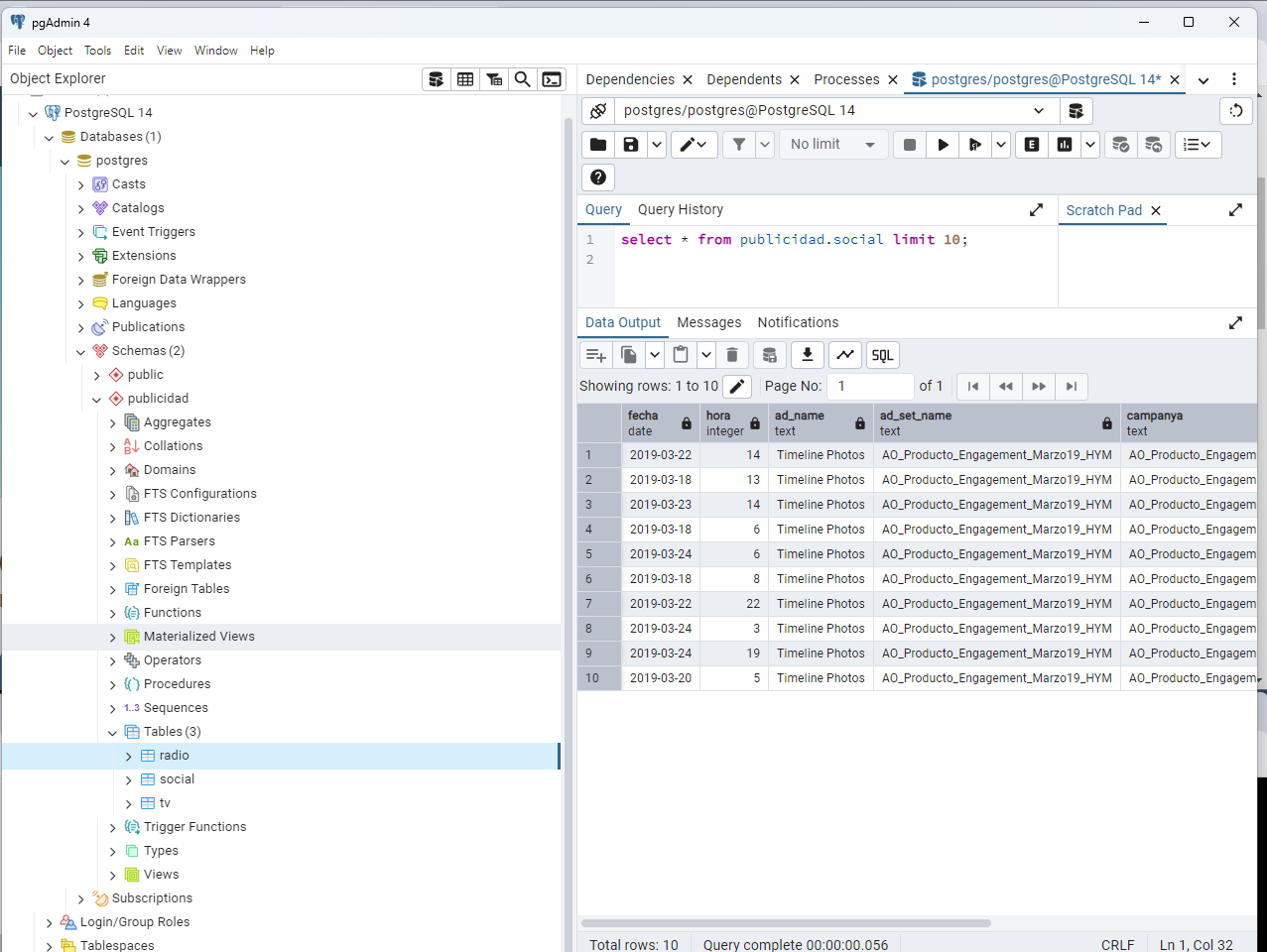
**Pascal, seihon**

**SPRINT1**

* **Queries utilizadas para crear cada una de las tablas.**
* **Para cada tabla, presentar un pantallazo con el resultado de una query que obtiene 10 registros.**
* **Validar que el número de registros cargados coincide con el número de filas de los ficheros en origen.**

****

* **Queries utilizadas para crear cada una de las tablas.**

-- Crear el schema "publicidad"

CREATE SCHEMA IF NOT EXISTS publicidad;

**-- Crear Tabla TV**

CREATE TABLE publicidad.tv (

fecha\_hora\_minuto TIMESTAMP,

campanya TEXT,

cadena TEXT,

franja TEXT,

creatividad TEXT,

duracion INT,

posicion TEXT,

grps\_ad16 NUMERIC,

inversion NUMERIC

);

**-- Crear Tabla Radio**

CREATE TABLE publicidad.radio (

fecha\_hora\_minuto TIMESTAMP,

franja TEXT,

emisora TEXT,

campanya TEXT,

formato TEXT,

posicion TEXT,

duracion INT,

grps\_ad16 NUMERIC,

inversion NUMERIC

);

**-- Crear Tabla Social**

CREATE TABLE publicidad.social (

fecha DATE,

hora INT,

ad\_name TEXT,

ad\_set\_name TEXT,

campanya TEXT,

impresiones INT,

clics INT,

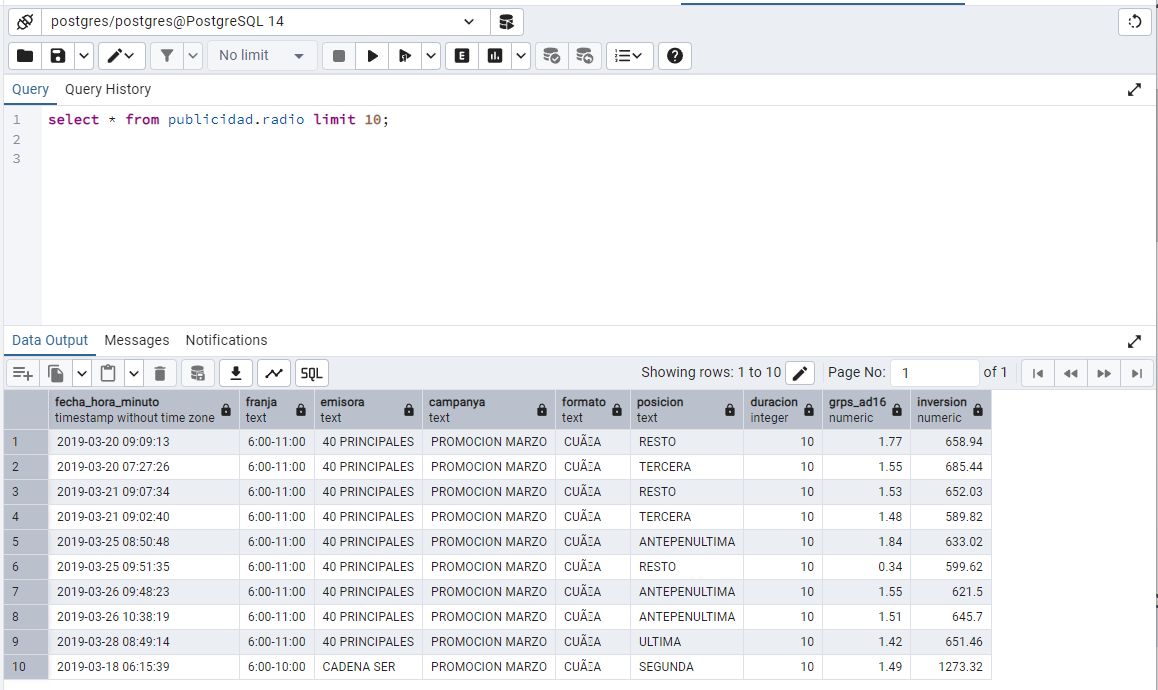
leads INT,

inversion NUMERIC

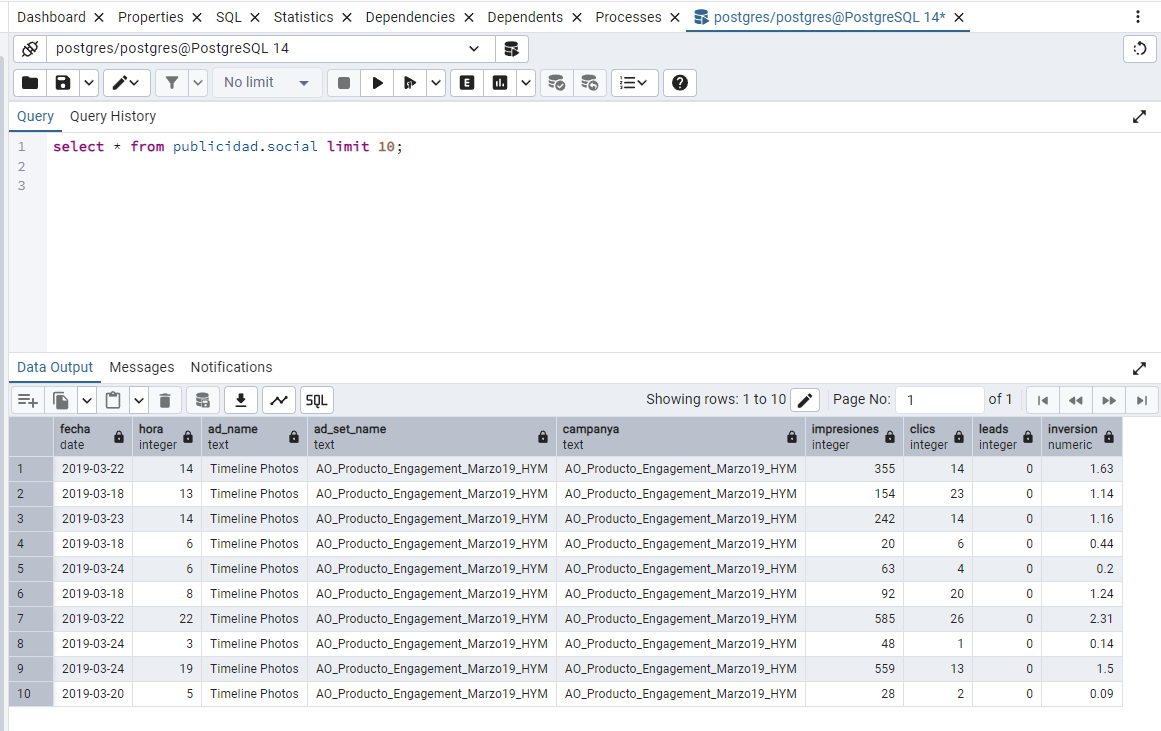
);

* **Para cada tabla, presentar un pantallazo con el resultado de una query que obtiene 10 registros.**

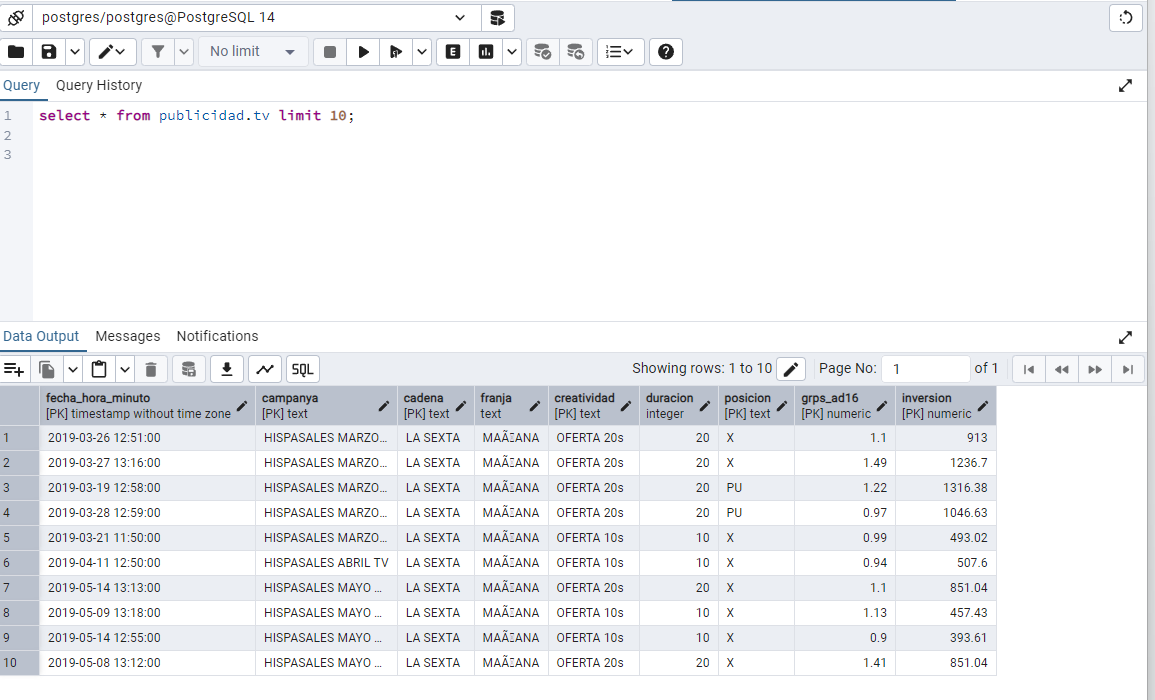
–Primeros 10 registros de la tabla radio

****

–Primeros 10 registros de la tabla social

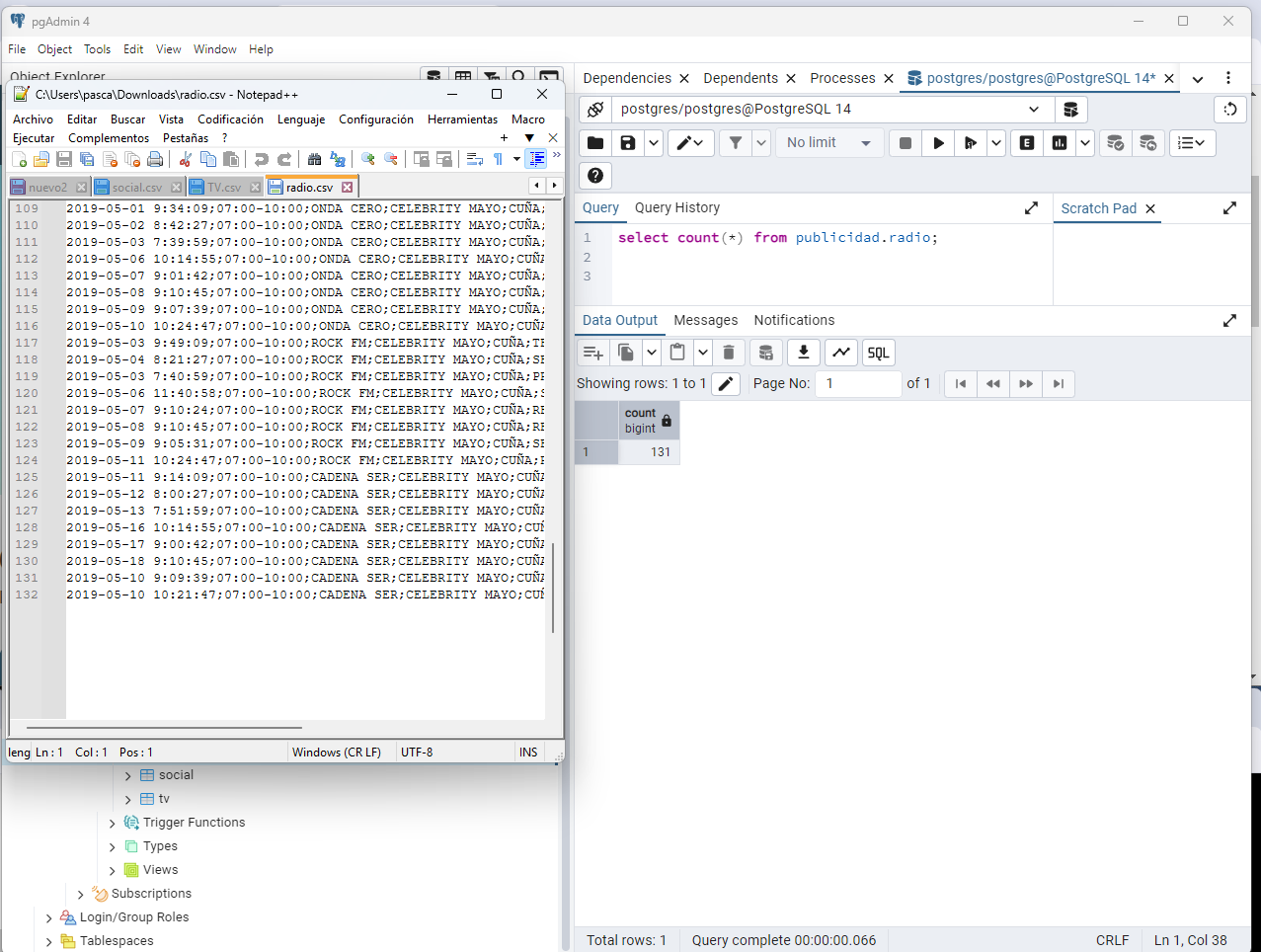
****

–Primeros 10 registros de la tabla tv

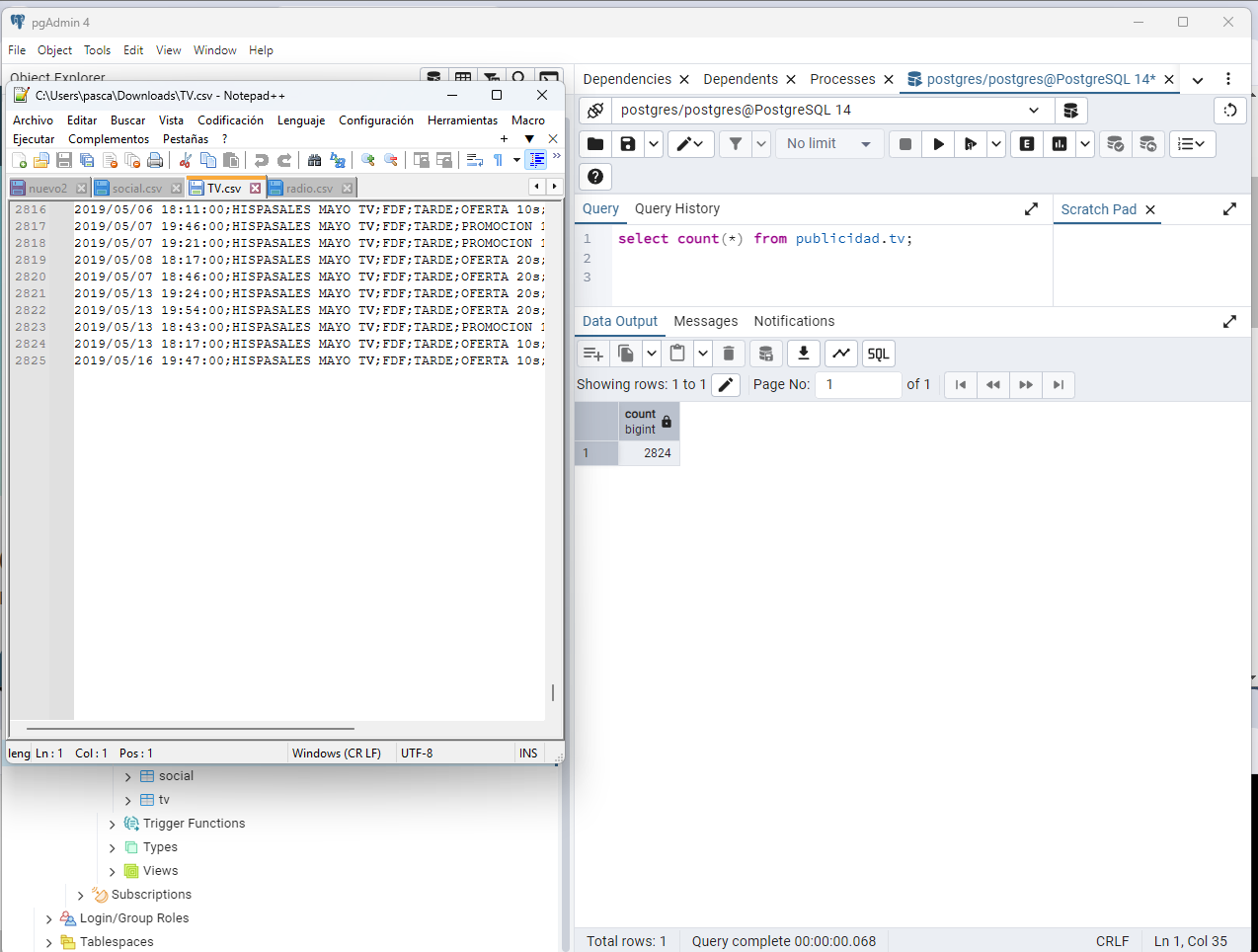
****

* **Validar que el número de registros cargados coincide con el número de filas de los ficheros en origen.**

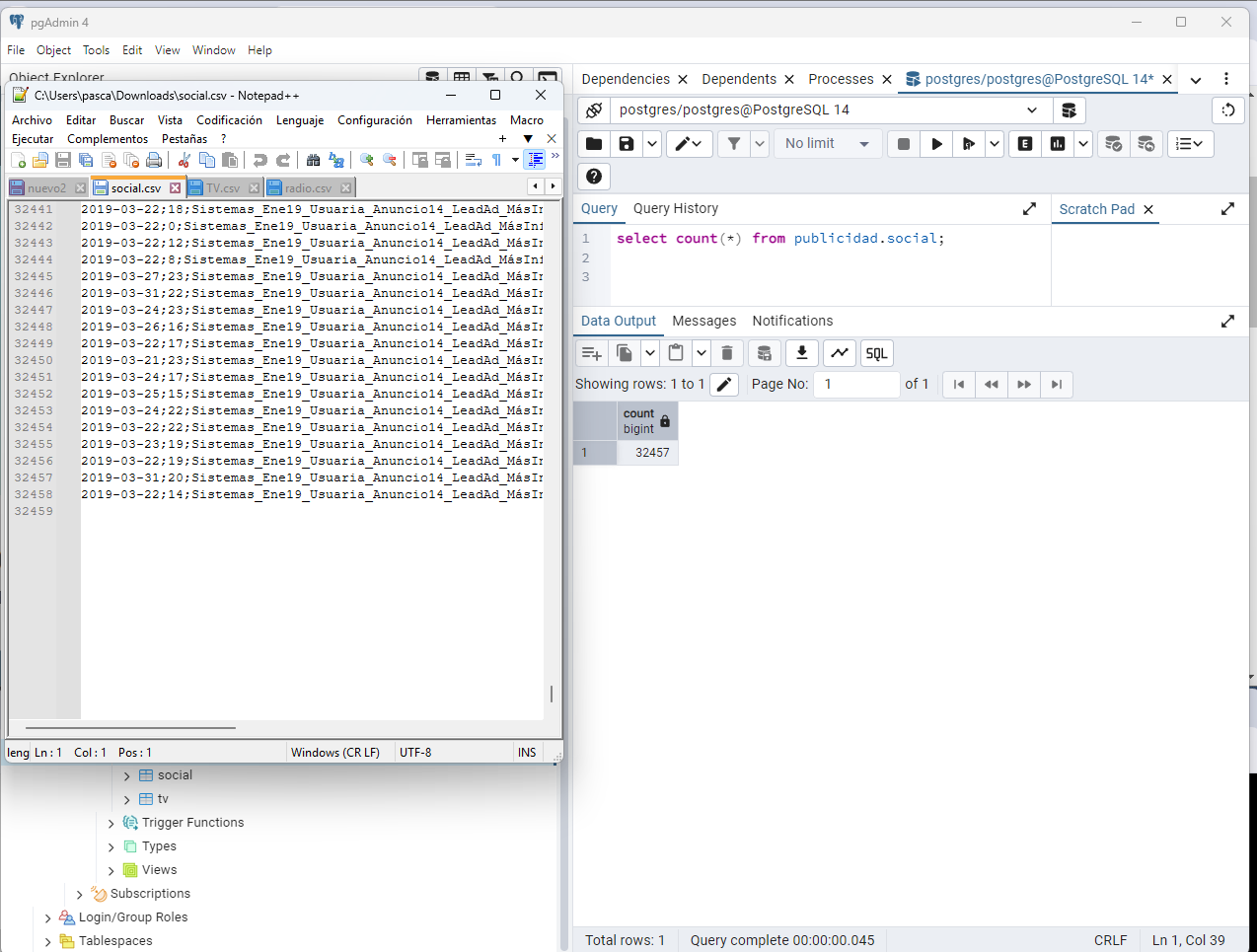
Conteo de numero registros de la tabla radio **VS** fichero

****

Conteo de numero registros de la tabla tv **VS** fichero

****

Conteo de numero registros de la tabla social **VS** fichero

****

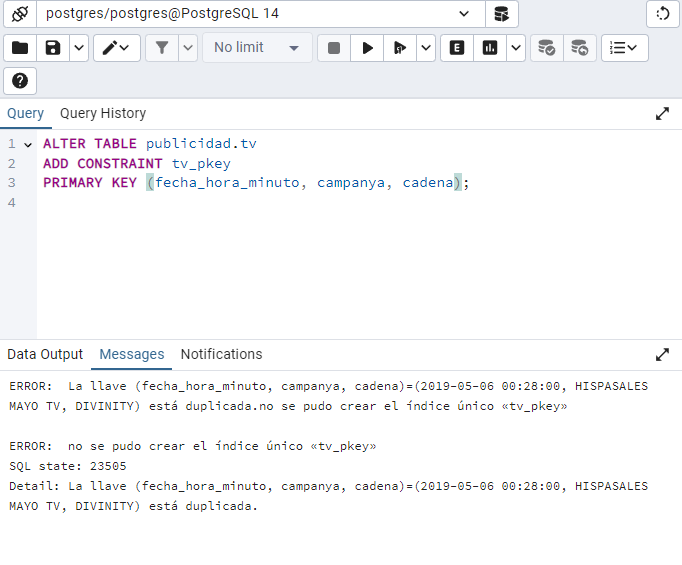
**Parte opcional:**

**Proponga una clave primaria para la tabla de TV y justifique su respuesta.**

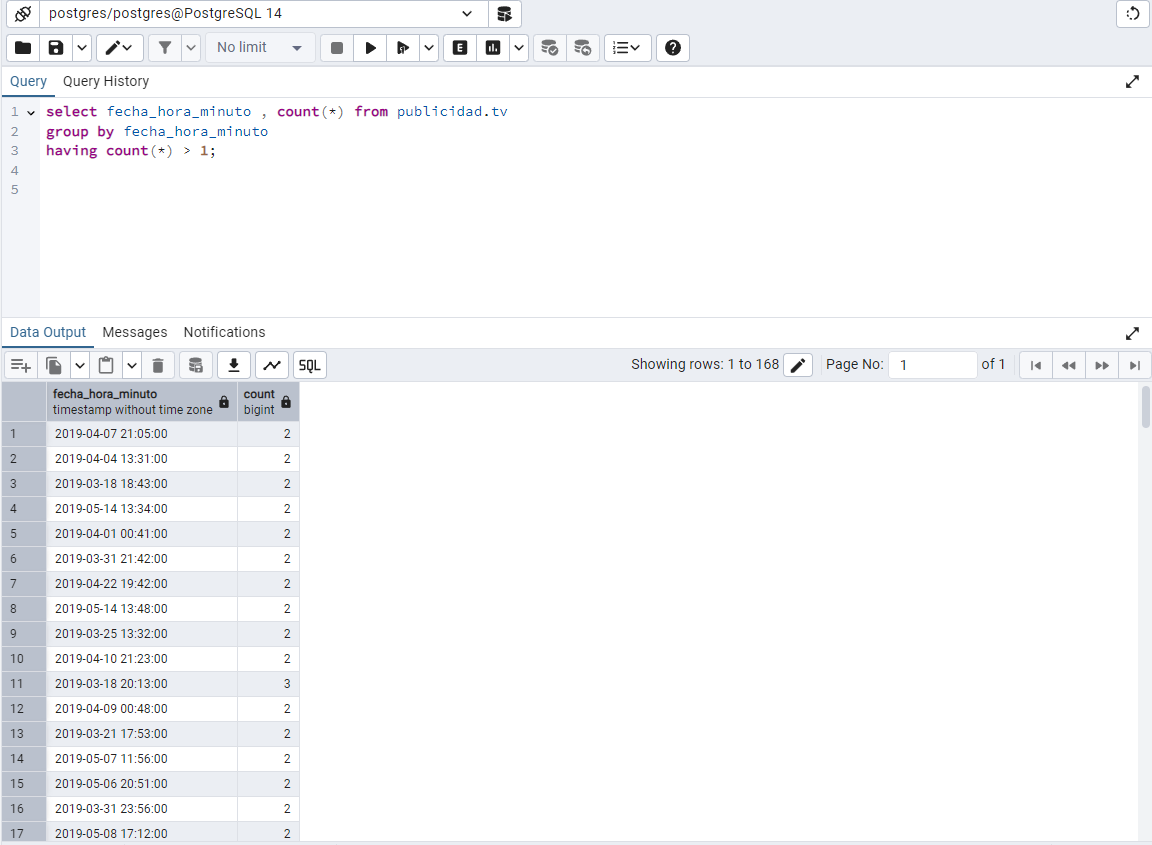
En miopinion, la combinación de **fecha\_hora\_minuto, campanya, cadena** permite identificar cada registro de manera única, evitando duplicados.

**Justificación**:

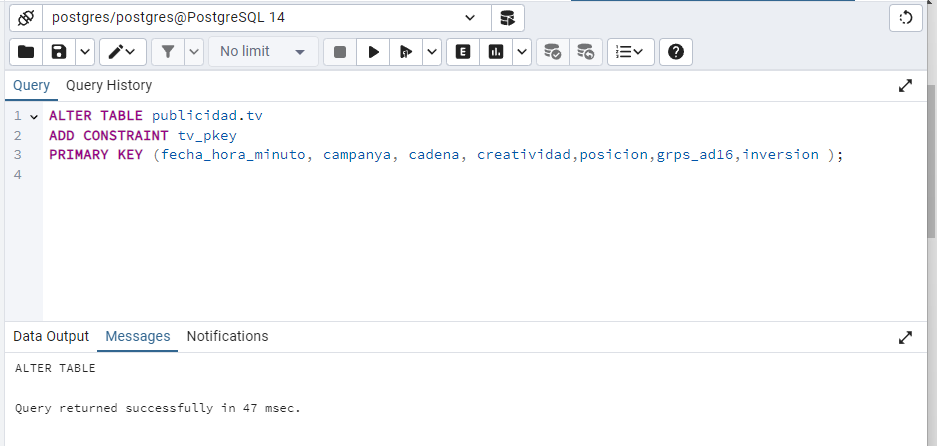
1. **fecha\_hora\_minuto**: La combinación de fecha y hora garantiza que cada registro tenga una marca temporal única.
2. **campanya**: Cada campaña puede tener múltiples anuncios, pero combinada con la fecha y hora ayuda a identificar anuncios específicos.
3. **cadena**: Una misma campaña puede transmitirse en varias cadenas de televisión, por lo que este campo ayuda a diferenciar registros.

****

**Los datos importados en la tabla tv contienes duplicados en fecha\_hora\_minutos**

****

Como mostra la siguiente la siguiente imagen he conseguido crear la clave primeria combinando las siguientes columnas .

**  
  
  
  
  
Sprint 2**

**--Creación de la tabla kpi**

**DROP TABLE IF EXISTS publicidad.kpi;**

**CREATE TABLE publicidad.kpi(**

**id SERIAL PRIMARY KEY,**

**fecha\_hora\_minuto TIMESTAMP NOT NULL,**

**dispositivo VARCHAR(20) NOT NULL,**

**visitas INT NOT NULL,**

**unidades\_vendidas INT NOT NULL,**

**importe DECIMAL(10,2) NOT NULL**

**);**

**Select \* from publicidad.kpi limit 10;**

**--Creación de la tabla publicitaria**

**DROP TABLE IF EXISTS publicidad.publicitaria;**

**CREATE TABLE publicidad.publicitaria (**

**id\_publicidad SERIAL PRIMARY KEY,**

**fecha DATE NOT NULL,**

**visitas\_web INT,**

**unidades\_vendidas INT,**

**importe DECIMAL(10,2),**

**inversion\_total DECIMAL(10,2),**

**inversion\_tv DECIMAL(10,2),**

**inversion\_radio DECIMAL(10,2),**

**inversion\_social DECIMAL(10,2)**

**);**

**-- Aprovisionemos la tabla publicitaria a partir de las tablas creadas en el sprint 1 (radio, social , tv y kpi)**

**INSERT INTO publicidad.publicitaria (fecha, visitas\_web, unidades\_vendidas, importe,inversion\_total, inversion\_tv,inversion\_radio,inversion\_social)**

**SELECT**

**kpi.Fecha as Fecha,**

**COALESCE( kpi.Visitas::NUMERIC, 0) as Visitas\_web,**

**COALESCE(kpi.Unidades\_vendidas::NUMERIC,0) as Unidades\_vendidas,**

**COALESCE(kpi.Importe::NUMERIC,0) as Importe,**

**(COALESCE(tv.Inversion\_tv::NUMERIC,0) + COALESCE(radio.Inversion\_radio::NUMERIC,0) + COALESCE(social.Inversion\_social::NUMERIC,0) ) as Interversion\_total,**

**COALESCE(tv.Inversion\_tv::NUMERIC,0) as Inversion\_TV,**

**COALESCE(radio.Inversion\_radio::NUMERIC,0) as Inversion\_radio,**

**COALESCE(social.Inversion\_social::NUMERIC,0) as Inversion\_social**

**FROM**

**(SELECT DATE("kpi"."fecha\_hora\_minuto") as Fecha,**

**SUM("kpi"."visitas") as Visitas,**

**SUM("kpi"."unidades\_vendidas") as Unidades\_vendidas,**

**SUM("kpi"."importe") as Importe**

**FROM "publicidad"."kpi"**

**GROUP BY Fecha**

**) kpi**

**LEFT JOIN (SELECT**

**DATE("tv"."fecha\_hora\_minuto") as Fecha,**

**SUM("tv"."inversion") as Inversion\_tv**

**FROM "publicidad"."tv"**

**GROUP BY fecha) tv**

**ON kpi.fecha = tv.fecha**

**LEFT JOIN (SELECT**

**DATE("radio"."fecha\_hora\_minuto") as Fecha,**

**SUM("radio"."inversion") as Inversion\_radio**

**FROM "publicidad"."radio"**

**GROUP BY fecha) radio**

**ON kpi.fecha = radio.fecha**

**LEFT JOIN (SELECT**

**DATE("social"."fecha") as Fecha,**

**SUM("social"."inversion") as Inversion\_social**

**FROM "publicidad"."social"**

**GROUP BY fecha) social**

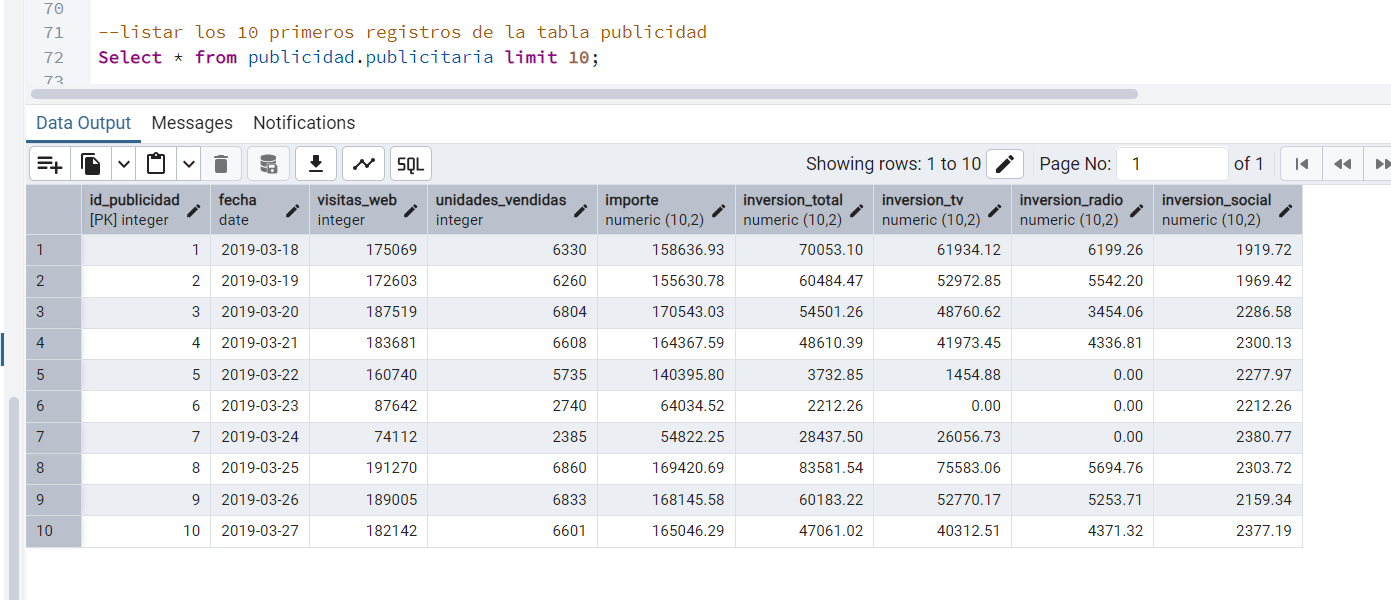
**ON kpi.fecha = social.fecha**

**ORDER BY**

**kpi.fecha;**

**--listar los 10 primeros registros de la tabla publicidad**

**Select \* from publicidad.publicitaria limit 10;**

****

**-- Medio con más gasto publicitario**

**SELECT 'TV' AS medio, SUM(inversion\_tv) AS total**

**FROM publicidad.publicitaria**

**UNION ALL**

**SELECT 'Radio'AS medio, SUM(inversion\_radio)**

**FROM publicidad.publicitaria**

**UNION ALL**

**SELECT 'Social'AS medio, SUM(inversion\_social)**

**FROM publicidad.publicitaria**

**ORDER BY total DESC**

**LIMIT 1;**



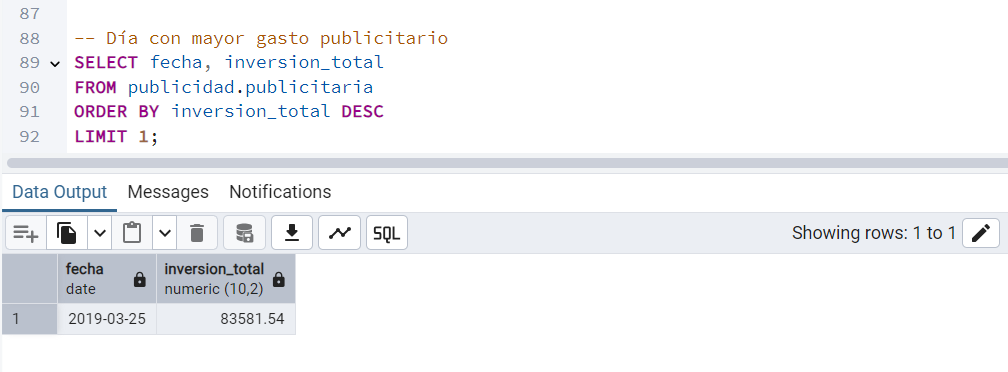
**-- Día con mayor gasto publicitario**

**SELECT fecha, inversion\_total**

**FROM publicidad.publicitaria**

**ORDER BY inversion\_totalDESC**

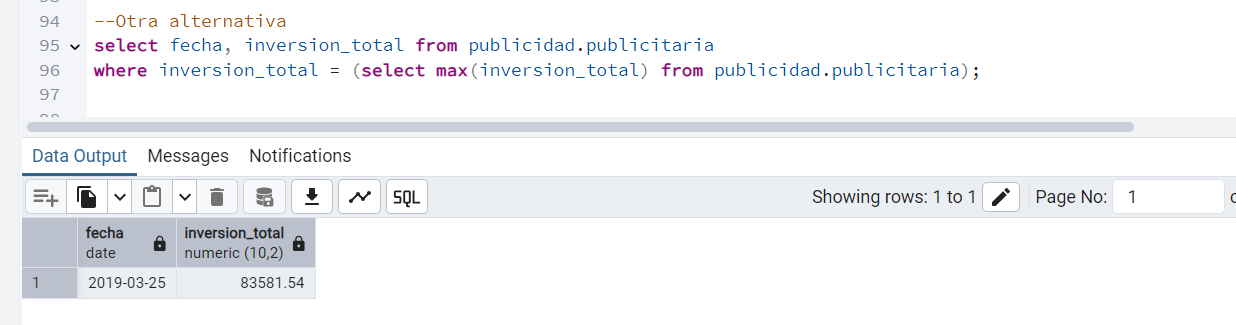
**LIMIT 1;**

****

**--Otra alternativa**

**select fecha, inversion\_totalfrom publicidad.publicitaria**

**where inversion\_total= (select max(inversion\_total) from publicidad.publicitaria);**

****

**-- Crear Vista**

**DROP view IF EXISTS publicidad.vista\_kpi;**

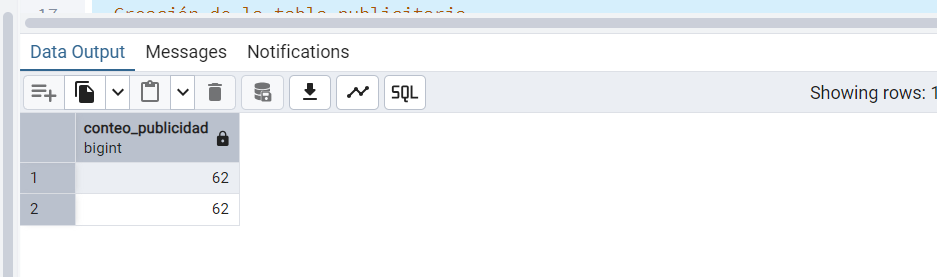
**CREATE VIEW publicidad.vista\_kpi AS**

**SELECT \* FROM publicidad.publicitaria;**

**select count (\*) as conteo\_publicidad from publicidad.publicitaria**

**Union all**

**select count (\*) as conteo\_Vista\_publicidad from publicidad.vista\_kpi;**

****

**--¿Qué ventajas tiene una vista sobre una tabla?**

**/\*Las vistas en SQL brindan numerosas ventajas en comparación con las tablas,**

**ya que contribuyen a una mejor seguridad, simplicidad y eficiencia en la gestión de la información. Permiten controlar**

**el acceso a datos sensibles, agilizar consultas complejas y promover la reutilización del código.**

**Asimismo, facilitan el mantenimiento del sistema, ya que los cambios en la base de datos pueden realizarse**

**sin afectar directamente a los usuarios. Aunque las vistas no almacenan datos por sí mismas (excepto las vistas materializadas),**

**su implementación adecuada puede mejorar el rendimiento y la organización de la información.**

**En definitiva, las vistas son una herramienta valiosa para optimizar la estructura, accesibilidad y**

**protección de los datos dentro de una base de datos.\*/**

**Sprint 3  
  
  
Opinión sobre la solución propuesta por el director de IT**

Desde el equipo de Data & Analytics, creemos que la propuesta de incrementar las capacidades del sistema RDB actual para almacenar y analizar noticias de otros periódicos, blogs y redes sociales no es la más idónea. Aunque las bases de datos relacionales (RDB) son excelentes para almacenar datos estructurados con relaciones bien definidas, tienen limitaciones cuando se trata de gestionar grandes volúmenes de datos no estructurados y semiestructurados, como texto libre en noticias y comentarios en redes sociales.

* **Limitaciones del enfoque basado en RDB:**

- **Escalabilidad**: Las bases de datos relacionales pueden tener dificultades para escalar horizontalmente, lo cual es crucial al manejar volúmenes crecientes de datos de múltiples fuentes externas.

- **Flexibilidad**: Los datos de noticias y redes sociales no tienen un esquema fijo y pueden variar en formato y estructura. Un RDB requeriría modificaciones constantes de esquema para adaptarse a estos cambios.

- **Procesamiento de Texto y Análisis Semántico:** Las bases de datos relacionales no están optimizadas para el procesamiento de lenguaje natural (NLP), minería de texto y análisis de sentimientos, tareas esenciales en este caso.

* **Propuesta Alternativa:** Uso de una combinación de tecnologías

Para enfrentar los desafíos mencionados, proponemos una arquitectura basada en tecnologías NoSQL y Big Data:

- B**ase de Datos NoSQL** (Ej. MongoDB o Elasticsearch):

- **MongoDB** permite almacenar documentos JSON de forma flexible, facilitando la integración de datos de múltiples fuentes con estructuras variadas.

- **Elasticsearch** es ideal para búsquedas en texto completo, agilizando la recuperación de noticias relevantes y el análisis de tendencias.

- **Data Lake y Procesamiento Distribuido (**Ej. Apache Hadoop o Apache Spark):

- Un Data Lake en Amazon S3, Google Cloud Storage o Hadoop HDFS puede almacenar datos en crudo para su posterior procesamiento.

- Apache Spark permite el análisis avanzado, incluyendo NLP, análisis de sentimientos y correlaciones entre noticias y tendencias en redes sociales.

- Pipeline de Ingesta de Datos (Ej. Apache Kafka o AWS Kinesis):

- Un sistema de mensajería en tiempo real como Kafka puede capturar datos de redes sociales y otras fuentes, asegurando que la plataforma pueda procesar y analizar información en tiempo real.

**Resumen:**

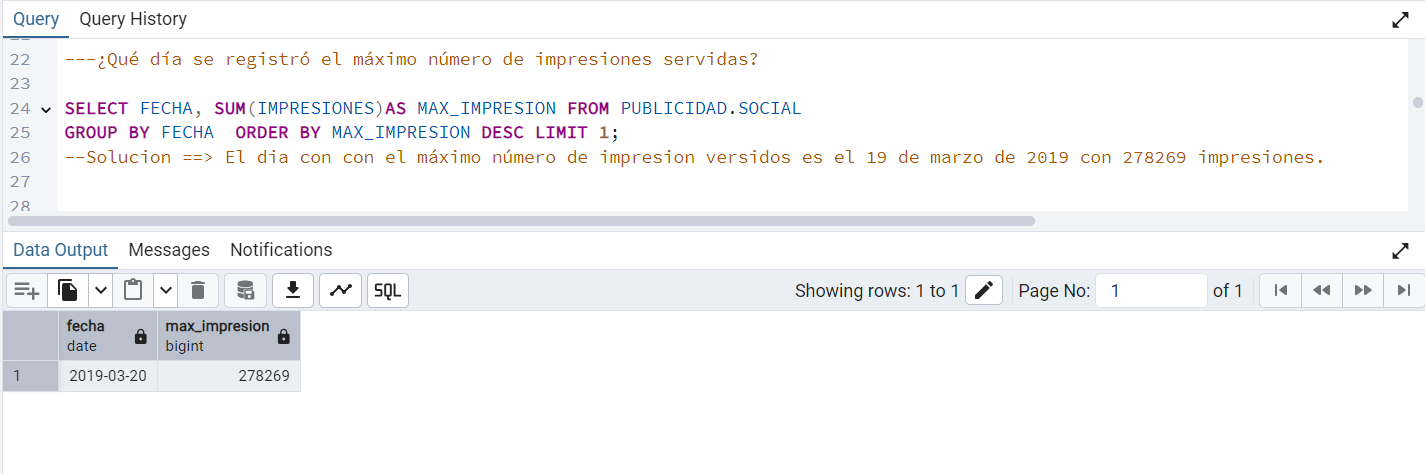
Aunque ampliar el sistema RDB actual podría ser una solución rápida, no es la más eficiente ni escalable a largo plazo. La combinación de bases de datos NoSQL, procesamiento distribuido y herramientas de streaming permitiría una ingesta, almacenamiento y análisis de datos más eficiente, asegurando que el periódico pueda obtener insights en tiempo real y adaptarse al crecimiento de datos en el futuro.  
  
  
  
**Aportacion Proyecto Final**

---¿Qué día se registró el máximo número de impresiones servidas?

SELECT FECHA, SUM(IMPRESIONES)AS MAX\_IMPRESION FROM PUBLICIDAD.SOCIAL

GROUP BY FECHA ORDER BY MAX\_IMPRESION DESC LIMIT 1;

--Solucion ==> El dia con con el máximo número de impresion versidos es el 19 de marzo de 2019 con 278269 impresiones.



---¿Cuál es el coste por GRP de cada una de las emisoras de radio en la que se emiten anuncios? Ordenar de mayor a menor.

SELECT EMISORA, SUM(INVERSION) AS INVERSION\_EMISORA

FROM PUBLICIDAD.RADIO

GROUP BY EMISORA

ORDER BY INVERSION\_EMISORA DESC;

--Solucion ==> --Cadena ser = 48689.41

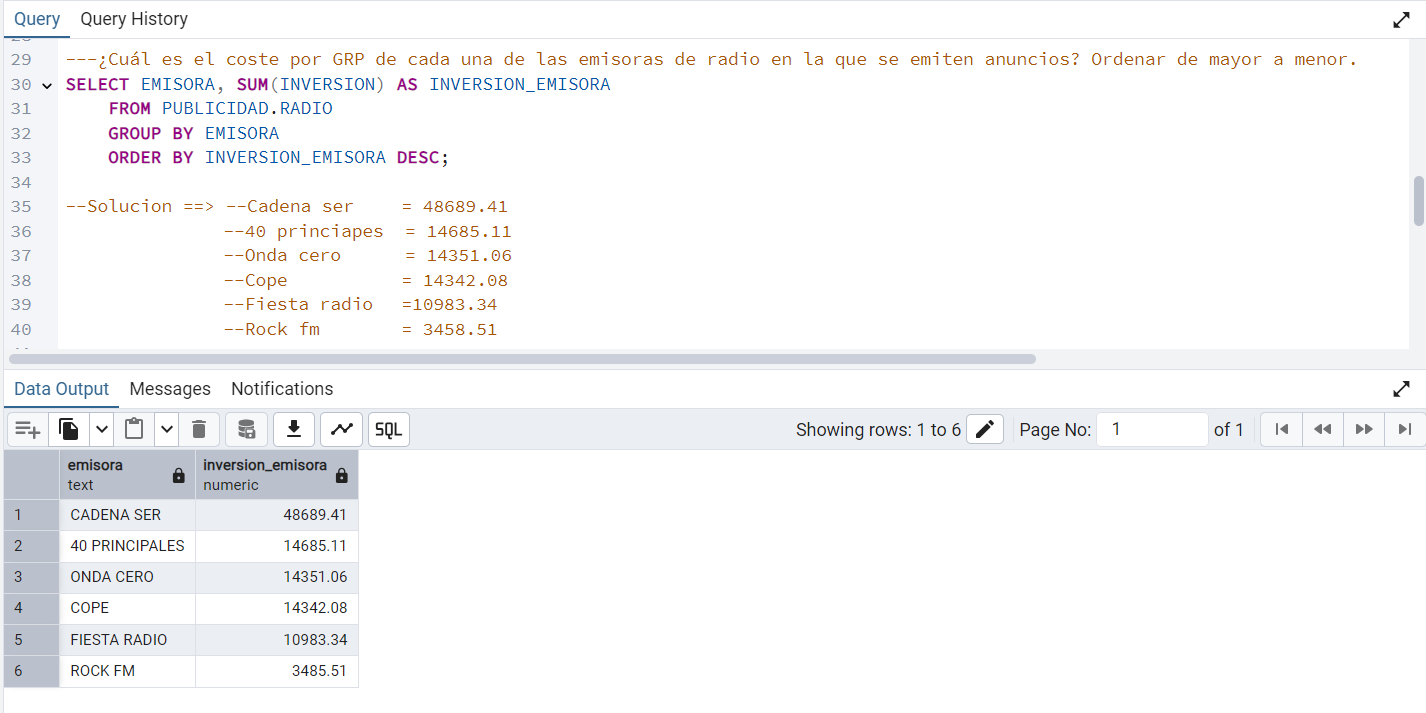
--40 princiapes = 14685.11

--Onda cero = 14351.06

--Cope = 14342.08

--Fiesta radio =10983.34

--Rock fm = 3458.51



--¿Qué día se registró el máximo presupuesto gastado en publicidad?

SELECT

kpi.Fecha as Fecha,

COALESCE( kpi.Visitas::NUMERIC, 0) as Visitas\_web,

COALESCE(kpi.Unidades\_vendidas::NUMERIC,0) as Unidades\_vendidas,

COALESCE(kpi.Importe::NUMERIC,0) as Importe,

(COALESCE(tv.Inversion\_tv::NUMERIC,0) + COALESCE(radio.Inversion\_radio::NUMERIC,0) + COALESCE(social.Inversion\_social::NUMERIC,0) ) as Interversion\_total,

COALESCE(tv.Inversion\_tv::NUMERIC,0) as Inversion\_TV,

COALESCE(radio.Inversion\_radio::NUMERIC,0) as Inversion\_radio,

COALESCE(social.Inversion\_social::NUMERIC,0) as Inversion\_social

FROM

((SELECT DATE("kpi"."fecha\_hora\_minuto") as Fecha, SUM("kpi"."visitas") as Visitas, SUM("kpi"."unidades\_vendidas") as Unidades\_vendidas,SUM("kpi"."importe") as Importe

FROM "publicidad"."kpi"

GROUP BY Fecha

) kpi

LEFT JOIN (SELECT DATE("tv"."fecha\_hora\_minuto") as Fecha,SUM("tv"."inversion") as Inversion\_tv FROM "publicidad"."tv"

GROUP BY fecha) tv ON kpi.fecha = tv.fecha

LEFT JOIN (SELECT DATE("radio"."fecha\_hora\_minuto") as Fecha, SUM("radio"."inversion") as Inversion\_radio

FROM "publicidad"."radio"

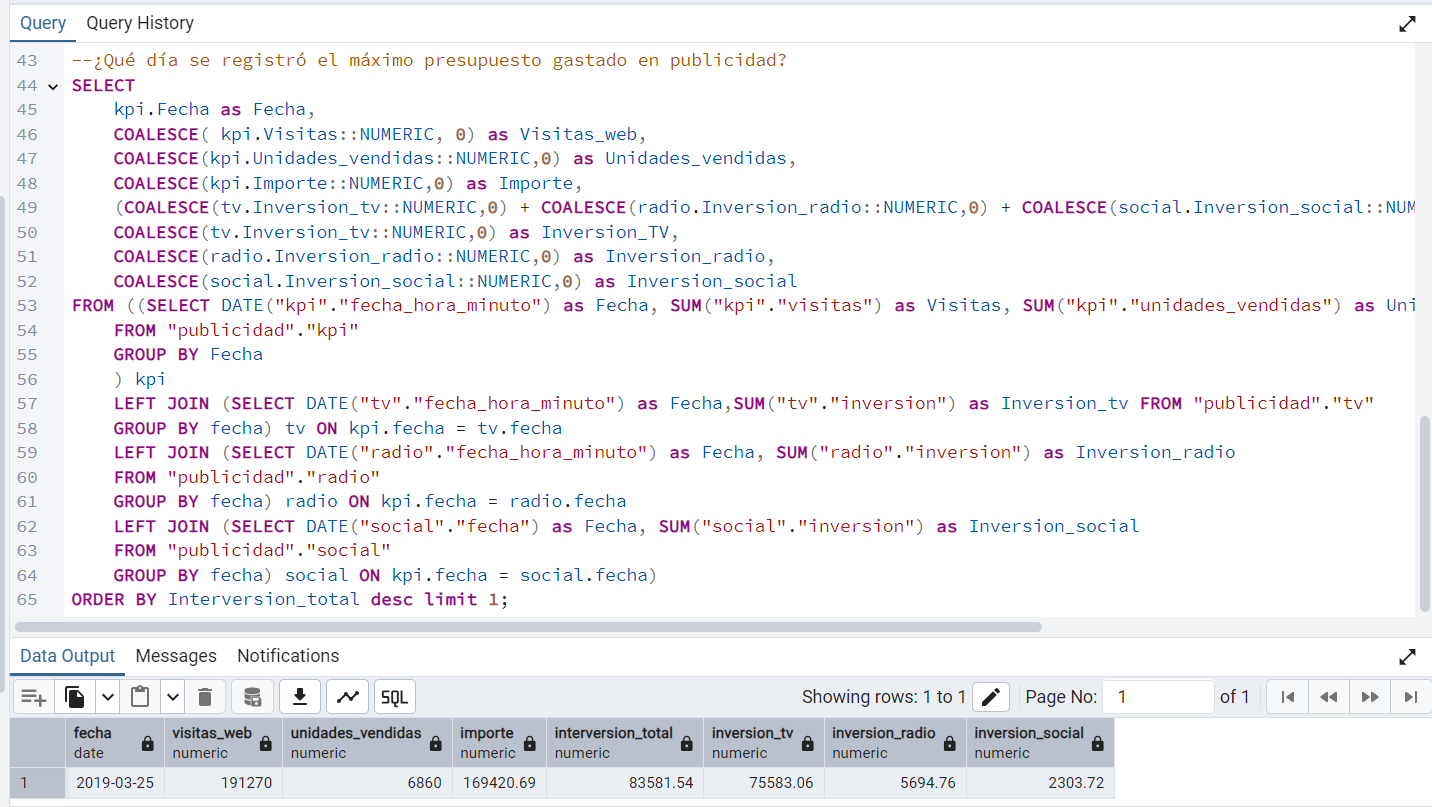
GROUP BY fecha) radio ON kpi.fecha = radio.fecha

LEFT JOIN (SELECT DATE("social"."fecha") as Fecha, SUM("social"."inversion") as Inversion\_social

FROM "publicidad"."social"

GROUP BY fecha) social ON kpi.fecha = social.fecha)

ORDER BY Interversion\_total desc limit 1;



**--Fase 3: Creación de una consulta que permita analizar el impacto de los anuncios de TV en los minutos posteriores a su emisión.**

**------- el impacto de los ultimos 30 minutos**

SELECT tv.fecha\_hora\_minuto AS fecha\_emision, kpi.fecha\_hora\_minuto AS fecha\_visita,

tv.campanya, tv.creatividad,tv.cadena,tv.franja, tv.grps\_ad16,sum(visitas) as total\_visitas\_posteriores

FROM publicidad.tv tv

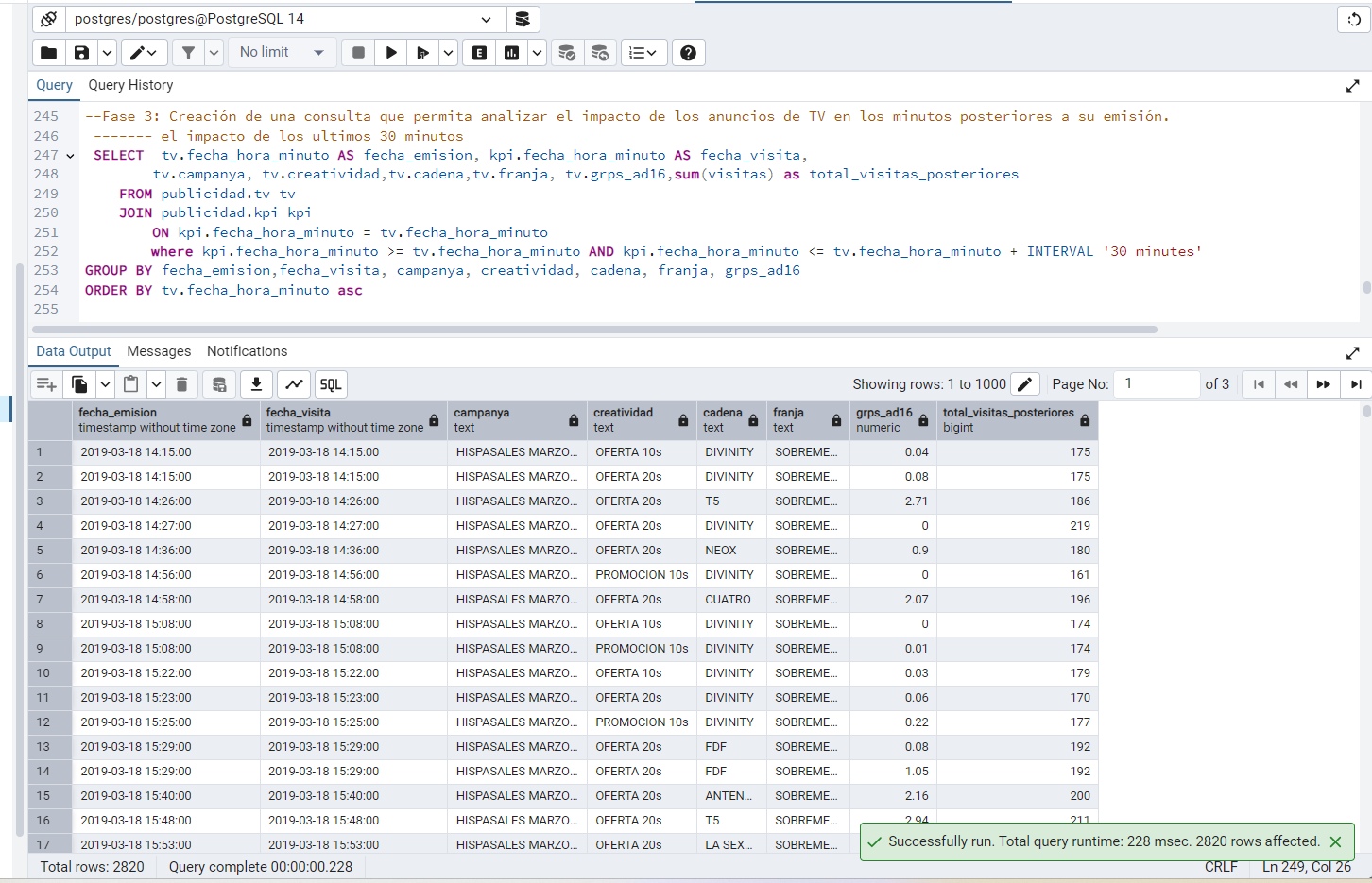
JOIN publicidad.kpi kpi

ON kpi.fecha\_hora\_minuto = tv.fecha\_hora\_minuto

where kpi.fecha\_hora\_minuto >= tv.fecha\_hora\_minuto AND kpi.fecha\_hora\_minuto <= tv.fecha\_hora\_minuto + INTERVAL '30 minutes'

GROUP BY fecha\_emision,fecha\_visita, campanya, creatividad, cadena, franja, grps\_ad16

ORDER BY tv.fecha\_hora\_minuto asc

****

**--Fase 4: Creación de una tabla con el resultado de la consulta y obtener un gráfico que permita visualizar la evolución de las visitas y los GRPs de los anuncios emitidos en TV.**

DROP TABLE IF EXISTS publicidad.impacto\_tv\_posterior\_emision;

CREATE TABLE publicidad.impacto\_tv\_posterior\_emision (

id SERIAL PRIMARY KEY,

fecha\_emision TIMESTAMP,

fecha\_visita TIMESTAMP,

campanya TEXT,

creatividad TEXT,

cadena TEXT,

franja TEXT,

grps\_ad16 NUMERIC,

total\_visitas\_posteriores INT

);

INSERT INTO publicidad.impacto\_tv\_posterior\_emision (fecha\_emision, fecha\_visita, campanya, creatividad, cadena, franja, grps\_ad16,total\_visitas\_posteriores) (

SELECT tv.fecha\_hora\_minuto AS fecha\_emision, kpi.fecha\_hora\_minuto AS fecha\_visita,

tv.campanya, tv.creatividad,tv.cadena,tv.franja, tv.grps\_ad16,sum(visitas) as total\_visitas\_posteriores

FROM publicidad.tv tv

JOIN publicidad.kpi kpi

ON kpi.fecha\_hora\_minuto = tv.fecha\_hora\_minuto

where kpi.fecha\_hora\_minuto >= tv.fecha\_hora\_minuto AND kpi.fecha\_hora\_minuto <= tv.fecha\_hora\_minuto + INTERVAL '30 minutes'

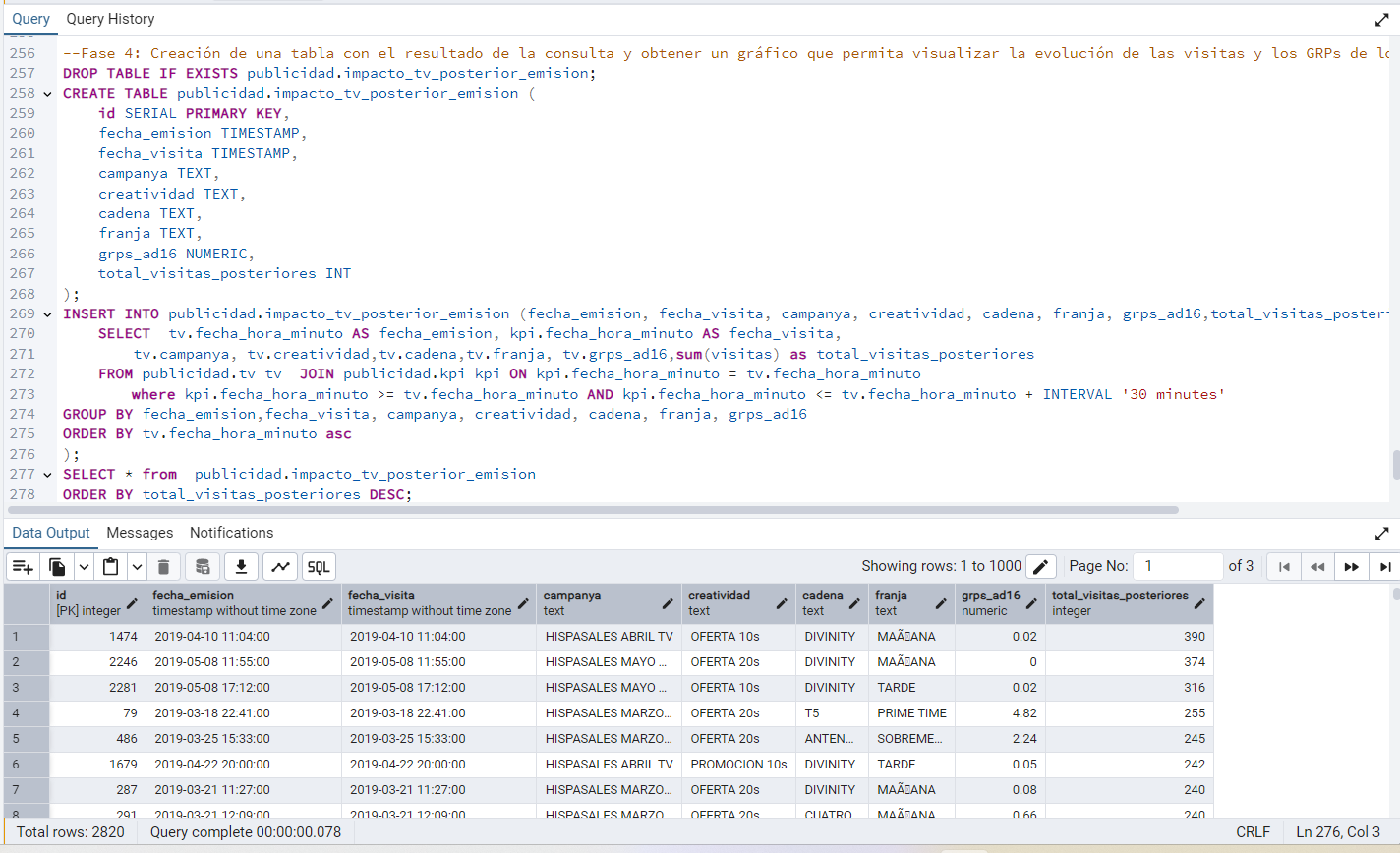
GROUP BY fecha\_emision,fecha\_visita, campanya, creatividad, cadena, franja, grps\_ad16

ORDER BY tv.fecha\_hora\_minuto asc

);

SELECT \* from publicidad.impacto\_tv\_posterior\_emision

ORDER BY total\_visitas\_posteriores DESC;

****

**--Fase adicional: Identificar qué anuncio de TV ha generado más visitas en la página web tras su emisión.**

**----------**

SELECT campanya, creatividad, cadena, franja, grps\_ad16, MAX(total\_visitas\_posteriores) AS max\_visitas

FROM publicidad.impacto\_tv\_posterior\_emision

GROUP BY campanya, creatividad, cadena, franja, grps\_ad16

ORDER BY max\_visitas DESC

LIMIT 1;

****