

Group No:- 17
Mukund Ladani(202003039)
Kashyap Halavadia(202003040)
Lab 11(11/11/2021)
Restaurant-Franchise Management System

Programming – Stored Procedures (Functions), Triggers, Cursors, Views

Q1) You may have various roles (admin, general user, guest etc) supporting different database content access. Create Views (if any) for each of these roles to access only the required content for the role.

Ans-1)

---1 EMP (User view for employee details is through the view 'emp')

--View created for employee relation so that only required info is provided to any user or customer of our restaurant

```
create or replace view emp as
select e.eid,e.branch_license_no,(e.first_name || ' '||e.last_name)as
name,e.company_employee,e.gender ,
e.job_position,e.city as residing_city,e.joining_date from employee e

select * from emp
```

---2 i) User wants to see the menu of a particular branch then we can use view

```
create or replace view menu_of_jamnagar_branch as
select d.* from branch_serving bs natural join dish d join branch b on
bs.branch_no=b.trading_license_no
where city='Jamnagar'
```

```
select * from menu_of_jamnagar_branch
```

---2 ii)

```
create or replace view menu_of_bhuj_branch as
select d.* from branch_serving bs natural join dish d join branch b on
bs.branch_no=b.trading_license_no
where city='Bhuj'
```

---2 iii)

```
create or replace view menu_of_junagadh_branch as
    select d.* from branch_serving bs natural join dish d join branch b on
bs.branch_no=b.trading_license_no
    where city='Junagadh'
```

---2 iv)

```
create or replace view menu_of_ahmedabad_branch as
    select d.* from branch_serving bs natural join dish d join branch b on
bs.branch_no=b.trading_license_no
    where city='Ahmedabad'
```

---2 v)

```
create or replace view menu_of_surat_branch as
    select d.* from branch_serving bs natural join dish d join branch b on
bs.branch_no=b.trading_license_no
    where city='Surat'
```

```
select * from menu_of_surat_branch
```

3) View for admin to see company_name, head_manager, cin, royalty_fees, branch_trading_license_no, rating of that branch , city of branch.

```
create or replace view franchisee_company_view as
    select company_name,head_manager,fc.cin,royalty_fees,b.trading_license_no,rating,city
    from franchiseecompany as fc natural join franchisee_owned_branch as fob natural join
    branch as b
```

```
select * from franchisee_company_view
```

4) View for franchisee company (having cin = U90167XY1111LNM777881) of franchisee owned branch and other details.

```
CREATE OR REPLACE VIEW COMPANY_VIEW AS
SELECT
    head_manager,royalty_fees,b.trading_license_no,rating,locality,region,city,pin_code
FROM branch as b natural join franchisee_owned_branch as fob natural join
franchiseecompany as fc where cin='U90167XY1111LNM777881'
```

```
select * from COMPANY_VIEW
```

5) Create a view which shows the dependents of employee id=1000000003.

Ans-5)

create or replace view dependent_info_of_emp as

select (d.first_name || ' ' || d.last_name) as name, d.gender, d.relationship from dependents d where
eid=1000000003;

select * from dependent_info_of_emp;

STORED PROCEDURE

- 1) **Create a procedure for getting the details of the franchisee company whose cin number is passed as an argument in the stored procedure.**

Ans-1)**Input:-**

CREATE or REPLACE FUNCTION company_having(ccin char(21))

RETURNS integer AS \$BODY\$

DECLARE

r record;

BEGIN

FOR r IN SELECT head_manager, royalty_fees, b.trading_license_no, rating, city, pin_code

FROM branch as b natural join franchisee_owned_branch as fob natural join

franchiseecompany as fc where fc.cin=ccin

LOOP

raise notice 'Head Manager: %, Royalty Fees Paying: %, Branch no: %, Rating of that

Branch: %, Branch city: %, Pincode of City: %',

r.head_manager, r.royalty_fees, r.trading_license_no, r.rating, r.city, r.pin_code;

END LOOP;

RETURN 1;

END;

\$BODY\$ LANGUAGE plpgsql

Test:-

Select * from company_having('U90167XY1111LNM777881'::char(21))

Output:-

```

529 FROM branch as b natural join franchisee_owned_branch as tob natural join franchiseecompany as tc where tc
530
531 LOOP
532 raise notice 'Head Manager: %, Royalty Fees Paying: %, Branch no: %, Rating of that Branch: %, Branch city
533 END LOOP;
534 RETURN 1;
535 END;
536 $BODY$ LANGUAGE plpgsql
537
538 Select * from company_having('U90167XY11111NM777881'::char(21))
539
540

```

Data Output Explain Messages Notifications

NOTICE: Head Manager: Aksat Srivastava, Royalty Fees Paying: 50000.00, Branch no: 111111, Rating of that Branch: 4, Branch city: Jamnagar, Pincode of City: 340245

Successfully run. Total query runtime: 68 msec.
1 rows affected.

- 2) Create a stored procedure for finding all the branches whose expenditure on ad >given amount. Amount is passed as an argument to the function. Also test it.

Ans-2)

Input:-

create or replace function

get_branch_whose_expend_on_ad_is_more_than_given_amt(amount

numeric(6,2))returns integer as

\$\$

declare

rec record;--same datatype as a tuple of branch relation

rec_branch branch%rowtype;

expend_on_ad numeric(8,2);

cur cursor for select a.branch_no,sum(monthly_charge)as total_expend from advertisement a

group by branch_no

having sum(monthly_charge)>amount

order by total_expend desc;

begin

open cur;

raise notice 'new run';

loop

fetch cur into rec;

select *into rec_branch from branch where trading_license_no =rec.branch_no;

exit when not found;

```

        raise notice '% , expend_on_ad=%',rec_branch,rec.total_expend;--expend_on_ad;

        --raise notice ",rec;
    end loop;
return 1;
end;
$$language plpgsql;

```

Test:-

select get_branch_whose_expend_on_ad_is_more_than_given_amt(12000)

Output:- Displays all those branches and their respective monthly expenditure on ads whose monthly_expenditure>12000

```

66 create or replace function get_branch_whose_expend_on_ad_is_more_than_given_amt(amount numeric
67 $$
68 declare
69 rec record;--same datatype as a tuple of branch relation
70 rec_branch branch%rowtype;
71 expend_on_ad numeric(8,2);
72 cur cursor for select a.branch_no,sum(monthly_charge)as total_expend from advertisement a
73 group by branch_no
74 having sum(monthly_charge)>amount
75 order by total_expend desc;
76
77 begin
78     open cur;
79     raise notice 'new run';
80     loop
81         fetch cur into rec;
82         select *into rec_branch from branch where trading_license_no =rec.branch_no;
83         exit when not found;
84

```

Results 1 | Execution Log | Variables | Output

```

new run
(111113,5,"Zanzarda Road",340246,Junagadh,Saurashtra) , expend_on_ad=19000.00
(111112,3,Madhapar,340248,Bhuj,Kutch) , expend_on_ad=15000.00

```

3) Create a stored procedure to update the First names of all the employees such that all the males have Mr appended before thier name and all females have Ms appended.

Ans-3)

Input:-

```

create or replace function append_Mr_Ms()returns integer as
$$
begin

```

```

    update employee set first_name='Mr '||first_name where gender='M';

```

```

        update employee set first_name='Ms '||first_name where gender='F';
        return 1;
end;
$$language plpgsql;

```

Function call:-

```
select append_Mr_Ms();
```

Output:-

The screenshot shows a SQL IDE with a script editor and a results pane. The script editor contains the following SQL code:

```

98 select * from employee e
99 select append_Mr_Ms();
100
101 create or replace function append_Mr_Ms() returns integer as
102 $$
103 begin
104
105     update employee set first_name='Mr '||first_name where gender='M';
106     update employee set first_name='Ms '||first_name where gender='F';
107     return 1;
108 end;
109 $$language plpgsql;

```

The results pane shows the output of the function call, which is 1. Below the output, the employee table is displayed with the following data:

eid	branch_license_no	first_name	middle_name	last_name	gender	monthly_salary	contact_no
1	1,000,000,107	Mr Nirmal	Pankajkumar	Patel	M	25,000	8,443,503
2	1,000,000,037	Mr Siddharth	Bhavesh	Parmar	M	10,000	8,719,160
3	1,000,000,003	Mr Rajesh	V	Panchal	M	15,000	9,527,318
4	1,000,000,050	Mr Bijoy	Mani	kakar	M	20,000	8,218,320
5	1,000,000,700	Mr Pankaj	M	Sheth	M	25,000	9,973,503
6	1,000,000,027	Ms Ragini	K.	Chauhan	F	10,000	8,202,718
7	1,000,000,370	Ms Sanjana	[NULL]	Singh	F	15,000	9,483,607
8	1,000,000,010	Ms Shalini	Thomas	Sebastian	F	25,000	7,789,305
9	1,000,000,036	Ms Shilpa	Vraj	Kunda	F	10,000	7,802,218
10	1,000,000,001	Ms Nagma	K	Chandra	F	20,000	7,555,444
11	1,000,000,019	Mr Manqan	Manqanlal	Patil	M	15,000	8,972,200

- 4) Create a stored procedure to update the First names of all the employees to remove Mr/Ms from their respective names.

Ans-4)

Input:-

```

create or replace function remove_Mr_Ms() returns integer as
$$
begin
    update employee set
        first_name=substring(first_name,3,char_length(first_name)-2);

```

```

        return 1;
end;
$$language plpgsql;

```

Function call:-

```
select remove_Mr_Ms()
```

Output:-

The screenshot shows a SQL IDE with a script editor and a results grid. The script editor contains the following code:

```

110
111 select *from employee
112
113 create or replace function remove_Mr_Ms() returns integer as
114 $$
115 begin
116     update employee set first_name=substring(first_name,3,char_length(first_name)-2);
117     return 1;
118 end;
119 $$language plpgsql;
120
121 select remove_Mr_Ms()

```

The results grid displays the output of the function call, which is a table with 11 rows and 10 columns. The columns are: eid, branch_license_no, first_name, middle_name, last_name, gender, monthly_salary, and contact_no. The data is as follows:

	eid	branch_license_no	first_name	middle_name	last_name	gender	monthly_salary	contact_no
1	1,000,000,107	111,114	Nirmal	Pankajkumar	Patel	M	25,000	8,443,503
2	1,000,000,037	111,113	Siddharth	Bhaves	Parmar	M	10,000	8,719,160
3	1,000,000,003	111,113	Rajesh	V	Panchal	M	15,000	9,527,318
4	1,000,000,050	111,113	Bijoy	Mani	kakar	M	20,000	8,218,320
5	1,000,000,700	111,115	Pankaj	M	Sheth	M	25,000	9,973,503
6	1,000,000,027	111,114	Ragini	K.	Chauhan	F	10,000	8,202,718
7	1,000,000,370	111,114	Sanjana	[NULL]	Singh	F	15,000	9,483,607
8	1,000,000,010	111,113	Shalini	Thomas	Sebastian	F	25,000	7,789,305
9	1,000,000,036	111,115	Shilpa	Vraj	Kunda	F	10,000	7,802,218
10	1,000,000,001	111,112	Nagma	K	Chandra	F	20,000	7,555,444
11	1,000,000,019	111,115	Manqan	Manqanlal	Patil	M	15,000	8,972,200

- 5) Create a procedure for getting the details of the supplier, supplying which raw-materials at which cost and amount to which branch whose name is passed as an argument in the stored procedure.

```

CREATE or REPLACE FUNCTION supplier_view(name varchar(20))
RETURNS integer AS $BODY$
DECLARE
r record;
BEGIN

```

```

FOR r IN SELECT
ss.supplier_name,sr.office_city,ss.raw_material_name,ss.cost,st.branch_no,st.quantity_bought
FROM supplies as ss natural join supplied_to as st join raw_material as ra on
ss.raw_material_name = ra.material_name join supplier as sr on (ss.supplier_name =
sr.supplier_name and ss.supplier_office_locality = sr.office_locality and
ss.supplier_office_city=sr.office_city ) where sr.supplier_name='Preet Patel'
LOOP
raise notice 'Supplier Name: %, City of office: %, Raw material: %, Cost: %, Branch no:
%, Quantity Provided: %',
r.supplier_name,r.office_city,r.raw_material_name,r.cost,r.branch_no,r.quantity_bought;
END LOOP;
RETURN 1;
END;
$BODY$ LANGUAGE plpgsql

```

Function call:

Select supplier_view('Preet Patel')

Output:

```

554 FROM supplies as ss natural join supplied_to as st join raw_material as ra on ss.raw_material_name = i
555 LOOP
556 raise notice 'Supplier Name: %, City of office: %, Raw material: %, Cost: %, Branch no: %, Quantity Pr
557 END LOOP;
558 RETURN 1;
559 END;
560 $BODY$ LANGUAGE plpgsql
561
562 select * from supplier
563
564 Select supplier_view('Preet Patel')
565

```

Data Output	Explain	Messages	Notifications
NOTICE: Supplier Name: Preet Patel, City of office: Bhuj, Raw material: Butter, Cost: 1200.00, Branch no: 111112, Quantity Provided: 200.00			
NOTICE: Supplier Name: Preet Patel, City of office: Bhuj, Raw material: Cheese, Cost: 2000.00, Branch no: 111112, Quantity Provided: 300.00			
NOTICE: Supplier Name: Preet Patel, City of office: Bhuj, Raw material: Chicken, Cost: 5000.00, Branch no: 111112, Quantity Provided: 20.00			
NOTICE: Supplier Name: Preet Patel, City of office: Bhuj, Raw material: Egg, Cost: 3000.00, Branch no: 111112, Quantity Provided: 30.00			
Successfully run. Total query runtime: 52 msec.			
1 rows affected.			

6) Create a stored procedure which on giving eid of an employee gives its name and its dependent details.

Ans-6)

Input:-

```
create or replace function get_dependent_details(eid_no numeric(10))returns integer as
$body$
declare
query varchar(1000)='select * from dependents d where eid='||eid_no;
emp_name text;
cur refcursor;
rec dependents%rowtype;

begin
    select (e.first_name||' '||e.last_name)into emp_name from employee e where eid=eid_no;
    raise notice 'Employee id=%, employee name=%',eid_no,emp_name;
    raise notice 'Following are dependent details:- ';
    open cur for execute query;
    loop
        fetch cur into rec;
        exit when not found;
        raise notice 'dep_name=%, dep_gender=%, dep_relationship= %',rec.first_name||
'||rec.last_name,rec.gender,rec.relationship;
    end loop;
    close cur;
return 1;
end;
$body$language plpgsql;
```

Function call:-

```
select get_dependent_details(1000000007)
```

Output:-

```

137 emp_name text;
138 cur refcursor;
139 rec dependents%rowtype;
140
141 begin
142     select (e.first_name||' '||e.last_name)into emp_name from employee e where eid=eid_no;
143     raise notice 'Employee id=%, employee name=%',eid_no,emp_name;
144     raise notice 'Following are dependent details:- ';
145     open cur for execute query;
146     loop
147         fetch cur into rec;
148         exit when not found;
149         raise notice 'dep_name=%, dep_gender=%, dep_relationship= %',rec.first_name||' '||rec.la
150     end loop;
151     close cur;
152 return 1;
153 end;
154 $body$language plpgsql;
155
156 select get_dependent_details(1000000007)

```

Results 1 Output

```

Employee id=1000000007, employee name= Vikrant Singh
Following are dependent details:-
dep_name=Aabhilash Singh , dep_gender=M , dep_relationship= Father
dep_name=Rajvi Parmar , dep_gender=F , dep_relationship= Sister
dep_name=Shruti Singh , dep_gender=F , dep_relationship= Wife

```

TRIGGERS:-

- 1) For ensuring that the two subtypes i.e.(company-owned branch and franchisee-owned branch) of the branch relation are following disjoint total-participation IS-A relationship constraint, we need to have a trigger on the franchisee-owned branch and company-owned branch table to ensure while insertion of a branch in any of these tables(company-owned branch and franchisee-owned branch) , that branch is present in only one one of these tables i.e. insertion will not be allowed in franchisee-owned branch if that branch is already present in the company-owned branch table.

Ans-1)

Procedure (which will be called by each of the triggers) :-

```

create or replace function trig_on_franchisee_owned_branch()returns trigger as
$body$
declare
read_branch_no numeric(6);
allow_insertion int:=1;
cur cursor for select trading_license_no from company_owned_branch;

```

```

begin
    if(tg_op='INSERT')then
        open cur;
        loop
            fetch cur into read_branch_no;
            exit when not found;
            if(new.trading_license_no=read_branch_no)then
                allow_insertion:=0;
                exit;
            end if;
        end loop;
        if(allow_insertion=1)then
            return new;
        end if;
        raise notice 'The branch cannot be inserted as it(its license_no) is already
present in the comapny-owned branch table';
        return new;--the new row will not be inserted in the farnchisee-branch
table if t is already inserted

    elsif(tg_op='DELETE')then
        delete from branch where trading_license_no =old.trading_license_no;
        return null;
    end if;
end;
$body$language plpgsql

```

Trigger definition:-

```

create trigger trig_for_insert_del_on_franchisee_branch_tbl
before insert or delete on franchisee_owned_branch
for each row execute procedure trig_on_franchisee_owned_branch();

```

Tables before invoking trigger:-

company_owned_branch Enter a SQL expression to filter results (use Ctrl+Space)			
	123 trading_license_no	123 revenue_generated_monthly	
1	111,113	1,500,000	
2	111,112	1,350,000	

franchisee_owned_branch Enter a SQL expression to filter results (use Ctrl+Space)				
	123 trading_license_no	ABC cin	123 royalty_fees	
1	111,111	U90167XY1111LNM777881	50,000	
2	111,114	L70167XY1122LNM777883	70,000	
3	111,115	L96576ZW1133IIT757693	90,000	

Trigger invoked:-

insert into franchisee_owned_branch
values(111113,'U90167XY1111LNM777881',60000)

Output:-

```

420      raise notice 'The branch cannot be inserted as it(its license_no) is already present in the company-owned branch table';
421      return new;--the new row will not be inserted in the franchisee-owned branch table if the license_no is already present in the company-owned branch table
422
423      elsif(tg_op='DELETE')then
424          delete from branch where trading_license_no =old.trading_license_no;
425          return null;
426      end if;
427 end;
428 $body$language plpgsql
429
430 create trigger trig_for_insert_del_on_franchisee_branch_tbl
431 before insert or delete on franchisee_owned_branch
432 for each row execute procedure trig_on_franchisee_owned_branch();
433
434 insert into franchisee_owned_branch values(111113,'U90167XY1111LNM777881',60000)
435
436
437
438

```

The branch cannot be inserted as it(its license_no) is already present in the company-owned branch table

Similarly there is trigger on company_owned branch table which checks for the same thing during insertion and deletion :-

Procedure (which will be called by each of the triggers) :-

```
create or replace function trig_on_company_owned_branch() returns trigger as
$body$
declare
read_branch_no numeric(6);
allow_insertion int:=1;
cur cursor for select trading_license_no from franchisee_owned_branch;

begin
    if(tg_op='INSERT')then
        open cur;
        loop
            fetch cur into read_branch_no;
            exit when not found;
            if(new.trading_license_no=read_branch_no)then
                allow_insertion:=0;
                exit;
            end if;
        end loop;
        if(allow_insertion=1)then
            return new;
        end if;
        raise notice 'The branch cannot be inserted as it(its license_no) is already
present in the franchisee-owned branch table';
        return null;--the new row will not be inserted in the franchisee-branch
table if it is already inserted

    elsif(tg_op='DELETE')then
        delete from branch where trading_license_no =old.trading_license_no;
        return old;
    end if;
end;
$body$language plpgsql
```

Trigger definition:-

```
create trigger trig_for_insert_del_on_company_owned_branch_tbl
before insert or delete on company_owned_branch
for each row execute procedure trig_on_company_owned_branch();
```

Trigger invoked:-

```
insert into company_owned_branch values(111111,1200000)
```

Output:-

Insertion will not be allowed in the company_owned_branch table as the branch license no is already present in the franchisee-owned branch.

- 2) **Create a special procedure so that each of the views of the individual branch menu becomes updatable. So that if one branch wants to add a particular dish in the menu from its own then that dish_name is automatically inserted in the dish (if not already present) relation and the dish_name and branch_no gets inserted in the branch_serving relation.**

Ans-1)

Input:-

Procedure (which will be called by each of the triggers) :-

```
create or replace function trig_for_making_view_of_menu_updatable()returns trigger as
$body$
```

```
declare
```

```
    dish_present varchar(30);
```

```
    branch_no numeric(6);
```

```
begin
```

```
    if(tg_op='INSERT') then
```

```
        select dish_name into dish_present from dish where
```

```
        dish_name=new.dish_name;
```

```
        select trading_license_no into branch_no from branch where
```

```
        city=tg_argv[0];
```

```
        if(dish_present isnull )then
```

```
            raise notice '%',new;
```

```
            insert into dish values(new.dish_name,new.price,new.veg_or_nonveg
```

```
,new.description,new.speciality,new.region,new.signatredish);
```

```
        end if;
```

```
        raise notice 'City name is %',tg_argv[0];
```

```
        insert into branch_serving values(branch_no,new.dish_name);
```

```

        return new;
    end if;
    if(tg_op='UPDATE')then
        update dish set
dish_name=new.dish_name,price=new.price,veg_or_nonveg=new.veg_or_nonveg
,description=new.description,speciality=new.speciality,region=new.region,signedish=
new.signedish
        where dish_name=new.dish_name;
        return new;
    end if;
    if(tg_op='DELETE')then
        raise notice 'Delete operation is allowed on this view as a particular dish
may be present in more than one branches and deleting it from one branch menu should
not delete it from the table of dish';
        --delete from dish where dish_name=old.dish_name;
        return null;
    end if;
end;
$body$language plpgsql;

```

Trigger definition for each of the views:-

```

create trigger trig_for_jam_branch_menu--instead of trigger for making views updatable
instead of insert or update or delete on menu_of_jamnagar_branch
for each row execute procedure trig_for_making_view_of_menu_updatable('Jamnagar');

```

```

create trigger trig_for_ahmedabad_branch_menu
instead of insert or update or delete on menu_of_ahmedabad_branch
for each row execute procedure
trig_for_making_view_of_menu_updatable('Ahmedabad');

```

```

create trigger trig_for_surat_branch_menu
instead of insert or update or delete on menu_of_surat_branch
for each row execute procedure trig_for_making_view_of_menu_updatable('Surat');

```

```

create trigger trig_for_junagadh_branch_menu
instead of insert or update or delete on menu_of_junagadh_branch
for each row execute procedure trig_for_making_view_of_menu_updatable('Junagadh');

```

```

create trigger trig_for_bhuj_branch_menu
instead of insert or update or delete on menu_of_bhuj_branch

```

for each row execute procedure trig_for_making_view_of_menu_updatable('Bhuj');

Trigger invoked:-

insert into menu_of_ahmedabad_branch values('Dalvada',65,'V','Crispy and crunchy Dalavada with onion','Famously known to be tasty in Amdavad','Central Guj',true);

insert into menu_of_jamnagar_branch values('Kutchi Dabeli',90,'V','Spicy and crispy','Has the taste of special Kutchi chutni','Kutch',true);

Tables before trigger was invoked:-

postgres			
<postgres> ...			
branch			
dish			
branch_serving			
Properties			
Data			
ER Diagram			
pos			
branch_serving			
Enter a SQL expression to filter results (use Ctrl+Space)			
Grid		branch_no	dish_name
	42	111,114	Garlic Bread
Text	43	111,115	Garlic Bread
	44	111,111	Dragon Potato
	45	111,112	Dragon Potato
	46	111,113	Dragon Potato
	47	111,114	Dragon Potato
	48	111,115	Dragon Potato
	49	111,111	Stuff Garlic Bread
	50	111,112	Stuff Garlic Bread
	51	111,113	Stuff Garlic Bread
	52	111,114	Stuff Garlic Bread
	53	111,115	Stuff Garlic Bread
	54	111,111	Mango Pineapple Smoothie
	55	111,111	Dry fruit kachori
	56	111,113	Dry fruit kachori
	57	111,112	Matka Kulfi
	58	111,113	Mango Pineapple Smoothie
	59	111,111	Paneer Tandori pizza
	60	111,112	Paneer Tandori pizza
	61	111,113	Paneer Tandori pizza
	62	111,114	Paneer Tandori pizza
Record	63	111,115	Paneer Tandori pizza

dish Enter a SQL expression to filter results (use Ctrl+Space)						
	ABC dish_name	123 price	ABC veg_or_nonveg	ABC description	ABC speciality	ABC region
1	Aloo Tikki Burger	65	V	Classic Indian Burger	[NULL]	[NULL]
2	French Fries	50	V	Made with crispy Indian Potato	[NULL]	[NULL]
3	Iced Coffee	40	V	Cold Coffee with classic flavour	[NULL]	[NULL]
4	Egg Burger	90	N	Made with fresh eggs	[NULL]	[NULL]
5	Kutchi Dabeli	90	V	Spicy and crispy	Has the taste of special Kutchi chutni	Kutch
6	Maska Bun	70	V	Burn served with fruit and jam	[NULL]	[NULL]
7	Fruit Ice cream	90	V	Made from Mixed real fruits	Natural fruit ice-cream	Central Guj
8	Surti Locho	50	V	Steamed Gujrati Farsan	Made from Gram flour	South Guj
9	Veg Cheese Sandwich	80	V	Perfect for a heavy lunch	[NULL]	[NULL]
10	Veg Cheese Pizza	100	V	A perfect food as it contains veggies	[NULL]	[NULL]
11	Garlic Bread	70	V	Roasted bread with Garlic layer	[NULL]	[NULL]
12	Dragon Potato	50	V	Crunch and spicy potatoes	[NULL]	[NULL]
13	Stuff Garlic Bread	70	V	Garlic bread stuffed with cheese	[NULL]	[NULL]
14	Dry fruit kachori	100	V	Crispy and filled with stuff	Dry-fruit stuffing is filled	Saurashtra
15	Matka Kulfi	80	V	Traditional ice-cream	Traditional recipe made in matka	Kutch
16	Mango Pineapple Smoothie	125	V	Made with Kesar Mango	Made from seasonal Kesar mangoes	Saurashtra
17	Paneer Tandoori pizza	140	V	Pizza loaded with spicy paneer	[NULL]	[NULL]

Tables after trigger was invoked:-

branch_serving Enter a SQL expression to filter results (use Ctrl+Space)			
Grid		123 branch_no	ABC dish_name
Text	44	111,111	Dragon Potato
	45	111,112	Dragon Potato
	46	111,113	Dragon Potato
	47	111,114	Dragon Potato
	48	111,115	Dragon Potato
	49	111,111	Stuff Garlic Bread
	50	111,112	Stuff Garlic Bread
	51	111,113	Stuff Garlic Bread
	52	111,114	Stuff Garlic Bread
	53	111,115	Stuff Garlic Bread
	54	111,111	Mango Pineapple Smoothie
	55	111,111	Dry fruit kachori
	56	111,113	Dry fruit kachori
	57	111,112	Matka Kulfi
	58	111,113	Mango Pineapple Smoothie
	59	111,111	Paneer Tandori pizza
	60	111,112	Paneer Tandori pizza
	61	111,113	Paneer Tandori pizza
	62	111,114	Paneer Tandori pizza
	63	111,115	Paneer Tandori pizza
Record	64	111,114	Dalvada
	65	111,111	Kutchi Dabeli

dish	Enter a SQL expression to filter results (use Ctrl+Space)								
	ABC dish_name	123 price	ABC veg_or_nonveg	ABC description	ABC speciality	ABC region			
1	Aloo Tikki Burger	65	V	Classic Indian Burger	[NULL]	[NULL]			
2	French Fries	50	V	Made with crispy Indian Potato	[NULL]	[NULL]			
3	Iced Coffee	40	V	Cold Coffee with classic flavour	[NULL]	[NULL]			
4	Egg Burger	90	N	Made with fresh eggs	[NULL]	[NULL]			
5	Kutchi Dabeli	90	V	Spicy and crispy	Has the taste of special Kutchi chutni	Kutch			
6	Maska Bun	70	V	Burn served with fruit and jam	[NULL]	[NULL]			
7	Fruit Ice cream	90	V	Made from Mixed real fruits	Natural fruit ice-cream	Central Guj			
8	Surti Locho	50	V	Steamed Gujrati Farsan	Made from Gram flour	South Guj			
9	Veg Cheese Sandwich	80	V	Perfect for a heavy lunch	[NULL]	[NULL]			
10	Veg Cheese Pizza	100	V	A perfect food as it contains veggies	[NULL]	[NULL]			
11	Garlic Bread	70	V	Roasted bread with Garlic layer	[NULL]	[NULL]			
12	Dragon Potato	50	V	Crunch and spicy potatoes	[NULL]	[NULL]			
13	Stuff Garlic Bread	70	V	Garlic bread stuffed with cheese	[NULL]	[NULL]			
14	Dry fruit kachori	100	V	Crispy and filled with stuff	Dry-fruit stuffing is filled	Saurashtra			
15	Matka Kulfi	80	V	Traditional ice-cream	Traditional recipe made in matka	Kutch			
16	Mango Pineapple Smoothie	125	V	Made with Kesar Mango	Made from seasonal Kesar mangoes	Saurashtra			
17	Paneer Tandori pizza	140	V	Pizza loaded with spicy paneer	[NULL]	[NULL]			
18	Dalvada	65	V	Crispy and crunchy Dalavada with onion	Famously known to be tasty in Amdam	Central Guj			

- 3) Create a trigger for storing the audit information in a separate table called emp_audit whenever some update(insert , update or delete) operation is performed on the employee table.

Ans-3)

Creating the audit table(DDL):-

```
create table emp_audit(
    operation char(1) not null,
    stamp timestamp not null,
    userid text not null,
    branch_license_no numeric(6) NOT NULL,
    first_name varchar(15) NOT NULL,
    middle_name varchar(15) NULL,
    last_name varchar(15) NOT NULL,
    gender bpchar(1) NOT NULL,
    monthly_salary numeric(7, 2) NOT NULL,
    contact_no numeric(10) NOT NULL,
    aadhar_no bpchar(12) NOT NULL,
    city varchar(12) NOT NULL,
    locality varchar(12) NOT NULL,
    pin_code bpchar(6) NOT NULL,
    job_position varchar(12) NOT NULL,
    joining_date date NOT NULL,
    company_employee bool NOT NULL,
    work_status varchar(8) NULL,
```

```
overall_performance varchar(14) NULL,  
promotion_eligibility bool NULL
```

```
);
```

Procedure which the trigger will call:-

```
CREATE OR REPLACE FUNCTION process_emp_audit() RETURNS TRIGGER AS  
$emp_audit$  
BEGIN  
    --  
    -- Create a row in emp_audit to reflect the operation performed on emp,  
    -- making use of the special variable TG_OP to work out the operation.  
    --  
    IF (TG_OP = 'INSERT') THEN  
        INSERT INTO emp_audit SELECT 'I', now(), user,  
new.branch_license_no,new.first_name,new.middle_name,new.last_name,new.gender,new.  
w.monthly_salary,new.contact_no,new.aadhar_no,new.city,new.locality,new.pin_code,new.  
w.job_position,new.joining_date,new.company_employee,new.work_status,new.overall_p  
erformance,new.promotion_eligibility;  
    ELSIF (TG_OP = 'UPDATE') THEN  
        INSERT INTO emp_audit SELECT 'U', now(), user,  
new.branch_license_no,new.first_name,new.middle_name,new.last_name,new.gender,new.  
w.monthly_salary,new.contact_no,new.aadhar_no,new.city,new.locality,new.pin_code,new.  
w.job_position,new.joining_date,new.company_employee,new.work_status,new.overall_p  
erformance,new.promotion_eligibility;  
    ELSIF (TG_OP = 'DELETE') THEN  
        INSERT INTO emp_audit SELECT 'D', now(), user,  
old.branch_license_no,old.first_name,old.middle_name,old.last_name,old.gender,old.mo  
nthly_salary,old.contact_no,old.aadhar_no,old.city,old.locality,old.pin_code,old.job_posit  
ion,old.joining_date,old.company_employee,old.work_status,old.overall_performance,ol  
d.promotion_eligibility;  
    END IF;  
    RETURN NULL; -- result is ignored since this is an AFTER trigger  
END;  
$emp_audit$ LANGUAGE plpgsql;
```

Trigger definition :-

create trigger trig_foremployee_audit
after insert or update or delete on employee
for each row execute procedure process_emp_audit();

Employee table for any update:-

1,000,000,107	111,114	Nirmal	Pankajkumar	Patel	M	
1,000,000,037	111,113	Siddharth	Bhavesb	Parmar	M	
1,000,000,003	111,113	Rajesh	V	Panchal	M	
1,000,000,050	111,113	Bijoy	Mani	kakar	M	
1,000,000,700	111,115	Pankaj	M	Sheth	M	
1,000,000,027	111,114	Ragini	K.	Chauhan	F	
1,000,000,370	111,114	Sanjana	[NULL]	Singh	F	
1,000,000,010	111,113	Shalini	Thomas	Sebastian	F	
1,000,000,036	111,115	Shilpa	Vraj	Kunda	F	
1,000,000,001	111,112	Nagma	K	Chandra	F	
1,000,000,019	111,115	Mangan	Manganlal	Patil	M	
1,000,000,201	111,115	Thomas	Bob	Roy	M	
1,000,000,007	111,111	Vikrant	Aabhilash	Singh	M	
1,000,000,011	111,111	Lalit	Jatinlal	Natt	M	
1,000,000,004	111,112	Bipin	J	Vaghasiya	M	
1,000,000,005	111,114	andrakant	Manganlal	Bhatt	M	

Trigger invoked by making an update statement:-

update employee set first_name ='Andrakant',locality ='Ghatlodia' where eid
=1000000005;

Output:

Emp_audit table after trigger invocation:-

```

233 BEGIN
234 --
235 -- Create a row in emp_audit to reflect the operation performed on emp,
236 -- making use of the special variable TG_OP to work out the operation.
237 --
238 IF (TG_OP = 'DELETE') THEN
239     INSERT INTO emp_audit SELECT 'D', now(), user, new.branch_license_no,new.first_n
240 ELSIF (TG_OP = 'UPDATE') THEN
241     INSERT INTO emp_audit SELECT 'U', now(), user, new.branch_license_no,new.first_n
242 ELSIF (TG_OP = 'INSERT') THEN
243     INSERT INTO emp_audit SELECT 'I', now(), user, old.branch_license_no,old.first_n
244 END IF;
245 RETURN NULL; -- result is ignored since this is an AFTER trigger
246 END;
247

```

emp_audit 1 Execution Log Variables Output

select * from emp_audit ea

ABC operation	stamp	ABC userid	123 branch_license_no	ABC first_name	ABC middle_name	ABC last_name	ABC gender
U	2021-11-14 18:09:46	postgres	111,114	Andrakant	Manganlal	Bhatt	M

Employee table after trigger invocation:-

123 eid	123 branch_license_no	ABC first_name	ABC middle_name	ABC last_name	ABC gender	123 monthly_salary	123 contact_no
1,000,000,107	111,114	Nirmal	Pankajkumar	Patel	M	25,000	8,443,503,1
1,000,000,037	111,113	Siddharth	Bhavesh	Parmar	M	10,000	8,719,160,7
1,000,000,003	111,113	Rajesh	V	Panchal	M	15,000	9,527,318,0
1,000,000,050	111,113	Bijoy	Mani	kakar	M	20,000	8,218,320,1
1,000,000,700	111,115	Pankaj	M	Sheth	M	25,000	9,973,503,5
1,000,000,027	111,114	Ragini	K.	Chauhan	F	10,000	8,202,718,4
1,000,000,370	111,114	Sanjana	[NULL]	Singh	F	15,000	9,483,607,3
1,000,000,010	111,113	Shalini	Thomas	Sebastian	F	25,000	7,789,305,3
1,000,000,036	111,115	Shilpa	Vraj	Kunda	F	10,000	7,802,218,5
1,000,000,001	111,112	Nagma	K	Chandra	F	20,000	7,555,444,3
1,000,000,005	111,114	Andrakant	Manganlal	Bhatt	M	20,000	6,475,548,2
1,000,000,019	111,115	Mangan	Manganlal	Patil	M	15,000	8,972,200,2
1,000,000,201	111,115	Thomas	Bob	Roy	M	20,000	6,983,510,3
1,000,000,007	111,111	Vikrant	Aabhilash	Singh	M	25,000	7,555,449,8
1,000,000,011	111,111	Lalit	Jatinlal	Natt	M	10,000	9,455,568,1
1,000,000,004	111,112	Bipin	J	Vaghasiya	M	10,000	9,455,568,1

- 4) Create a stored procedure which ensures that while inserting values in the franchisee_company relation , the cin(a 21 character alpha-numeric no) is a valid one(it will check it by checking if the alphabets and numbers are present at the correct position or not .)

Ans-4)

Input:-

Procedures which the trigger will call:-

create or replace function trig_proc_for_franch_comp() returns trigger as \$body\$

declare

 cin_no char(21):=new.cin;

 alpha char(1);

 index int:=1;

 valid int:=1;

begin

 loop

 alpha:=substring(cin_no,index,1);

 --raise notice 'index=%',index;

 exit when index=22;

 if(index=1)then

 if(not(check_alphabet_or_not(alpha)))then

 valid:=0;

 exit;

 end if;

 elsif(index>=2 and index<=6)then

 if(not(check_number_or_not(alpha)))then

 valid:=0;

 exit;

 end if;

 elsif(index=7 or index=8)then

 if(not(check_alphabet_or_not(alpha)))then

 valid:=0;

 exit;

 end if;

 elsif(index>=9 and index<=12)then

 if(not(check_number_or_not(alpha)))then

 valid:=0;

 exit;

 end if;

 elsif(index>=13 and index<=15)then

 if(not(check_alphabet_or_not(alpha)))then

 valid:=0;

 exit;

 end if;

 elsif(index>=16 and index<=21)then

 if(not(check_number_or_not(alpha)))then


```

                                valid:=0;
                                exit;
                            end if;
                        end if;
                        index:=index+1;
                    end loop;
                if(valid=1)then
                    return new;
                else
                    raise notice 'The cin no which you are trying to enter is in invalid format';
                    return null;
                end if;
            end;
        $$body$language plpgsql

```

Supporting procedures:-

(1)

create or replace function check_alphabet_or_not(alpha char(1))returns bool as

\$\$

declare

 str char(26)='ABCDEFGHIJKLMNOPQRSTUVWXYZ';

 ch char(1);

 index integer:=1;

 present bool:=false;

begin

 loop

 exit when index=27;

 ch=substring(str,index,1);

 if(alpha=ch)then

 present:=true;

 exit;

 end if;

 index:=index+1;

 end loop;

 return present;

end;

\$\$language plpgsql;

(2)

create or replace function check_number_or_not(num char(1))returns bool as

```

$$
declare
    str char(26)='0123456789';
    ch char(1);
    index integer:=1;
    present bool:=false;
begin
    loop
        exit when index=11;
        ch=substring(str,index,1);
        if(num=ch)then
            present:=true;
            exit;
        end if;
        index:=index+1;
    end loop;
    return present;
end;
$$language plpgsql;

```

Trigger definition:-

```

create trigger trig_for_franch_comp
before insert on franchiseecompany
for each row execute procedure trig_proc_for_franch_comp());

```

Franchisee company Table before trigger invocation:-

Grid	cin	company_name	head_manager	office_locality	office_pin_code	office_city
1	U90167XY1111LNM777881	Acme Corporation	Aksat Srivastava	Bopal	340,200	Ahmedabad
2	L70167XY1122LNM777883	Globex Corporation	Aayush Kumar	Ghatgopal	400,004	Mumbai
3	U70167ZW1122DAI757663	Initech	Yash Sanghvi	Adajan Patia	340,280	Surat
4	L96576ZW1133IIT757693	Wayne Enterprises	Shreeji Joshi	SG Highway	340,200	Ahemdabad
5	U96546FG1133NIT757393	Soylent Corporation	Harsh Madani	Dumas	340,280	Surat

Function call which invoked trigger:-

```

insert into franchiseecompany values('UA0167XY1111LNM777881','L and T','Mukund
Ladani','PritamPura','360012','New Delhi');

```

insert into franchiseecompany values('X78985AB1245UHJ785621','MnC and company','Kashyap Halavadia','Bodakdev','340200','Ahmedabad')

Output:-

```

336         end if;
337         index:=index+1;
338     end loop;
339     if(valid=1) then
340         return new;
341     else
342         raise notice 'The cin no which you are trying to enter is in invalid format';
343         return null;
344     end if;
345 end;
346 $$body$language plpgsql
347
348
349 create trigger trig_for_franch_comp
350 before insert on franchiseecompany
351 for each row execute procedure trig_proc_for_franch_comp();
352
353 insert into franchiseecompany values('UA0167XY1111LNM777881','L and T','Mukund Ladani','
354 insert into franchiseecompany values('X78985AB1245UHJ785621','MnC and company','Kashyap
355
356
357

```

Statistics 1 | Execution Log | Variables | Output

The cin no which you are trying to enter is in invalid format

Franchisee company table after trigger call:-

	cin	company_name	head_manager	office_locality	office_pin_code	office_city
1	U90167XY1111LNM777881	Acme Corporation	Aksat Srivastava	Bopal	340,200	Ahmedabad
2	L70167XY1122LNM777883	Globex Corporation	Aayush Kumar	Ghatgopal	400,004	Mumbai
3	U70167ZW1122DAI757663	Initech	Yash Sanghvi	Adajan Patia	340,280	Surat
4	L96576ZW1133IIT757693	Wayne Enterprises	Shreeji Joshi	SG Highway	340,200	Ahmedabad
5	U96546FG1133NIT757393	Soylent Corporation	Harsh Madani	Dumas	340,280	Surat
6	X78985AB1245UHJ785621	MnC and company	Kashyap Halavadia	Bodakdev	340,200	Ahmedabad

- In the advertisement table , the ad_code is of specific format , it is a 5 letter alpha-numeric number whose first two letters should be alphabets denoting mode of ads and the last three letters will be integers. So we need to check before a tuple is inserted in the advertisement table that if it is valid or not .Insertion will be allowed only if it is valid.This can be implemented using triggers.

Ans-5)

Procedure which the trigger will call:-

```
create or replace function trig_on_advertisement_tbl()returns trigger as
$body$
declare
    ad_code char(5):=new.advertisement_code;
    alpha char(1);
    str varchar(2);
    index int:=3;
begin
    if(tg_op='INSERT')then
        str:=substring(ad_code,1,2);
        if(not(str='PO' or str='SM' or str='TV'))then
            raise notice 'Insertion not allowed as the first two letters in the
ad_code are invalid as they can be only from = {PO,SM,TV}';
            return null;
        end if;
        loop
            exit when index=6;
            alpha:=substring(ad_code,index,1);
            if(not(check_number_or_not(alpha)))then
                raise notice 'Insertion not allowed as % index letter
in the ad_code is not number',index;
                raise notice 'Letters from index=3 to 6 in the
ad_code must be integers';
                return null;
                exit;
            end if;
            index:=index+1;
        end loop;
        return new;
    end if;
end;
$body$ language plpgsql
```

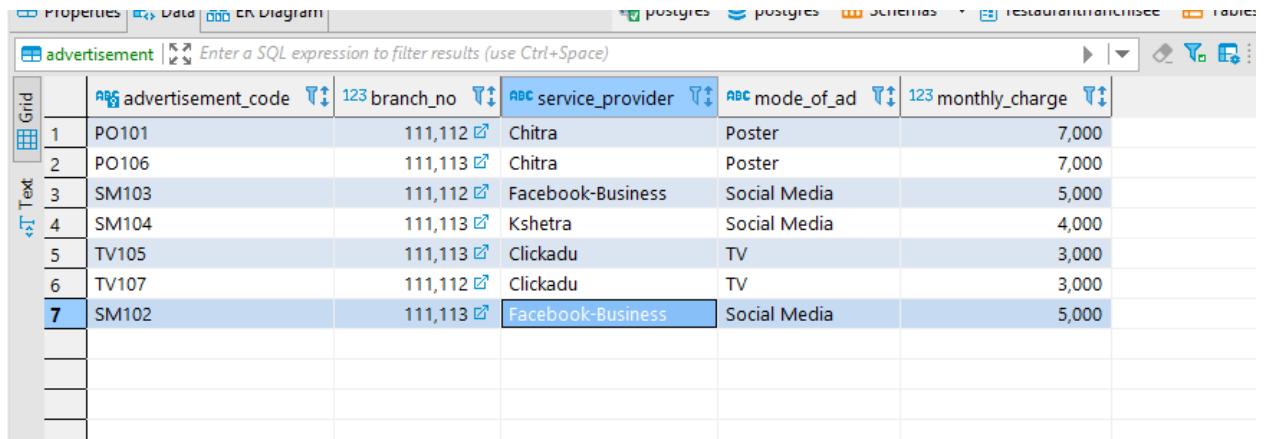
Trigger definition:-

create trigger trig_on_advertisement

before insert on advertisement

for each row execute procedure trig_on_advertisement_tbl()


Advertisement table before calling triggers function:-



	advertisement_code	branch_no	service_provider	mode_of_ad	monthly_charge
1	PO101	111,112	Chitra	Poster	7,000
2	PO106	111,113	Chitra	Poster	7,000
3	SM103	111,112	Facebook-Business	Social Media	5,000
4	SM104	111,113	Kshetra	Social Media	4,000
5	TV105	111,113	Clickadu	TV	3,000
6	TV107	111,112	Clickadu	TV	3,000
7	SM102	111,113	Facebook-Business	Social Media	5,000

Trigger invoked and its output:-

- 1) insert into advertisement values('P1108','111112','RG Entertainment','Poster',6000)



```
540         end if;
541         index:=index+1;
542     end loop;
543     return new;
544 end if;
545 end;
546 $body$ language plpgsql
547
548 create trigger trig_on_advertisement
549 before insert on advertisement
550 for each row execute procedure trig_on_advertisement_tbl()
551
552 insert into advertisement values('P1108','111112','RG Entertainment','Poster',6000)
553
554
555
556
557
558
```

Insertion not allowed as the first two letters in the ad_code are invalid as they can be only from ={PO,SM,TV}

- 2) insert into advertisement values('PO108','111112','RG Entertainment','Poster',6000)

Trigger call output:-

The screenshot shows a PostgreSQL IDE with a trigger function defined in PL/pgSQL. The function, named `trig_on_advertisement_tbl()`, is designed to be executed before an insert on the `advertisement` table. It includes a loop that iterates from `index=3` to `6`, checking if the `ad_code` must be an integer. A notice is raised if the condition is not met. The function is then called within a trigger definition. Below the code, the execution output is displayed, showing the successful insertion of a new record into the `advertisement` table.

```
537         raise notice 'Letters from index=3 to 6 in the ad_code must be integer';
538         return null;
539         exit;
540     end if;
541     index:=index+1;
542 end loop;
543 return new;
544 end if;
545 end;
546 $$ language plpgsql
547
548 create trigger trig_on_advertisement
549 before insert on advertisement
550 for each row execute procedure trig_on_advertisement_tbl()
551
552 insert into advertisement values('PO108','111112','RG Entertainment','Poster',6000)
```

Statistics 1 | Output

Enter a SQL expression to filter results (use Ctrl+Space)

Name	Value
Updated Rows	1
Query	insert into advertisement values('PO108','111112','RG Entertainment','Poster',6000)
Finish time	Mon Nov 22 10:57:12 IST 2021

Advertisement table after insertion:-

The screenshot shows the 'advertisement' table in a PostgreSQL IDE. The table has six columns: `advertisement_code`, `branch_no`, `service_provider`, `mode_of_ad`, and `monthly_charge`. The table contains eight rows of data, including the newly inserted record with `advertisement_code` 'PO108' and `monthly_charge` 6,000.

	advertisement_code	branch_no	service_provider	mode_of_ad	monthly_charge
1	PO101	111,112	Chitra	Poster	7,000
2	PO106	111,113	Chitra	Poster	7,000
3	SM103	111,112	Facebook-Business	Social Media	5,000
4	SM104	111,113	Kshetra	Social Media	4,000
5	TV105	111,113	Clickadu	TV	3,000
6	TV107	111,112	Clickadu	TV	3,000
7	SM102	111,113	Facebook-Business	Social Media	5,000
8	PO108	111,112	RG Entertainment	Poster	6,000