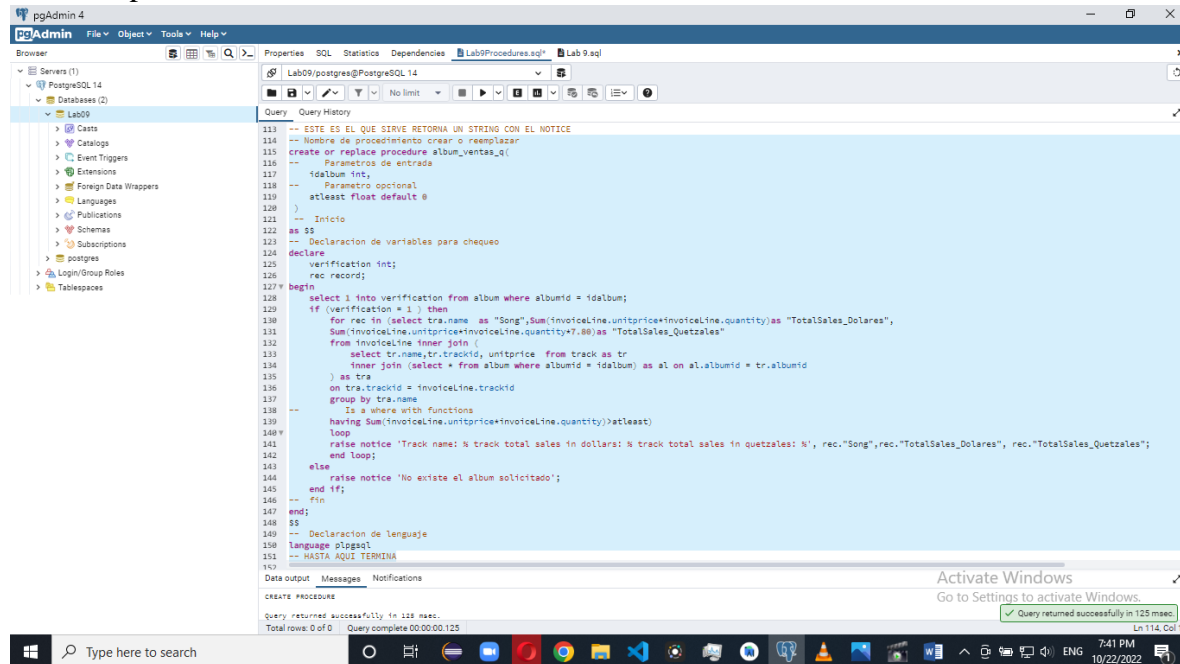


Laboratorio 09

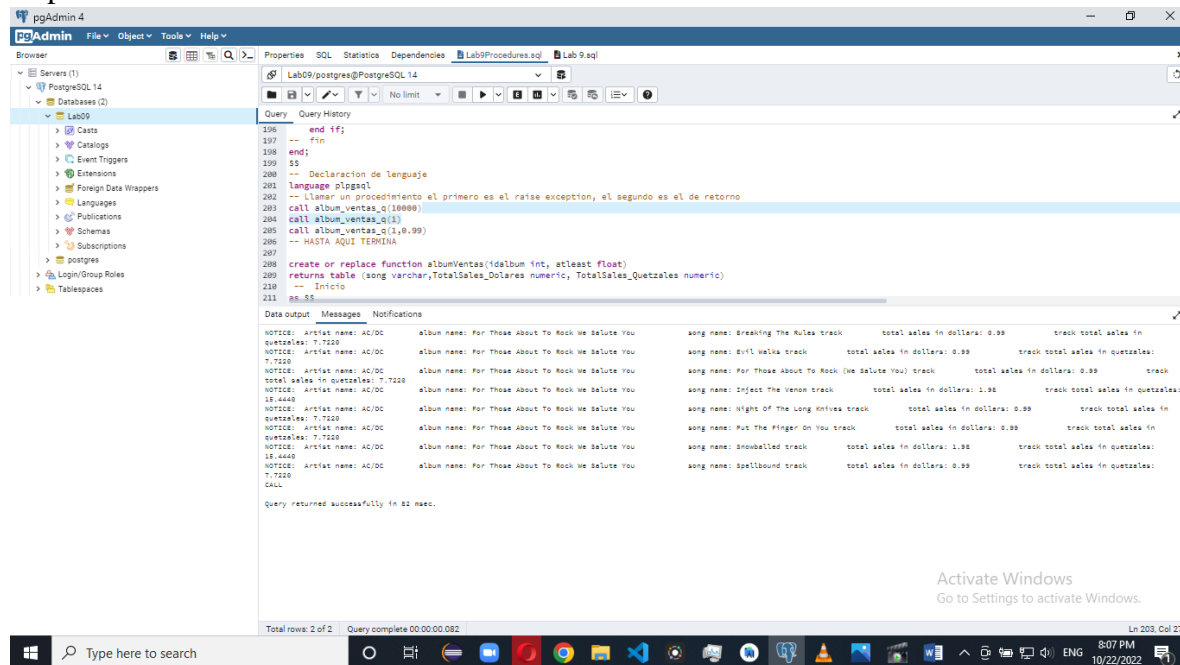
1. Crear un procedimiento



```
-- 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 $$
-- Declaracion de variables para chequeo
declare
    verification int;
    rec record;
begin
    select 1 into verification from album where albumid = idalbum;
    if (verification = 1) then
        for rec in (select tra.name as "Song", Sum(invoiceLine.unitprice*invoiceLine.quantity) as "TotalSales_Dolares",
            Sum(invoiceLine.unitprice*invoiceLine.quantity*7.86) as "TotalSales_Quetzales"
            from invoiceLine inner join (
                select tr.name, tr.trackid, unitprice from track as tr
                inner join (select * from album where albumid = idalbum) as al on al.albumid = tr.albumid
            ) as tra
            on tra.trackid = invoiceLine.trackid
            group by tra.name
        -- Is a where with functions
        having Sum(invoiceLine.unitprice*invoiceLine.quantity)>atleast)
        loop
            raise notice 'Track name: % track total sales in dollars: % track total sales in quetzales: %', rec."Song", rec."TotalSales_Dolares", rec."TotalSales_Quetzales";
        end loop;
    else
        raise notice 'No existe el album solicitado';
    end if;
-- fin
end;
$$
-- Declaracion de lenguaje
language plpgsql
-- HASTA AQUI TERMINA
--
```

Query returned successfully in 125 msec.
Total rows: 0 of 0 | Query complete 00:00:00.125

El procedimiento funciona

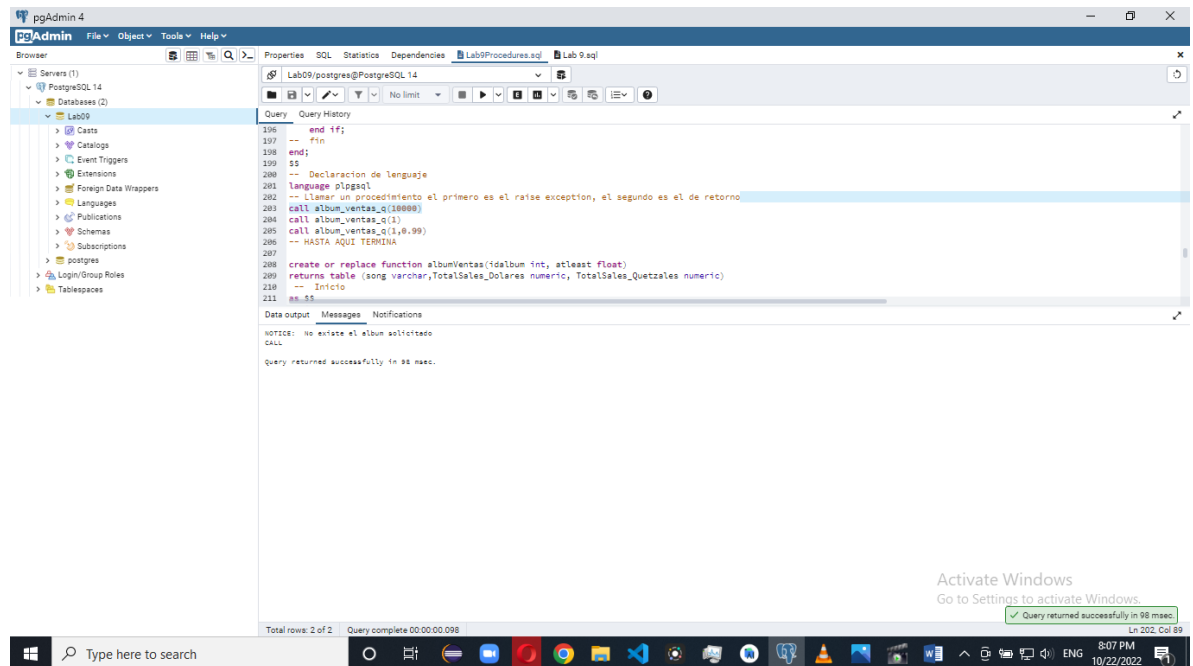


```
-- end if;
-- fin
end;
-- Declaracion de lenguaje
language plpgsql
-- Llamar un procedimiento el primero es el raise exception, el segundo es el de retorno
call album_ventas_q(100000);
call album_ventas_q(1);
call album_ventas_q(1,0.99);
-- HASTA AQUI TERMINA
--
```

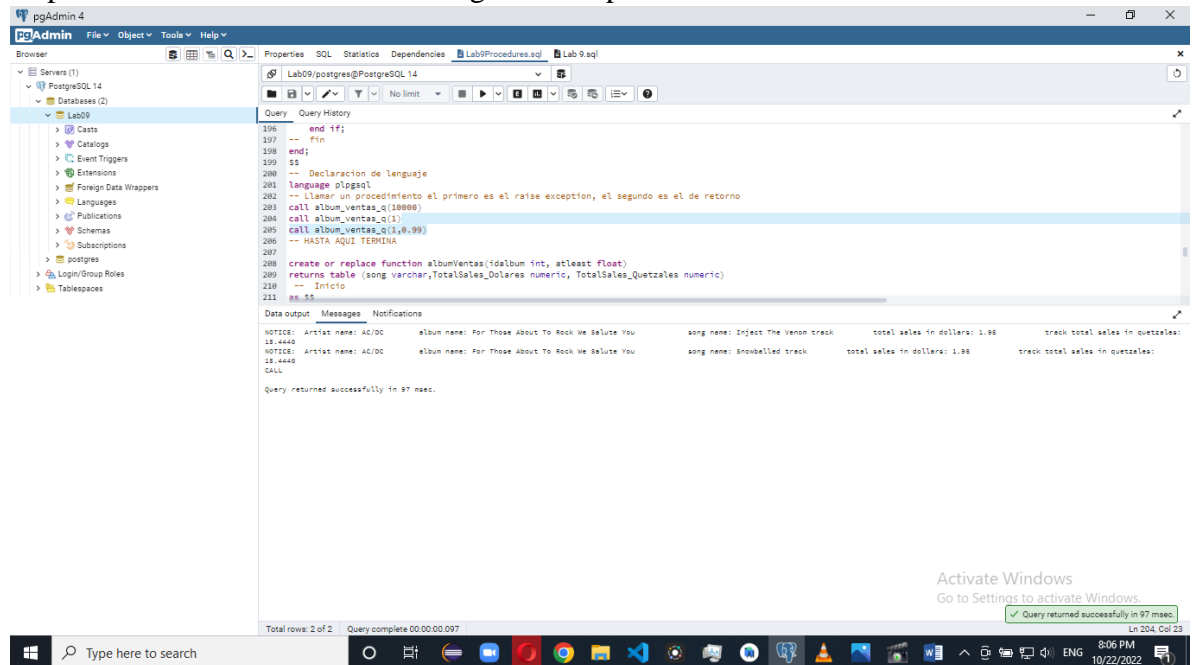
Query returned successfully in 82 msec.

Artist name:	Album name:	Song name:	total sales in dollars:	track total sales in quetzales:
AC/DC	For Those About To Rock We Salute You	Breaking the Rules track	0.99	7.7220
AC/DC	For Those About To Rock We Salute You	Evil Walla track	0.99	7.7220
AC/DC	For Those About To Rock We Salute You	For Those About To Rock (We Salute You) track	0.99	7.7220
AC/DC	For Those About To Rock We Salute You	Inject the Venom track	1.99	15.4440
AC/DC	For Those About To Rock We Salute You	High of The Long Knives track	0.99	7.7220
AC/DC	For Those About To Rock We Salute You	Put The Finger On You track	0.99	7.7220
AC/DC	For Those About To Rock We Salute You	Snowballed track	1.99	15.4440
AC/DC	For Those About To Rock We Salute You	Spellbound track	0.99	7.7220

El procedimiento lanza la excepción

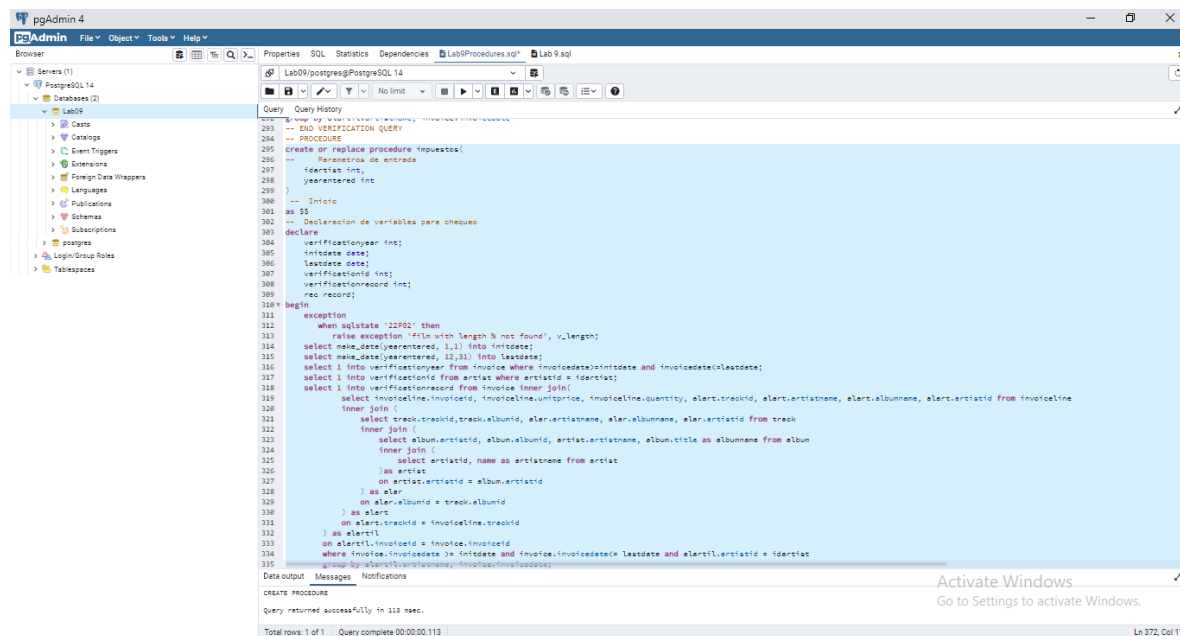


El procedimiento funciona con el argumento opcional



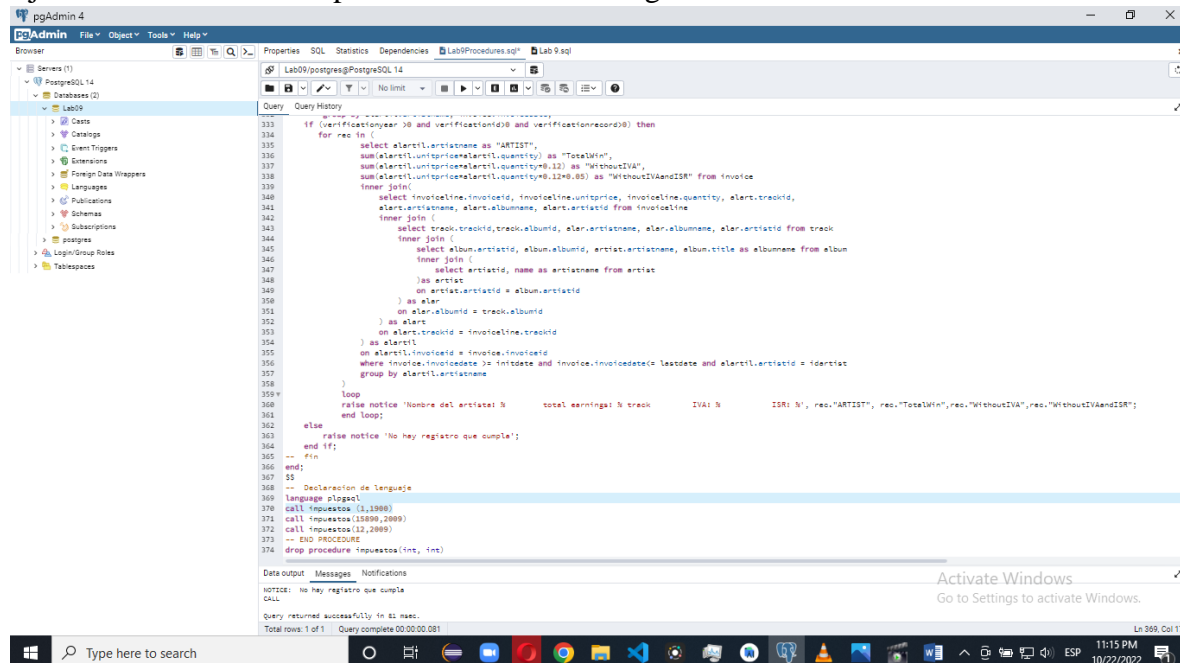
2.

Creación del procedimiento



The screenshot shows the pgAdmin 4 interface with the 'Lab9' database selected. The 'Query' tab is active, displaying a PL/pgSQL procedure named 'impuestos'. The procedure is designed to calculate taxes based on invoice data. It includes a 'BEGIN' block, a 'DECLARE' section for variables like 'verificationyear', 'intdate', 'lastdate', 'verificationid', and 'rec record'. The main logic is enclosed in a 'BEGIN' block with a 'when sqlstate '22P02' then' clause. Inside this clause, there are several 'SELECT' statements to retrieve data from 'invoice', 'invoiceinner', 'track', 'album', and 'artist' tables. The procedure ends with a 'DROP PROCEDURE' statement. The 'Data output' section shows the message 'CREATE PROCEDURE' and 'Query returned successfully in 113 msec.'.

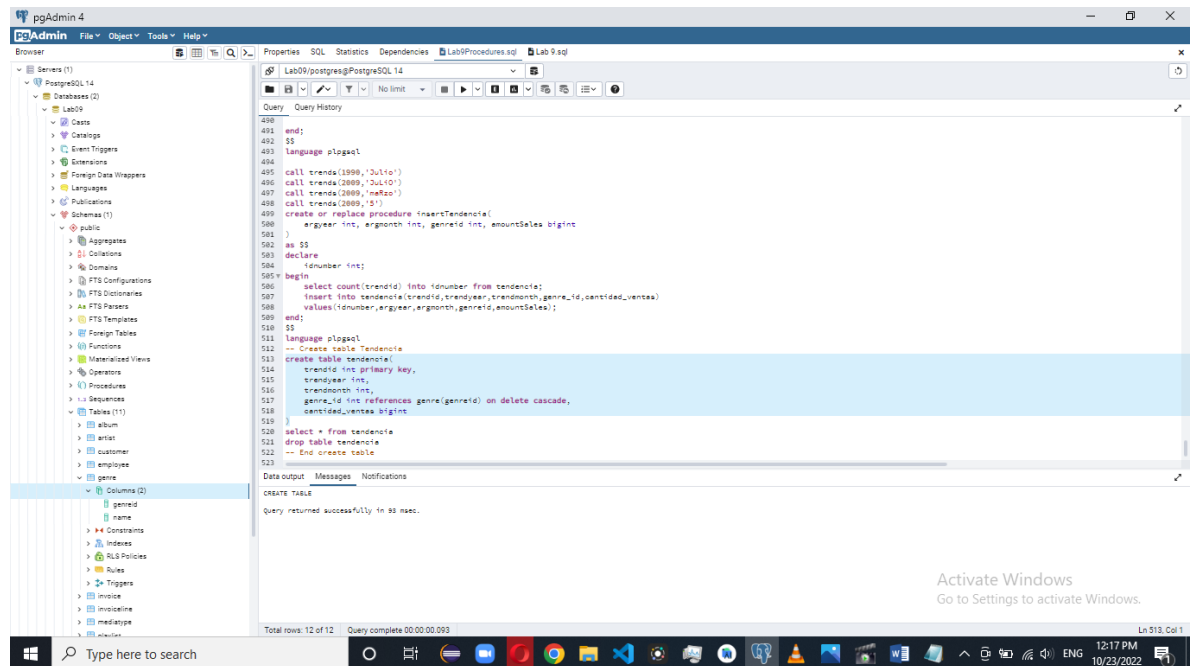
Ejecución de un error de procedimiento con un registro invalido de



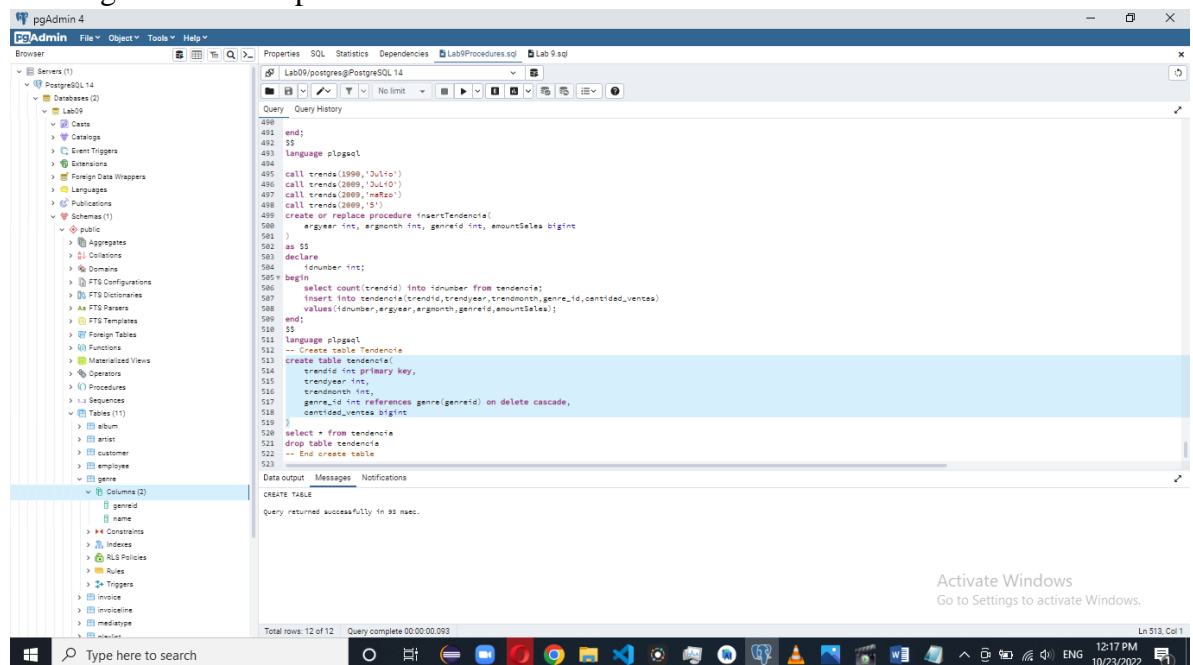
The screenshot shows the pgAdmin 4 interface with the 'Lab9' database selected. The 'Query' tab is active, displaying a PL/pgSQL procedure named 'impuestos'. The procedure is designed to calculate taxes based on invoice data. It includes a 'BEGIN' block, a 'DECLARE' section for variables like 'verificationyear', 'intdate', 'lastdate', 'verificationid', and 'rec record'. The main logic is enclosed in a 'BEGIN' block with a 'when sqlstate '22P02' then' clause. Inside this clause, there are several 'SELECT' statements to retrieve data from 'invoice', 'invoiceinner', 'track', 'album', and 'artist' tables. The procedure ends with a 'DROP PROCEDURE' statement. The 'Data output' section shows the message 'CREATE PROCEDURE' and 'Query returned successfully in 113 msec.'.

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.



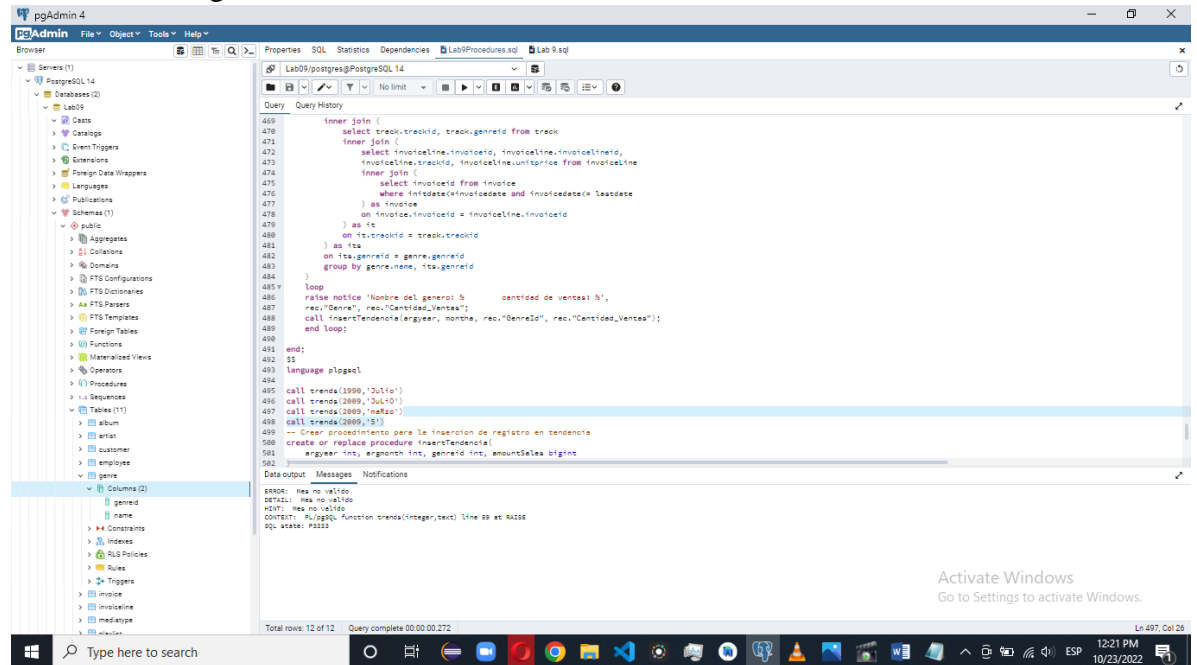
Crear procedimiento para obtener la cantidad de ventas generada según el genero basado en un mes y año ingresados.

The screenshot shows the pgAdmin 4 interface. On the left, the 'Servers' tree is expanded to 'PostgreSQL 14' > 'Lab09' > 'Schemas (1)' > 'public' > 'Columns (2)', with 'genreid' selected. The main pane displays a SQL query (lines 395-428) that creates or replaces a procedure named 'trends'. The query uses conditional logic to insert data into a table based on the month of the year. The status bar at the bottom indicates 'Query complete 00:00:00.096' and 'Ln 431, Col 42'. An 'Activate Windows' watermark is visible on the right side of the window.

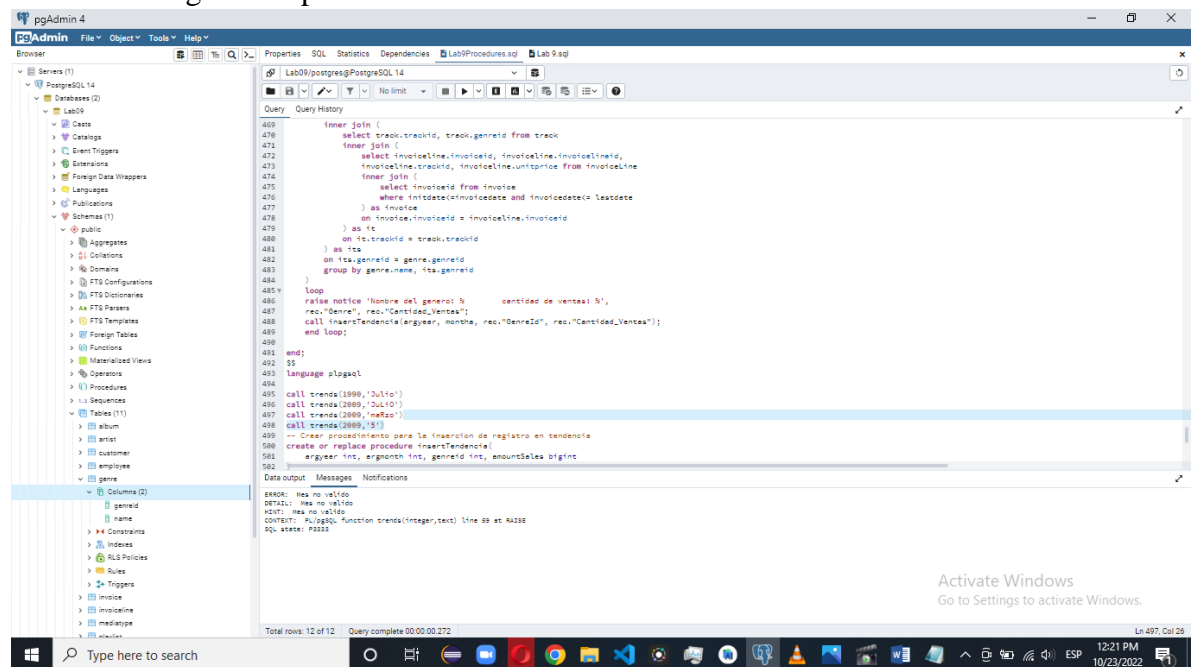
Pruebas de que funciona Error de año invalido

The screenshot shows the pgAdmin 4 interface with a different SQL query (lines 469-502) that calls the 'trends' procedure for the year 1999. The status bar indicates 'Query complete 00:00:00.121'. An error message is displayed in the 'Data output' pane: 'ERROR: año no válido' (ERROR: invalid year). The error details specify 'DETAIL: año no válido' and 'CONTEXT: PSQL Function trends(integer,text) line 66 at STATEMENT: PSQL'. An 'Activate Windows' watermark is visible on the right side of the window.

Error de mes ingresado invalido



Inserción e ingreso de procedimiento valido



The screenshot displays the pgAdmin 4 interface with a SQL query executed in the 'Query' tab. The query creates a table named 'Tendencia' and inserts data into it. The results are shown in the 'Data output' tab.

Query:

```
493 $$
494 language plpgsql
495 call trends(1999,'Juli@')
496 call trends(2009,'Juli@')
497 call trends(2009,'Metal')
498 call trends(2009,'S')
499 -- Crear procedimiento para la inserción de registro en tendencia
500 create or replace procedure insertTendencia(
501     argyear int, argmonth int, genrefid int, amountSales bigint
502 )
503 as $$
504 declare
505     tnumber int;
506 begin
507     select count(trendid) into tnumber from tendencia;
508     insert into tendencia(trendid,trendyear,trendmonth,genre_id,cantidad_ventas)
509     values(tnumber,argyear,argmonth,genrefid,amountSales);
510 end;
511 $$
512 language plpgsql
513 -- Create table Tendencia
514 create table Tendencia(
515     trendid int primary key,
516     trendyear int,
517     trendmonth int,
518     genre_id int references genre(genrefid) on delete cascade,
519     cantidad_ventas bigint
520 );
521 select * from tendencia
522 drop table tendencia
523 -- End create table
524
```

Data output:

trendid	trendyear	trendmonth	genre_id	cantidad_ventas
1	0	2009	2	9
2	1	2009	3	1
3	2	2009	3	7
4	3	2009	3	4
5	4	2009	3	11

Query complete 00:00:00.091

Activate Windows
Go to Settings to activate Windows.
Successfully run. Total query runtime: 91 msec. 5 rows affected.
Ln 520, Col 2

Type here to search

pgAdmin 4

Query:

```
481 ) as ita
482 on ita.genreid = genre.genrefid
483 group by genre.name, ita.genreid
484 )
485 loop
486     raise notice 'Nombre del genero: % cantidad de ventas: %',
487     rec."Genre", rec."Cantidad_Ventas";
488     call insertTendencia(argyear, month, rec."GenreId", rec."Cantidad_Ventas");
489 end loop;
490 end;
491 $$
492 language plpgsql
493 call trends(1999,'Juli@')
494 call trends(2009,'Juli@')
495 call trends(2009,'Metal')
496 call trends(2009,'S')
497 -- Crear procedimiento para la inserción de registro en tendencia
498 create or replace procedure insertTendencia(
499     argyear int, argmonth int, genrefid int, amountSales bigint
500 )
501 as $$
502 declare
503     tnumber int;
504 begin
505     select count(trendid) into tnumber from tendencia;
506     insert into tendencia(trendid,trendyear,trendmonth,genre_id,cantidad_ventas)
507     values(tnumber,argyear,argmonth,genrefid,amountSales);
508 end;
509 $$
510 language plpgsql
511 -- Create table Tendencia
512 create table Tendencia(
513     trendid int primary key,
514     trendyear int,
515     trendmonth int,
516     genre_id int references genre(genrefid) on delete cascade,
517     cantidad_ventas bigint
518 );
519 select * from tendencia
520 drop table tendencia
521 -- End create table
522
```

Data output:

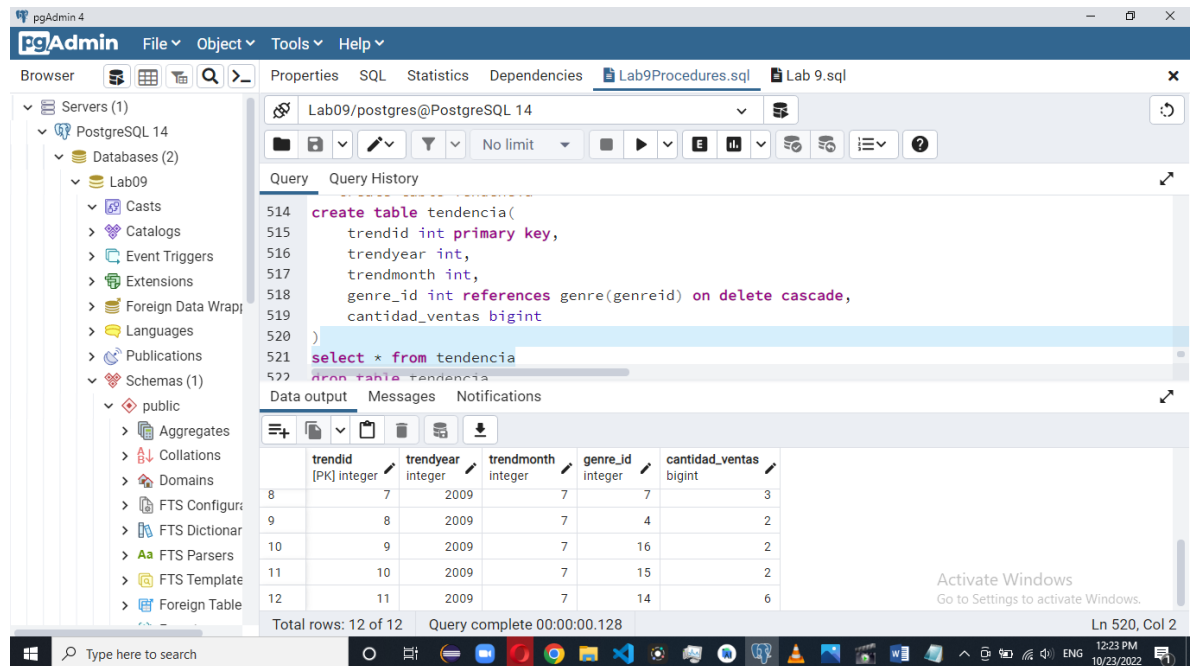
NOTICE: Nombre del genero: Metal	cantidad de ventas: 4
NOTICE: Nombre del genero: Rock	cantidad de ventas: 19
NOTICE: Nombre del genero: Latin	cantidad de ventas: 3
NOTICE: Nombre del genero: Alternative & Punk	cantidad de ventas: 2
NOTICE: Nombre del genero: Rap	cantidad de ventas: 2
NOTICE: Nombre del genero: Electronica/Dance	cantidad de ventas: 2
NOTICE: Nombre del genero: R&B/Soul	cantidad de ventas: 9

Query returned successfully in 100 msec.

Query complete 00:00:00.100

Activate Windows
Go to Settings to activate Windows.
Query returned successfully in 100 msec.
Ln 495, Col 28

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,
alart.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 invoiceid, invoiceid.invoicelineid,
        invoiceid.trackid, invoiceid.unitprice from invoiceid
        inner join (
            select invoiceid from invoice where '2009-01-01'<=invoicedate and
invoicedate<= '2009-01-31'
        ) as invoice
        on invoiceid.invoiceid = invoiceid.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;
elseif(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
```