id serial not null,
   name text not null,
   quantity integer not null,
   per_unit_price real not null
);

--create order summary table
create table order_summary(
   num_items integer not null,
   total_price real not null
);

--Trigger function
create function inc_order_summ()
   returns trigger
   language PLPGSQL
as $$
BEGIN
   update order_summary set num_items = num_items + new.quantity;
   update order_summary set total_price = total_price +
(new.per_unit_price*new.quantity);
   return null;
END;
$$;

create function dec_order_summ()
   returns trigger
   language PLPGSQL
as $$
BEGIN
   update order_summary set num_items = num_items - old.quantity;
```

```
   update order_summary set total_price = total_price -
(old.per_unit_price*old.quantity);
   return old;
END;
$$;

--Create the triggers
create trigger tr_tblOrderItem_ForInsert
after insert
on order_item
for each row
execute procedure inc_order_summ();

create trigger tr_tblOrderItem_ForDelete
after delete
on order_item
for each row
execute procedure dec_order_summ();

--insert statements
insert into order_summary(num_items,total_price) values(0,0);
insert into order_item(name,quantity,per_unit_price)
values('redmi_note7pro',1,14000),('apples',7,35),('oreo',3,20),('pens',5,1.5),('soc
ks',2,50),('apple_s6',1,20000),('hair_gel',2,100);

--display tables
select * from order_item;
select * from order_summary;

--delete commands
delete from order_item where name like 'socks';
delete from order_item where name like 'apple_s6';
delete from order_item where name like 'hair_gel';

--display tables
select * from order_item;
select * from order_summary;
```

Outputs of the above commands:

1) Table after insertion of records:

```
test1db=# --display tables
test1db=# select * from order_item;
 id |      name       | quantity | per_unit_price
----+-----------------+----------+----------------
  1 | redmi_note7pro |        1 |          14000
  2 | apples          |        7 |             35
  3 | oreo            |        3 |             20
  4 | pens            |        5 |            1.5
  8 | redmi_note7pro |        1 |          14000
  9 | apples          |        7 |             35
 10 | oreo            |        3 |             20
 11 | pens            |        5 |            1.5
 12 | socks           |        2 |             50
 13 | apple_s6        |        1 |          20000
 14 | hair_gel        |        2 |            100
(11 rows)

test1db=# select * from order_summary;
 num_items | total_price
-----------+-------------
        21 |     34612.5
(1 row)
```

- ➔ **When a new item with desired number of quantities is added into the cart,it is also added into the order_item table.**
- ➔ **Using trigger function,we also automatically add it to the order_summary table.**
- ➔ **We increment already existing quantity by the new item quantity and the price by (new item quantity * new item per unit price) since the price is "per" unit of the item.**

2)Tables after deleting records:

```
test1db=# --display tables
test1db=# select * from order_item;
 id |      name       | quantity | per_unit_price
----+-----------------+----------+----------------
  1 | redmi_note7pro |        1 |          14000
  2 | apples          |        7 |             35
  3 | oreo            |        3 |             20
  4 | pens            |        5 |            1.5
  8 | redmi_note7pro |        1 |          14000
  9 | apples          |        7 |             35
 10 | oreo            |        3 |             20
 11 | pens            |        5 |            1.5
(8 rows)

test1db=# select * from order_summary;
 num_items | total_price
-----------+-------------
        16 |     14312.5
(1 row)
```

- ➔ **When an item from the cart is delete into the cart,it should also be deleted from the order_item table.**
- ➔ **Using trigger function,we also automatically delete it from the order_summary table.**
- ➔ **We decrement already existing quantity by the item quantity we want to delete and the price by (item quantity *item per unit price) since the price is "per" unit of the item.**
- ➔ **This way the order summary is updated with the new cart.**