Assignment 4 : Error and Exception Handling

Name: Datta Panchal

Roll no: 13370

SET A

Person-Area Database

Person `(pno integer, pname varchar (20), birthdate date, income money)`
Area `(aname varchar (20), area-type varchar (5))`
An area can have one or more persons living in it, but a person belongs to exactly one area. The attribute `area_type` can have values either ‘*urban*’ or ‘*rural*’.

```
postgres=# create table Area(aname varchar(20) Primary Key, area_type varchar(5) check(area_type in ('urban', 'rural')));
CREATE TABLE
postgres=# \d Area
                        Table "public.area"
  Column   |         Type          | Collation | Nullable | Default
-----------+-----------------------+-----------+----------+---------
 aname     | character varying(20) |           | not null |
 area_type | character varying(5)  |           |          |
Indexes:
    "area_pkey" PRIMARY KEY, btree (aname)
Check constraints:
    "area_area_type_check" CHECK (area_type::text = ANY (ARRAY['urban'::character varying, 'rural'::character varying]::text[]))
```

```
postgres=# create table Person(pnumber int primary key, pname varchar(20), birthdate date, income int, aname varchar(30) references Area
CREATE TABLE
postgres=# \d Person
                       Table "public.person"
  Column   |         Type          | Collation | Nullable | Default
-----------+-----------------------+-----------+----------+---------
 pnumber   | integer               |           | not null |
 pname     | character varying(20) |           |          |
 birthdate | date                  |           |          |
 income    | integer               |           |          |
 aname     | character varying(30) |           |          |
Indexes:
    "person_pkey" PRIMARY KEY, btree (pnumber)
Foreign-key constraints:
    "person_aname_fkey" FOREIGN KEY (aname) REFERENCES area(aname) ON UPDATE CASCADE ON DELETE CASCADE
```

1. Write a stored function to print total number of person of a particular area. *(Accept `area_name` as input parameter)*. Display appropriate message for invalid area name.

```
Create or Replace FUNCTION setA1ass4 (varchar) Returns Int AS'
Declare
cnt int;
BEGIN
select into cnt count(pnumber) from Person where aname=$1;
if cnt = 0 then
  Raise Exception ''Invalid Area Name'';
end if;
return cnt;
END;
'LANGUAGE 'plpgsql';

seta1ass4
-----------
         4
(1 row)
```

2. Write a stored function to print sum of income of person living in “_____” area type. *(Accept area type as input parameter)*. Display appropriate message for invalid area type.

```
Create or Replace FUNCTION setA2ass4 (varchar) Returns Int AS'
Declare
cnt int;
BEGIN
if $1 = ''urban'' OR $1 = ''rural'' then
  select into cnt sum(income) from Person where aname IN(select aname from Area where area_type=$1);
  return cnt;
else
  Raise Exception ''Invalid Area Name'';
end if;

END;
'LANGUAGE 'plpgsql';

CREATE FUNCTION
postgres=# select setA2ass4('urban');
 seta2ass4
-----------
    133000
(1 row)


postgres=# select setA2ass4('rural');
 seta2ass4
-----------
```

```
     62500
(1 row)
```

3. Write a stored function to display details of person along with area name whose birthday falls in the month of “ ”. `(Accept month as input parameter)`. Display error message for invalid month name.

```
Create or Replace FUNCTION setA3ass4 (varchar) Returns void AS'
Declare
rec record;
BEGIN
for rec in select * from Person where CAST(birthdate As Char(10))=$1 loop
  raise notice ''% % % % %'', rec.pnumber, rec.pname, rec.birthdate, rec.income, rec.aname;
end loop;
END;
'LANGUAGE 'plpgsql';

postgres=# select setA3ass4('2002-05-05');
NOTICE:  105 Anuja 2002-05-05 15000 Shahada
 seta3ass4
-----------

(1 row)
```

SET B

Bus Driver Database

BUS `(bus_no int , capacity int , depot_name varchar(20))`
ROUTE `(route_no int, source char(20), destination char(20),no_of_stations int)`
DRIVER `(driver_no int , driver_name char(20), license_no int, address char(20), d_age int , salary float)`
The relationships are as follows:
BUS_ROUTE: M-1
BUS_DRIVER: M-M with descriptive attributes Date of duty allotted and Shift – it can be (*Morning*) or (*Evening*).

Constraints:

1. `License_no` is unique.
2. Bus capacity is not null

1. Write a stored function to accept the bus_no and date and print its allotted drivers. Raise an exception in case of invalid bus number.

```
create or replace function print_dri_bus (no integer, dt date ) returns void AS'
declare
rec record;
begin
select into rec from bus_driver where bus_no=no and date_of_duty_allotted=dt;
if NOT FOUND then raise exception ''Invalid bus no'';
end if;
for rec in select driver_name, bus_no
from driver a, bus_driver b
where b.bus_no and b.date_of_duty and a.dno-b.dno
loop
raise notice ''% %'',rec.dname, rec.bus_no;
end loop;
END;
'LANGUAGE 'plpgsql';

/--
select print_dri_bus(10, '2020-06-01');
```

2. Write a stored function to display the all Dates on which a driver has driven any bus.(*Accept driver name as input parameter*). Raise an exception in case of invalid driver name.

```
Create or Replace FUNCTION setA3ass4 (varchar) Returns void AS'
Declare
rec record;
cnt int;
BEGIN
select into cnt count(driver_no) from Driver where driver_name=$1;
if cnt < 1 then
  raise Exception ''Invalid Driver Name'';
end if;
for rec in select date_of_duty_allotted from bus_driver where driver_no IN (select driver_no from Driver where driver_name=$1) loop
  raise notice ''%'', rec.date_of_duty_allotted;
end loop;
END;
'LANGUAGE 'plpgsql';

/--
postgres=# select setA3ass4('Siddharthfwsswf');
ERROR:  Invalid Driver Name
CONTEXT:  PL/pgSQL function seta3ass4(character varying) line 8 at RAISE
postgres=# select setA3ass4('Siddharth');
NOTICE:  2020-06-01
```

3. Write a stored function to display the details of the buses that run on `route_no` = “ ”. (*accept* `route_no` *as input parameter*). Raise an error in case of invalid `route_no`.

```
Create or Replace FUNCTION setB3ass4 (int) Returns void AS'
Declare
rec record;
cnt int;
BEGIN
select into cnt count(bus_no) from Bus where route_no=$1;
if cnt < 1 then
  raise Exception ''Invalid Route No'';
end if;
for rec in select * from Bus where route_no=$1 loop
  raise notice ''% % % %'', rec.bus_no, rec.b_capacity, rec.depot_name, rec.route_no;
end loop;
END;
'LANGUAGE 'plpgsql';

/--
postgres=# select setB3ass4(2);
NOTICE:  11 30 Kothrud 2
 setb3ass4
-----------

(1 row)


postgres=# select setB3ass4(232233);
ERROR:  Invalid Route No
CONTEXT:  PL/pgSQL function setb3ass4(integer) line 8 at RAISE
```