-- SALES

--nugraha : procedure addtocart()

--sidiq : update status sales order header

--yogi : function ref cursor

-- 1. procedure addtocart()

--# procedure addtocart()

create or replace procedure sales.add\_to\_cart(

in p\_customer\_id integer,

in p\_salesperson\_id integer,

in p\_product\_id integer,

in p\_quantity integer

--p = parameter

)

language plpgsql as $$

declare

v\_cart\_id integer;

v\_order\_id integer;

v\_item\_count integer;

v\_order\_status varchar(50);

v\_existing\_order boolean;

v\_existing\_cart boolean;

v\_existing\_item boolean;

v\_total\_price numeric;

--v = vari lokal

begin

v\_existing\_order := false;

v\_existing\_cart := false;

v\_existing\_item := false;

-- cek jika ada order status open, disimpan ke v\_item\_count

select count(\*)

into v\_item\_count

from sales.salesorderheader soh

join sales.salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = ('open');

if v\_item\_count > 0 then

v\_existing\_order := true; --jika ada open, jadi true

select soh.salesorderid, soh.status

into v\_order\_id, v\_order\_status --simpan ke sini

from salesorderheader soh

join salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = sales.orderstatus('open')

limit 1; --jika ada open, ambil 1 baris

end if;

-- jika ada order open, tambah ke salesorderdetail

if v\_existing\_order then

insert into salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

v\_existing\_item := true;

-- jika tak ada open, tambah baru ke salesorderdetail

else

insert into salesorderheader (customerid, salespersonid, orderdate, status)

values (p\_customer\_id, p\_salesperson\_id, now(), 'open')

returning salesorderid into v\_order\_id;

insert into salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

end if;

-- hitung item di cart

select count(\*)

into v\_item\_count

from cart

where customerid = p\_customer\_id

and salespersonid = p\_salesperson\_id

and closedate is null;

if v\_item\_count > 0 then

v\_existing\_cart := true; --jika item > 0, jadi true

end if;

-- jika ada item di cart, update date jadi sekarang

if v\_existing\_cart then

update cart

set modifieddate = now()

where customerid = p\_customer\_id

and salespersonid = p\_salesperson\_id

and closedate is null;

-- jika tak ada, buat baru

else

insert into cart (customerid, salespersonid, createdate)

values (p\_customer\_id, p\_salesperson\_id, now());

end if;

-- hitung total price

select listprice \* p\_quantity

into v\_total\_price

from product

where productid = p\_product\_id;

-- return total price

p\_total\_price := v\_total\_price; --hasil disimpan ke v\_total\_price

end; $$;

call sales.add\_to\_cart(1, 2, 3, 4);

select \* from salesorderheader;

select \* from salesorderdetail;

drop procedure sales.add\_to\_cart();

/\* PERBAIKAN QUERY procedure sales.add\_to\_cart()

create or replace procedure sales.add\_to\_cart(

in p\_customer\_id integer,

in p\_salesperson\_id integer,

in p\_product\_id integer,

in p\_quantity integer

--p = parameter

)

language plpgsql as $$

declare

v\_cart\_id integer;

v\_order\_id integer;

v\_item\_count integer;

v\_order\_status varchar(50);

v\_existing\_order boolean;

v\_existing\_cart boolean;

v\_existing\_item boolean;

v\_total\_price numeric;

--v = vari lokal

begin

v\_existing\_order := false;

v\_existing\_cart := false;

v\_existing\_item := false;

-- cek jika ada order status open, disimpan ke v\_item\_count

select count(\*)

into v\_item\_count

from sales.salesorderheader soh

join sales.salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = 1; --status 1 = open. jika pakai string 'open' muncul eror

if v\_item\_count > 0 then

v\_existing\_order := true; --jika ada open, jadi true

select soh.salesorderid, soh.status

into v\_order\_id, v\_order\_status --simpan ke sini

from sales.salesorderheader soh

join sales.salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = 1

limit 1; --jika ada open, ambil 1 baris

end if;

-- jika ada order open, tambah ke salesorderdetail

if v\_existing\_order then

insert into sales.salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

v\_existing\_item := true;

-- jika tak ada open, tambah baru ke salesorderdetail

else

insert into sales.salesorderheader (customerid, salespersonid, orderdate, duedate, status)

values (p\_customer\_id, p\_salesperson\_id, now(), now(), 1)

returning salesorderid into v\_order\_id;

insert into sales.salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

end if;

-- hitung item di cart

select count(\*)

into v\_item\_count

from sales.salesorderheader soh

join sales.salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.duedate is null;

if v\_item\_count > 0 then

v\_existing\_cart := true; --jika item > 0, jadi true

end if;

-- jika ada item di cart, update modifieddate jadi sekarang

if v\_existing\_cart then

update sales.salesorderdetail

set modifieddate = now()

where customerid = p\_customer\_id

and salespersonid = p\_salesperson\_id

and duedate is null;

-- jika tak ada, buat baru

else

insert into sales.salesorderheader (customerid, salespersonid, orderdate)

values (p\_customer\_id, p\_salesperson\_id, now());

end if;

-- hitung total price

select unitprice \* p\_quantity

into v\_total\_price

from sales.salesorderdetail

where productid = p\_product\_id;

end; $$;

--call sales.add\_to\_cart(1, 279, 776, 1);

--select\*from sales.salesorderdetail

--select\*from sales.salesorderheader

/\*contoh pemanggilan

select \* from sales.salesorderheader soh

join sales.salesorderdetail sod on soh.salesorderid = sod.salesorderid

where soh.customerid = 1

and soh.salespersonid = 279

and soh.duedate is null;

\*/

--drop procedure if exists sales.add\_to\_cart;

--perubahan/alter table

--alter table sales.salesorderheader alter column billtoaddressid drop not NULL;

--alter table sales.salesorderheader alter column shiptoaddressid drop not NULL;

--alter table sales.salesorderheader alter column shipmethodid drop not NULL;

--alter table sales.salesorderheader alter column creditcardid drop not NULL;

--alter table sales.salesorderdetail alter column specialofferid drop not NULL;

--alter table sales.salesorderdetail alter column unitprice drop not NULL;

--alter table sales.salesorderheader alter column duedate drop not NULL;

--alter table sales.salesorderheader alter column billtoaddressid set not null;

--alter table sales.salesorderheader alter column shiptoaddressid set not null;

--alter table sales.salesorderheader alter column shipmethodid set not null;

--alter table sales.salesorderheader alter column creditcardid set not null;

--alter table sales.salesorderdetail alter column specialofferid set not null;

--alter table sales.salesorderdetail alter column unitprice set not null;

--alter table sales.salesorderheader alter column duedate set not null;

--# cek status (ini ga perlu demo karena cuma kondisi, dan sdh ada dan jadi satu dgn procedure)

--perkiraan sementara

create or replace function update\_sales\_order\_detail() returns trigger as $$

--mengembalikan trigger, otomatis berjalan di db

declare order\_id integer; --deklarasi variabel, yg akan menyimpan ke salesorderid

begin

-- cek apakah ada order open pada customerid dan salespersonid yang sama

select salesorderid into order\_id from salesorderheader --mengambil data dari salesorderid ke order\_id

where customerid = new.customer\_id and salespersonid = new.salesperson\_id and status = 'open';

if order\_id is null then

-- jika tidak ada order open, buat order baru dan tambah item ke salesorderdetail

insert into salesorderheader (customerid, salespersonid, orderdate, status)

values (new.customer\_id, new.salesperson\_id, now(), 'open')

returning salesorderid into order\_id; --kembalikan nilai ke order\_id

insert into salesorderdetail (salesorderid, productid, orderqty) values (order\_id, new.product\_id, new.quantity);

else

-- jika ada order open, tambah ke salesorderdetail tanpa buat order baru

insert into salesorderdetail (salesorderid, productid, orderqty) values (order\_id, new.product\_id, new.quantity);

end if;

return new; end; $$ language plpgsql;

create trigger update\_sales\_order\_detail\_trigger after insert on sales

for each row

when (new.quantity > 0) --buat trigger jika row > 0, jika ada pertambahan

execute function update\_sales\_order\_detail(); --eksekusi

-- 2. update status sales order header

-- 3. function ref cursor

--perintah 1 versi function

create or replace function sales.add\_to\_cart(

p\_customer\_id integer,

p\_salesperson\_id integer,

p\_product\_id integer,

p\_quantity integer

)

returns numeric

language plpgsql as $$

declare

v\_cart\_id integer;

v\_order\_id integer;

v\_item\_count integer;

v\_order\_status varchar(50);

v\_existing\_order boolean;

v\_existing\_cart boolean;

v\_existing\_item boolean;

v\_total\_price numeric;

begin

v\_existing\_order := false;

v\_existing\_cart := false;

v\_existing\_item := false;

-- check if there is an open order

select count(\*)

into v\_item\_count

from sales.salesorderheader soh

join sales.salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = ('open');

if v\_item\_count > 0 then

v\_existing\_order := true;

select soh.salesorderid, soh.status

into v\_order\_id, v\_order\_status

from salesorderheader soh

join salesorderdetail sod

on soh.salesorderid = sod.salesorderid

where soh.customerid = p\_customer\_id

and soh.salespersonid = p\_salesperson\_id

and soh.status = sales.orderstatus('open')

limit 1;

end if;

-- if there is an open order, add item to salesorderdetail

if v\_existing\_order then

insert into salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

v\_existing\_item := true;

-- if there is no open order, create a new one and add item to salesorderdetail

else

insert into salesorderheader (customerid, salespersonid, orderdate, status)

values (p\_customer\_id, p\_salesperson\_id, now(), 'open')

returning salesorderid into v\_order\_id;

insert into salesorderdetail (salesorderid, productid, orderqty)

values (v\_order\_id, p\_product\_id, p\_quantity);

end if;

-- check if there is a cart

select count(\*)

into v\_item\_count

from cart

where customerid = p\_customer\_id

and salespersonid = p\_salesperson\_id

and closedate is null;

if v\_item\_count > 0 then

v\_existing\_cart := true;

end if;

-- if there is a cart, update modifieddate

if v\_existing\_cart then

update cart

set modifieddate = now()

where customerid = p\_customer\_id

and salespersonid = p\_salesperson\_id

and closedate is null;

-- if there is no cart, create a new one

else

insert into cart (customerid, salespersonid, createdate)

values (p\_customer\_id, p\_salesperson\_id, now());

end if;

-- calculate total price

select listprice \* p\_quantity

into v\_total\_price

from product

where productid = p\_product\_id;

-- return total price

return v\_total\_price;

end; $$;