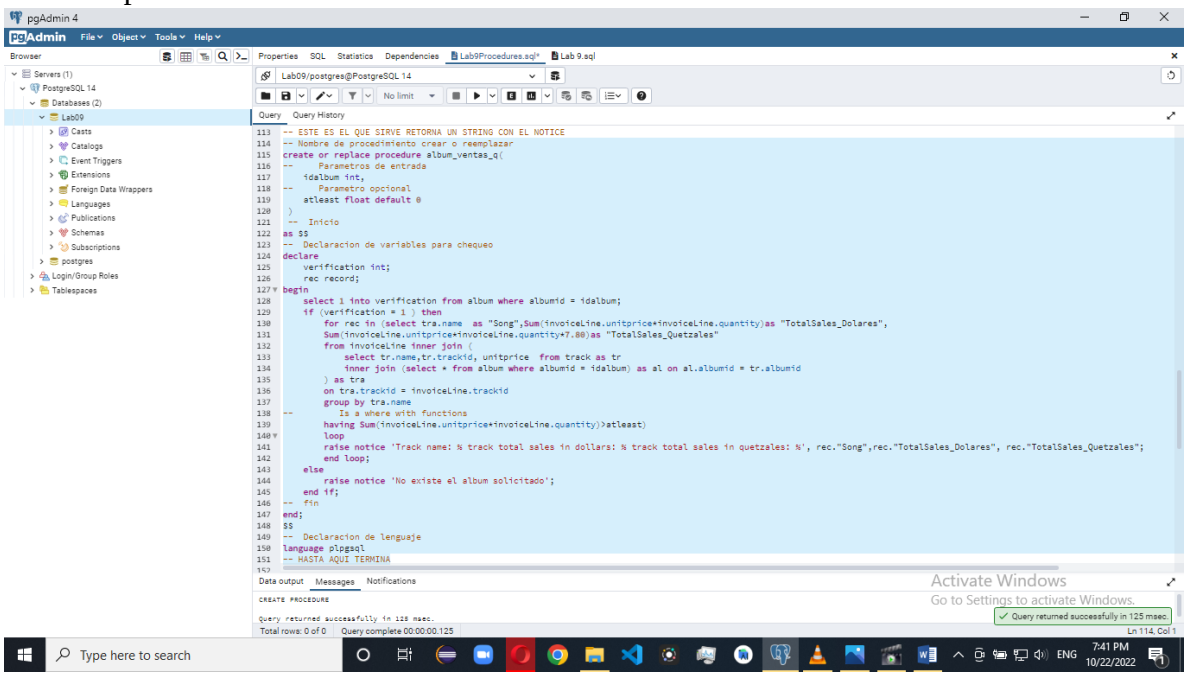
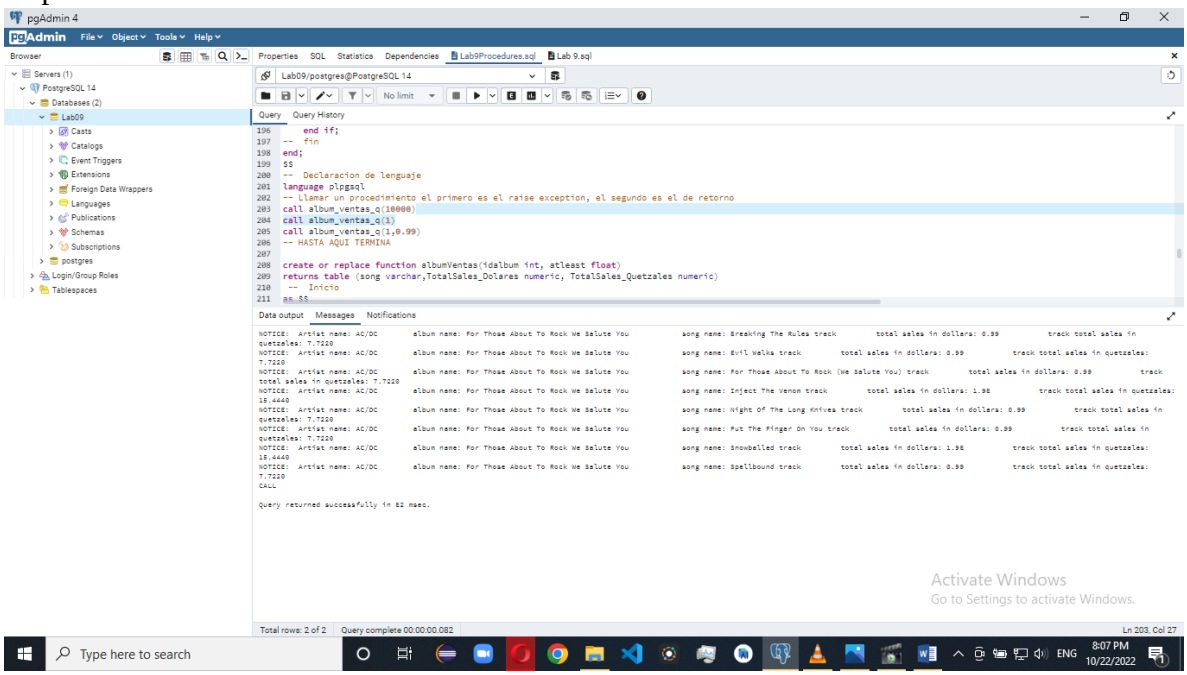
Laboratorio 09

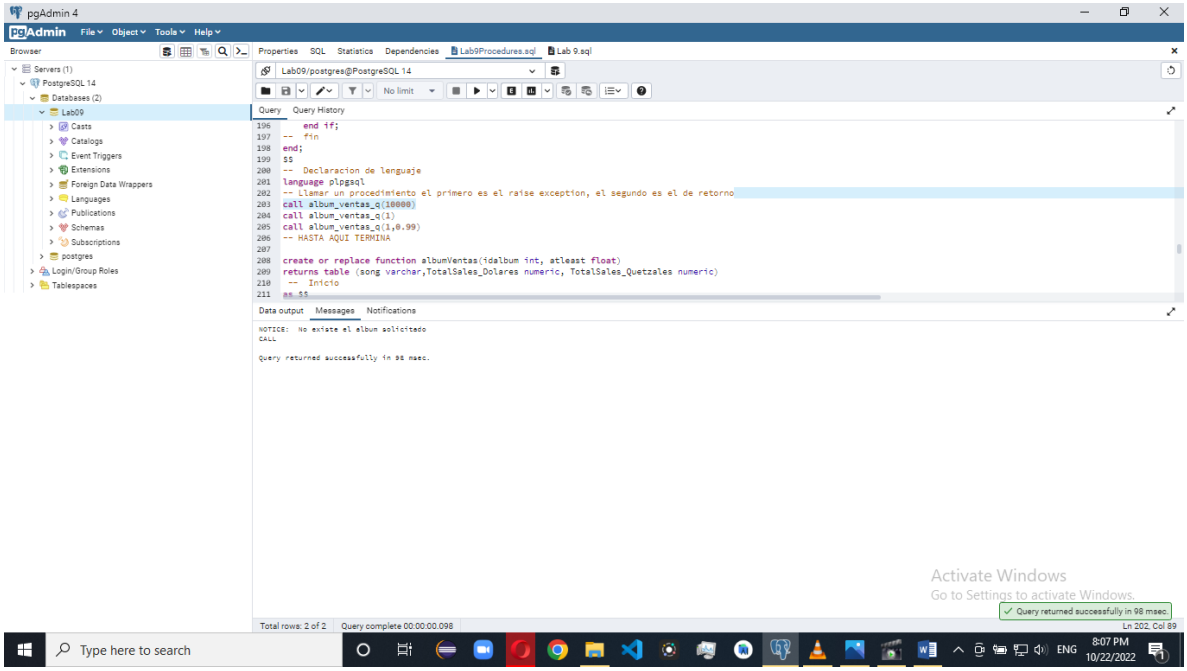
1. Crear un procedimiento



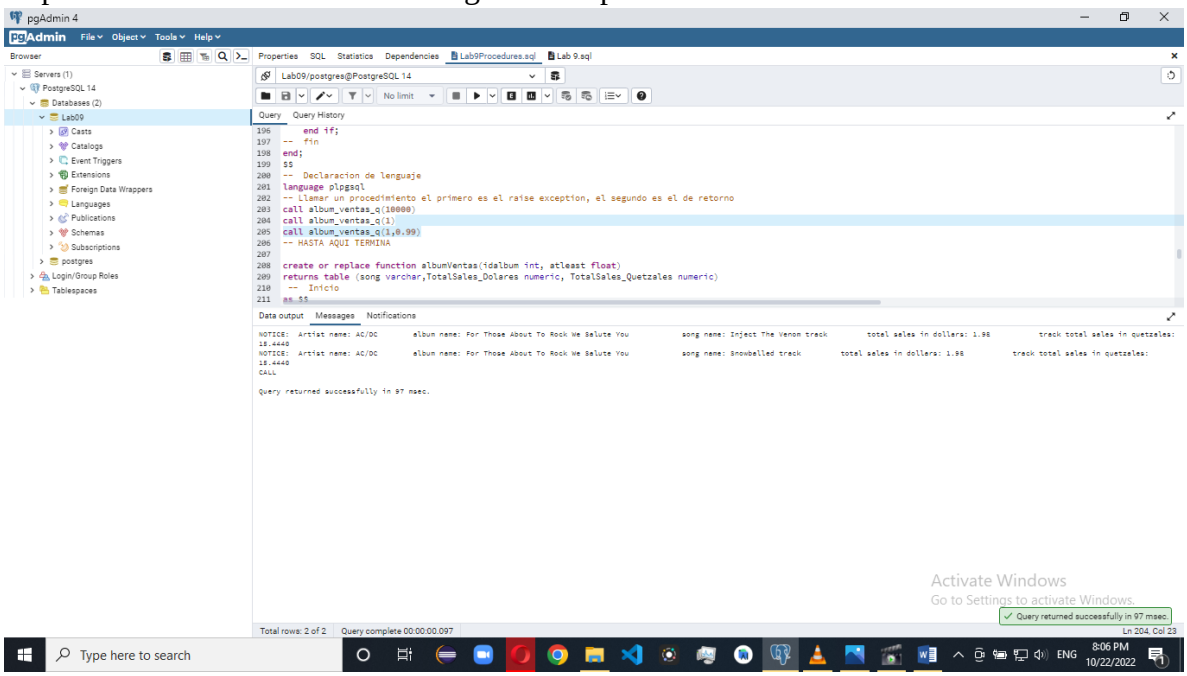
El procedimiento funciona



El procedimiento lanza la excepción

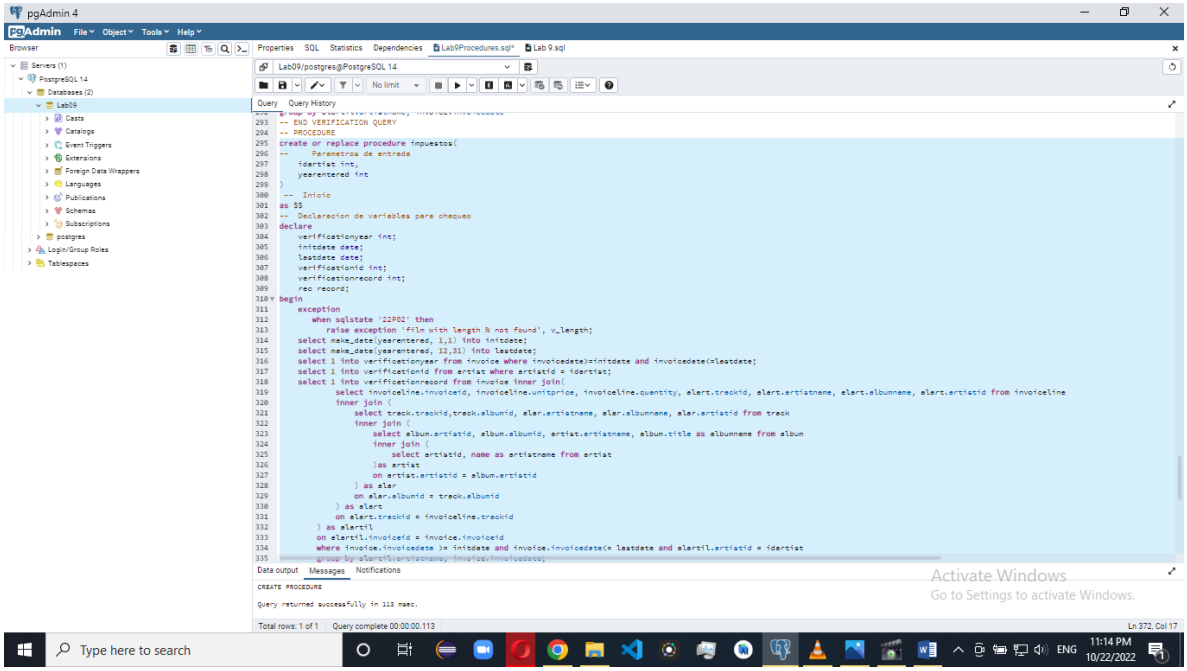


El procedimiento funciona con el argumento opcional

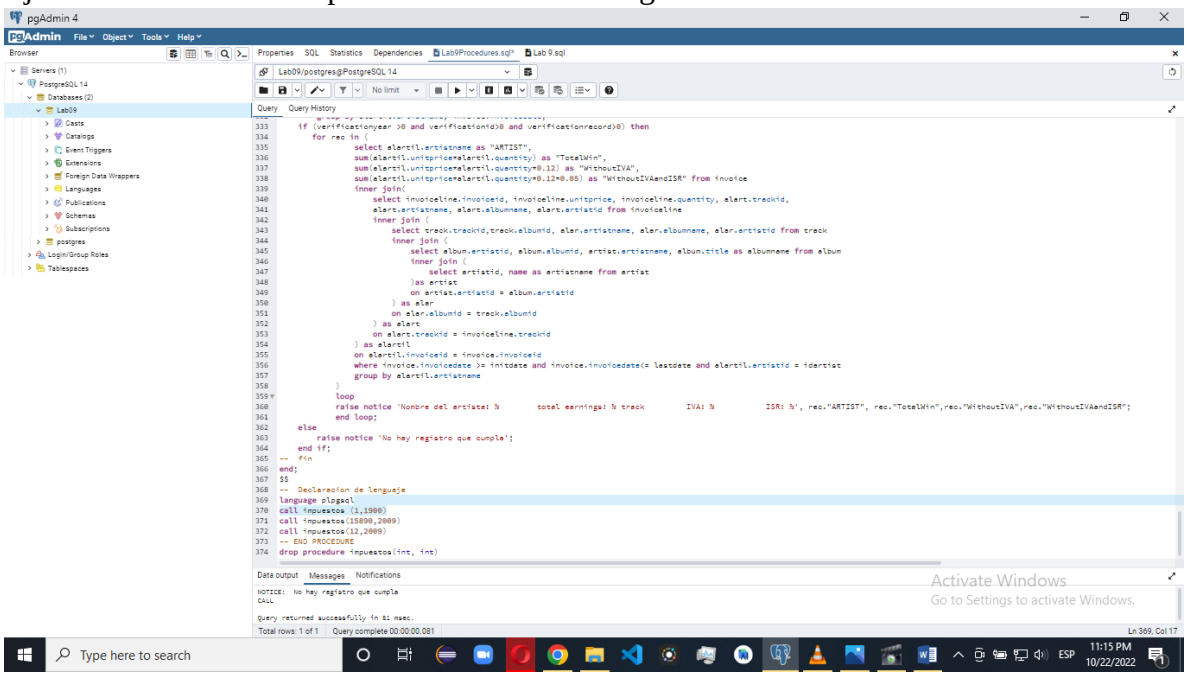


2.

Creación del procedimiento

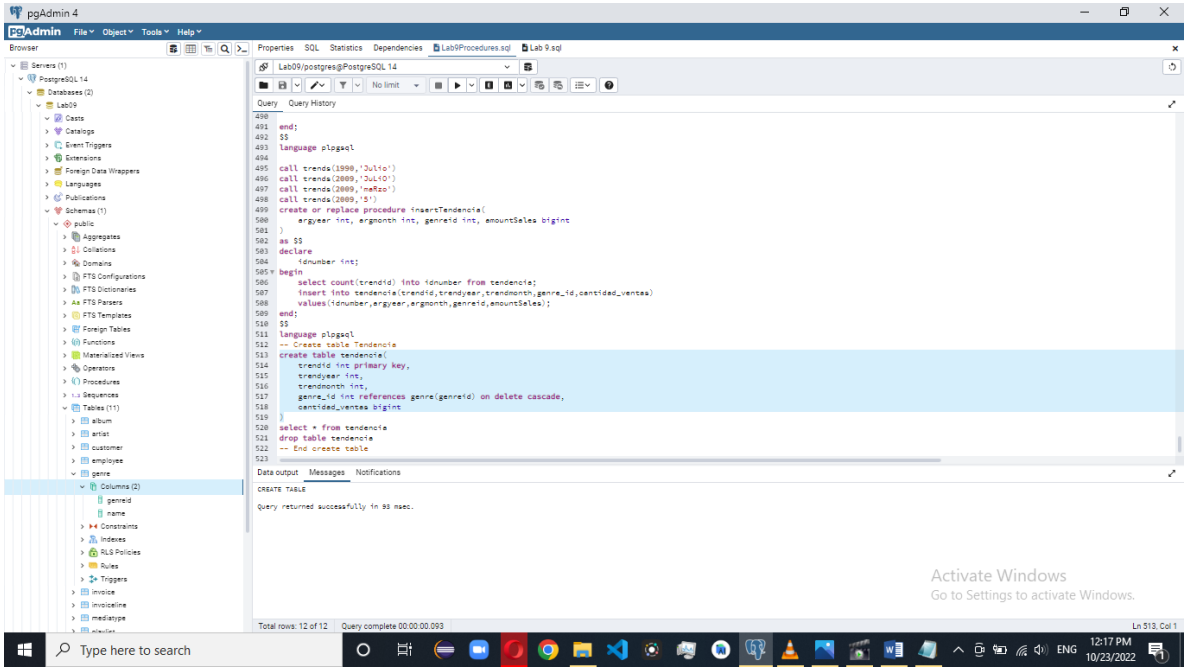


Ejecución de un error de procedimiento con un registro invalido de

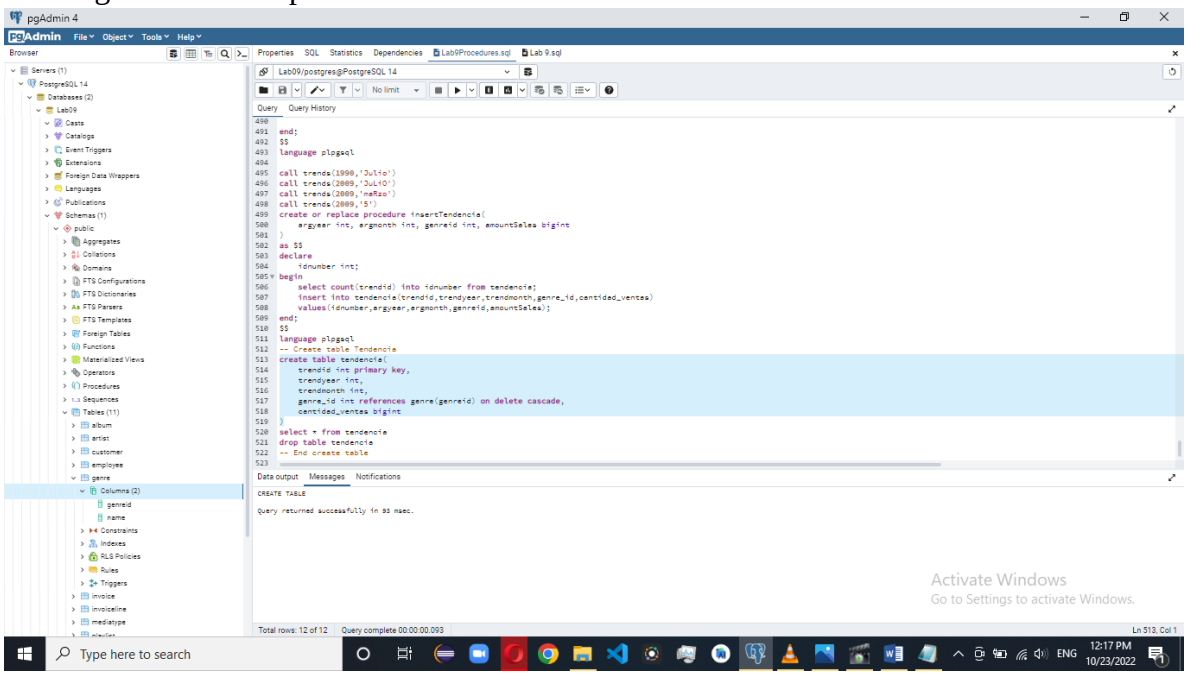


3.

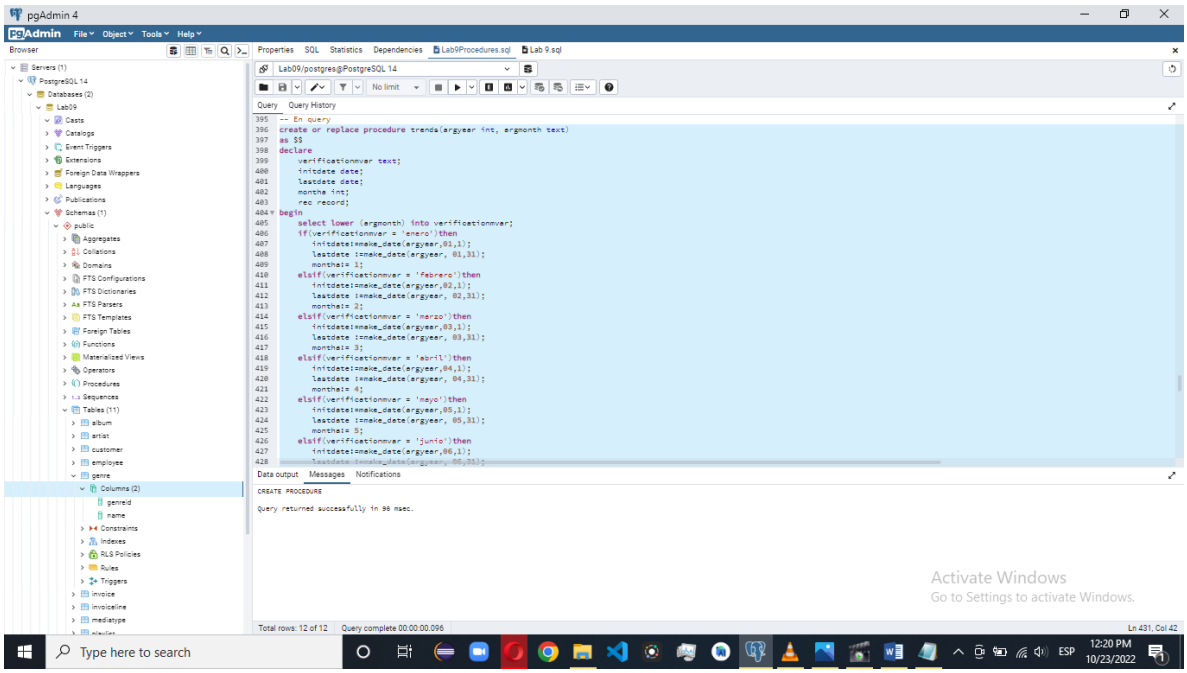
Crear la tabla de tendencia.



Crear procedimiento para la inserción de datos en la tabla de tendencia, nótese que se autogenera la llave primaria.

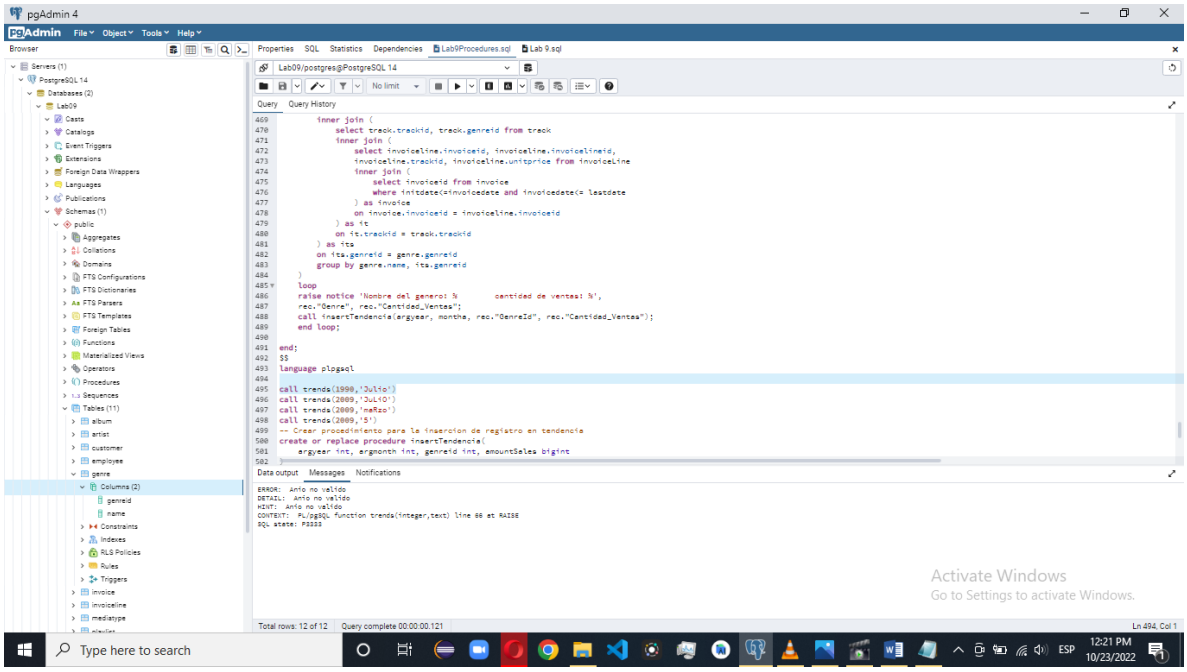


Crear procedimiento para obtener la cantidad de ventas generada según el genero basado en un mes y año ingresados.

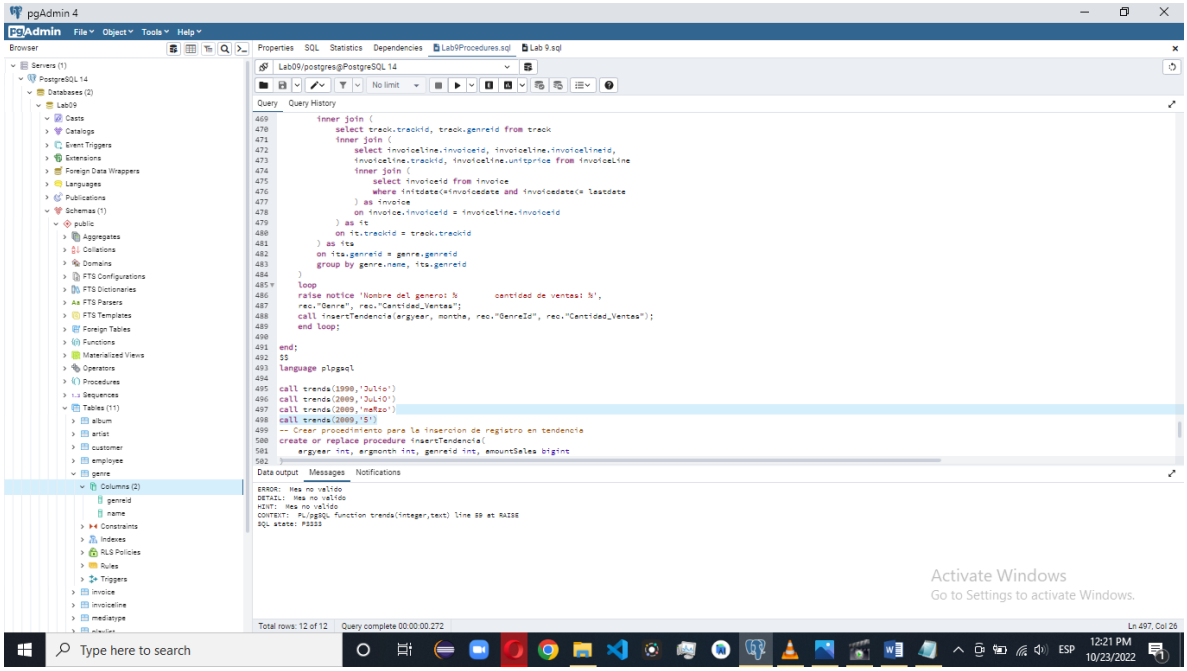


Pruebas de que funciona

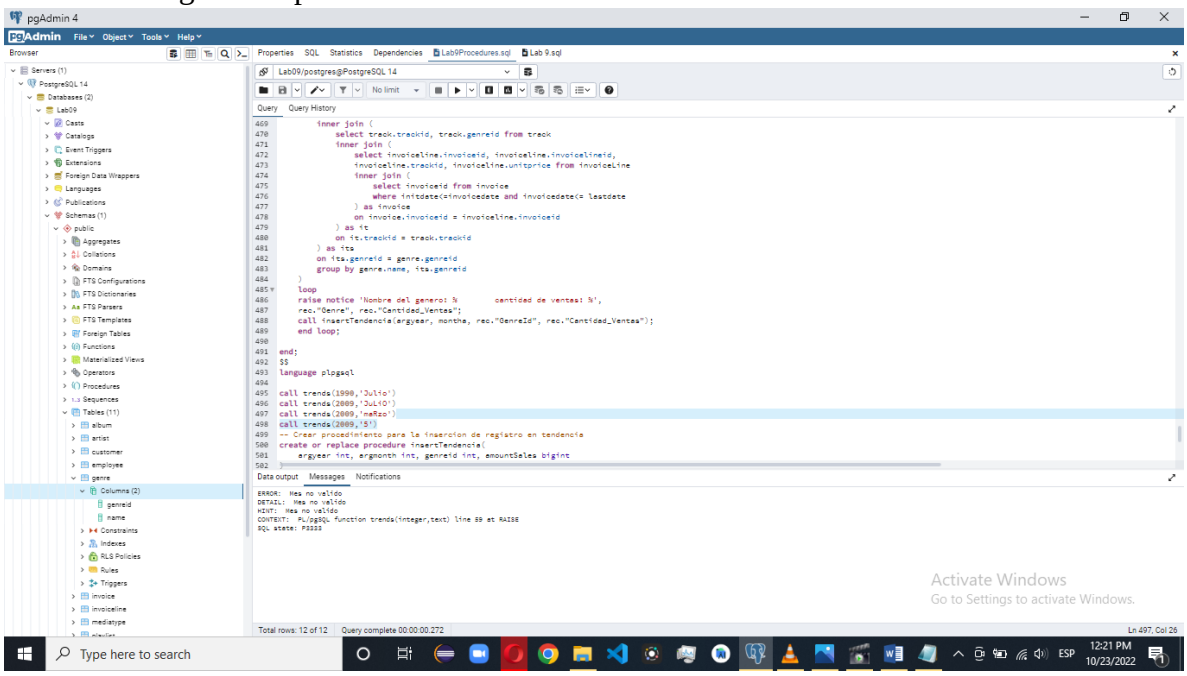
Error de año invalido

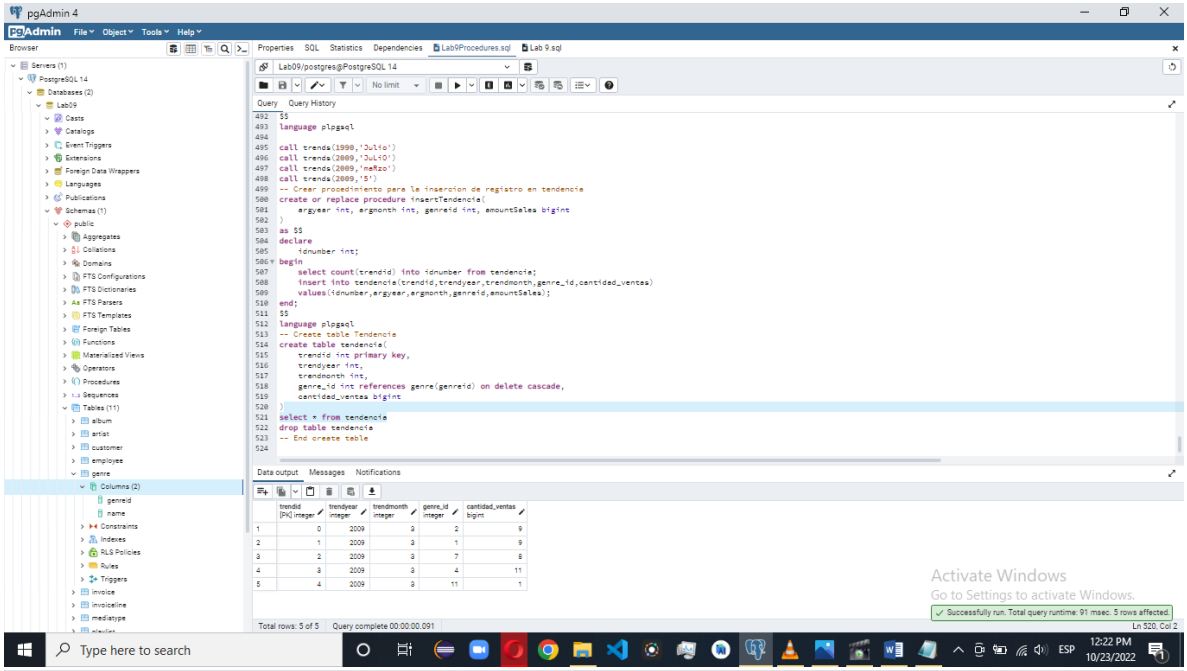


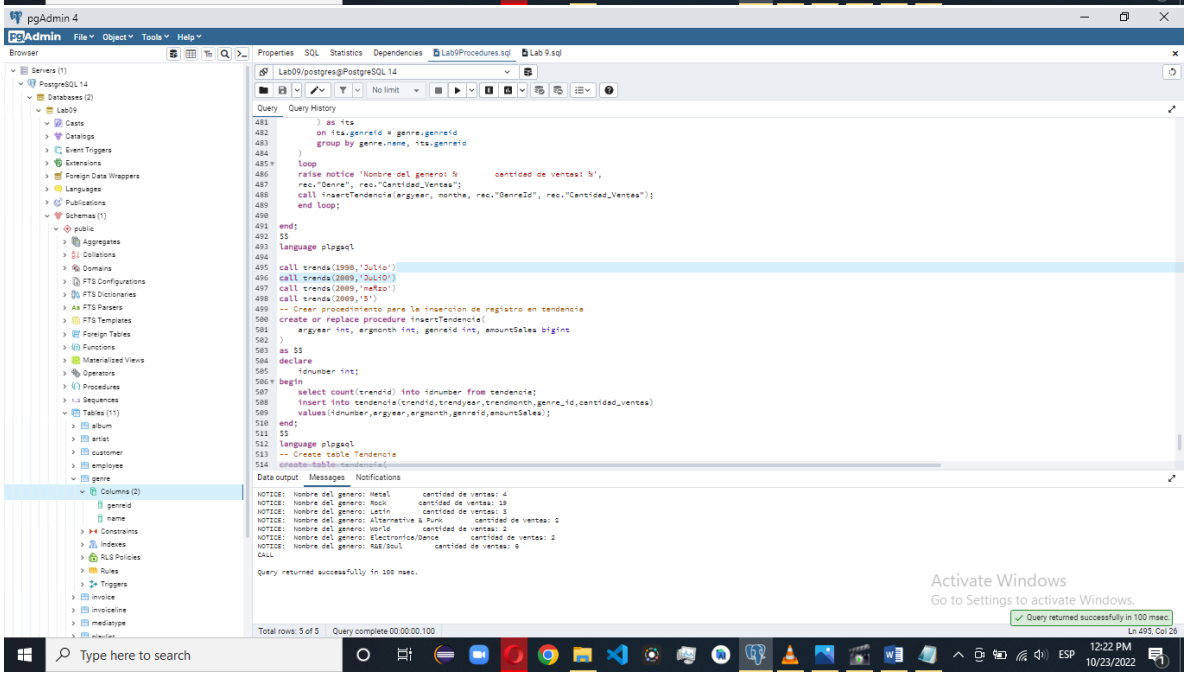
Error de mes ingresado invalido

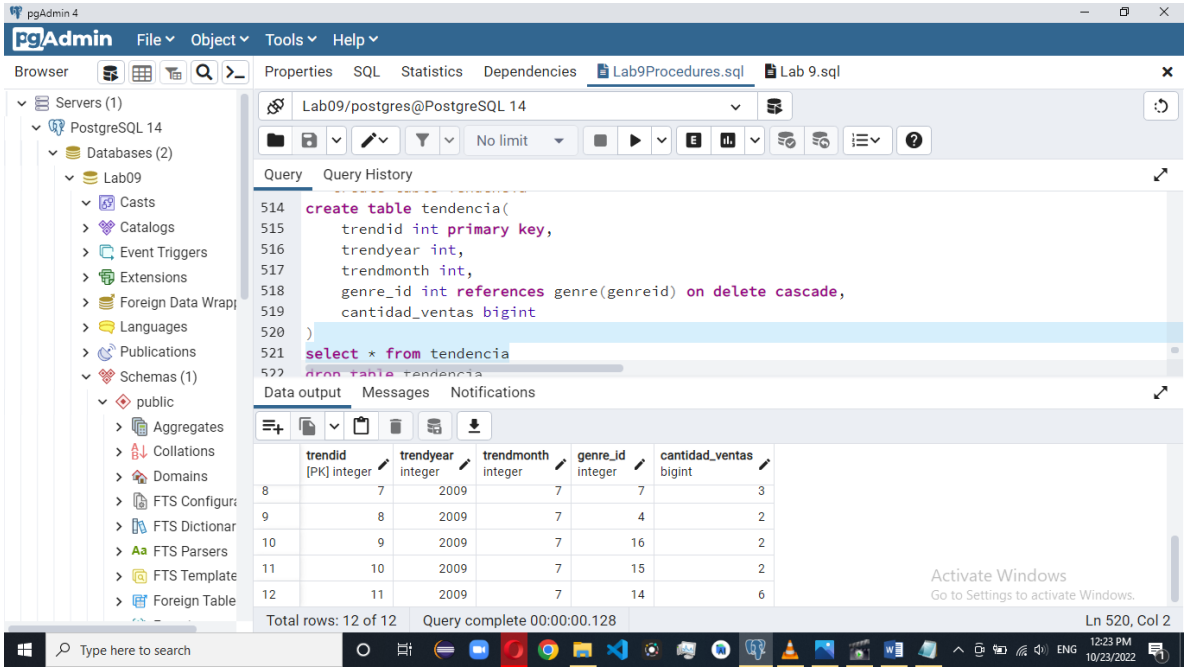


Inserción e ingreso de procedimiento valido




pgAdmin 4
Lab09/postgres@PostgreSQL 14
$$
language plpgsql
call trends(1990,'Julio')
call trends(2009,'JuLiO')
call trends(2009,'mAYo')
call trends(2009,'5')
-- Crear procedimiento para la insercion de registro en tendencia
create or replace procedure insertTendencia(
argyear int, argmonth int, genreid int, amountSales bigint
)
as $$
declare
idnumber int;
begin
select count(trendid) into idnumber from tendencia;
insert into tendencia(trendid,trendyear,trendmonth,genre_id,cantidad_ventas)
values(idnumber,argyear,argmonth,genreid,amountSales);
end;
$$
language plpgsql
-- Create table Tendencia
create table tendencia(
trendid int primary key,
trendyear int,
trendmonth int,
genre_id int references genre(genreid) on delete cascade,
cantidad_ventas bigint
)
select * from tendencia
drop table tendencia
-- End create table
Data output
Total rows: 5 of 5
Query complete 00:00:00.091
Activate Windows
Go to Settings to activate Windows.
Successfully run. Total query runtime: 91 msec. 5 rows affected.
Type here to search


pgAdmin 4
Lab09/postgres@PostgreSQL 14
) as tts
on tts.genreid = genre.genreid
group by genre.name, tts.genreid
loop
raise notice 'Nombre del genero: % cantidad de ventas: %',
rec."Genre", rec."Cantidad_Ventas";
call insertTendencia(argyear, months, rec."GenreId", rec."Cantidad_Ventas");
end loop;
end;
$$
language plpgsql
call trends(1990,'Julio')
call trends(2009,'JuLiO')
call trends(2009,'mAYo')
call trends(2009,'5')
-- Crear procedimiento para la insercion de registro en tendencia
create or replace procedure insertTendencia(
argyear int, argmonth int, genreid int, amountSales bigint
)
as $$
declare
idnumber int;
begin
select count(trendid) into idnumber from tendencia;
insert into tendencia(trendid,trendyear,trendmonth,genre_id,cantidad_ventas)
values(idnumber,argyear,argmonth,genreid,amountSales);
end;
$$
language plpgsql
-- Create table Tendencia
Data output
NOTICE: Nombre del genero: Metal cantidad de ventas: 4
NOTICE: Nombre del genero: Rock cantidad de ventas: 19
NOTICE: Nombre del genero: Latin cantidad de ventas: 3
NOTICE: Nombre del genero: Alternative & Punk cantidad de ventas: 2
NOTICE: Nombre del genero: World cantidad de ventas: 2
NOTICE: Nombre del genero: Electronica/Dance cantidad de ventas: 2
NOTICE: Nombre del genero: R&B/Soul cantidad de ventas: 6
CALL
Query returned successfully in 100 msec.
Total rows: 5 of 5
Query complete 00:00:00.100
Activate Windows
Go to Settings to activate Windows.
Query returned successfully in 100 msec.
Type here to search



Sql instrucciones

```
-- Nombre de procedimiento crear o reemplazar
create or replace procedure album_ventas_q(
--     Parametros de entrada
   idalbum int,
--     Parametro opcional
   atleast float default 0
 )

 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
   verification int;
begin
   select 1 into verification from album where albumid = idalbum;
   if (verification = 1 ) then
      select tra.name  as Song,Sum(invoiceLine.unitprice*invoiceLine.quantity)as
TotalSales_Dolares,
      Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as
TotalSales_Quetzales
      from invoiceLine inner join (
         select tr.name,tr.trackid, unitprice  from track as tr
         inner join (select * from album where albumid = idalbum) as al on
al.albumid = tr.albumid
      ) as tra
```

```
        on tra.trackid = invoiceLine.trackid
        group by tra.name
--          Is a where with functions
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast;
        commit;
    else
        raise notice 'No existe el album solicitado';
    end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql



-- Nombre de procedimiento crear o reemplazar
create or replace procedure album_ventas_q(
--      Parametros de entrada
    idalbum int,
--      Parametro opcional
    atleast float default 0
 )
 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
    verification int;
begin
    select 1 into verification from album where albumid = idalbum;
    if (verification = 1 ) then
        perform tra.name  as Song,Sum(invoiceLine.unitprice*invoiceLine.quantity)as
TotalSales_Dolares,
        Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as
TotalSales_Quetzales
        from invoiceLine inner join (
            select tr.name,tr.trackid, unitprice  from track as tr
            inner join (select * from album where albumid = idalbum) as al on
al.albumid = tr.albumid
        ) as tra
        on tra.trackid = invoiceLine.trackid
        group by tra.name
--          Is a where with functions
```

```
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast;
        commit;
    else
        raise notice 'No existe el album solicitado';
    end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql


-- Nombre de procedimiento crear o reemplazar
create or replace procedure album_ventas_q(
--     Parametros de entrada
    idalbum int,
--     Parametro opcional
    atleast float default 0
 )
 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
    verification int;
begin
    select 1 into verification from album where albumid = idalbum;
    if (verification = 1 ) then
        perform tra.name  as Song,Sum(invoiceLine.unitprice*invoiceLine.quantity)as
TotalSales_Dolares,
        Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as
TotalSales_Quetzales
        from invoiceLine inner join (
            select tr.name,tr.trackid, unitprice  from track as tr
            inner join (select * from album where albumid = idalbum) as al on
al.albumid = tr.albumid
        ) as tra
        on tra.trackid = invoiceLine.trackid
        group by tra.name
--         Is a where with functions
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast;
        commit;
    else
        raise notice 'No existe el album solicitado';
```

```
    end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql




-- ESTE ES EL QUE SIRVE RETORNA UN STRING CON EL NOTICE
-- Nombre de procedimiento crear o reemplazar
create or replace procedure album_ventas_q(
--     Parametros de entrada
    idalbum int,
--     Parametro opcional
    atleast float default 0
 )
 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
    verification int;
    rec record;
begin
    select 1 into verification from album where albumid = idalbum;
    if (verification = 1 ) then
        for rec in (select tra.name  as
"Song",Sum(invoiceLine.unitprice*invoiceLine.quantity)as "TotalSales_Dolares",
        Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as
"TotalSales_Quetzales"
        from invoiceLine inner join (
            select tr.name,tr.trackid, unitprice  from track as tr
            inner join (select * from album where albumid = idalbum) as al on
al.albumid = tr.albumid
        ) as tra
        on tra.trackid = invoiceLine.trackid
        group by tra.name
--          Is a where with functions
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast)
        loop
        raise notice 'Track name: % track total sales in dollars: % track total sales in
quetzales: %', rec."Song",rec."TotalSales_Dolares", rec."TotalSales_Quetzales";
```

```
        end loop;
    else
        raise notice 'No existe el album solicitado';
    end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql
-- Llamar un procedimiento el primero es el raise exception, el segundo es el de
retorno
call album_ventas_q(10000)
call album_ventas_q(1)
call album_ventas_q(1,0.99)
-- HASTA AQUI TERMINA

-- ESTE ES EL QUE SIRVE RETORNA UN STRING CON EL NOTICE ESTE
ES EL FINAL
-- Nombre de procedimiento crear o reemplazar
create or replace procedure album_ventas_q(
--     Parametros de entrada
    idalbum int,
--     Parametro opcional
    atleast float default 0
 )
 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
    verification int;
    rec record;
begin
    select 1 into verification from album where albumid = idalbum;
    if (verification = 1 ) then
        for rec in (
            select tra.ArtistName as "Artist", tra.AlbumName as "Album",
tra.TrackName  as "Song",
            Sum(invoiceLine.unitprice*invoiceLine.quantity)as "TotalSales_Dolares",
            Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as
"TotalSales_Quetzales" from invoiceLine inner join (
                select al.name as ArtistName, al.title as AlbumName, tr.name as
TrackName, tr.trackid, unitprice  from track as tr inner join (
                    select album.albumid, album.title, artist.name from album
```

```
                inner join (
                    select name, artistid from artist
                ) as artist
                on artist.artistid = album.artistid
                where albumid = idalbum
            ) as al on al.albumid = tr.albumid
        ) as tra
        on tra.trackid = invoiceLine.trackid
        group by tra.ArtistName, tra.AlbumName, tra.TrackName
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast
    )
    loop
    raise notice 'Artist name: %        album name: %            song name: % track
total sales in dollars: %           track total sales in quetzales: %', rec."Artist",
rec."Album",rec."Song",rec."TotalSales_Dolares", rec."TotalSales_Quetzales";
    end loop;
  else
    raise notice 'No existe el album solicitado';
  end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql
-- Llamar un procedimiento el primero es el raise exception, el segundo es el de
retorno
call album_ventas_q(10000)
call album_ventas_q(1)
call album_ventas_q(1,0.99)
-- HASTA AQUI TERMINA

create or replace function albumVentas(idalbum int, atleast float)
returns table (song varchar,TotalSales_Dolares numeric, TotalSales_Quetzales
numeric)
 --  Inicio
as $$
begin
  return query select tra.name  as
Song,Sum(invoiceLine.unitprice*invoiceLine.quantity)as TotalSales_Dolares,
  Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as TotalSales_Quetzales
  from invoiceLine inner join (
    select tr.name,tr.trackid, unitprice  from track as tr
```

```
        inner join (select * from album where albumid = idalbum) as al on al.albumid
= tr.albumid
    ) as tra
    on tra.trackid = invoiceLine.trackid
    group by tra.name
--        Is a where with functions
    having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast;
end;
$$
--  Declaracion de lenguaje
language plpgsql
-- Llamar una funcion
select album_ventas_q(10000)
-- Eliminar un procedimiento
drop procedure album_ventas_q(int, float)
-- Eliminar una funcion
drop function album_ventas_q(int,float)
drop function albumVentas(int,float)
-- Query
select tra.ArtistName, tra.AlbumName, tra.TrackName  as Song,
Sum(invoiceLine.unitprice*invoiceLine.quantity)as TotalSales_Dolares,
Sum(invoiceLine.unitprice*invoiceLine.quantity*7.80)as TotalSales_Quetzales
from invoiceLine inner join (
    select al.name as ArtistName, al.title as AlbumName, tr.name as TrackName,
tr.trackid, unitprice  from track as tr inner join (
       select album.albumid, album.title, artist.name from album
        inner join (
            select name, artistid from artist
        ) as artist
        on artist.artistid = album.artistid
        where albumid = 1
    ) as al on al.albumid = tr.albumid
) as tra
on tra.trackid = invoiceLine.trackid
group by tra.ArtistName, tra.AlbumName, tra.TrackName
having Sum(invoiceLine.unitprice*invoiceLine.quantity)>0.99
-- Query end
-- INIT THE SECOND PART
-- Query
select alartil.artistname as ARTIST, sum(alartil.unitprice*alartil.quantity) as
TotalWin,sum(alartil.unitprice*alartil.quantity*0.12) as WithoutIVA,
sum(alartil.unitprice*alartil.quantity*0.12*0.05) as WithoutIVAandISR from
invoice
```

```
inner join(
   select invoiceline.invoiceid, invoiceline.unitprice, invoiceline.quantity,
alart.trackid, alart.artistname, alart.albumname, alart.artistid from invoiceline
   inner join (
      select track.trackid,track.albumid, alar.artistname, alar.albumname, alar.artistid
from track
      inner join (
         select album.artistid, album.albumid, artist.artistname, album.title as
albumname from album
         inner join (
            select artistid, name as artistname from artist
         )as artist
         on artist.artistid = album.artistid
      ) as alar
      on alar.albumid = track.albumid
   ) as alart
   on alart.trackid = invoiceline.trackid
) as alartil
on alartil.invoiceid = invoice.invoiceid
where invoice.invoicedate >= '2009-01-01'and invoice.invoicedate<= '2009-12-
31'and alartil.artistid = '12'
group by alartil.artistname
-- END QUERY
-- VERIFICATION QUERY
select alartil.artistname as ARTIST, invoice.invoicedate as DATE from invoice
inner join(
   select invoiceline.invoiceid, invoiceline.unitprice, invoiceline.quantity,
alart.trackid, alart.artistname, alart.albumname, alart.artistid from invoiceline
   inner join (
      select track.trackid,track.albumid, alar.artistname, alar.albumname, alar.artistid
from track
      inner join (
         select album.artistid, album.albumid, artist.artistname, album.title as
albumname from album
         inner join (
            select artistid, name as artistname from artist
         )as artist
         on artist.artistid = album.artistid
      ) as alar
      on alar.albumid = track.albumid
   ) as alart
   on alart.trackid = invoiceline.trackid
) as alartil
```

```
on alartil.invoiceid = invoice.invoiceid
where invoice.invoicedate >= '2009-01-01'and invoice.invoicedate<= '2009-12-
31'and alartil.artistid = '12'
group by alartil.artistname, invoice.invoicedate
-- END VERIFICATION QUERY
-- PROCEDURE
create or replace procedure impuestos(
--     Parametros de entrada
    idartist int,
    yearentered int
)
 --  Inicio
as $$
--  Declaracion de variables para chequeo
declare
    verificationyear int;
    initdate date;
    lastdate date;
    verificationid int;
    verificationrecord int;
    rec record;
begin
    select make_date(yearentered, 1,1) into initdate;
    select make_date(yearentered, 12,31) into lastdate;
    select 1 into verificationyear from invoice where invoicedate>=initdate and
invoicedate<=lastdate;
    select 1 into verificationid from artist where artistid = idartist;
    select 1 into verificationrecord from invoice inner join(
            select invoiceline.invoiceid, invoiceline.unitprice, invoiceline.quantity,
alart.trackid, alart.artistname, alart.albumname, alart.artistid from invoiceline
            inner join (
                select track.trackid,track.albumid, alar.artistname, alar.albumname,
alar.artistid from track
                inner join (
                    select album.artistid, album.albumid, artist.artistname, album.title as
albumname from album
                    inner join (
                        select artistid, name as artistname from artist
                    )as artist
                    on artist.artistid = album.artistid
                ) as alar
                on alar.albumid = track.albumid
            ) as alart
```

```
            on alart.trackid = invoiceline.trackid
        ) as alartil
        on alartil.invoiceid = invoice.invoiceid
        where invoice.invoicedate >= initdate and invoice.invoicedate<= lastdate and
alartil.artistid = idartist
        group by alartil.artistname, invoice.invoicedate;
    if (verificationyear >0 and verificationid>0 and verificationrecord>0) then
       for rec in (
              select alartil.artistname as "ARTIST",
              sum(alartil.unitprice*alartil.quantity) as "TotalWin",
              sum(alartil.unitprice*alartil.quantity*0.12) as "WithoutIVA",
              sum(alartil.unitprice*alartil.quantity*0.12*0.05) as "WithoutIVAandISR"
from invoice
              inner join(
                 select invoiceline.invoiceid, invoiceline.unitprice, invoiceline.quantity,
alart.trackid,
                 alart.artistname, alart.albumname, alart.artistid from invoiceline
                 inner join (
                    select track.trackid,track.albumid, alar.artistname, alar.albumname,
alar.artistid from track
                    inner join (
                       select album.artistid, album.albumid, artist.artistname, album.title
as albumname from album
                       inner join (
                          select artistid, name as artistname from artist
                       )as artist
                       on artist.artistid = album.artistid
                    ) as alar
                    on alar.albumid = track.albumid
                 ) as alart
                 on alart.trackid = invoiceline.trackid
              ) as alartil
              on alartil.invoiceid = invoice.invoiceid
              where invoice.invoicedate >= initdate and invoice.invoicedate<= lastdate
and alartil.artistid = idartist
              group by alartil.artistname
           )
           loop
           raise notice 'Nombre del artista: %       total earnings: % track        IVA: %
ISR: %', rec."ARTIST",
rec."TotalWin",rec."WithoutIVA",rec."WithoutIVAandISR";
           end loop;
    else
```

```
        raise notice 'No hay registro que cumpla';
    end if;
--  fin
end;
$$
--  Declaracion de lenguaje
language plpgsql
call impuestos (1,1900)
call impuestos(15890,2009)
call impuestos(12,2009)
-- END PROCEDURE
drop procedure impuestos(int, int)
-- END SECOND PART
-- INIT THIRD PART

-- Query
select genre.name,count(genre.name) from genre
inner join (
    select track.trackid, track.genreid from track
    inner join (
        select invoiceline.invoiceid, invoiceline.invoicelineid,
        invoiceline.trackid, invoiceline.unitprice from invoiceLine
        inner join (
            select invoiceid from invoice where '2009-01-01'<=invoicedate and
invoicedate<= '2009-01-31'
        ) as invoice
        on invoice.invoiceid = invoiceline.invoiceid
    ) as it
    on it.trackid = track.trackid
) as its
on its.genreid = genre.genreid
group by genre.name

-- En query
create or replace procedure trends(argyear int, argmonth text)
as $$
declare
    verificationmvar text;
    initdate date;
    lastdate date;
    montha int;
    rec record;
begin
```

```
select lower (argmonth) into verificationmvar;
if(verificationmvar = 'enero')then
   initdate:=make_date(argyear,01,1);
   lastdate :=make_date(argyear, 01,31);
   montha:= 1;
elsif(verificationmvar = 'febrero')then
   initdate:=make_date(argyear,02,1);
   lastdate :=make_date(argyear, 02,31);
   montha:= 2;
elsif(verificationmvar = 'marzo')then
   initdate:=make_date(argyear,03,1);
   lastdate :=make_date(argyear, 03,31);
   montha:= 3;
elsif(verificationmvar = 'abril')then
   initdate:=make_date(argyear,04,1);
   lastdate :=make_date(argyear, 04,31);
   montha:= 4;
elsif(verificationmvar = 'mayo')then
   initdate:=make_date(argyear,05,1);
   lastdate :=make_date(argyear, 05,31);
   montha:= 5;
elsif(verificationmvar = 'junio')then
   initdate:=make_date(argyear,06,1);
   lastdate :=make_date(argyear, 06,31);
   montha:= 6;
elsif(verificationmvar = 'julio')then
   initdate:=make_date(argyear,07,1);
   lastdate :=make_date(argyear, 07,31);
   montha:= 7;
elsif(verificationmvar = 'agosto')then
   initdate:=make_date(argyear,08,1);
   lastdate :=make_date(argyear, 08,31);
   montha:= 8;
elsif(verificationmvar = 'septiembre')then
   initdate:=make_date(argyear,09,1);
   lastdate :=make_date(argyear,09,31);
   montha:= 9;
elsif(verificationmvar = 'octubre')then
   initdate:=make_date(argyear,10,1);
   lastdate :=make_date(argyear, 10,31);
   montha:= 10;
elsif(verificationmvar = 'noviembre')then
   initdate:=make_date(argyear,11,1);
```

```
        lastdate :=make_date(argyear, 11,31);
        montha:= 11;
    elsif(verificationmvar = 'diciembre')then
        initdate:=make_date(argyear,12,1);
        lastdate :=make_date(argyear, 12,31);
        montha:= 12;
    else
        raise exception  using message = 'Mes no valido',
        detail = 'Mes no valido',
        hint = 'Mes no valido',
        errcode = 'P3333';
    end if;
    if (2000<=argyear and argyear<=2016) then
    else
        raise exception  using message = 'Anio no valido' ,
        detail = 'Anio no valido' ,
        hint = 'Anio no valido' ,
        errcode = 'P3333';
    end if;
    for rec in (
        select genre.name as "Genre",count(genre.name) as "Cantidad_Ventas",
its.genreid as "GenreId" from genre
        inner join (
            select track.trackid, track.genreid from track
            inner join (
                select invoiceline.invoiceid, invoiceline.invoicelineid,
                invoiceline.trackid, invoiceline.unitprice from invoiceLine
                inner join (
                    select invoiceid from invoice
                    where initdate<=invoicedate and invoicedate<= lastdate
                ) as invoice
                on invoice.invoiceid = invoiceline.invoiceid
            ) as it
            on it.trackid = track.trackid
        ) as its
        on its.genreid = genre.genreid
        group by genre.name, its.genreid
    )
    loop
    raise notice 'Nombre del genero: %        cantidad de ventas: %',
    rec."Genre", rec."Cantidad_Ventas";
    call insertTendencia(argyear, montha, rec."GenreId", rec."Cantidad_Ventas");
    end loop;
```

```
end;
$$
language plpgsql

call trends(1990,'Julio')
call trends(2009,'JuLiO')
call trends(2009,'maRzo')
call trends(2009,'5')
-- Crear procedimiento para la insercion de registro en tendencia
create or replace procedure insertTendencia(
    argyear int, argmonth int, genreid int, amountSales bigint
)
as $$
declare
    idnumber int;
begin
    select count(trendid) into idnumber from tendencia;
    insert into tendencia(trendid,trendyear,trendmonth,genre_id,cantidad_ventas)
    values(idnumber,argyear,argmonth,genreid,amountSales);
end;
$$
language plpgsql
-- Create table Tendencia
create table tendencia(
    trendid int primary key,
    trendyear int,
    trendmonth int,
    genre_id int references genre(genreid) on delete cascade,
    cantidad_ventas bigint
)
select * from tendencia
drop table tendencia
-- End create table
```