Program Studi : Teknik Informatika

Laporan Praktikum : Basis Data 2

Praktikum 5

Muhammad Azhar Rasyad
0110217029

STT Terpadu Nurul Fikri
Tahun 2018

Triggers

Tugas Pendahuluan

1. Jelaskan apa yang dimaksud dengan triggers !

Triggers adalah aksi yang dilakukan jika terjadi perubahan pada row data suatu table.

2. Jelaskan keuntungan dan kerugian dari triggers !

Keuntungan Triggers :

- Trigger menyediakan cara alternative untuk memeriksa integritas.
- Trigger blas menangkap kesalahan dalam business logic pada tingkat database.
- Trigger menyediakan cara alternative untuk menjalankan tugas-tugas yang dijadwalkan.
- Trigger sangat berguna untuk mengaudit perubahan data dalam table database

Kerugian Triggers :

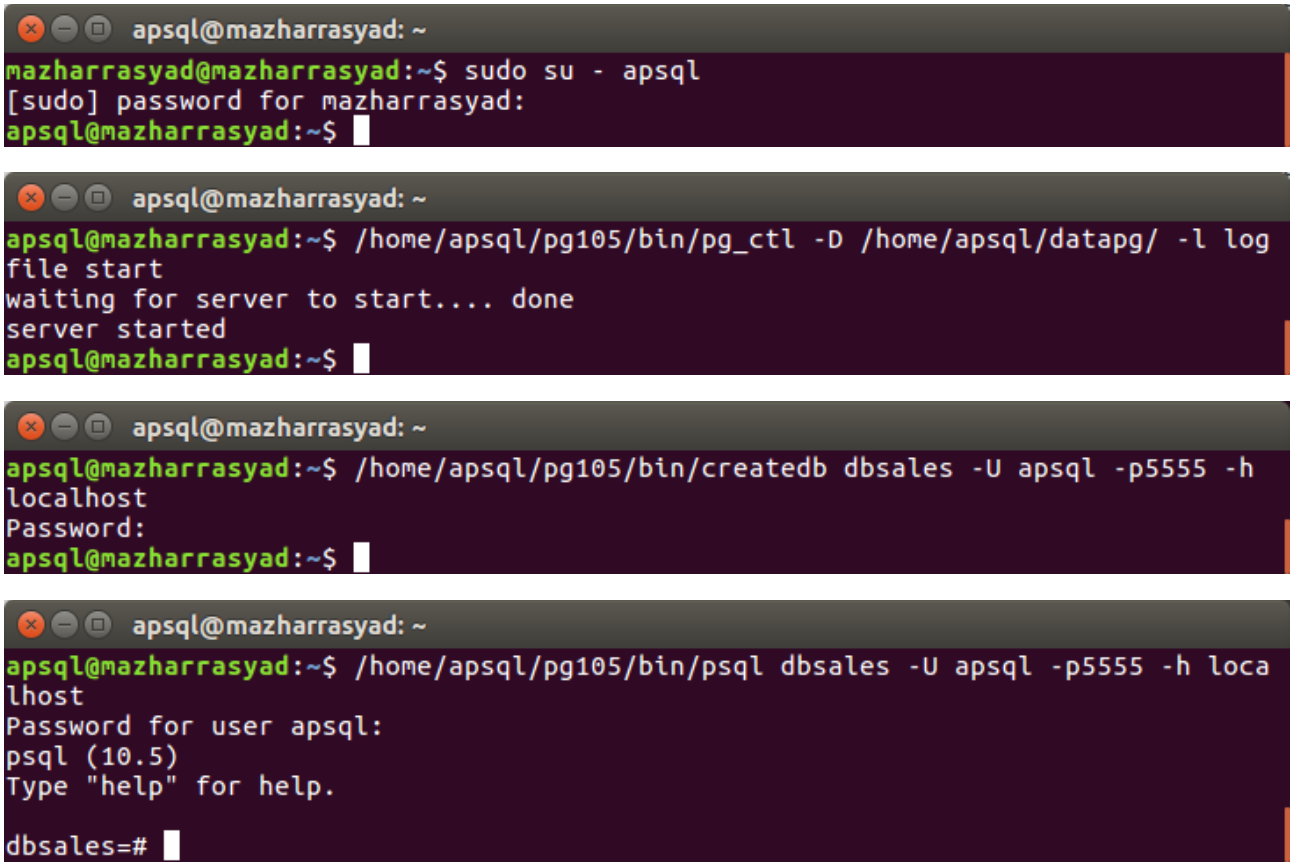
- Trigger hanya bisa menyediakan validasi tambahan tapi tidak dapat menggantikan semua validasi.
- Beberapa validasi sederhana dapat dilakukan di level aplikasi.
- Trigger mengeksekusi secara tak terlihat dari klien-aplikasi yang terhubung ke database server sehingga sulit untuk mencari tahu apa yang terjadi di level database.
- Trigger berjalan setiap update yang dibuat ke table sehingga menambah beban kerja ke database dan menyebabkan system berjalan lebih lambat.

Sumber :

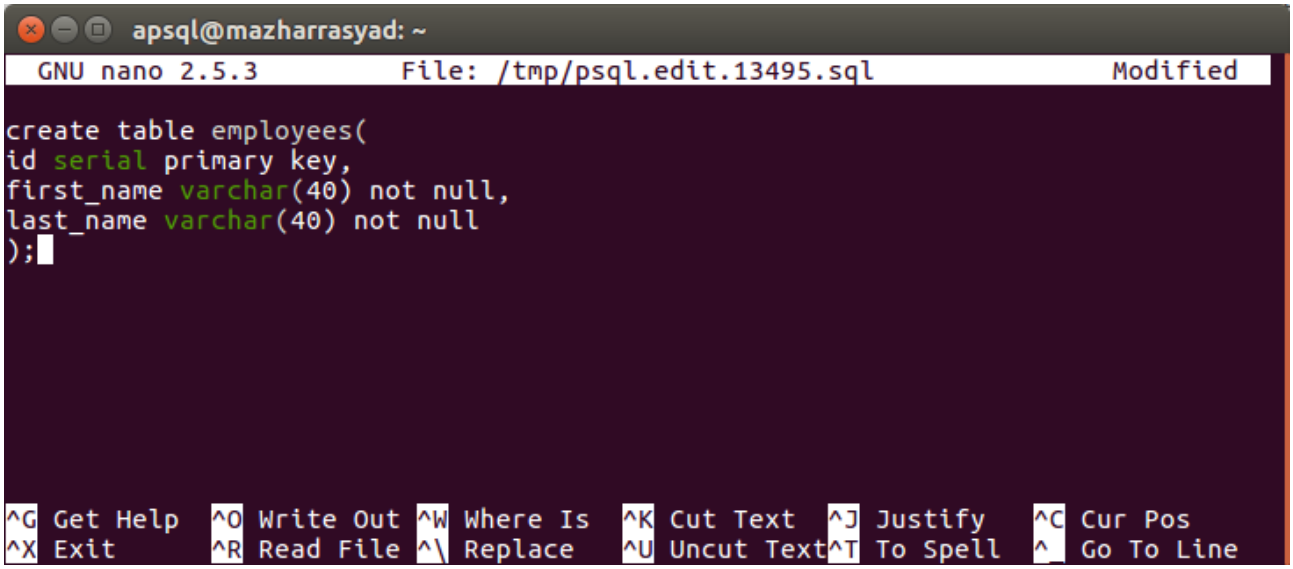
http://apanyaa.blogspot.com/2012/03/laporan-matakuliah-pengolahan-basis_15.html

Percobaan 1 : Skema Table

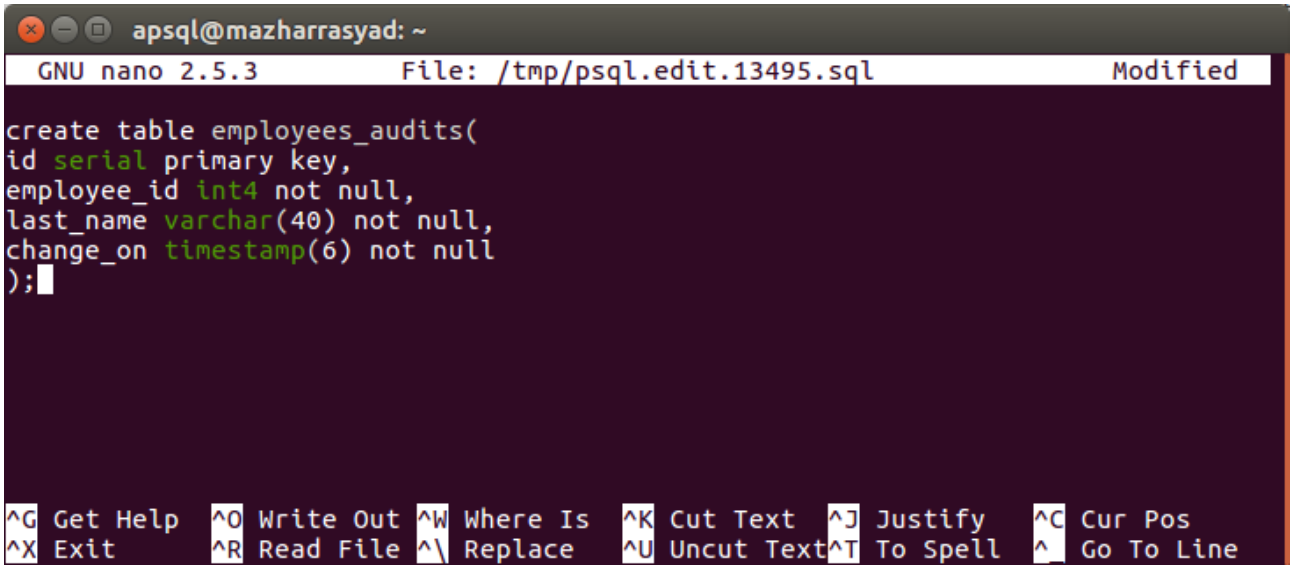
1. Buat database dbsales



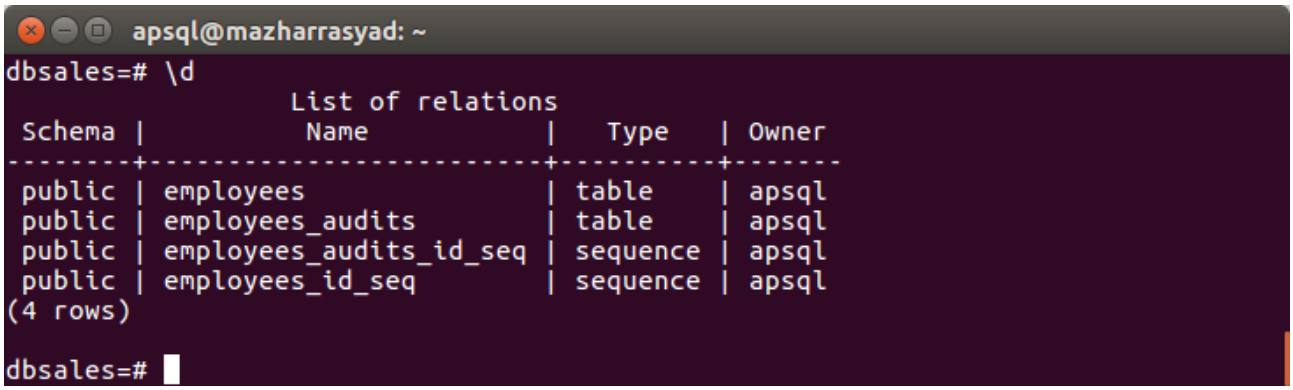
2. Buat table employees dengan perintah berikut :



3. Buat table employees_audits dengan perintah berikut :

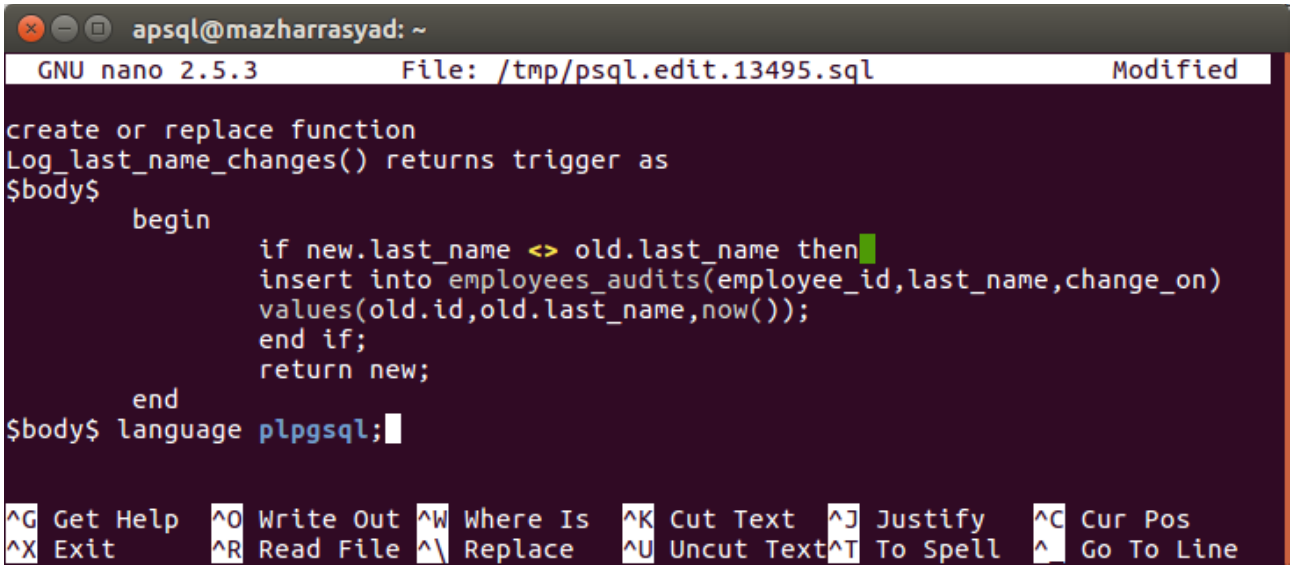


4. List table :

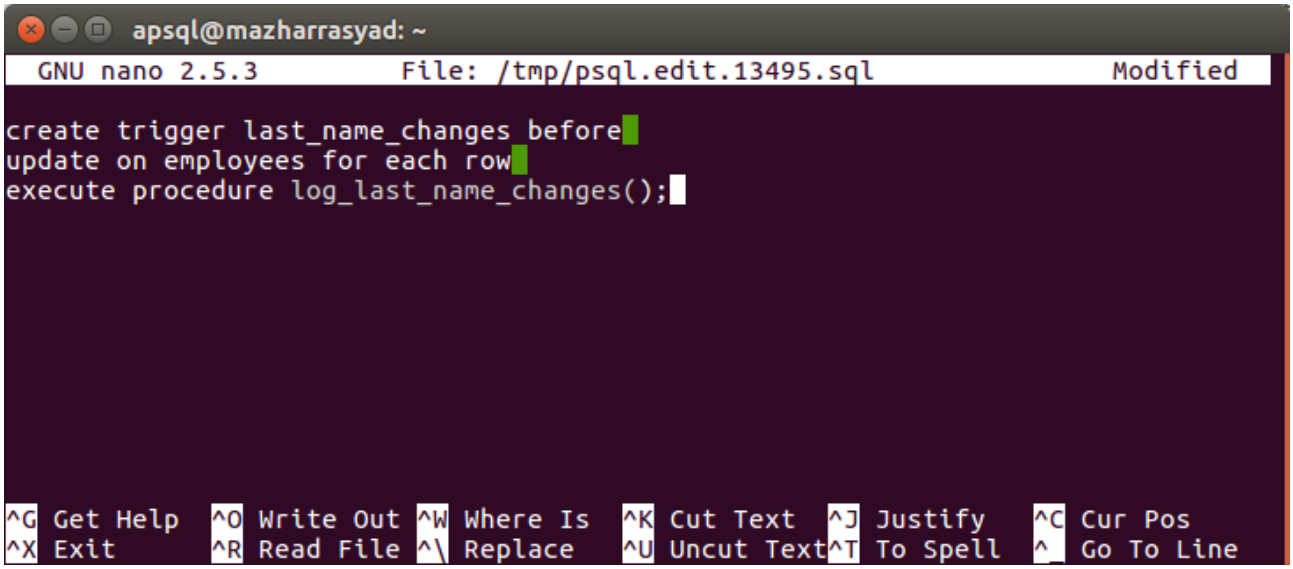


Percobaan 2 : Buat Triggers

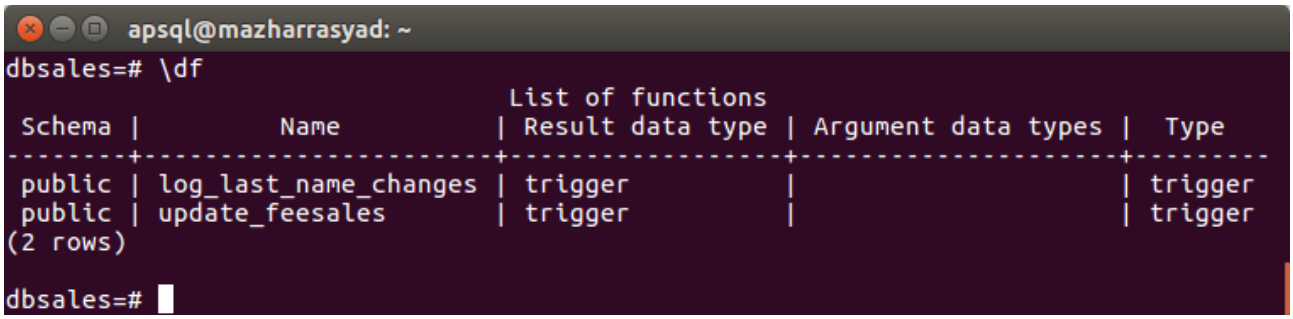
1. Buat fungsi dengan nama log_last_name_changes()



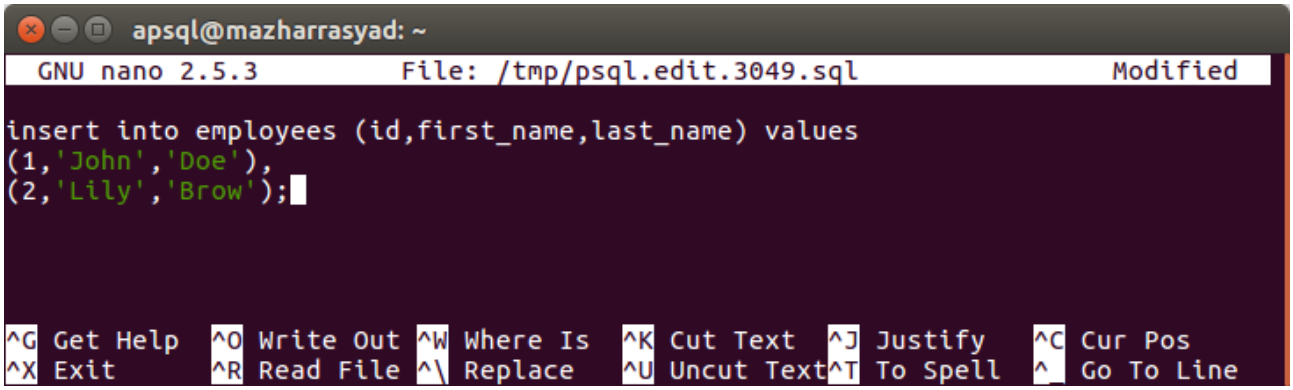
2. Buat trigger yang akan memanggil fungsi log_last_name_changes()



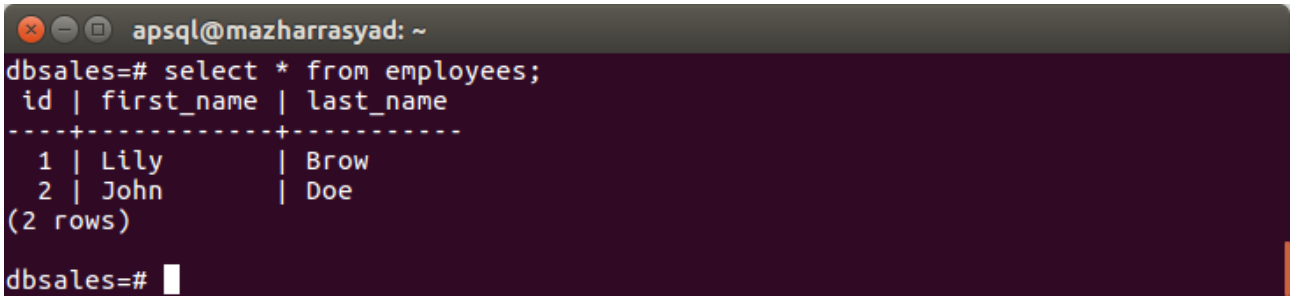
3. List Function



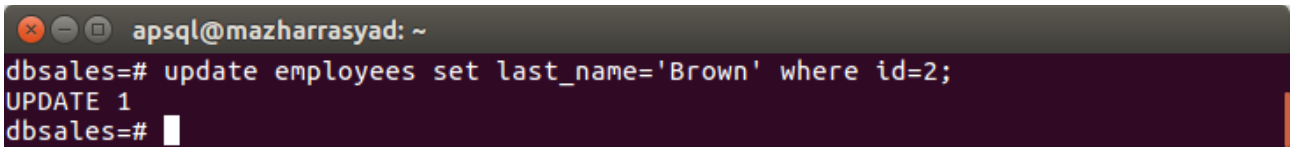
4. Menambahkan data table employees



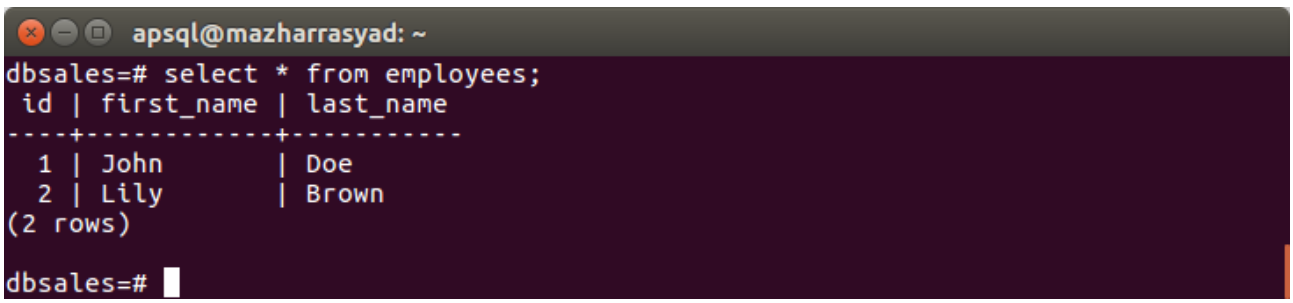
5. Menampilkan data table employees



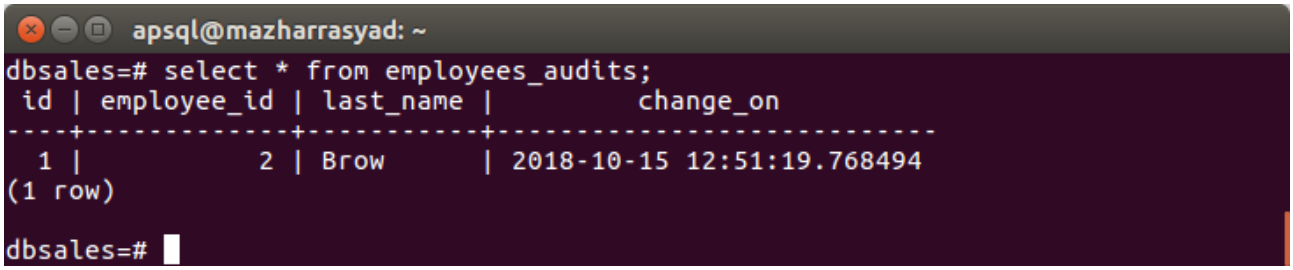
6. Mengubah data Brown menjadi Brown



7. Tampilan setelah diubah



8. Tampilan log setelah diupdate



Percobaan 3 : Studi kasus transaksi sales

1. Buat tabel transaksi dengan skema berikut ini : **(field id SERIAL, sales_id foreign key ke table employees)**

```
apsql@mazharrasyad: ~
  GNU nano 2.5.3          File: /tmp/psql.edit.3049.sql                Modified

create table transaksi(
id serial primary key,
tanggal date default now(),
jumlah double precision,
sales_id integer references employees(id)
);

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

```
apsql@mazharrasyad: ~
dbsales=# \d transaksi
                                  Table "public.transaksi"
  Column  |       Type       | Collation | Nullable |                 Default
----------+------------------+-----------+----------+-----------------------------------
-
 id       | integer          |           | not null | nextval('transaksi_id_seq'::regclass)
 tanggal  | date             |           |          |
 jumlah   | double precision |           |          |
 sales_id | integer          |           |          |
Indexes:
    "transaksi_pkey" PRIMARY KEY, btree (id)
Foreign-key constraints:
    "transaksi_sales_id_fkey" FOREIGN KEY (sales_id) REFERENCES employees(id)
Triggers:
    trig_update_fee AFTER INSERT OR UPDATE ON transaksi FOR EACH ROW EXECUTE PROCEDURE upda
te_feesales()

dbsales=#
```

2. Tambahkan field persen_fee dan total_fee dengan tipe data double precision pada table employees dan update datanya menjadi seperti berikut ini

```
apsql@mazharrasyad: ~
dbsales=# alter table employees add persen_fee double precision;
ALTER TABLE
dbsales=# alter table employees add total_fee double precision;
ALTER TABLE
dbsales=#
```

```
apsql@mazharrasyad: ~
dbsales=# select * from employees;
 id | first_name | last_name | persen_fee | total_fee
----+------------+-----------+------------+-----------
  1 | John       | Doe       |            |
  2 | Lily       | Brown     |            |
(2 rows)

dbsales=#
```

```
apsql@mazharrasyad: ~
  GNU nano 2.5.3          File: /tmp/psql.edit.3049.sql                Modified

update employees set persen_fee=0.2 where id=1;
update employees set persen_fee=0.15 where id=2;
update employees set total_fee=0 where id in(1, 2);



^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

```
apsql@mazharrasyad: ~
dbsales=# select * from employees;
 id | first_name | last_name | persen_fee | total_fee
----+------------+-----------+------------+-----------
  1 | John       | Doe       |        0.2 |         0
  2 | Lily       | Brown     |       0.15 |         0
(2 rows)

dbsales=#
```

3. Buat fungsi untuk update total_fee untuk setiap transaksi yang pernah dilakukan oleh sales

```
apsql@mazharrasyad: ~
  GNU nano 2.5.3                File: /tmp/psql.edit.15952.sql                     Modified

create or replace function
update_feesales() returns trigger as
$body$
        begin
                update employees set total_fee = total_fee + (persen_fee * new.jumlah) where id = new.sales_id;
                return new;
        end
$body$ language plpgsql;


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     ^Y Prev Page
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line  ^V Next Page
```

4. Buat trigger yang akan menjalankan fungsi update_feesales() ketika data baru dimasukan ke table transaksi

```
apsql@mazharrasyad: ~
  GNU nano 2.5.3          File: /tmp/psql.edit.15952.sql                Modified

create trigger trig_update_fee after
insert or update on transaksi for each row
execute procedure update_feesales();


^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

5. Insert data ke table transaksi !!

```
apsql@mazharrasyad: ~
  GNU nano 2.5.3          File: /tmp/psql.edit.3049.sql                 Modified

insert into transaksi (jumlah,sales_id) values
(20000,1),
(500000,1),
(2500000,2);


^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

6. Tampilkan data hasil transaksi !!

```
apsql@mazharrasyad: ~
dbsales=# select * from transaksi;
 id |  tanggal   | jumlah  | sales_id
----+------------+---------+----------
  1 | 2018-10-15 |   20000 |        1
  2 | 2018-10-15 |  500000 |        1
  3 | 2018-10-15 | 2500000 |        2
(3 rows)

dbsales=#
```

7. Tampilkan data employees, apakah total_fee telah terupdate !!

```
apsql@mazharrasyad: ~
dbsales=# select * from employees;
 id | first_name | last_name | persen_fee | total_fee
----+------------+-----------+------------+-----------
  1 | John       | Doe       |        0.2 |    104000
  2 | Lily       | Brown     |       0.15 |    375000
(2 rows)

dbsales=#
```

---------- *Selesai* ----------