****

**UNIVERSIDAD TÉCNICA DE MACHALA**

Maestría en Software

**Asignatura:**

Ciencia de Datos e Inteligencia de Negocios

**Tema:**

## Tarea Práctica1. Diseño DW, proceso ETL y dashboard BI del caso Camaronera.

Docente: Ing. Bertha Mazón, Mg. Inf.

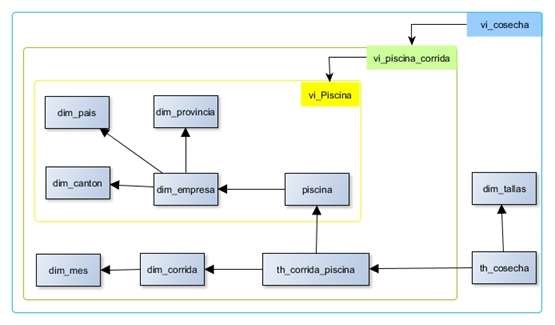
Estudiantes:

Ing. Jimmy Fernando Castillo Crespín

Ing. Paquita Alejandra Cuadros García

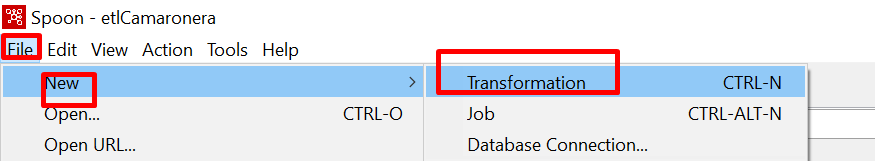
2021-2022

**Diseño del DataWarehouse**

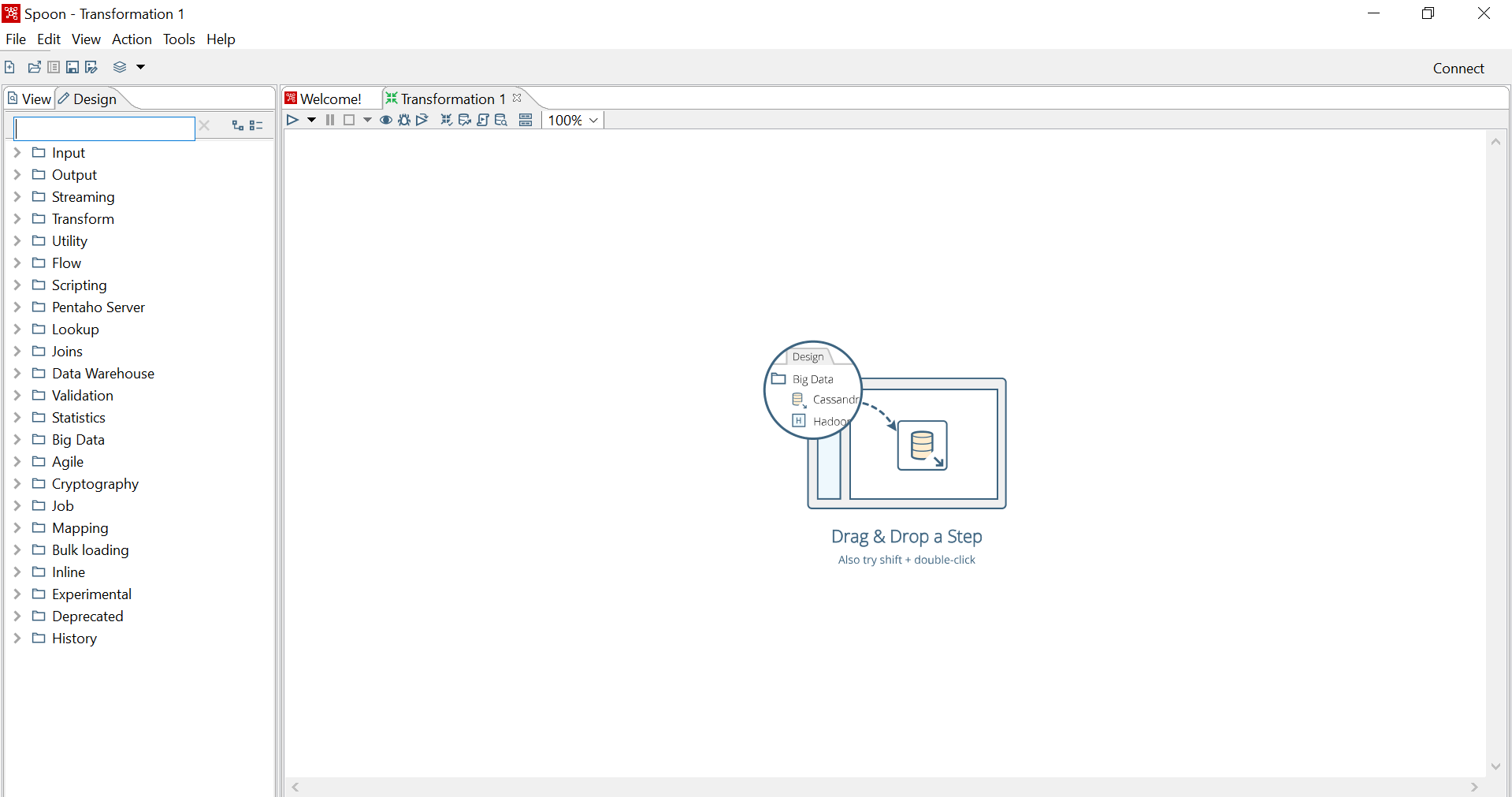


**Proceso ETL**

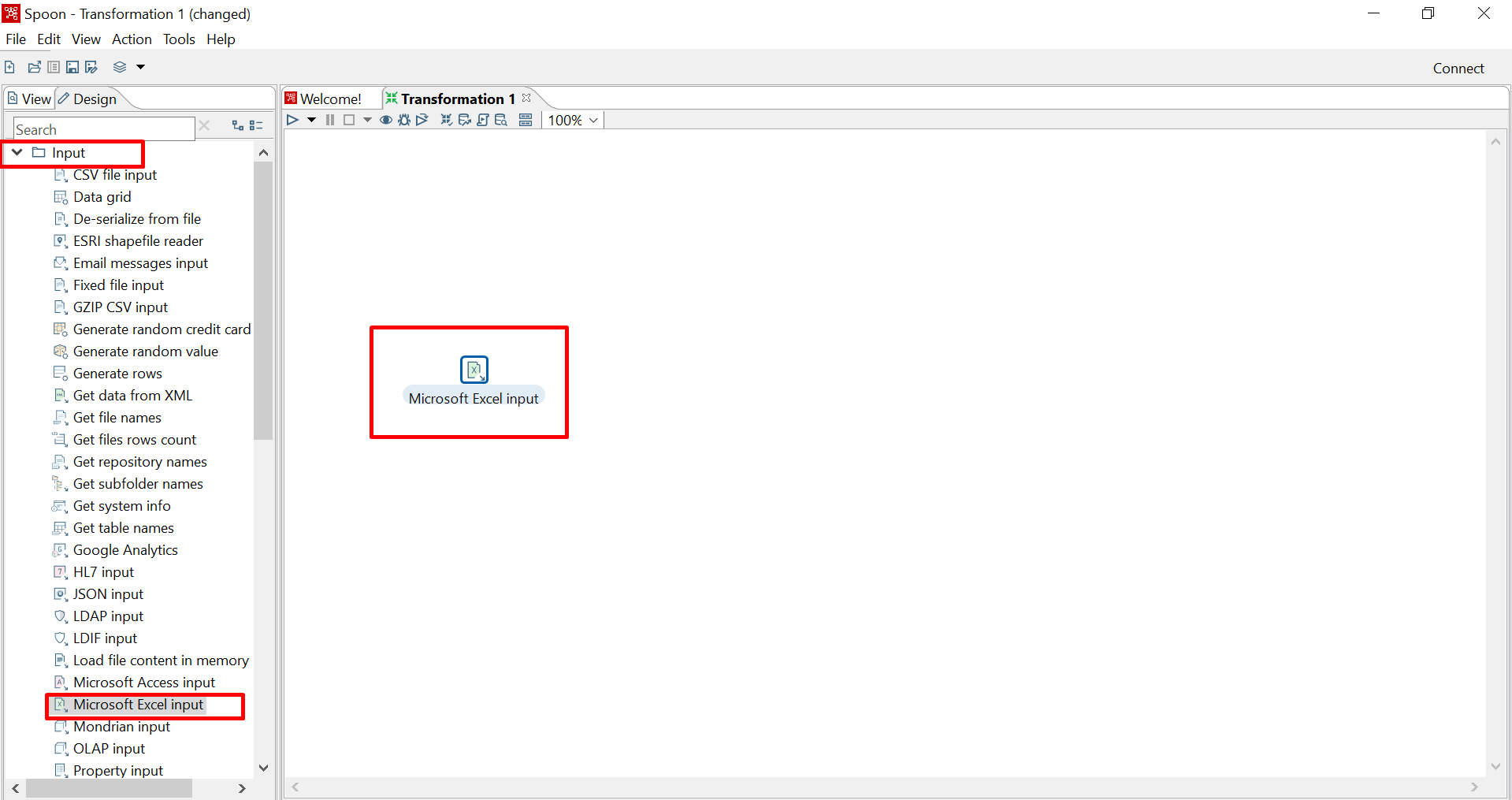
Primeramente hay que dirigirnos al menú File, luego a la opción New y elegir la opción de Transformation.



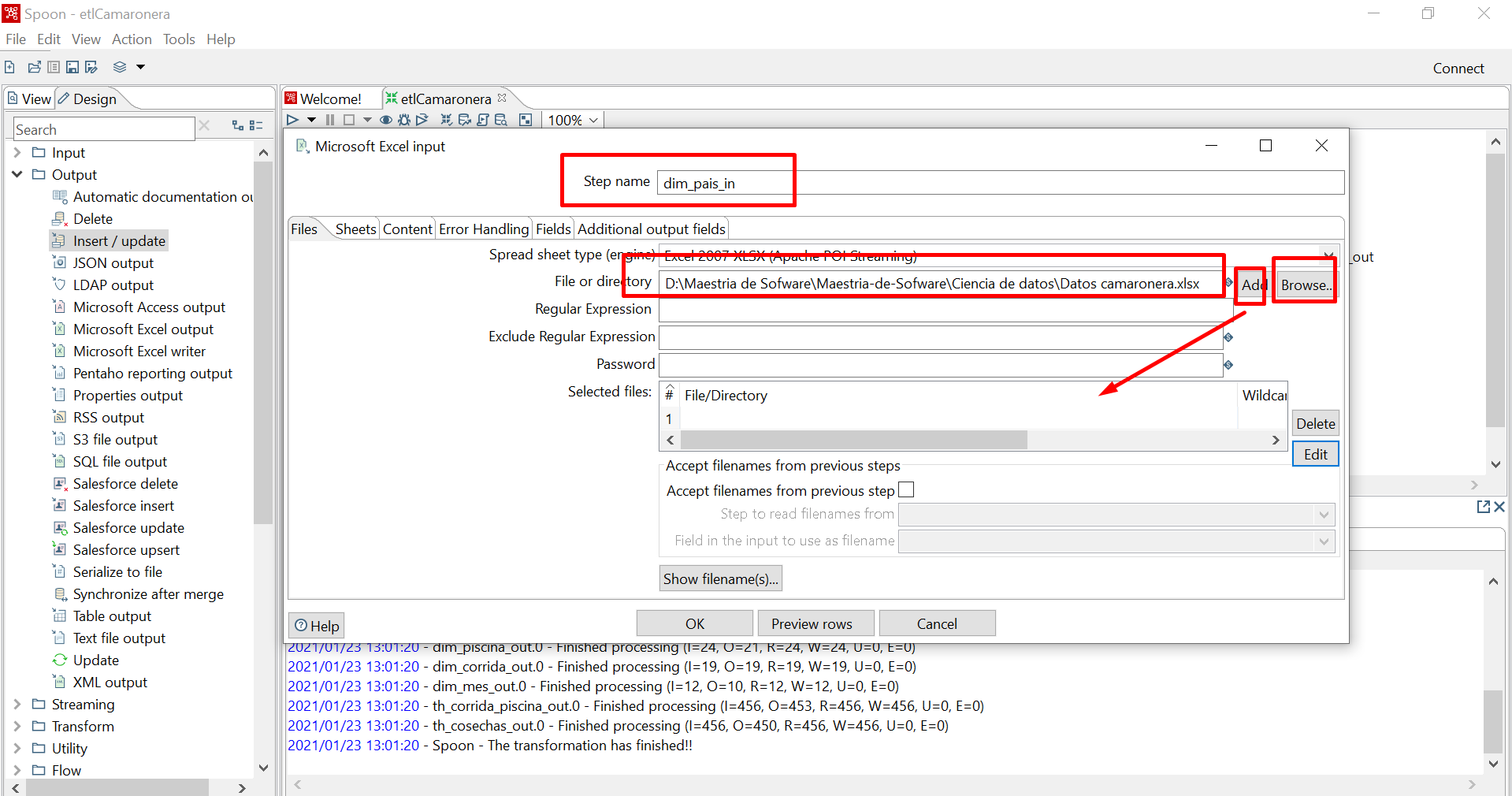
Esta pantalla nos permite realizar la Transformación de los datos que van a ser cargados desde un archivo externo.



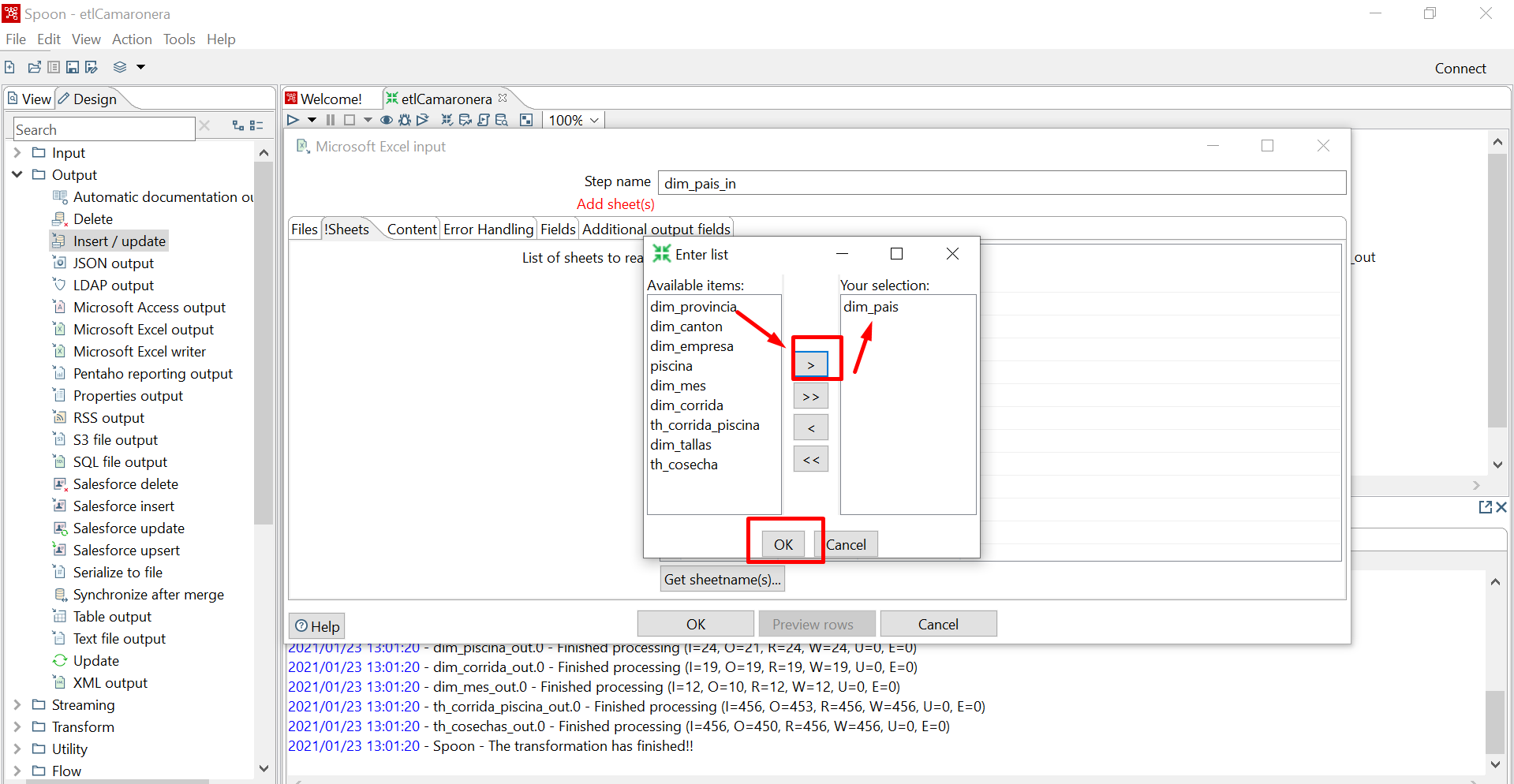
En esta pantalla realizamos la selección de la entrada de un archivo de Excel



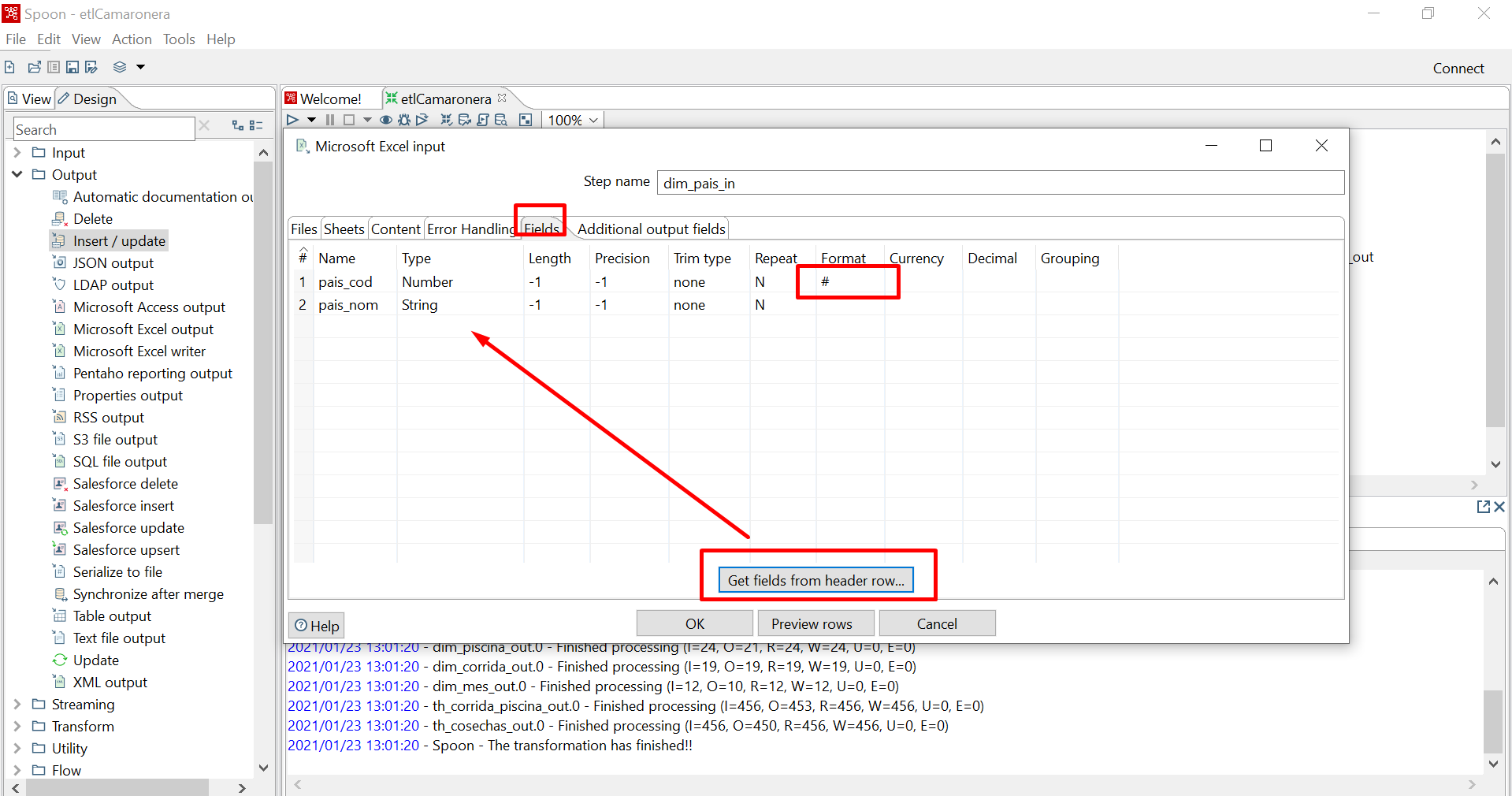
Esta ventana nos aparece haciendo doble clic en el icono “Microsoft Excel Input”, donde vamos a indicar el tipo de fuente externa a seleccionar, y seleccionamos la ruta desde donde va a ser cargados los datos externos.



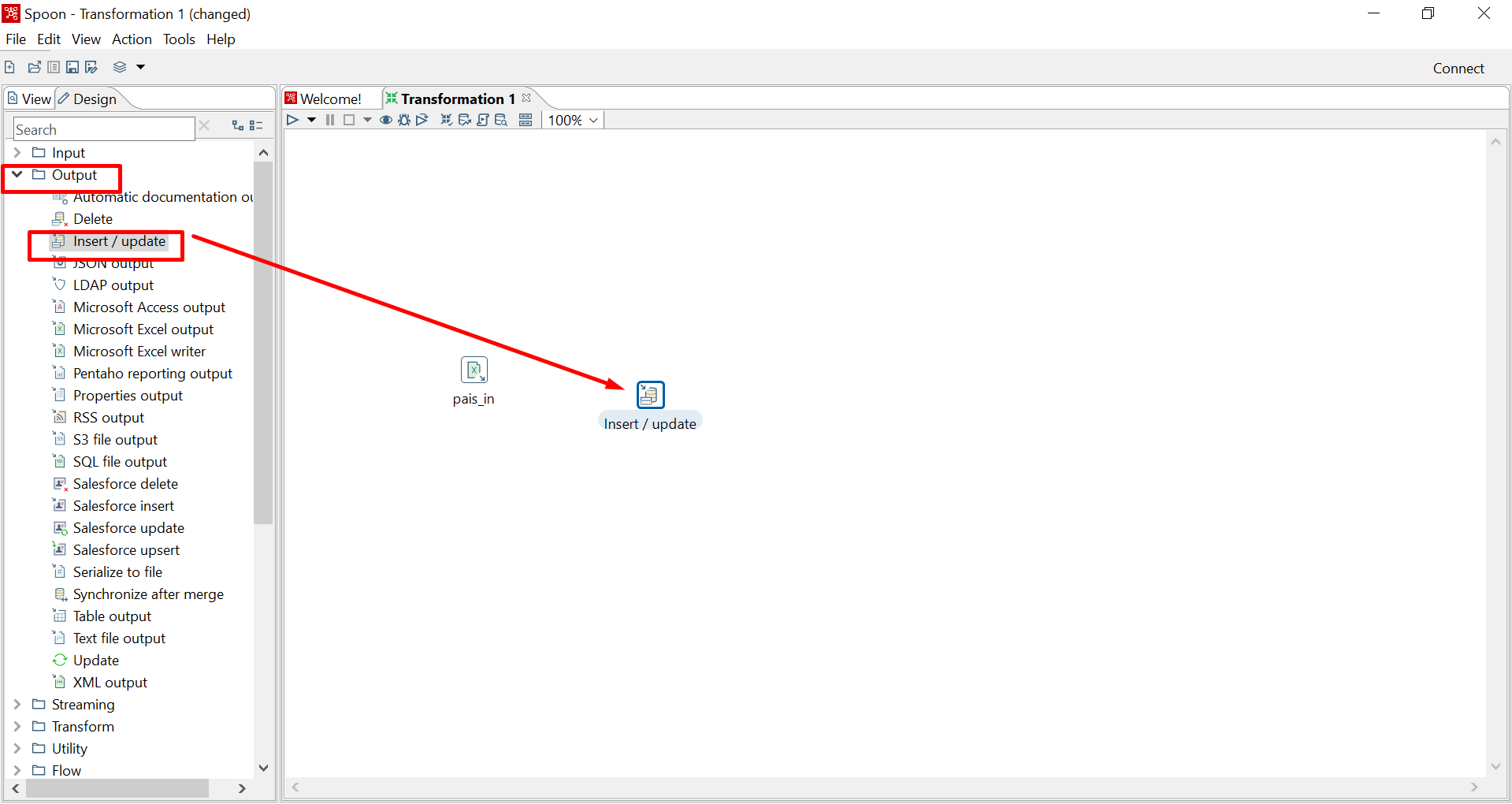
En esta ventana seleccionamos la pestaña Sheets y obtenemos el nombre de hojas que serán agregadas desde la fuente externa, es decir desde el archivo de excel, mismas que debemos agregar una por una en este caso agregamos la tabla dim\_pais.



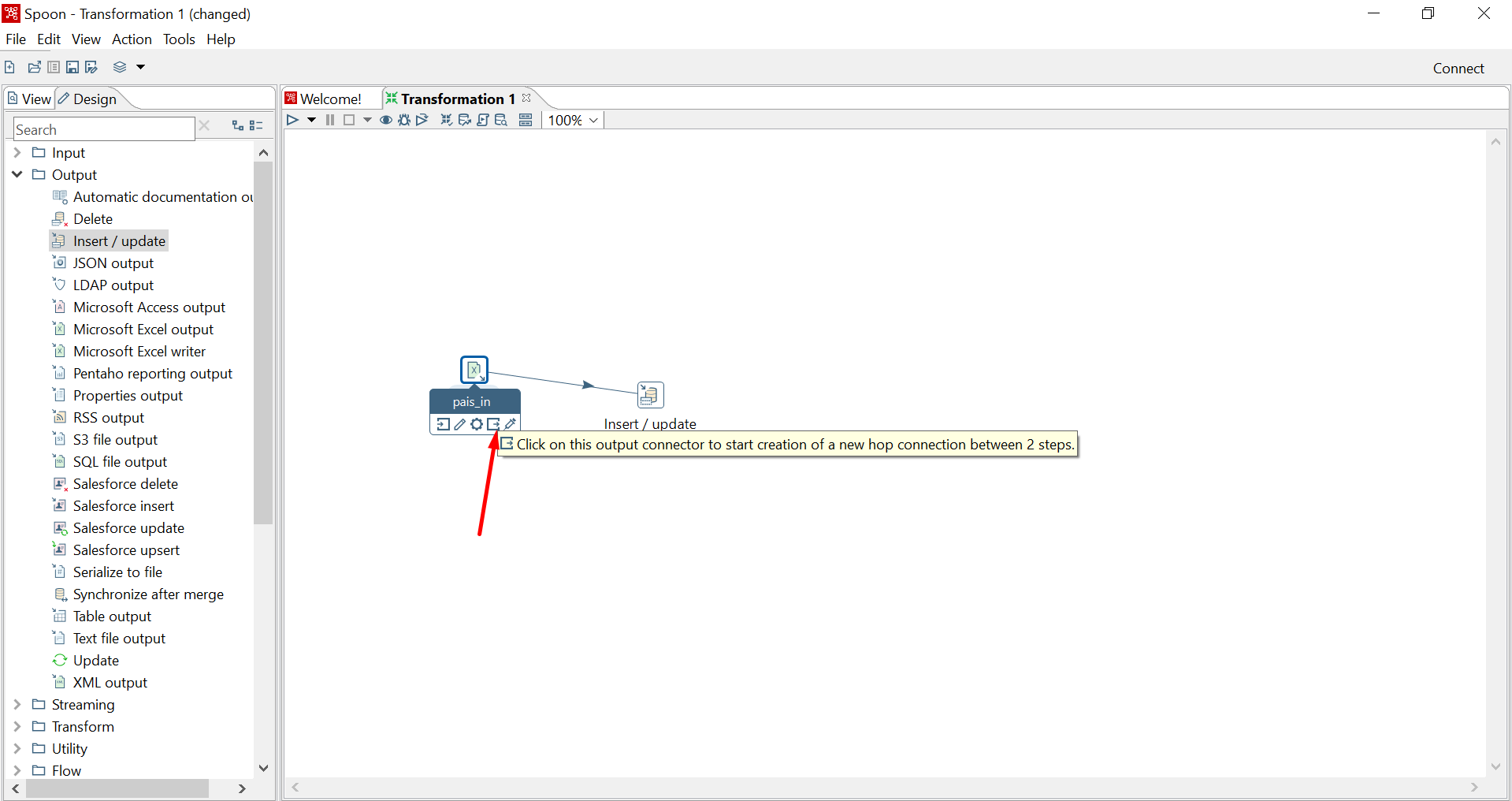
Seleccionamos la pestaña Fields en la cual vamos a cargar los nombres de encabezados de fila desde la fuente externa para eso damos click en Get fields from header row y luego vamos a dar formato a los campos cargados.



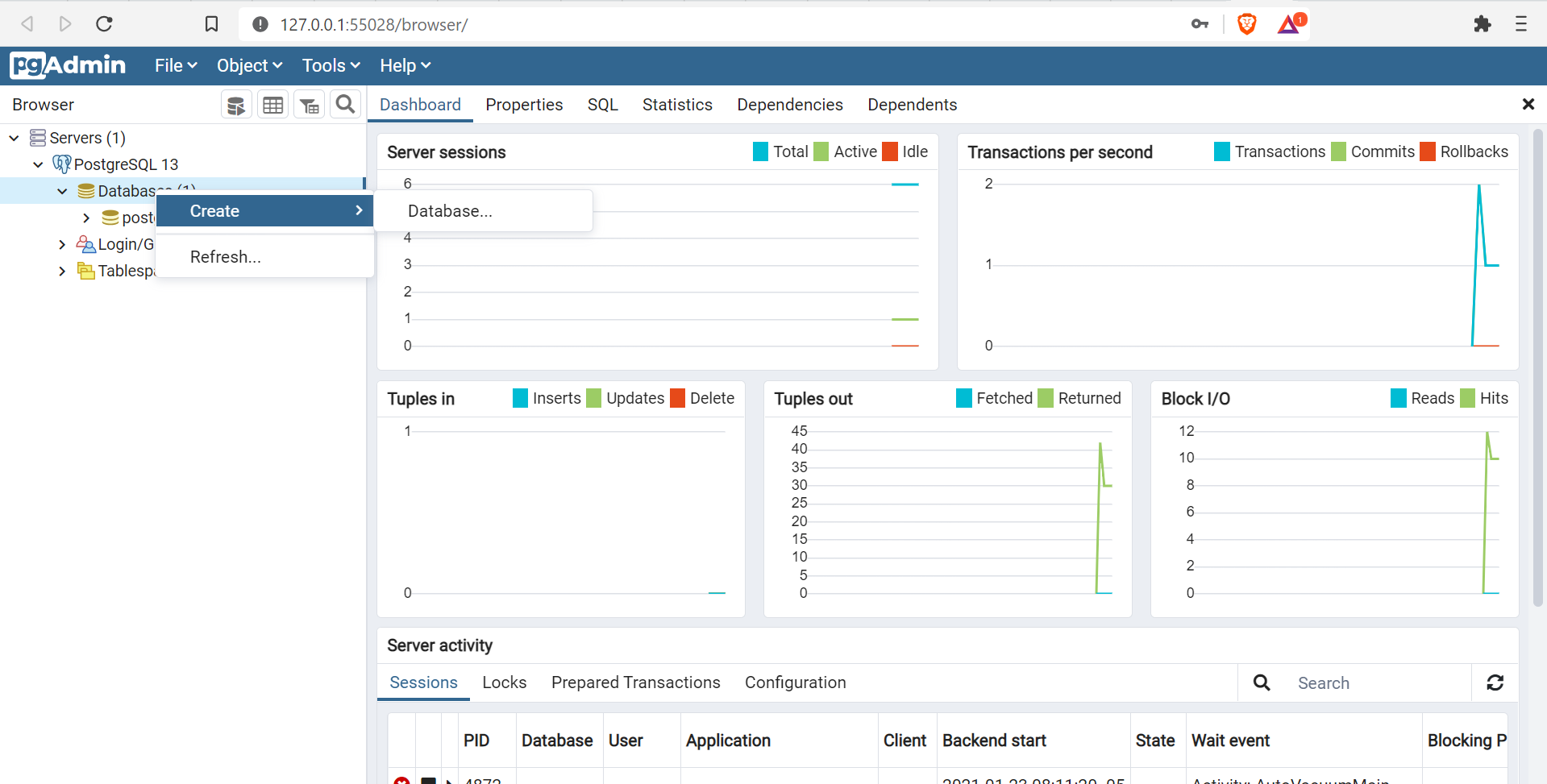
En esta parte seleccionamos las salidas (Output) de las cuales tenemos dos posibilidades de la cual arrastramos la opción Insert/Update y antes de cualquier configuración enlazamos la entrada con la salida.



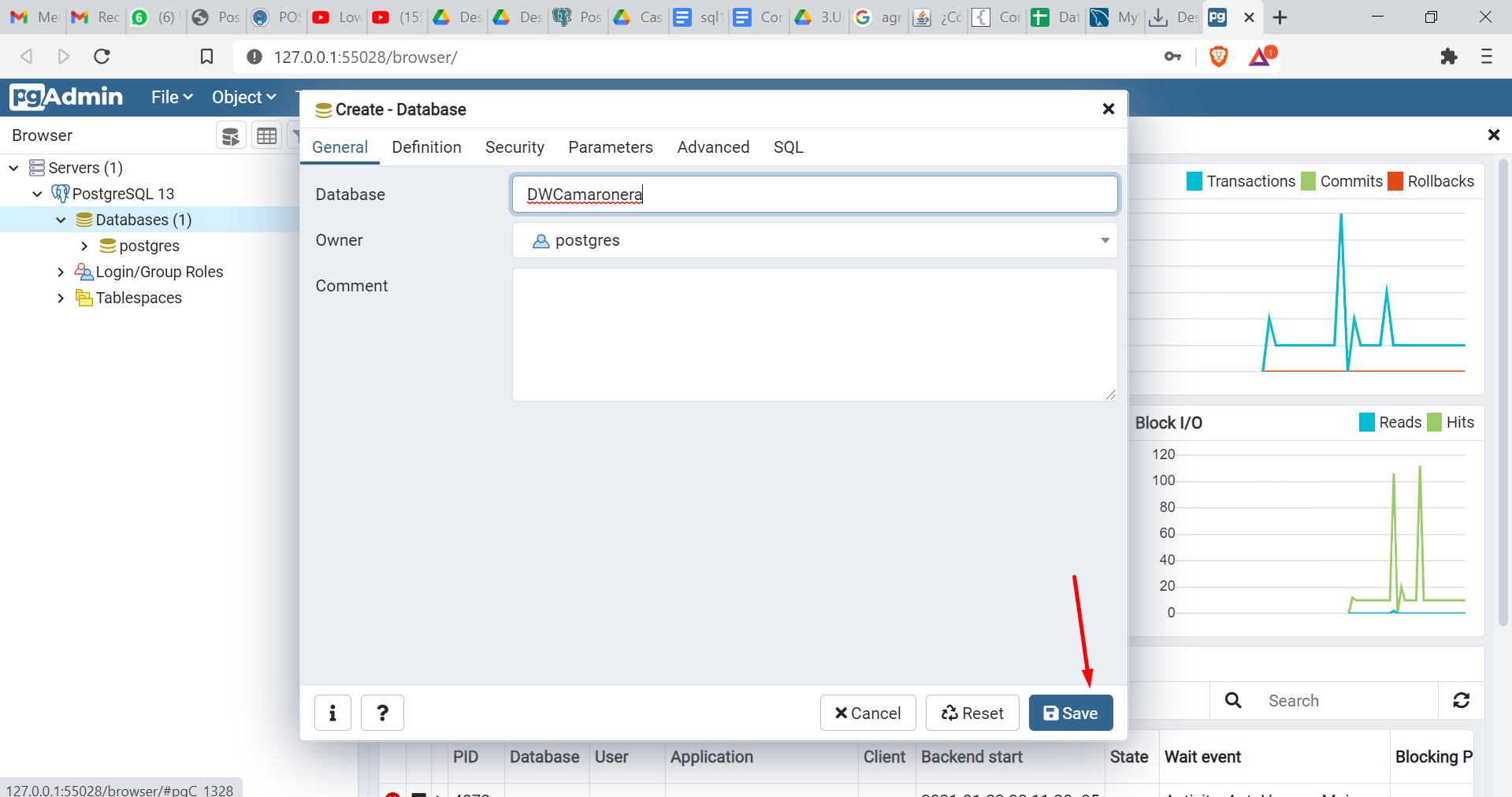
Esta pantalla nos muestra el proceso de enlazar la entrada con la salida misma que se puede observar gráficamente a través de una línea de conexión entre ambas.



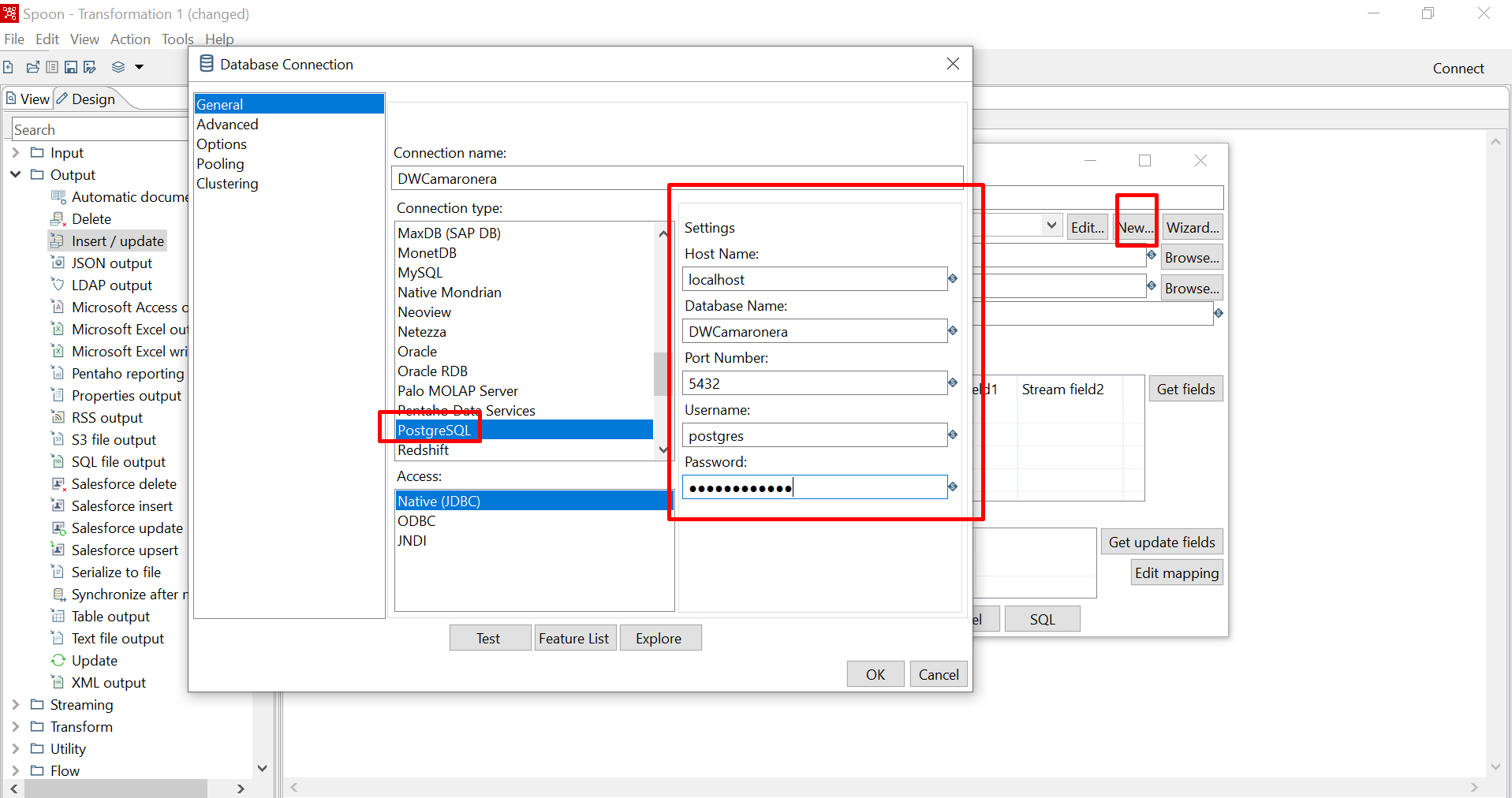
Esta pantalla nos muestra la ventana principal de Postgresql donde procedemos a crear nuestra base de datos.



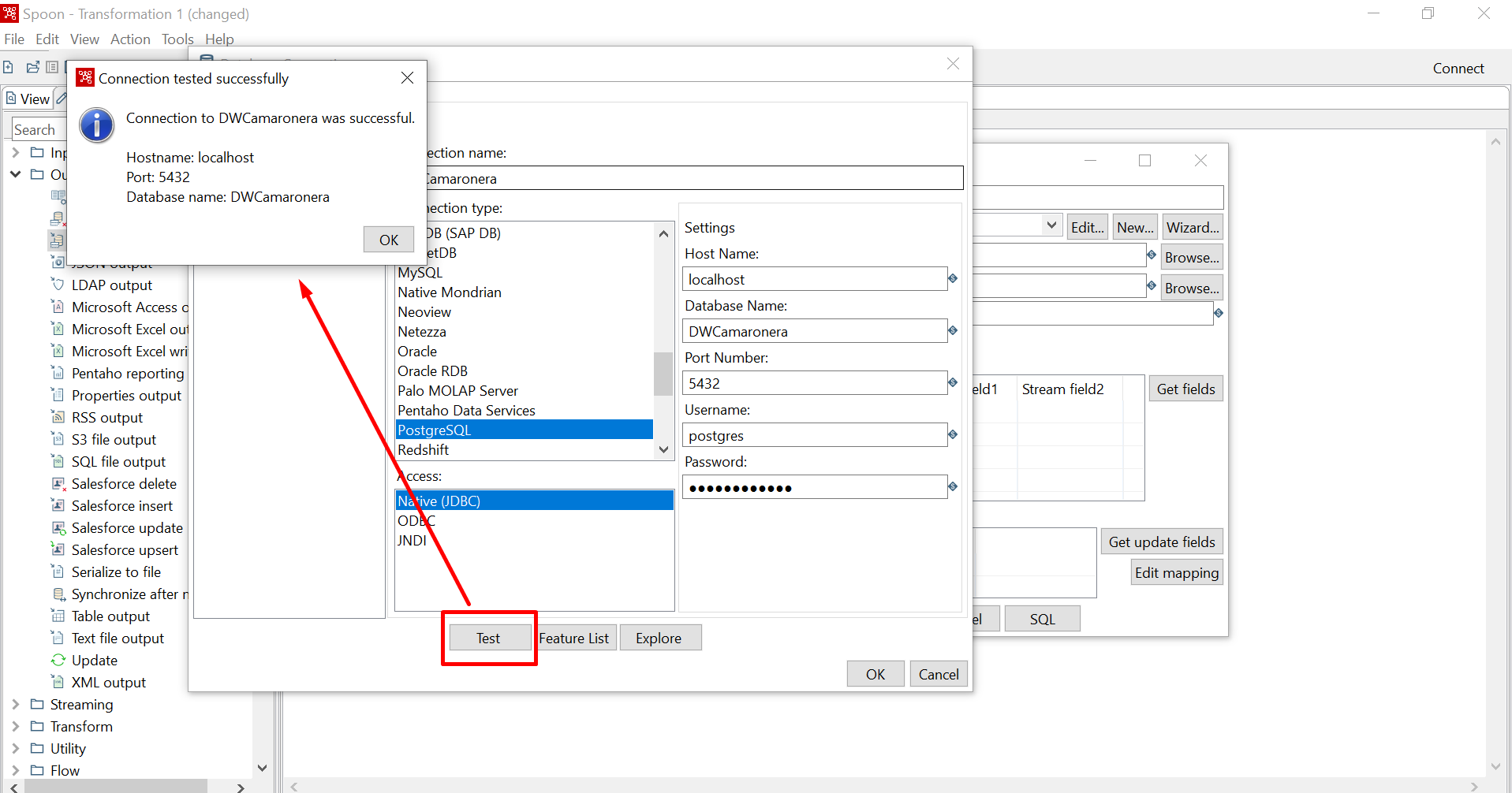
En esta ventana escribimos el nombre de la base de datos y procedemos a guardar la misma haciendo clic en el botón SAVE.



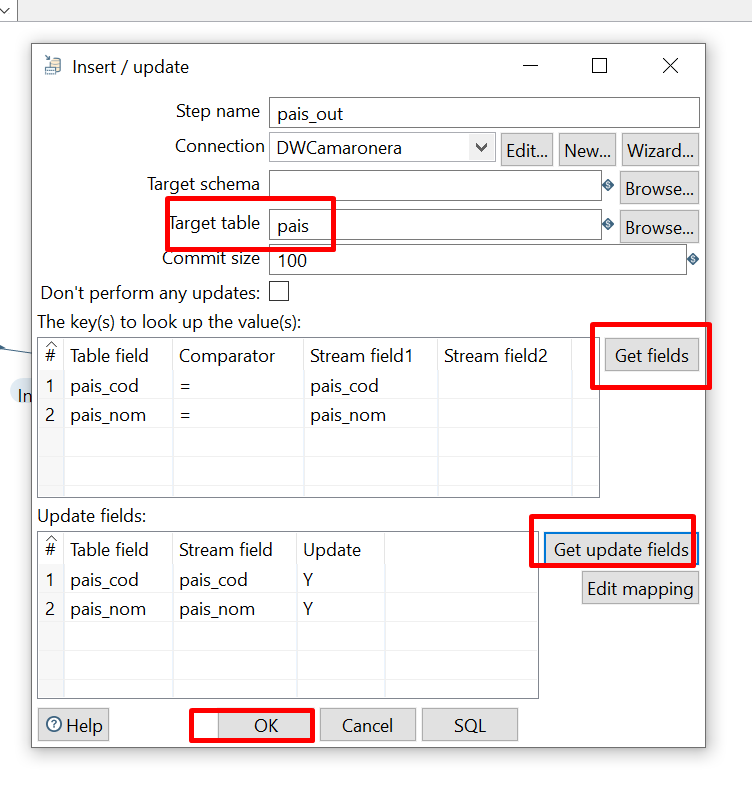
En esta ventana realizamos la conexión con la base de datos postgresql escribiendo los parámetros de configuración de la misma. Para comprobar que esta sea correcta hacemos clic en el botón TEST.



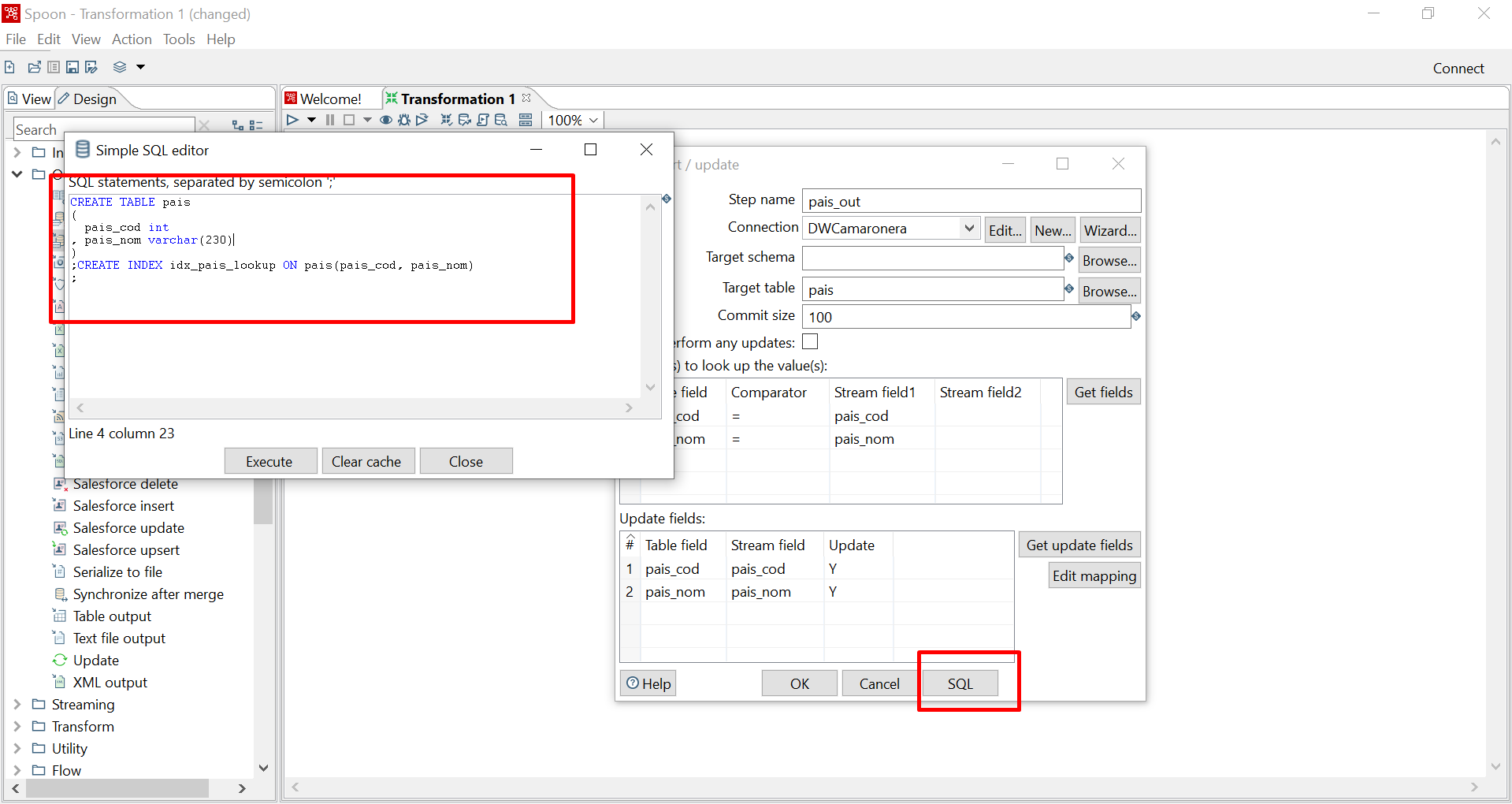
En esta ventana se observa que la conexión con la base de datos ha sido exitosa.



Procedemos a crear un schema propio agregando las tablas una por una, en este paso crearemos la tabla “pais”.



Aquí definimos cómo va a crearse la estructura de la tabla en la base de datos a través del botón SQL y agregando las sentencias correspondientes para lo cual pegamos el siguiente codigo.



el código sql generado es el siguiente:

CREATE TABLE pais

(

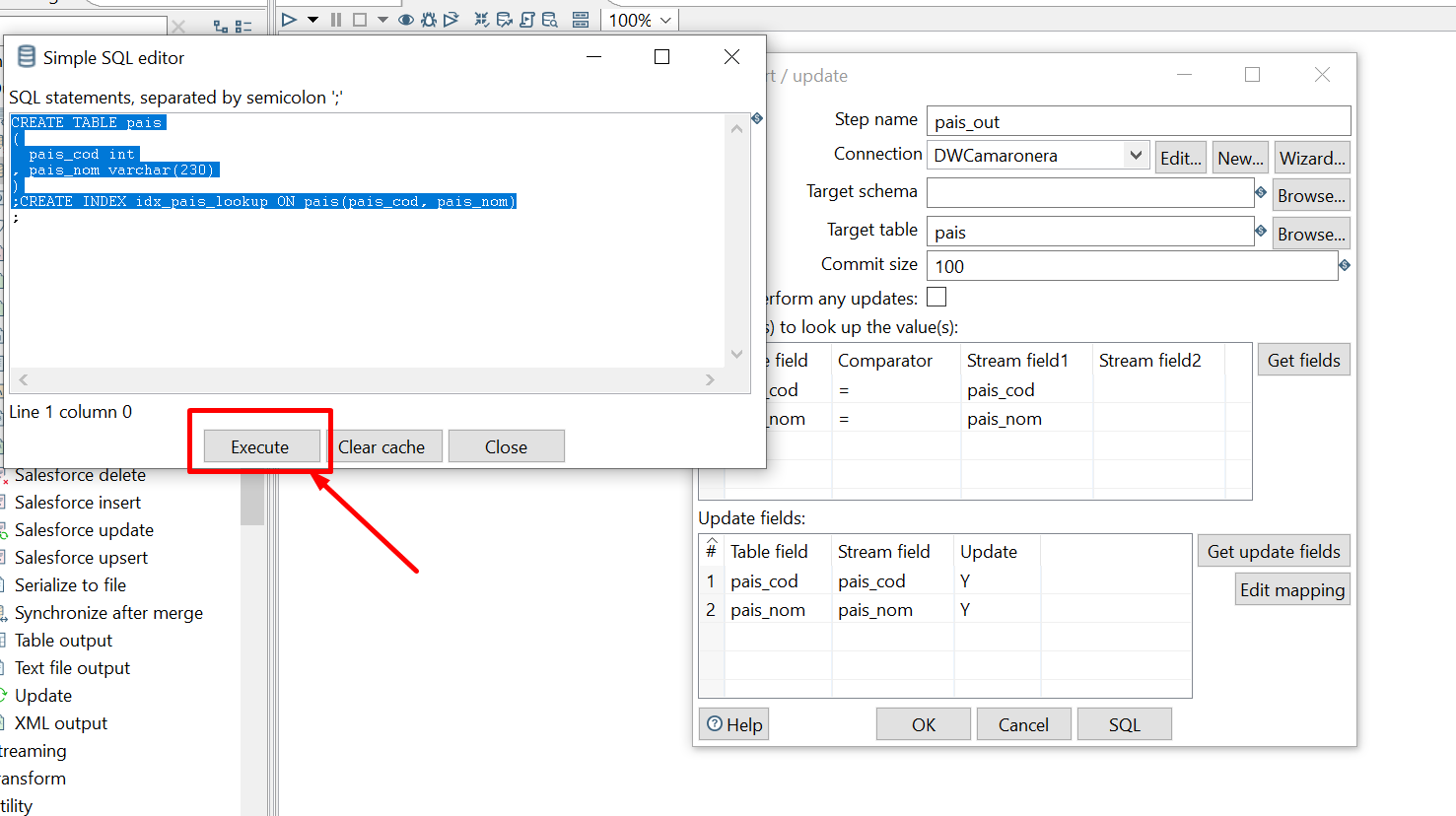
pais\_cod int

, pais\_nom varchar(230)

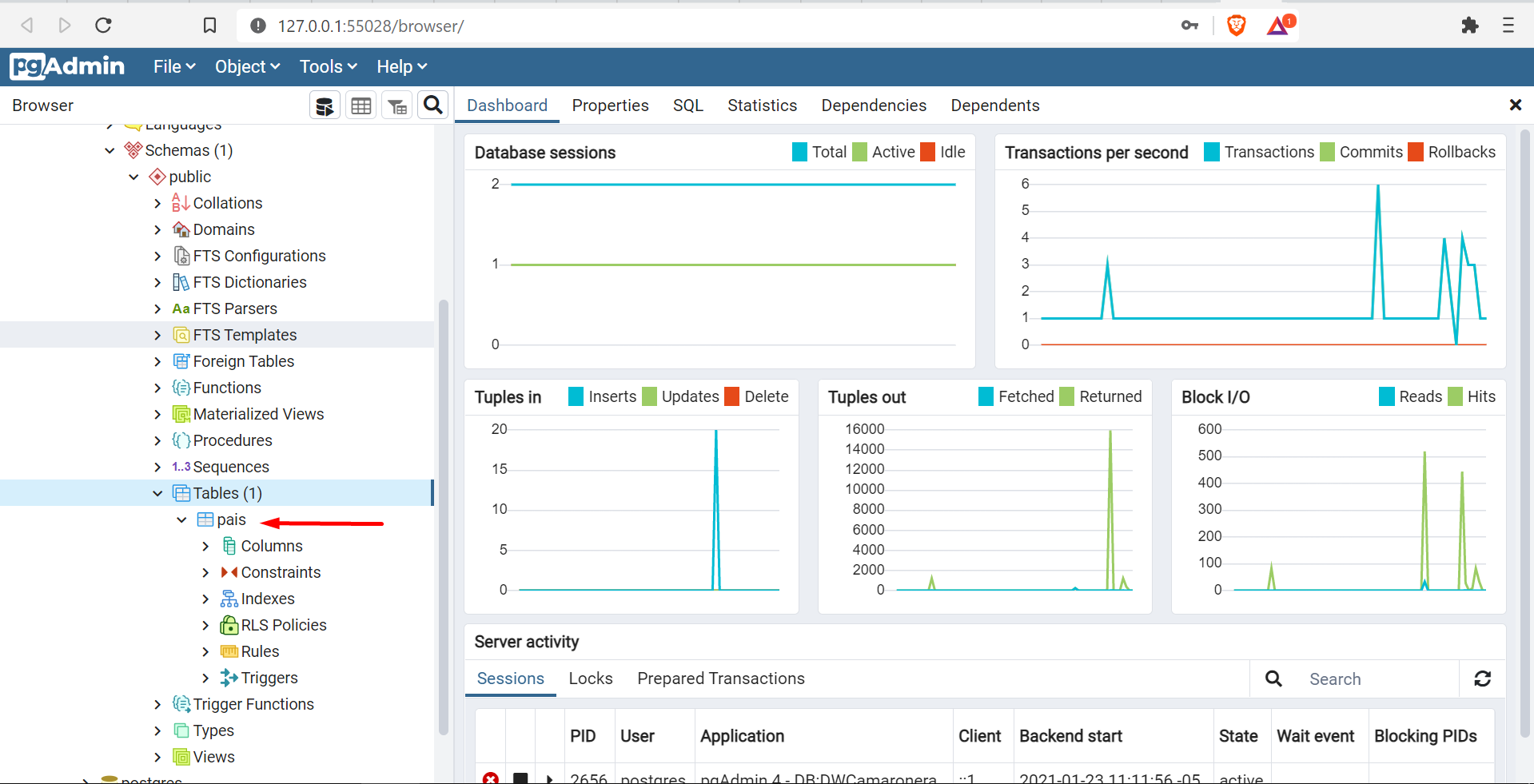
)

;CREATE INDEX idx\_pais\_lookup ON pais(pais\_cod, pais\_nom)

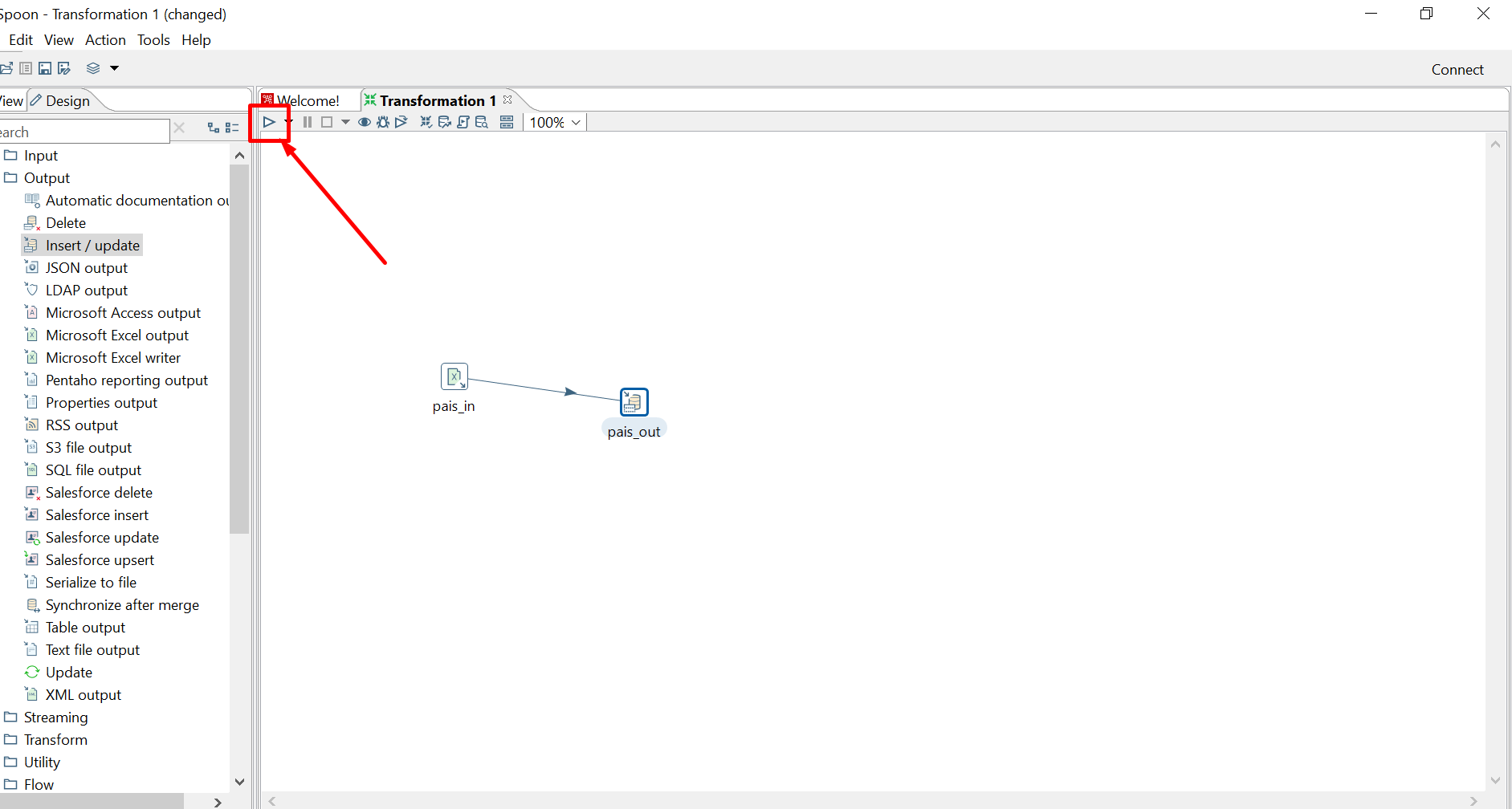
Una vez copiado el código SQL hacemos clic en el botón EXECUTE para crear la estructura de la tabla.



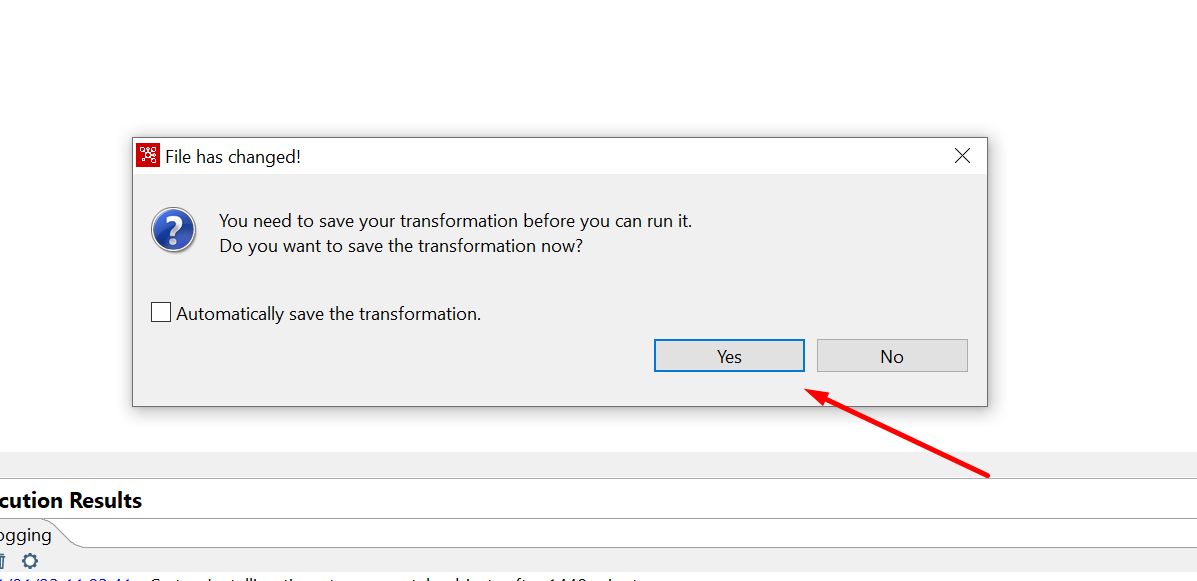
En esta pantalla de postgreslq nos muestra la tabla previamente creada la misma que se encuentra vacía.



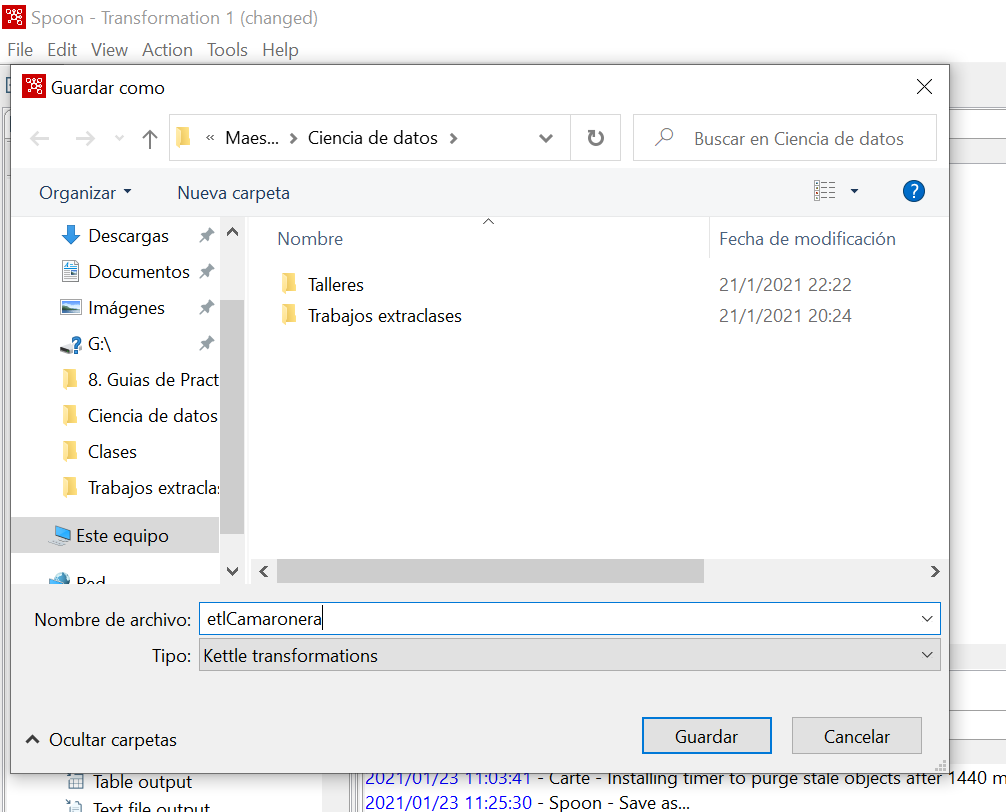
Aquí se comprueba los datos haciendo clic en el botón RUN.



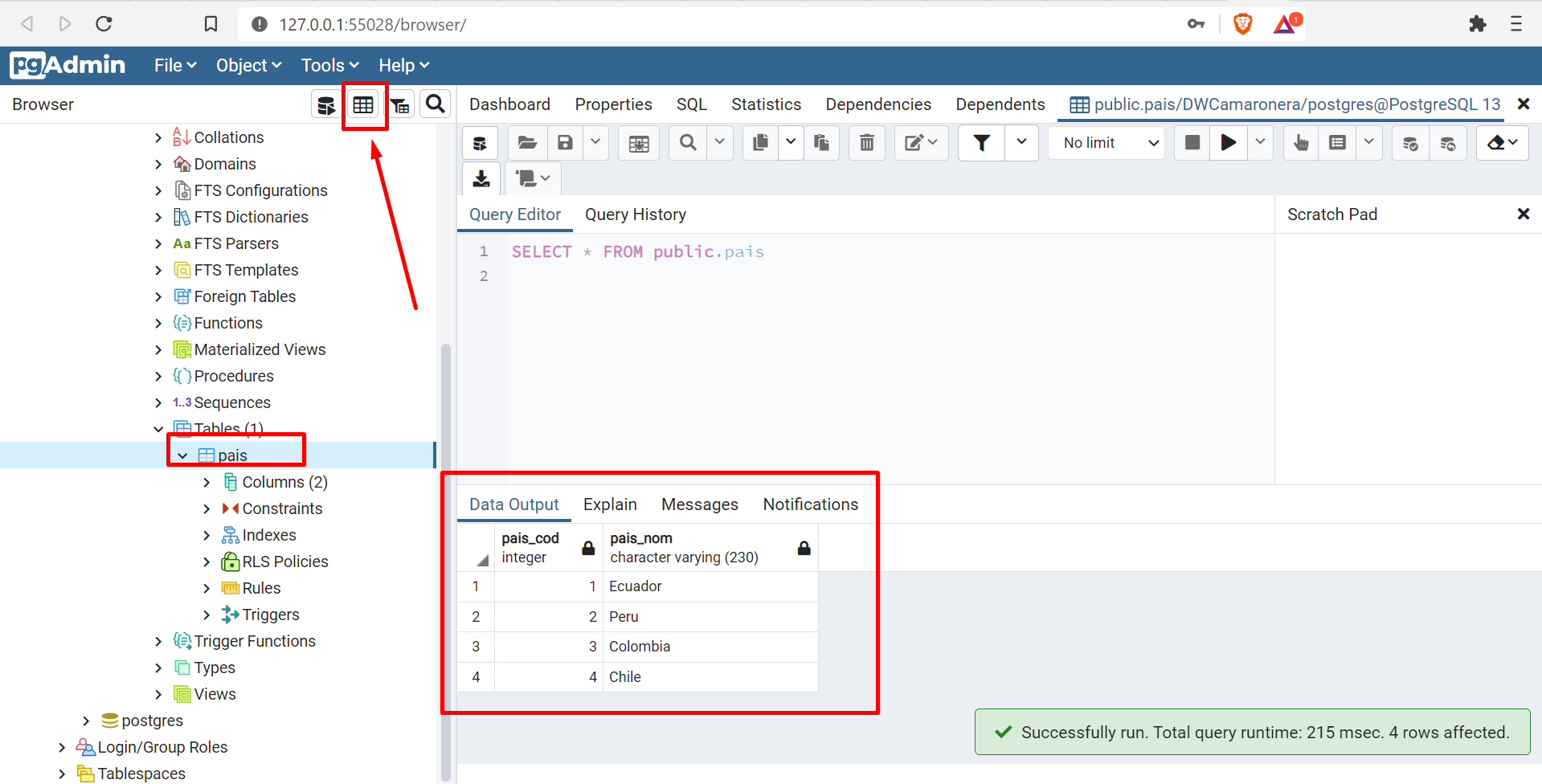
Este mensaje solicita guardar la transformación previamente a la carga de los datos.



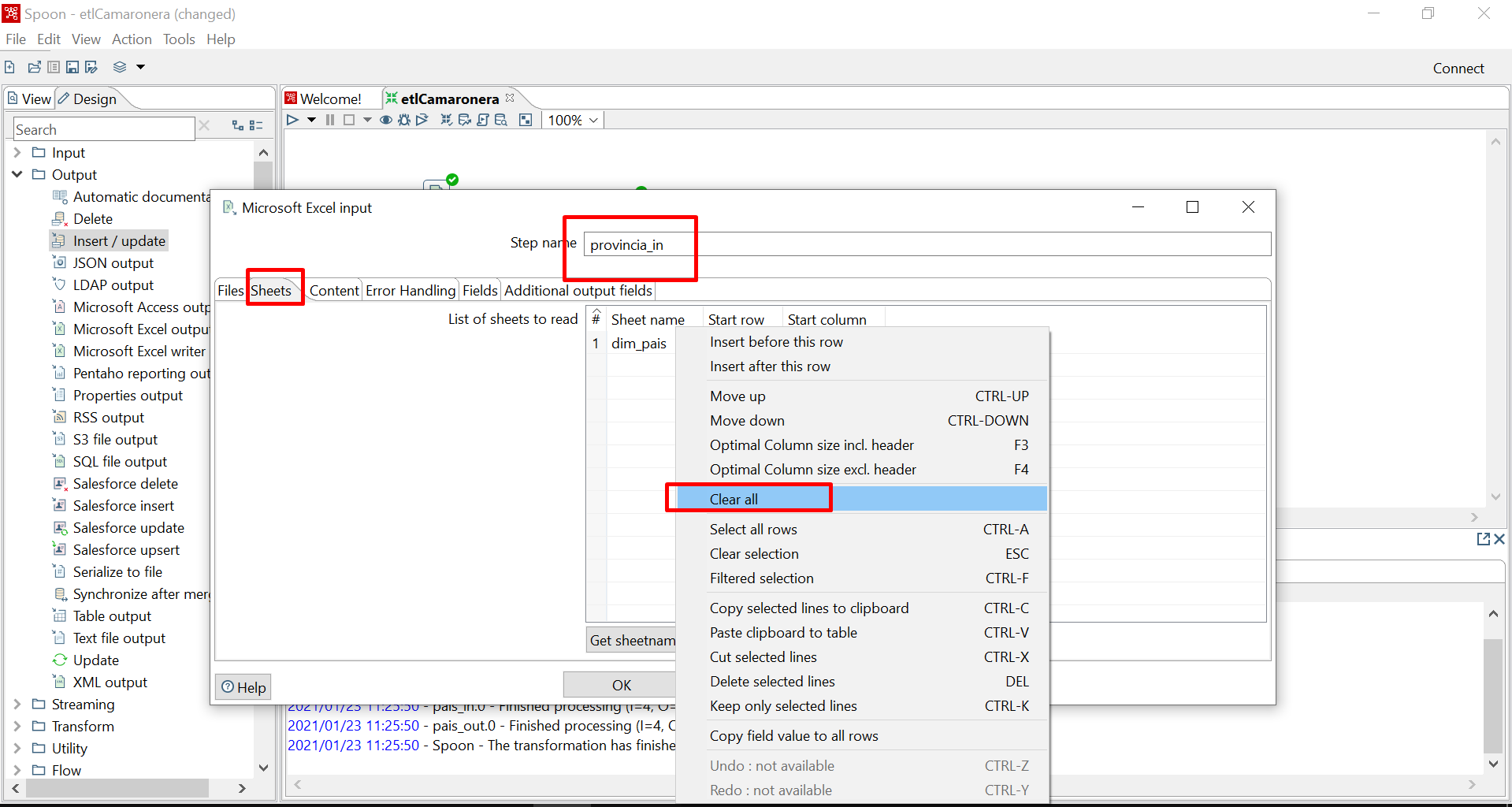
En este cuadro escribimos el nombre del archivo de la transformación.



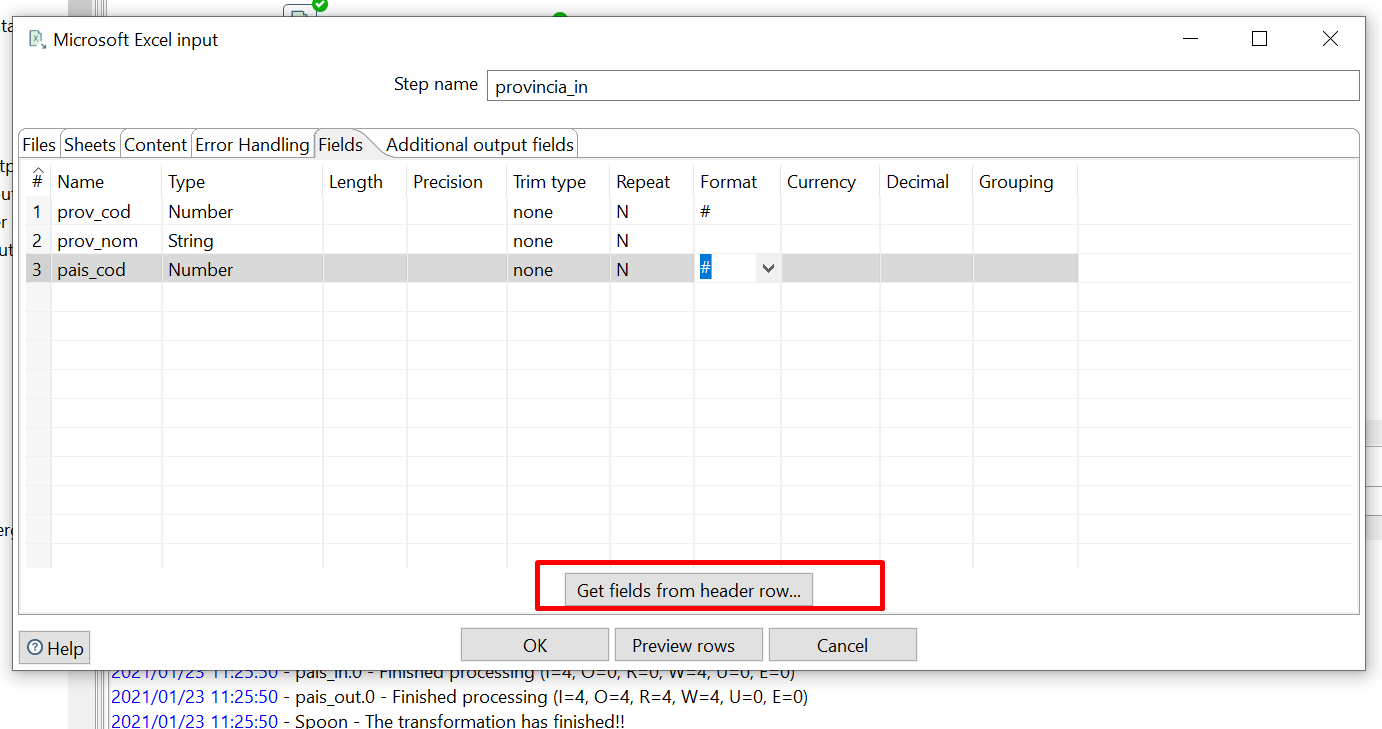
Esta pantalla nos muestra los datos que ya se encuentran cargados en la tabla de la base de datos.



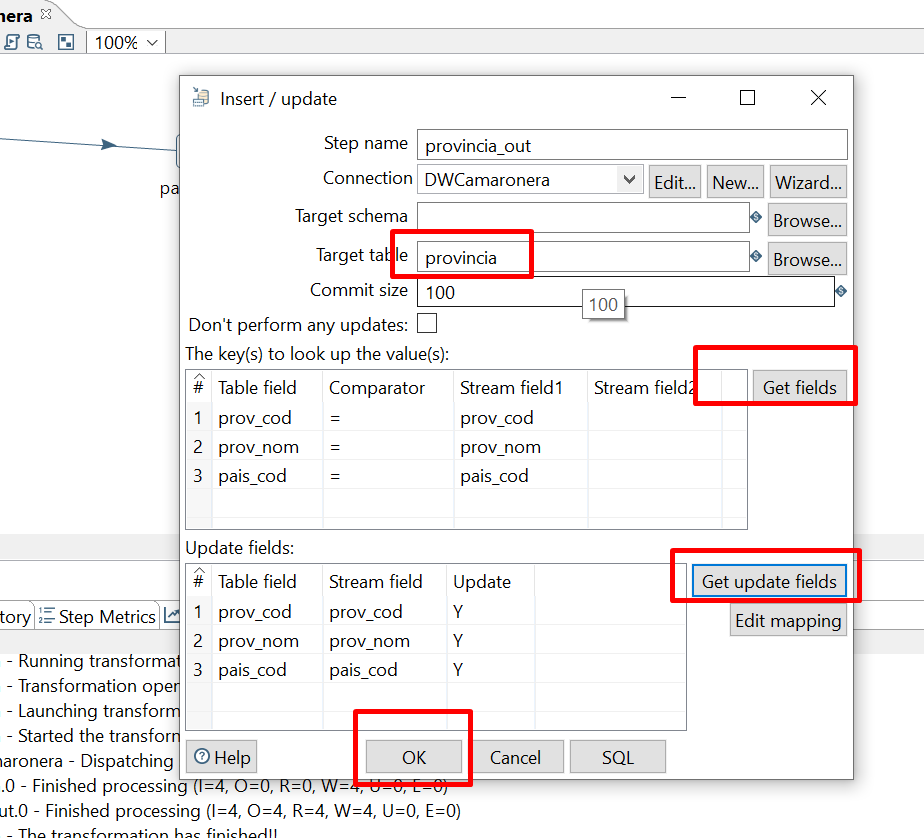
Procedemos a cargar los datos de la siguiente hoja que se encuentra en el archivo externo. , para eso limpiamos los datos del anterior hoja de excel.



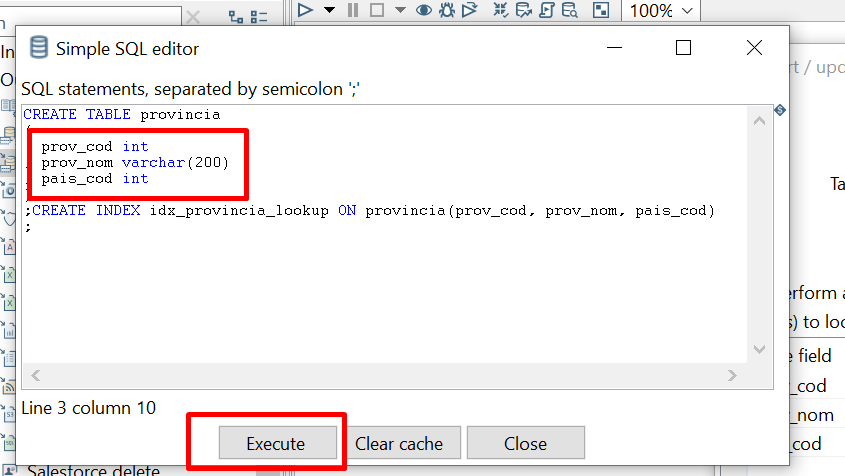
Procedemos a dar formato a los campos.



cambiamos el nombre de la tabla y colocamos provincia, obtenemos los nuevos campos dando click en get fields y en get update fields.



Modificamos si queremos el tipo de dato de las variables sql y damos click en ejecutar.



CREATE TABLE provincia

(

prov\_cod int

, prov\_nom varchar(200)

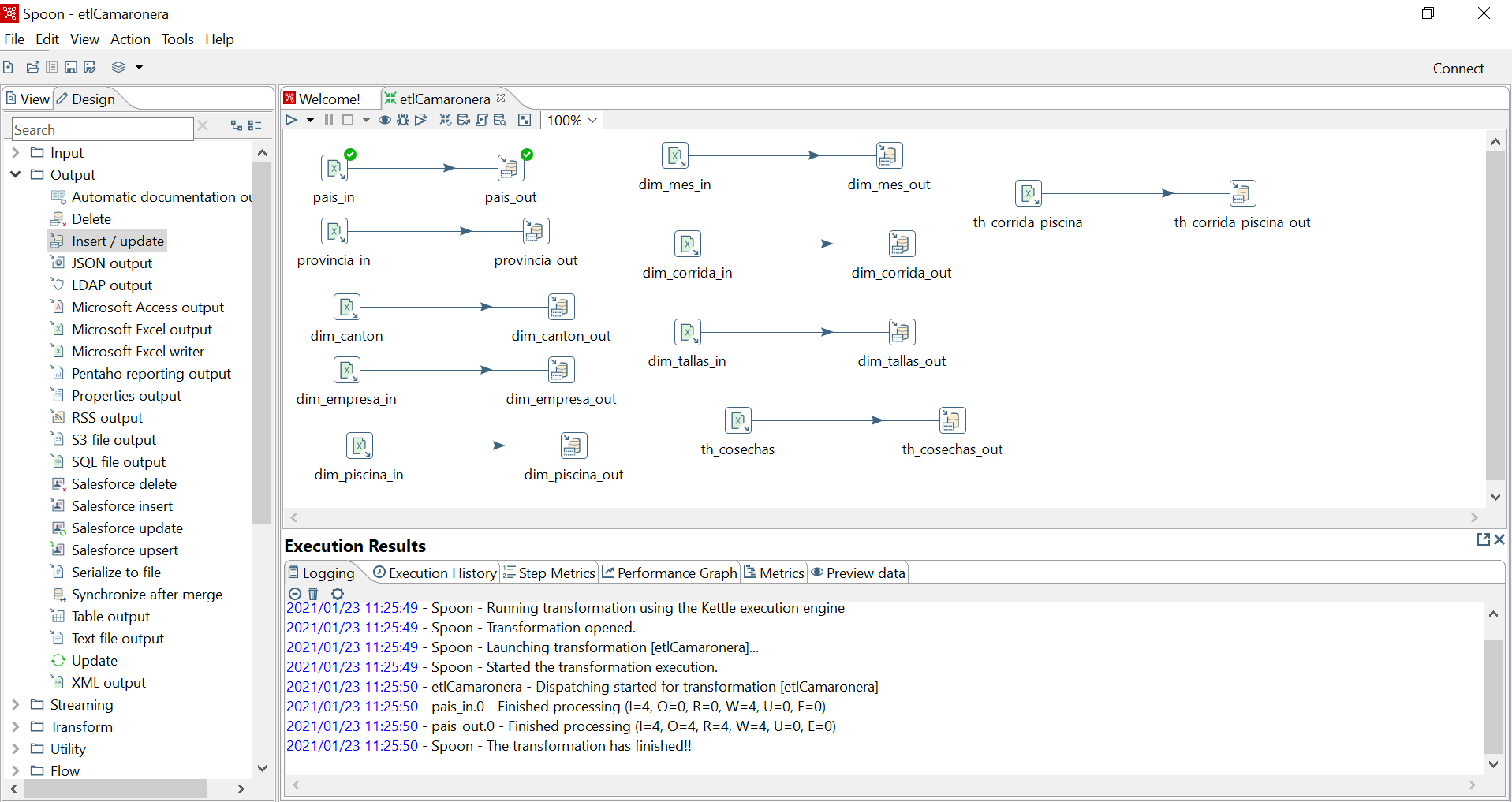
, pais\_cod int

)

;CREATE INDEX idx\_provincia\_lookup ON provincia(prov\_cod, prov\_nom, pais\_cod)

;

Se repetirá este mismo proceso ETL para las demás hojas de excel quedando como resultado lo siguiente:



Las sentencias SQL proporcionadas por este proceso:

CREATE TABLE dim\_canton

(

can\_cod int, can\_nom varchar(150), prov\_cod int

);CREATE INDEX idx\_dim\_canton\_lookup ON dim\_canton(can\_cod, can\_nom, prov\_cod)

CREATE TABLE dim\_empresa

(

emp\_cod int

, emp\_nom varchar(100)

, emp\_nom\_cam varchar(100)

, pais\_cod int

, prov\_cod int

, can\_cod int

);CREATE INDEX idx\_dim\_empresa\_lookup ON dim\_empresa(emp\_cod, emp\_nom, emp\_nom\_cam, pais\_cod, prov\_cod, can\_cod);

CREATE TABLE dim\_piscina

(

pis\_cod int

, pis\_nom varchar(20)

, pis\_num\_hect DOUBLE PRECISION

, emp\_cod int

)

;CREATE INDEX idx\_dim\_piscina\_lookup ON dim\_piscina(pis\_cod, pis\_nom, pis\_num\_hect, emp\_cod);

CREATE TABLE dim\_mes

(

mes\_codigo int

, mes varchar(20)

, trimestre varchar(10)

, semestre varchar(10)

);CREATE INDEX idx\_dim\_mes\_lookup ON dim\_mes(mes\_codigo, mes, trimestre, semestre);

CREATE TABLE dim\_corrida

(

cor\_cod int

, cor\_fec\_ini TIMESTAMP

, cor\_fec\_fin TIMESTAMP

, cor\_est varchar(20)

, anio int

, mes\_codigo int

);CREATE INDEX idx\_dim\_corrida\_lookup ON dim\_corrida(cor\_cod, cor\_fec\_ini, cor\_fec\_fin, cor\_est, año, mes\_codigo);

CREATE TABLE dim\_tallas

(

tall\_cod int

, tall\_nombre varchar(100)

, tall\_cam varchar(100)

);CREATE INDEX idx\_dim\_tallas\_lookup ON dim\_tallas(tall\_cod, tall\_nombre, tall\_cam);

CREATE TABLE th\_cosecha

(

cos\_cant\_kg DOUBLE PRECISION

, cos\_pre\_kg DOUBLE PRECISION

, cos\_ing DOUBLE PRECISION

, cor\_cod int

, pis\_cod int

, tal\_cod int

, emp\_cod int

);CREATE INDEX idx\_th\_cosecha\_lookup ON th\_cosecha(cos\_cant\_kg, cos\_pre\_kg, cos\_ing, cor\_cod, pis\_cod, tal\_cod, emp\_cod);

CREATE TABLE th\_corrida\_piscina

(

cp\_num\_larv int

, cp\_cos\_inv DOUBLE PRECISION

, cp\_total\_ing DOUBLE PRECISION

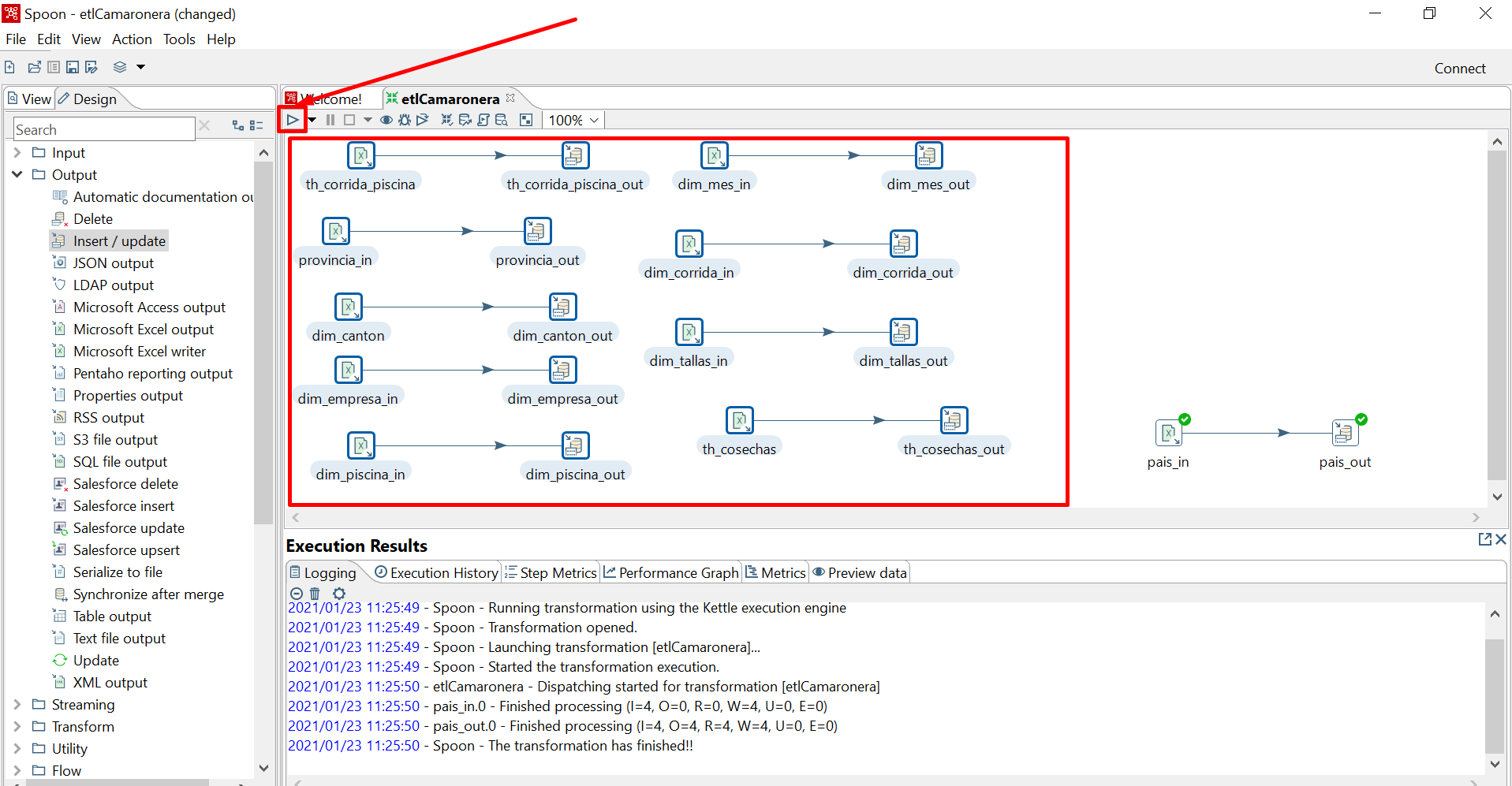
, emp\_cod int

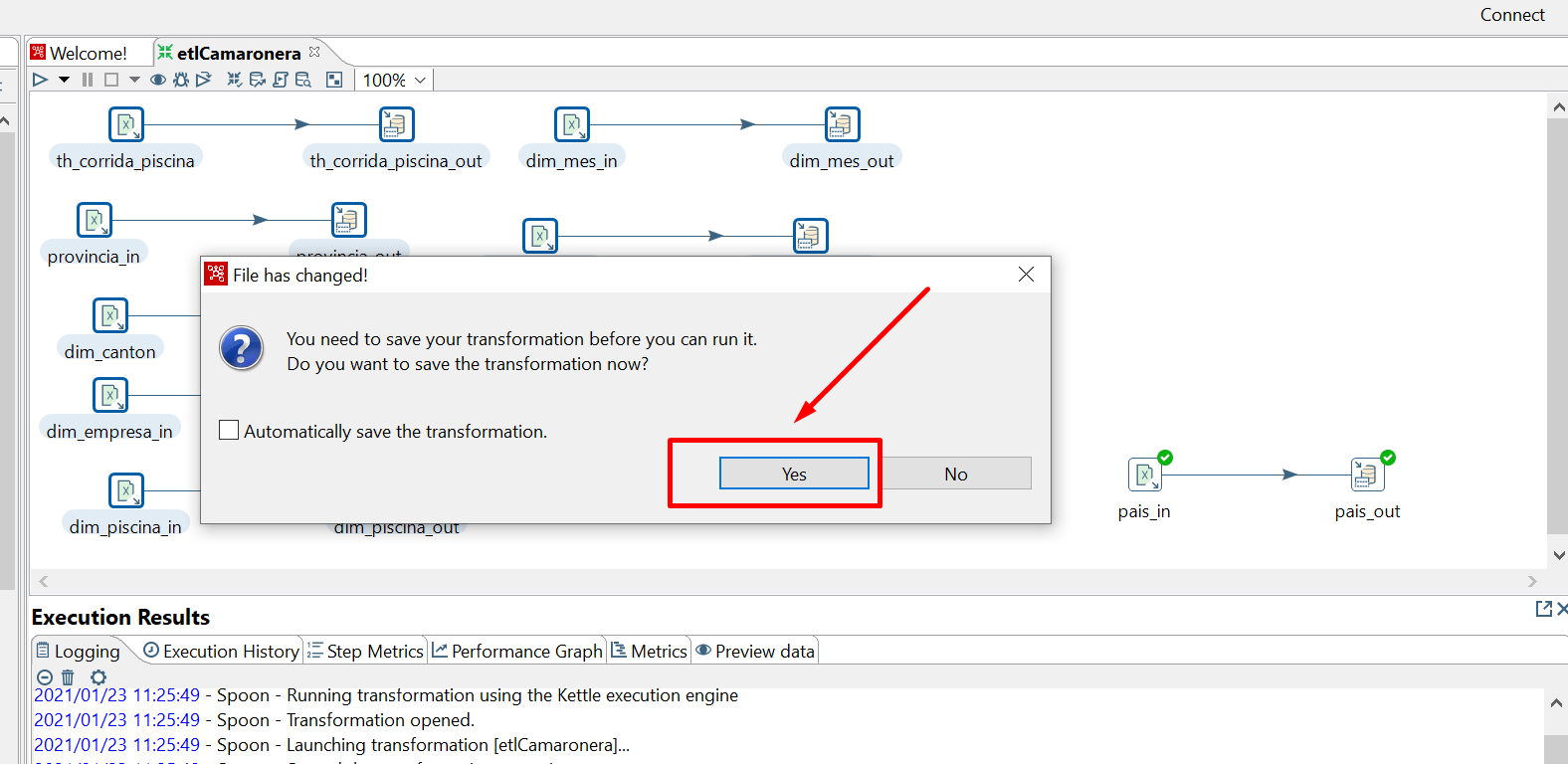
, pis\_cod int

, cor\_cod int

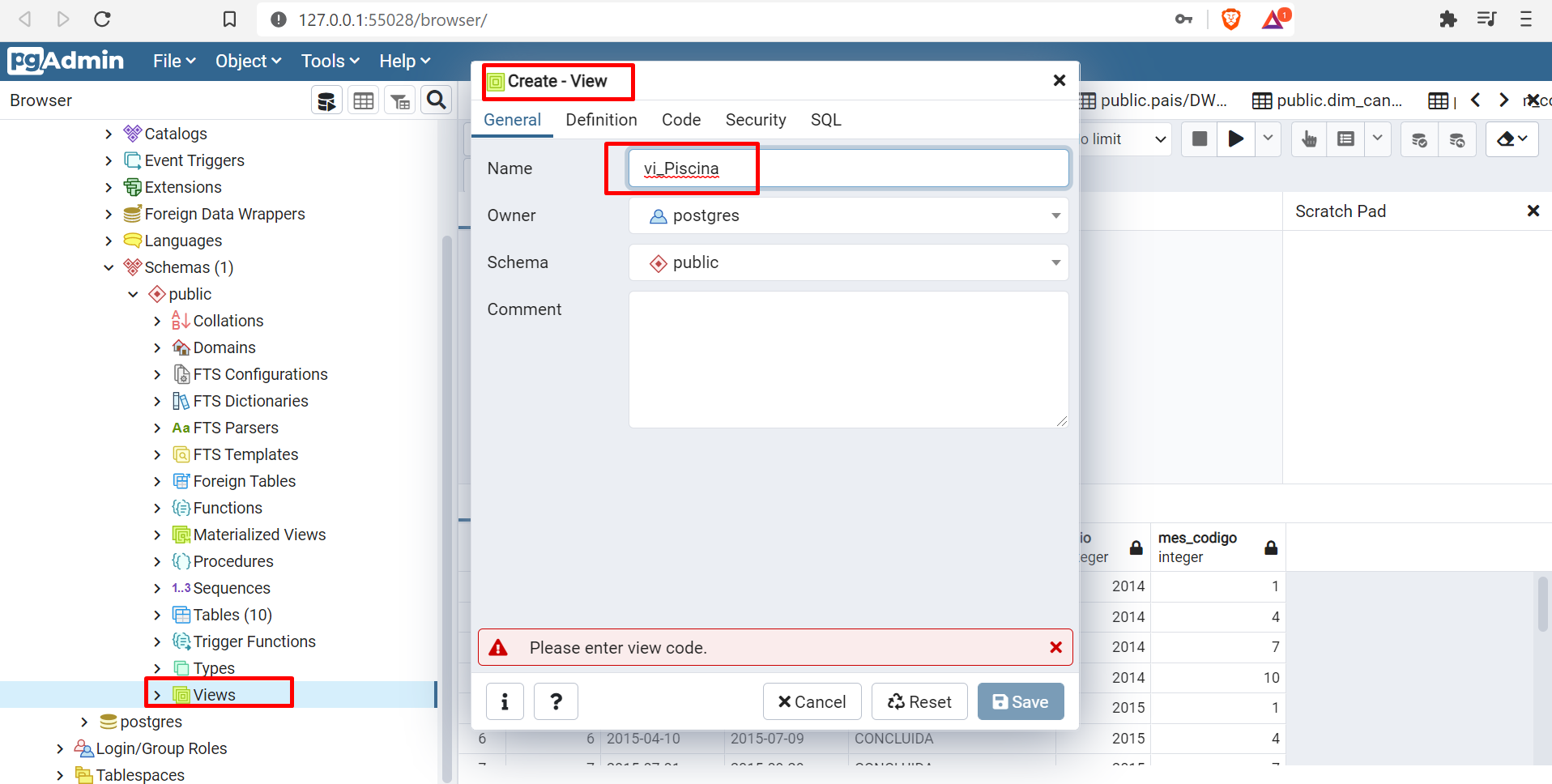
);CREATE INDEX idx\_th\_corrida\_piscina\_lookup ON th\_corrida\_piscina(cp\_num\_larv, cp\_cos\_inv, cp\_total\_ing, emp\_cod, pis\_cod, cor\_cod);

Ahora se procederá con el proceso de carga de los datos desde Pentaho Data Integration hacia la BD Postgresql, para eso se selecciona los archivos que desea ejecutar y dar click en el icono del play (run).

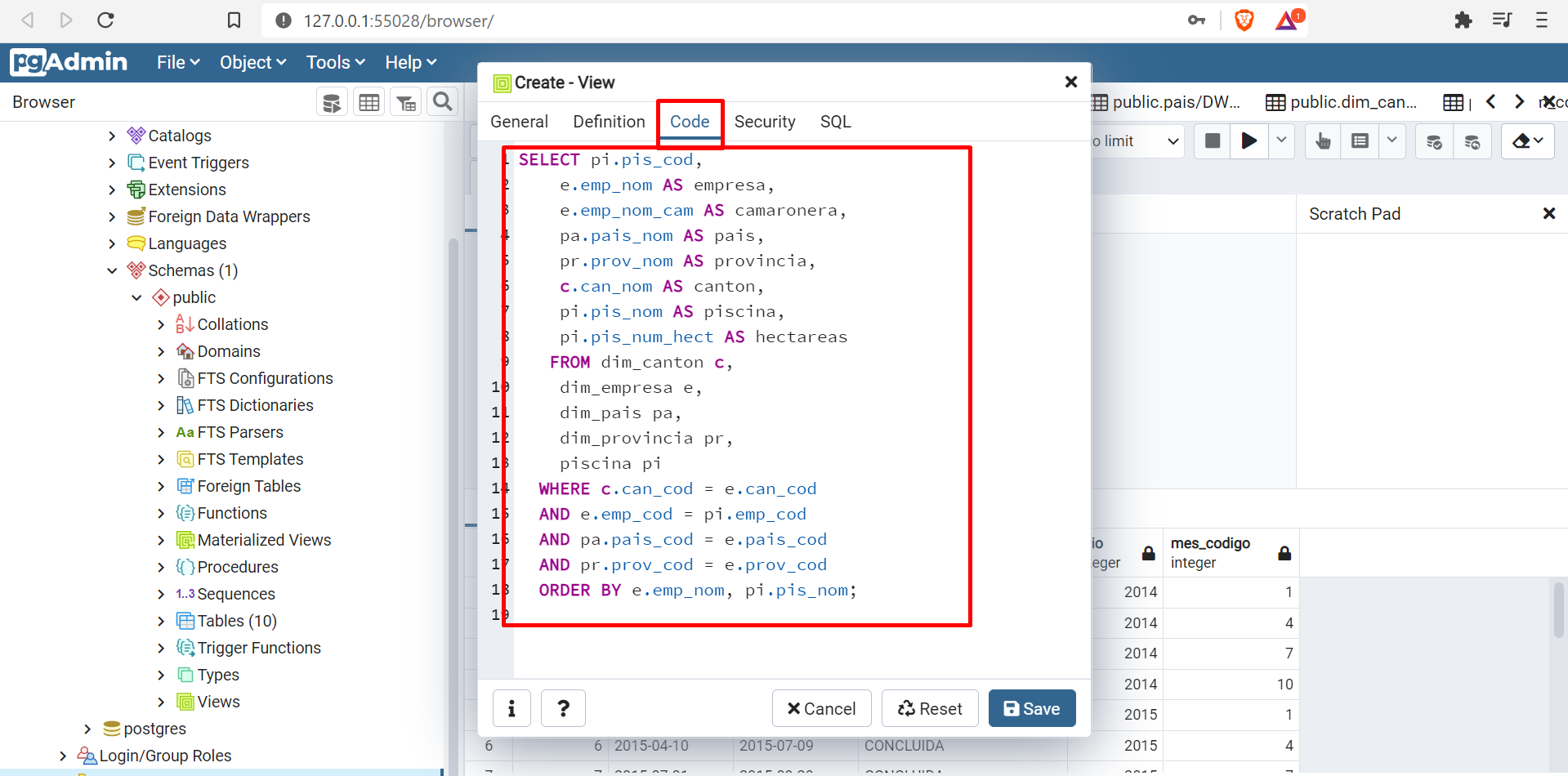
  
Le pedirá que guarde los cambios nuevamente del archivo de transformación.



Ahora se procederá con la creación de las Vistas en PostgreSql, como ejemplo se tomará la vista para la piscina (vi\_Piscina), se debe dar click derecho en la Views, luego dar click en create y dar nombre a la vista, en este caso se llamará vi\_Piscina.



Luego en la pestaña code, codificamos la sentencia sql para crear la vista.



SCRIPT SQL VISTA **vi\_Piscina**

SELECT pi.pis\_cod,

e.emp\_nom AS empresa,

e.emp\_nom\_cam AS camaronera,

pa.pais\_nom AS pais,

pr.prov\_nom AS provincia,

c.can\_nom AS canton,

pi.pis\_nom AS piscina,

pi.pis\_num\_hect AS hectareas

FROM dim\_canton c,

dim\_empresa e,

dim\_pais pa,

dim\_provincia pr,

dim\_piscina pi

WHERE c.can\_cod = e.can\_cod

AND e.emp\_cod = pi.emp\_cod

AND pa.pais\_cod = e.pais\_cod

AND pr.prov\_cod = e.prov\_cod

ORDER BY e.emp\_nom, pi.pis\_nom;

Repetimos este proceso con el resto de vistas.

SCRIPT SQL VISTA vi\_piscina\_corrida

SELECT pi.empresa,

pi.camaronera,

pi.pis\_cod,

pi.piscina,

cr.cor\_cod AS corrida,

cr.cor\_est AS estado,

pi.pais,

pi.provincia,

pi.canton,

cr.anio AS "Año",

mes.semestre,

mes.trimestre,

mes.mes,

cp.cp\_num\_larv AS "Num larvas",

cp.cp\_num\_larv::double precision / pi.hectareas AS "Larvas por Hectarea",

cp.cp\_cos\_inv AS "Costo de inversion",

i.ingresos AS "Total ingresos",

round(( i.ingresos - cp.cp\_cos\_inv)::numeric, 3) AS "Utilidad monto",

round((( i.ingresos - cp.cp\_cos\_inv) / cp.cp\_cos\_inv \* 100::double precision)::numeric, 3) AS "Utilidad %"

FROM

dim\_mes mes,

"vi\_Piscina" pi,

dim\_corrida cr,

th\_corrida\_piscina cp,

(SELECT

c.pis\_cod,

c.cor\_cod,

sum(c.cos\_cant\_kg \* c.cos\_pre\_kg) AS ingresos

FROM th\_cosecha c

GROUP BY c.emp\_cod, c.pis\_cod, c.cor\_cod

ORDER BY c.emp\_cod, c.pis\_cod, c.cor\_cod) i

WHERE mes.mes\_codigo = cr.mes\_codigo

AND pi.pis\_cod = cp.pis\_cod

AND cr.cor\_cod = cp.cor\_cod

AND cp.pis\_cod = i.pis\_cod

AND cp.cor\_cod = i.cor\_cod

ORDER BY pi.empresa, pi.piscina, cr.cor\_cod;

SCRIPT SQL VISTA vi\_cosecha

SELECT pc.empresa,

pc.camaronera,

pc.piscina,

pc.corrida,

pc.estado,

pc.pais,

pc.provincia,

pc.canton,

pc."Año",

pc.semestre,

pc.trimestre,

pc.mes,

t.tall\_nombre AS "Talla",

c.cos\_cant\_kg AS "Cantidad kg",

c.cos\_pre\_kg AS "Precio kg",

c.cos\_cant\_kg \* c.cos\_pre\_kg AS "Total ingreso"

FROM "vi\_piscina\_corrida" pc,

dim\_tallas t,

th\_cosecha c

WHERE c.tal\_cod = t.tall\_cod AND c.pis\_cod = pc.pis\_cod AND c.cor\_cod = pc.corrida

ORDER BY pc.empresa, pc.piscina, pc.corrida;

El resultado final tendremos 3 vistas creadas en nuestra base de datos.

