Laboratorio #4 ETL



Evaluación de herramientas ETL

Autores:

Miguel Ángel Acosta (201914976)

Andrés Felipe Rincón (201914118)

Ángela Liliana Jiménez (201912941)

Tabla de Contenidos



[**Bono #1 Preparación de datos**](#_ald5g73x0ebh) **2**

[**Bono #2 Datos Completos**](#_uwzrg3aechq4) **4**

[**Diagrama de alto nivel del proceso de ETL**](#_qpoyr3k5llcm) **5**

[**Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en Spoon**](#_vtzt6dc0ikx0) **5**

[**Resultados Spoon**](#_w4ikq3ostkne) **16**

[**Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en BigQuery**](#_ns201pb4zjyh) **18**

[**Resultados BigQuery**](#_qmve6gj7q2h4) **18**

[**Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en la 3ra herramienta**](#_2fxm05bz9ijv) **18**

[**Resultados 3ra herramienta**](#_byxe6duj92o4) **18**

[**Respuestas a las preguntas del laboratorio**](#_o7zddr5cdalg) **18**

[**Comparación de herramientas**](#_82pcy1vdofn0) **18**

# Bono #1 Preparación de datos

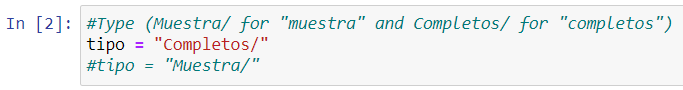


El código python necesario para el preprocesamiento de datos está en Preparacion\_Datos.ipynb, para crear los archivos con los datos pre procesados, es necesario que los archivos se encuentren en la carpeta /data/Muestra o /data/Completos, esto con relación al notebook, es decir, la idea es que al momento de ejecutar el notebook los archivos estén con la siguiente distribución:



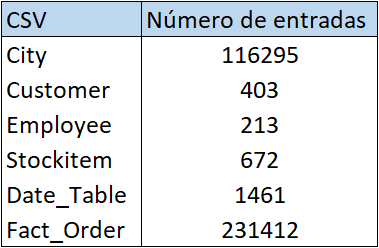
Los archivos serán creados en su respectiva carpeta, es decir, si queremos crear los datos preprocesados de los csv de Muestra, estos serán creados en la carpeta Muestra, con el mismo nombre que el csv original y el sufijo *\_ preprocessed*.

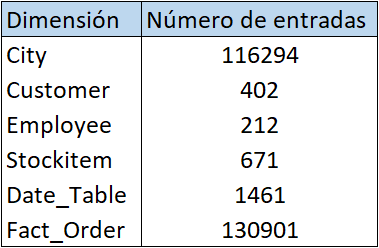
En In[2] se puede especificar si el preprocesamiento se realizará para Muestra o para Completos



Con respecto al preprocesamiento, encontramos lo siguiente:

* Inicialmente recibimos la siguiente cantidad de entradas de los csv originales:



* Varias dimensiones tienen entradas que son completamente nulas, por lo que se decide remover estas entradas. Es importante resaltar que si eliminamos una entrada de una dimensión tenemos que garantizar la consistencia en la tabla de hechos, a continuación damos un ejemplo con la dimensión city:  
    
  En este punto hemos removido entradas del dataframe df\_city, (dimensión city) ahora nos quedaremos con las entradas de df\_order (tabla de hechos) cuya llave foránea correspondiente a ciudad se encuentre en la dimensión ciudad, de tal forma eliminamos las entradas de la tabla de hechos que se refirieran a una entrada en la dimensión city que hayamos eliminado.
* En la dimensión Stockitem tenemos muchos valores nulos para los valores de color, brand y size\_val (90%, 47%, 30% de valores nulos respectivamente) retirar las entradas que tengan valores nulos en este caso puede ser perjudicial por que perderiamos una cantidad de datos considerable, no solo en esta dimensión sino en la tabla de hechos, por lo que se decide llenar estos valores nulos por “unknown” y se procede a eliminar las entradas con registros nulos (ya sin contar estos valores anteriores como nulos)
* Se elimina el valor WWI\_Stock\_Item\_ID de todos las entradas de la dimensión Stockitem porque este valor no existe en el modelo
* Finalmente después de retirar las inconsistencias tenemos la siguiente cantidad de entradas:  
  

# Bono #2 Datos Completos

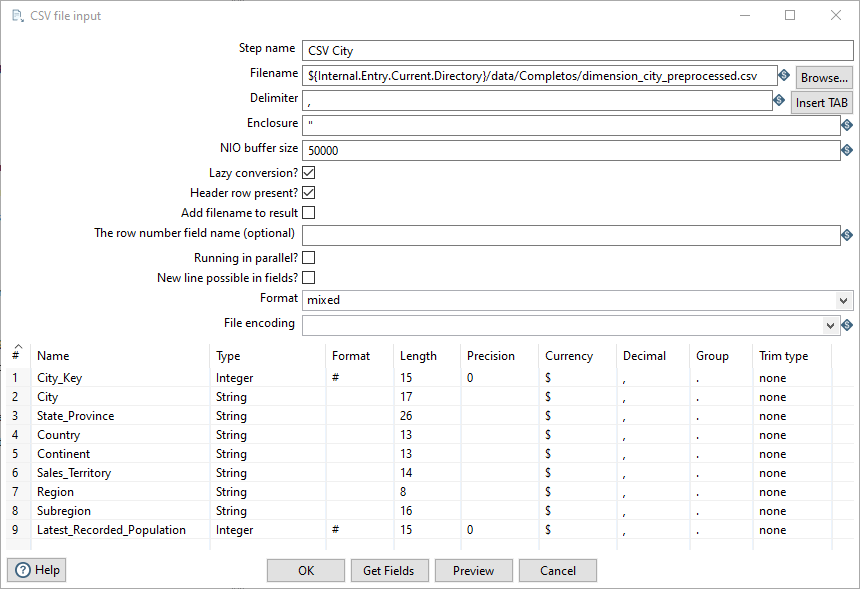


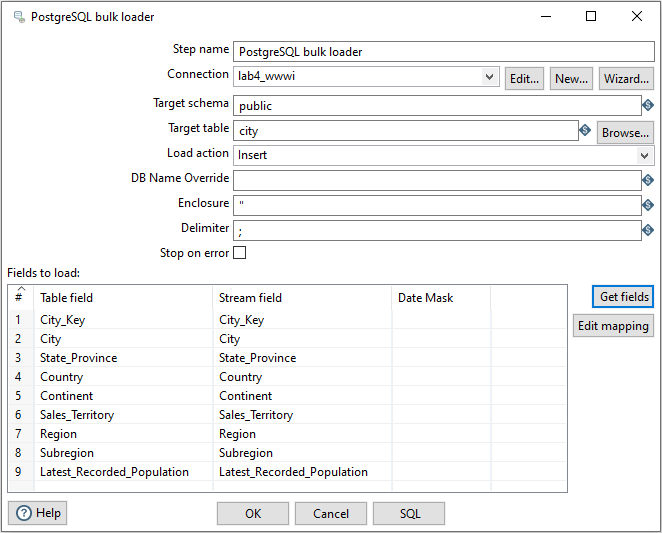
Para el proceso de ETL en las distintas herramientas usamos los datos Completos (luego de preprocesamiento), se puede evidenciar esto en los resultados de las consultas para cada herramienta, en donde se ve como la cantidad de datos corresponde a lo reportado anteriormente.

Particularmente para pentaho spoon Usamos la forma de carga de datos normal. Existía la forma de cargar los datos con la herramienta bulkload, que básicamente consiste en cambiar el proceso de transformación a uno como el siguiente (ponemos como ejemplo la dimensión city):



Y trabajamos en *CSV City* y en *PostgreSQL bulk loader*





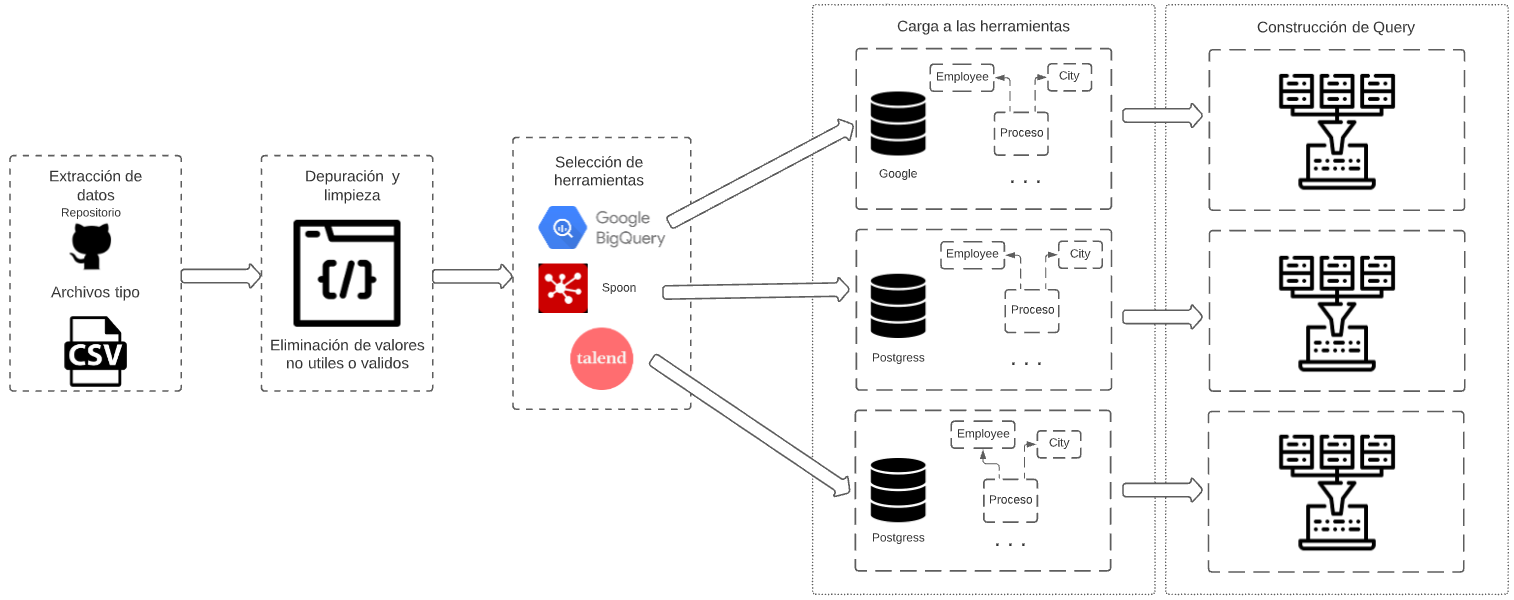
Realizar el proceso de ETL con este esquema demostró cargar los datos mucho más rápido que el esquema normal, pero tiene un problema que causó que descartamos este método y es que bulkload es un método de carga de datos que solo realiza operaciones insert/truncate, es decir o inserta los datos o borra todo para insertar datos, esto es problemático porque:

* Para la primera carga no hay problemas, pero para una segunda ejecución del job, las llaves chocaran con las existentes.
* Eso se puede arreglar eliminando los datos, sin embargo por la existencia de relaciones entre tablas es necesario revisar las reglas de estas llaves.
* Luego de lo mencionado, que son principalmente, problemas operativos la razón por la que principalmente lo descartamos es una razón de negocio y es que estamos hablando de eliminar todos los registros de la base de datos, lo que no es algo aceptable en este contexto.

En resumen, estamos usando los datos Completos, los usamos con la herramienta de carga normal, que aunque carga un poco más lento que bulk load funciona, y no presenta los problemas antes mencionados.

# Diagrama de alto nivel del proceso de ETL

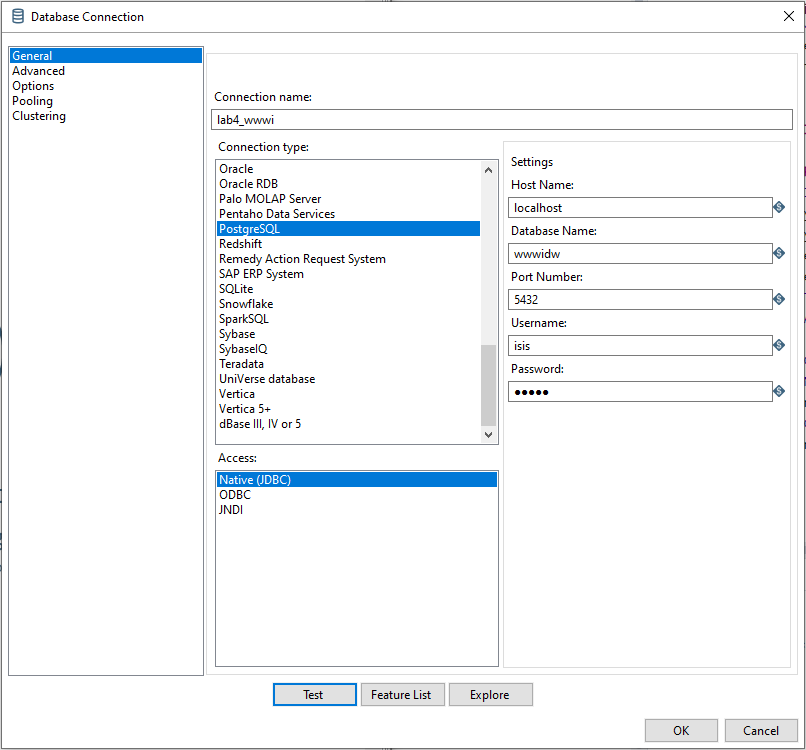
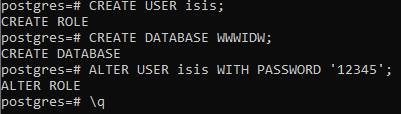




# Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en Spoon



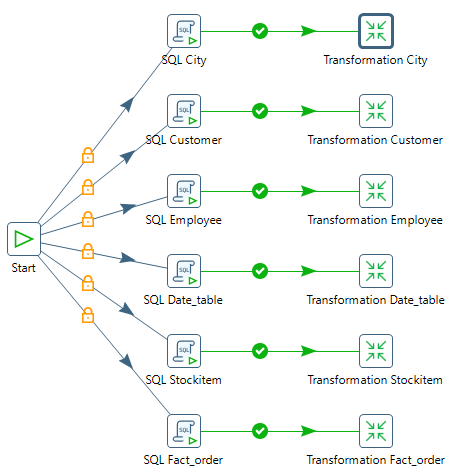
Creamos una base de datos en postgresql y nos conectamos a ella en pentaho



Luego creamos un job *job\_dimensiones* y creamos un esquema Start → SQL → Transformation, como se ve a continuación para date\_table:

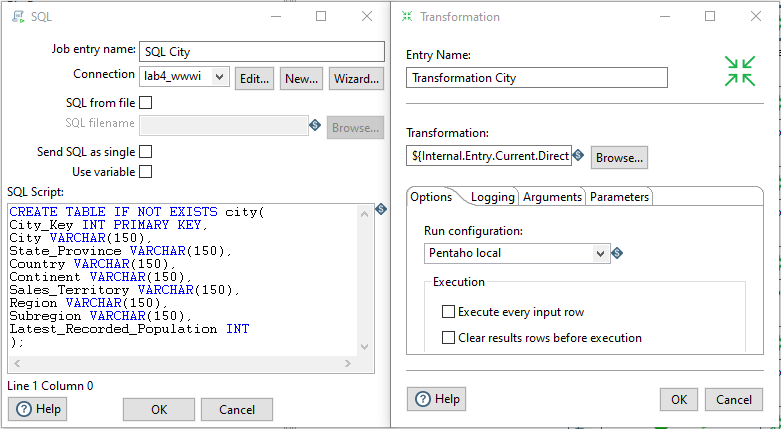
****

Para al final tener el siguiente esquema:



**Trabajo para City:**

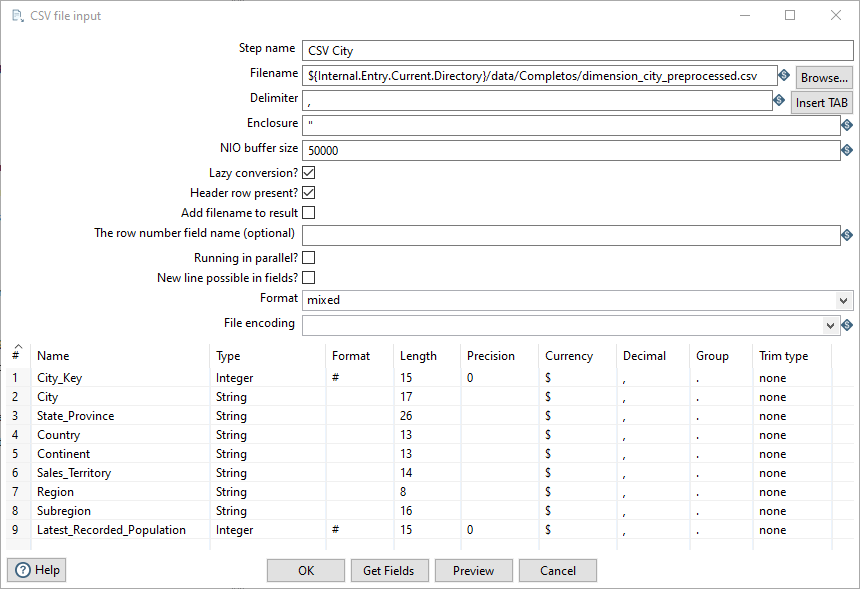
Trabajamos en el sql *SQL City* y en la transformación *Transformation City*

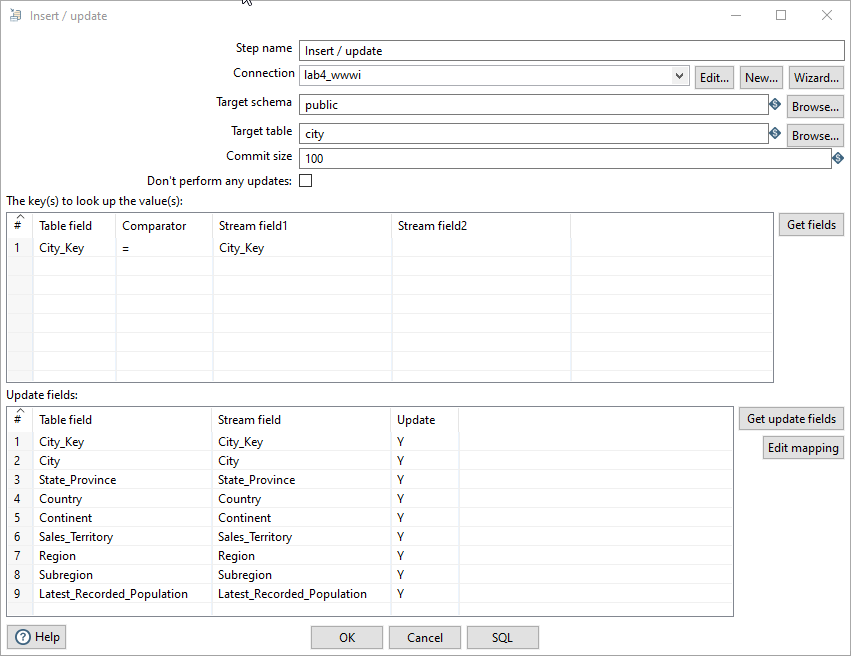


Creamos la transformación *Transformation City*



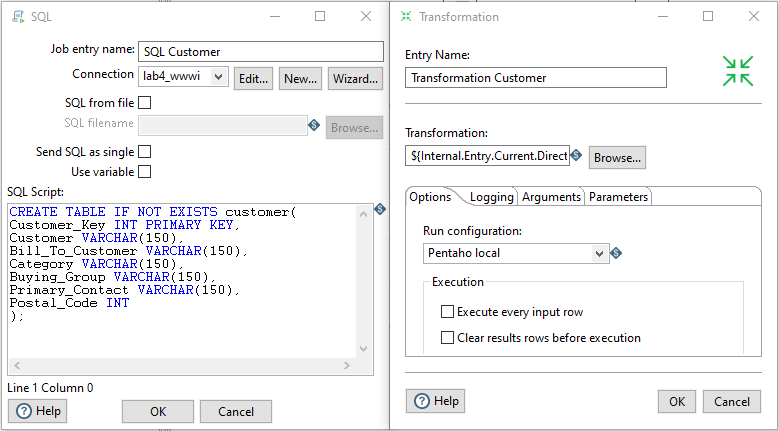
Trabajamos en *CSV City* y en *Insert / update*





**Trabajo para Customer:**

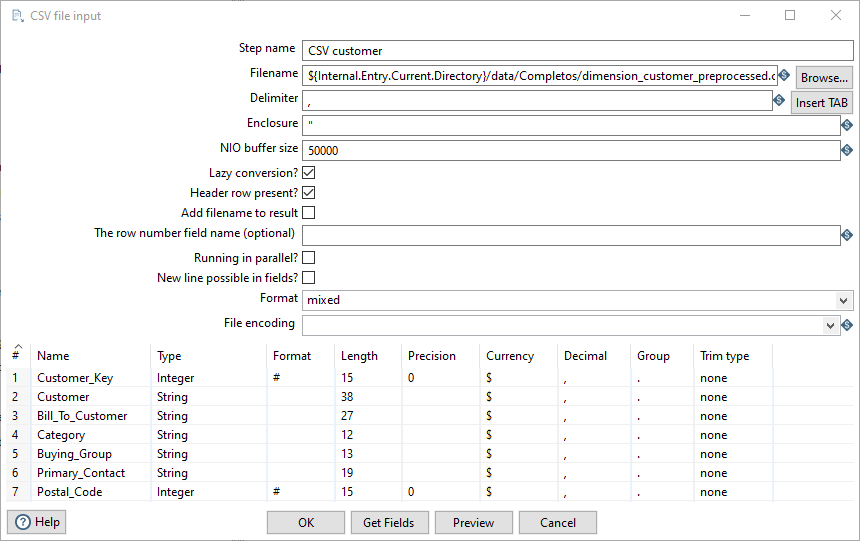
Trabajamos en el sql *SQL Customer* y en la transformación *Transformation Customer*

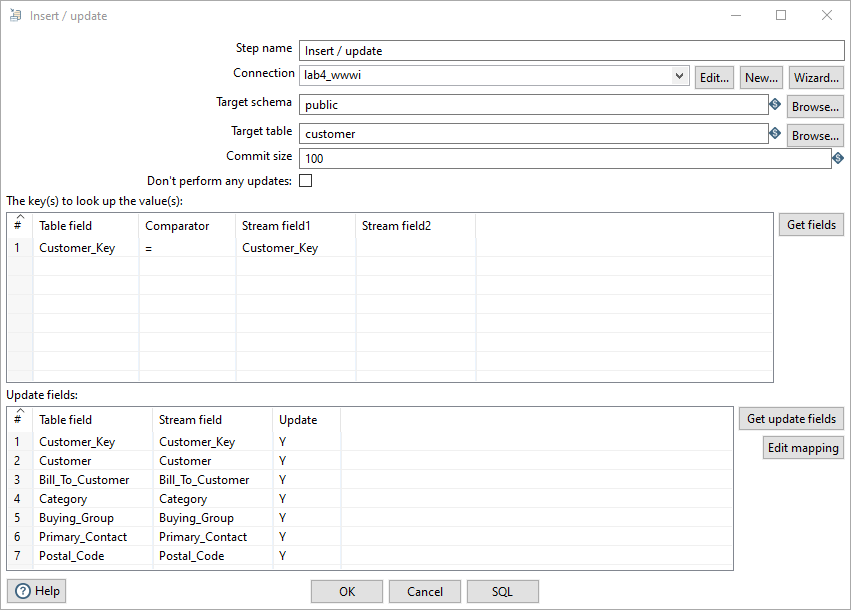


Creamos la transformación *Transformation Customer*

**

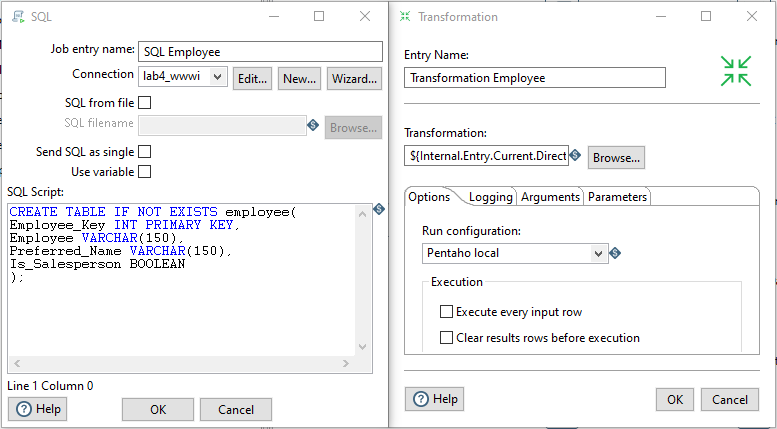
Trabajamos en *CSV Customer* y en *Insert / update*

**

**

**Trabajo para Employee:**

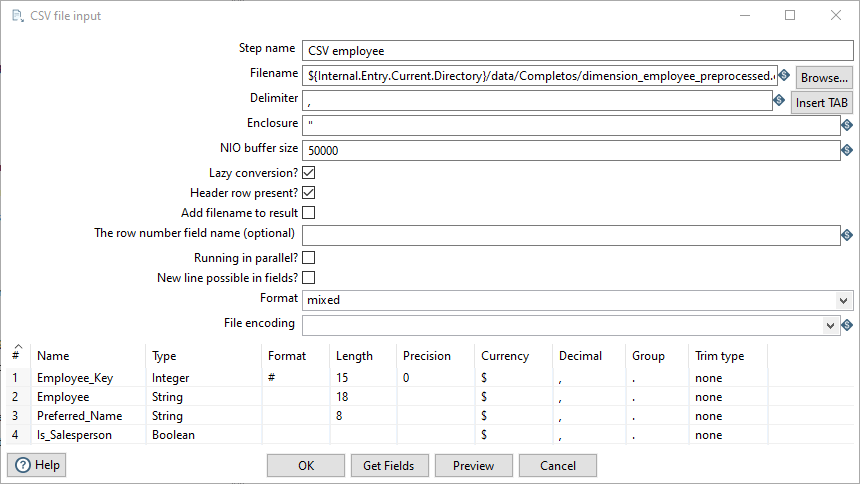
Trabajamos en el sql *SQL Employee* y en la transformación *Transformation Employee*



Creamos la transformación *Transformation Employee*



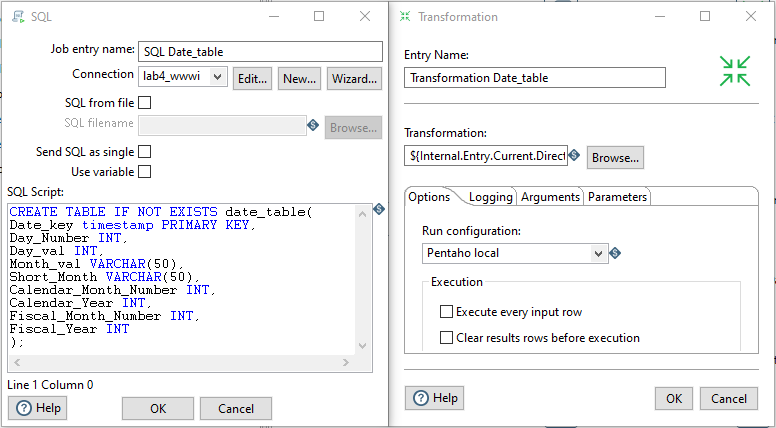
Trabajamos en *CSV Employee* y en *Insert / update*





**Trabajo para Date\_Table:**

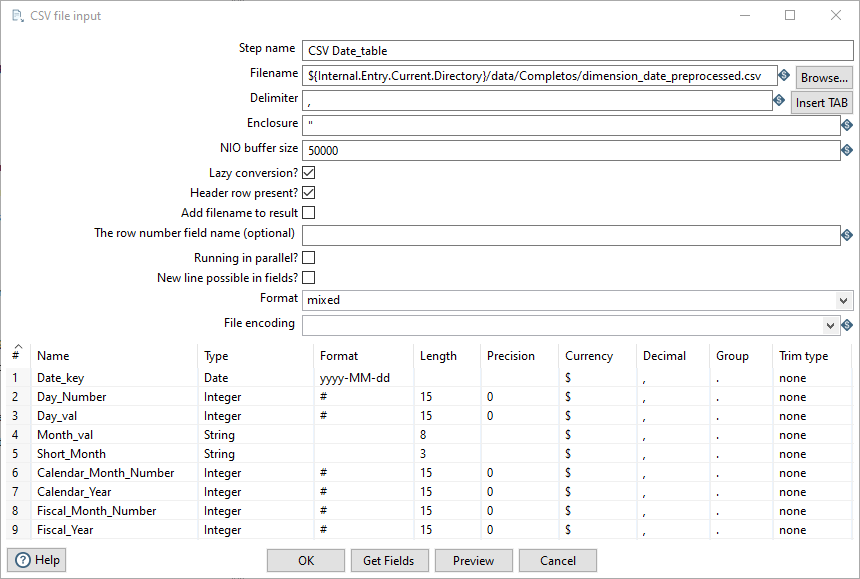
Trabajamos en el sql *SQL Date\_table* y en la transformación *Transformation Date\_table*

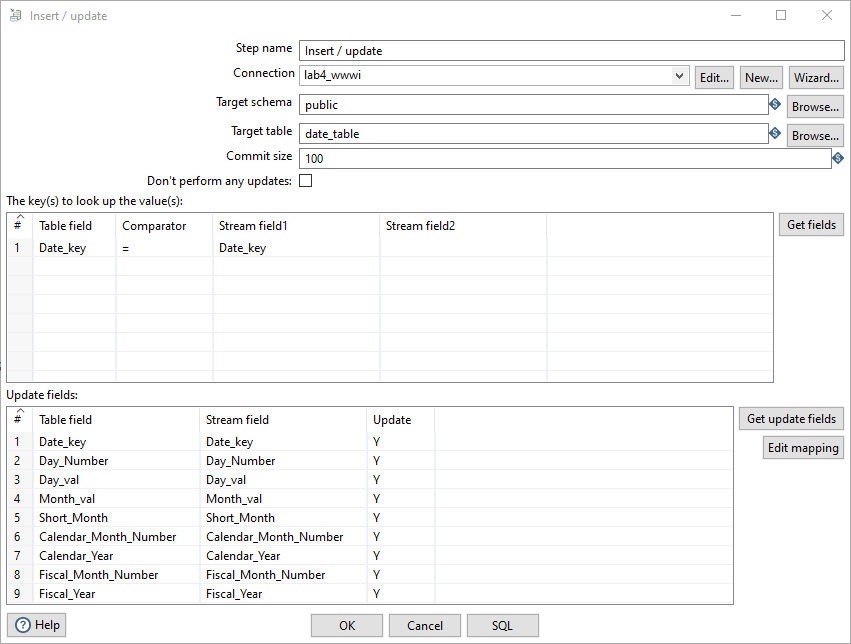


Creamos la transformación *Transformation Date\_Table*



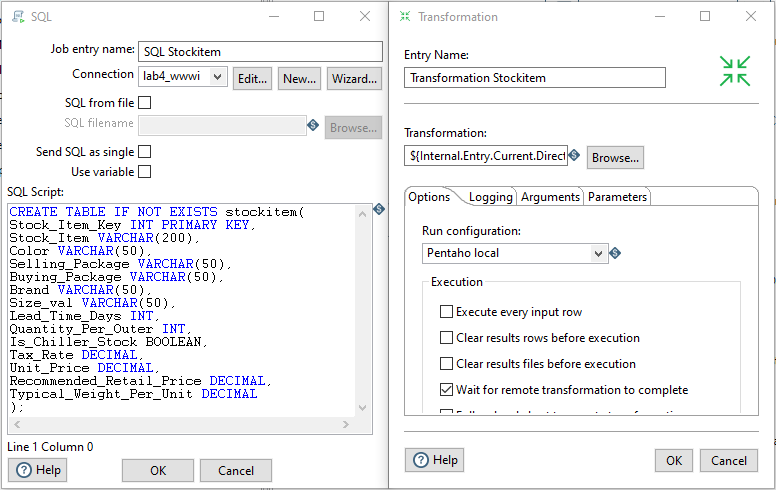
Trabajamos en *CSV Date\_Table* y en *Insert / update*





**Trabajo para Stockitem:**

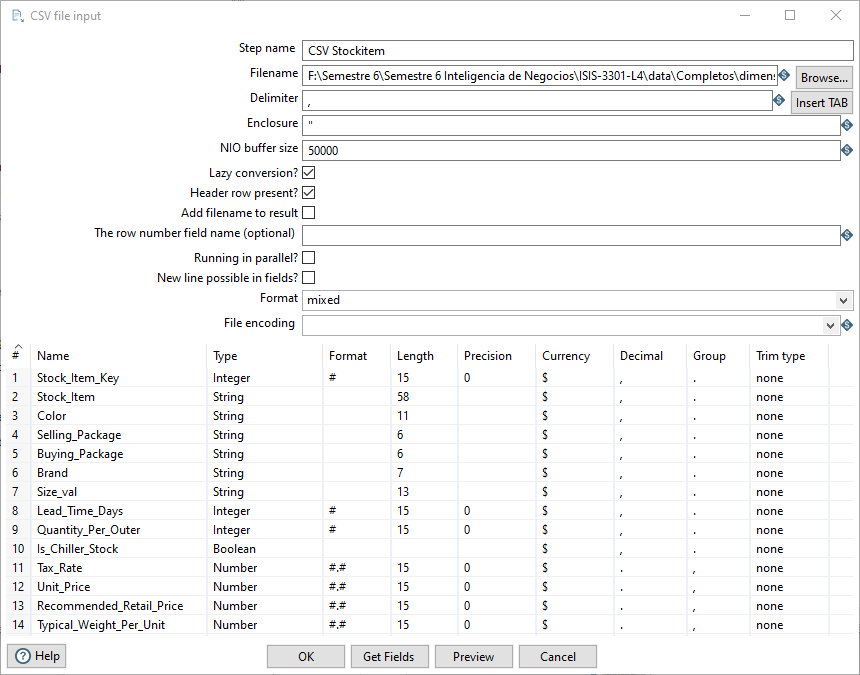
Trabajamos en el sql *SQL Stockitem* y en la transformación *Transformation Stockitem*

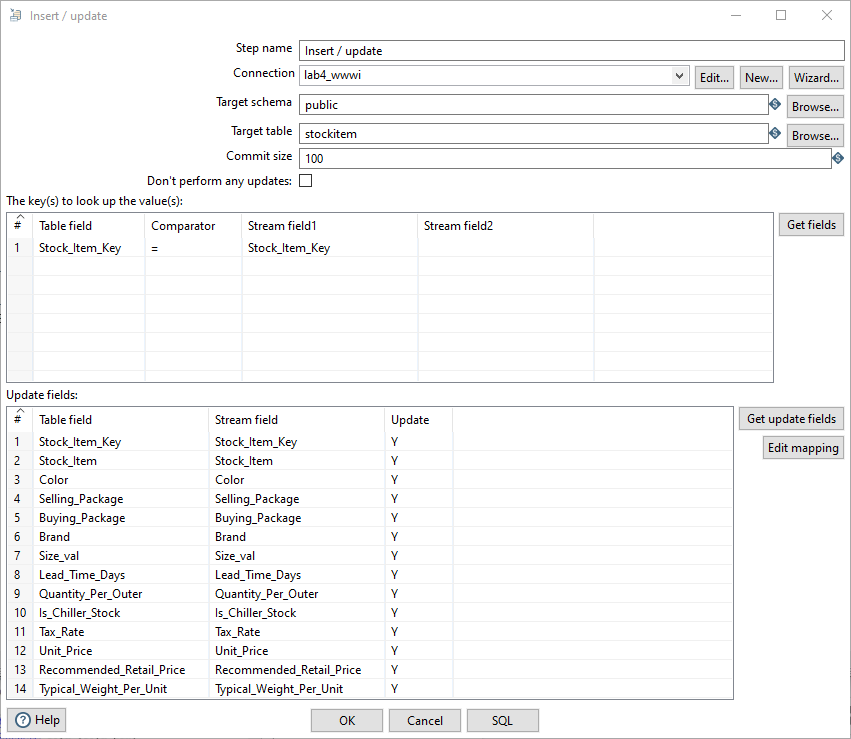


Creamos la transformación *Transformation Stockitem*



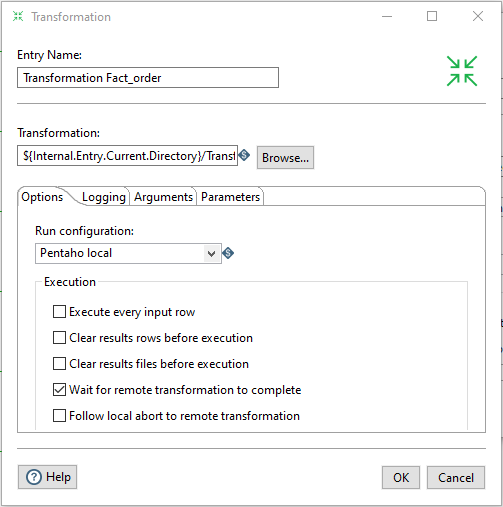
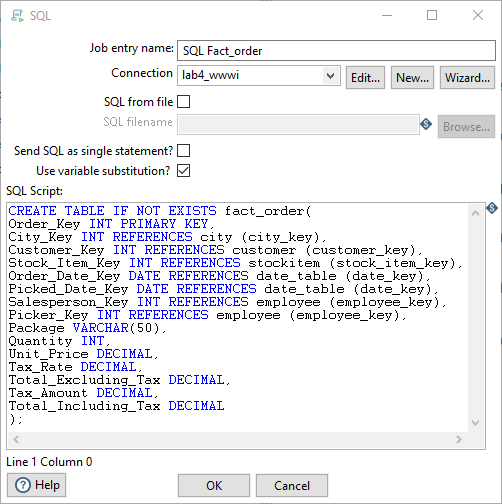
Trabajamos en *CSV Stockitem* y en *Insert / update*





**Trabajo para Fact\_order:**

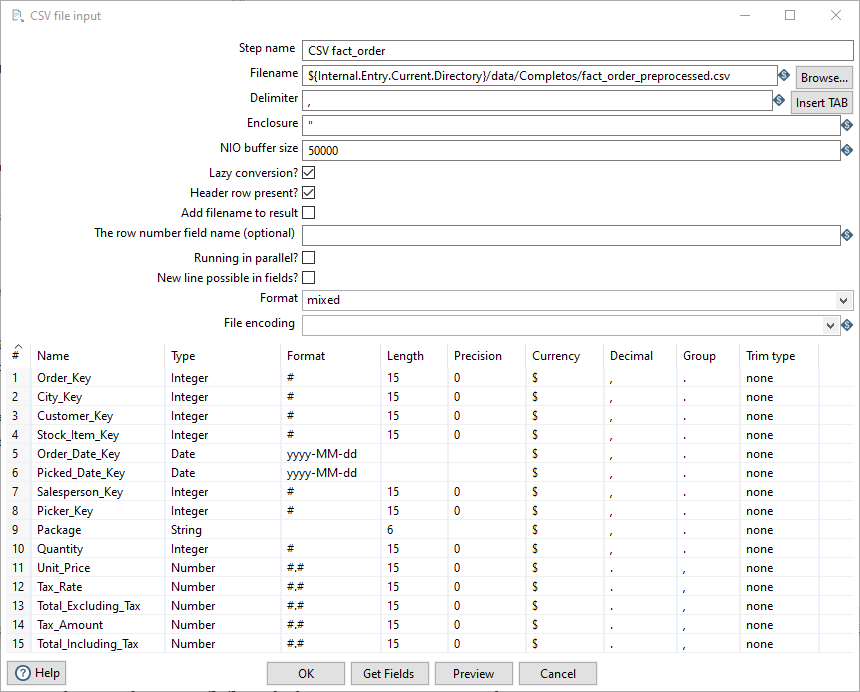
Trabajamos en el sql *SQL Fact\_order* y en la transformación *Transformation Fact\_order*

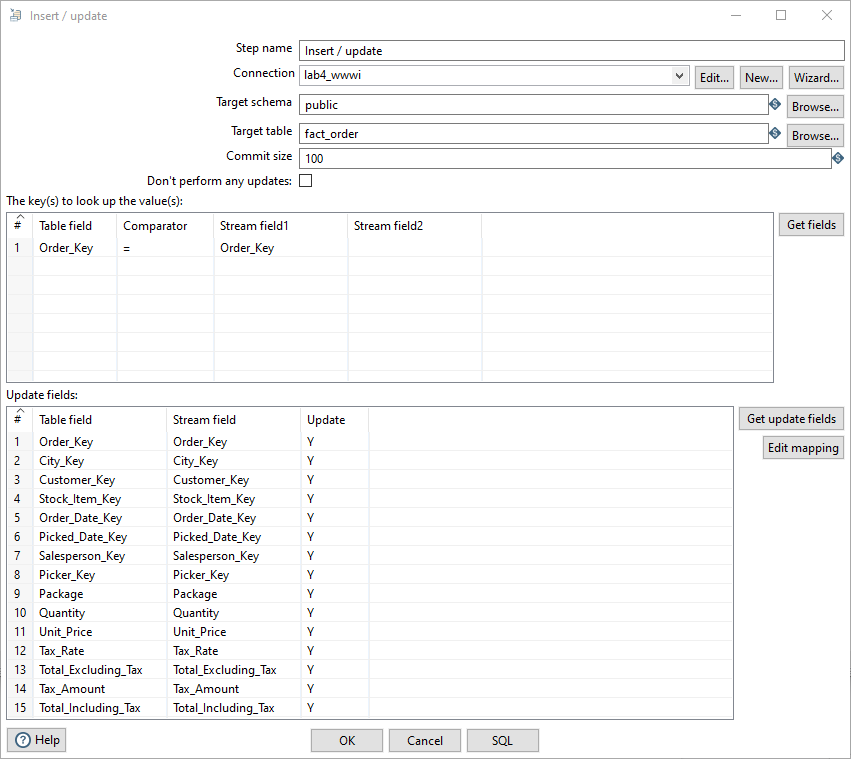


Creamos la transformación *Transformation Fact\_order*



Trabajamos en *CSV Fact\_order* y en *Insert / update*

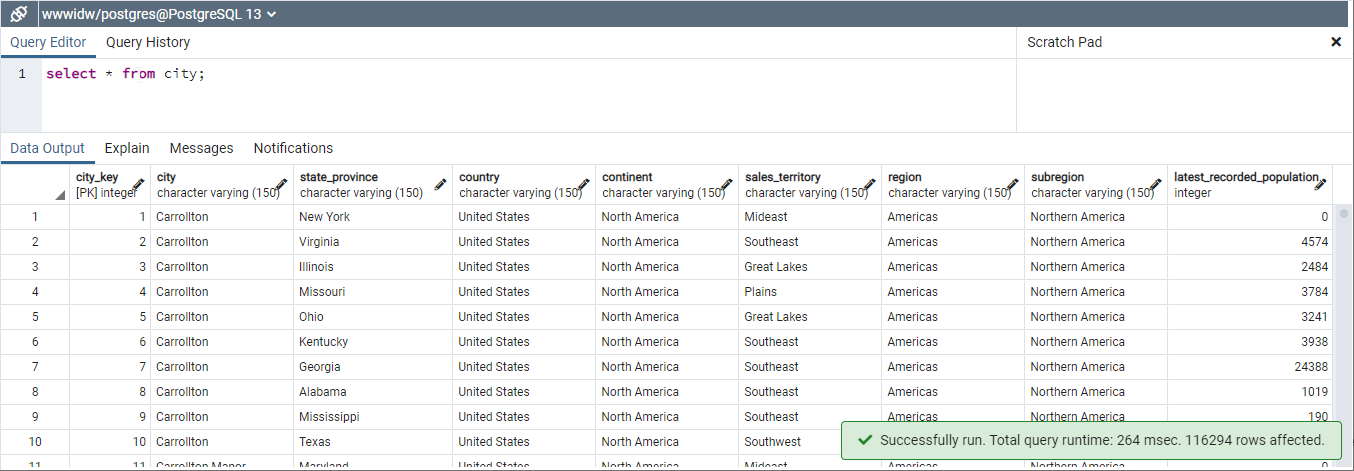




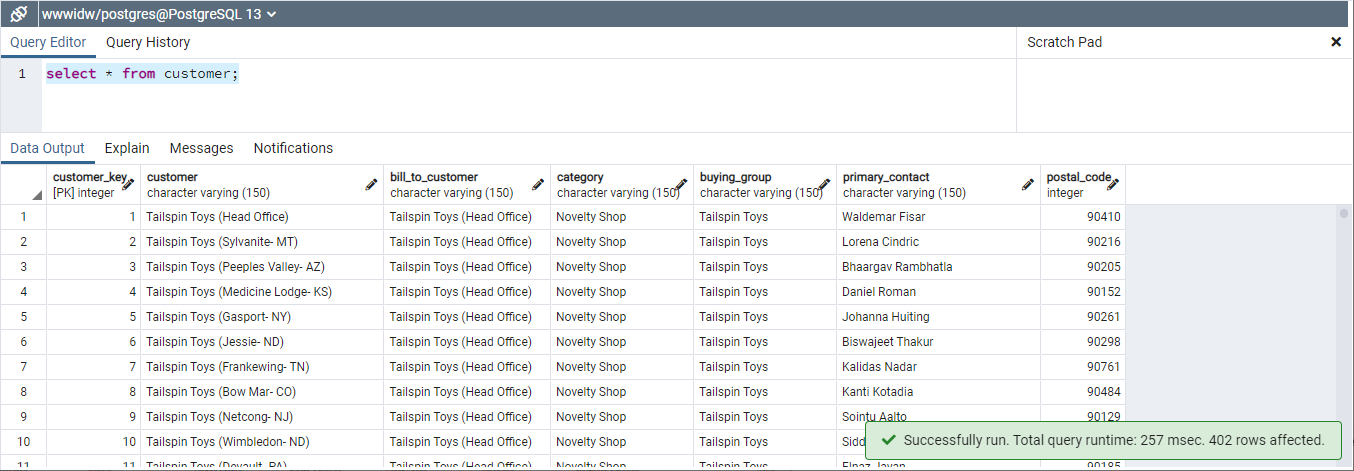
# Resultados Spoon



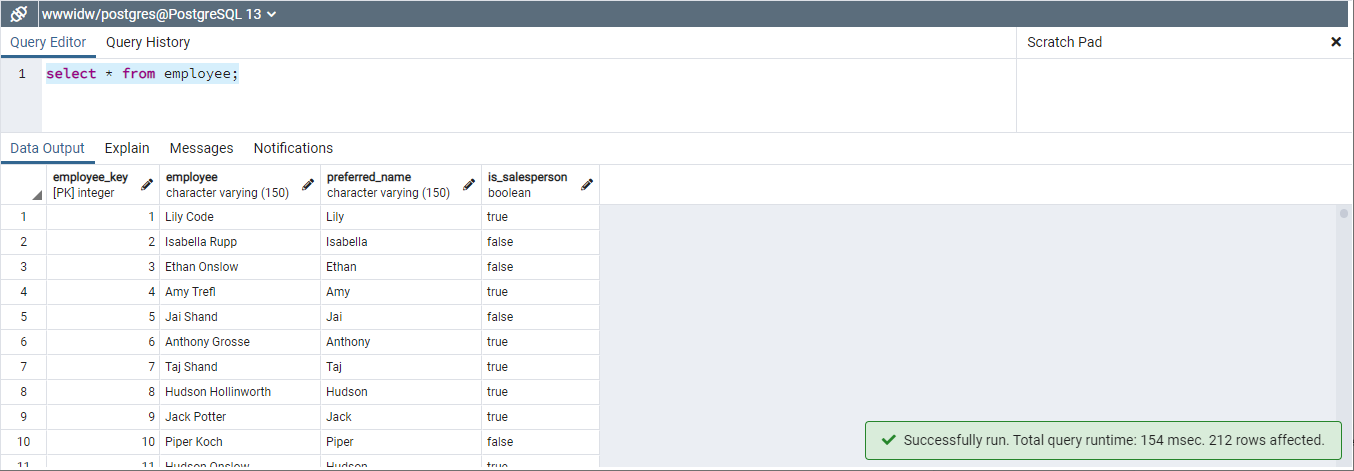
**Consultas para City:**



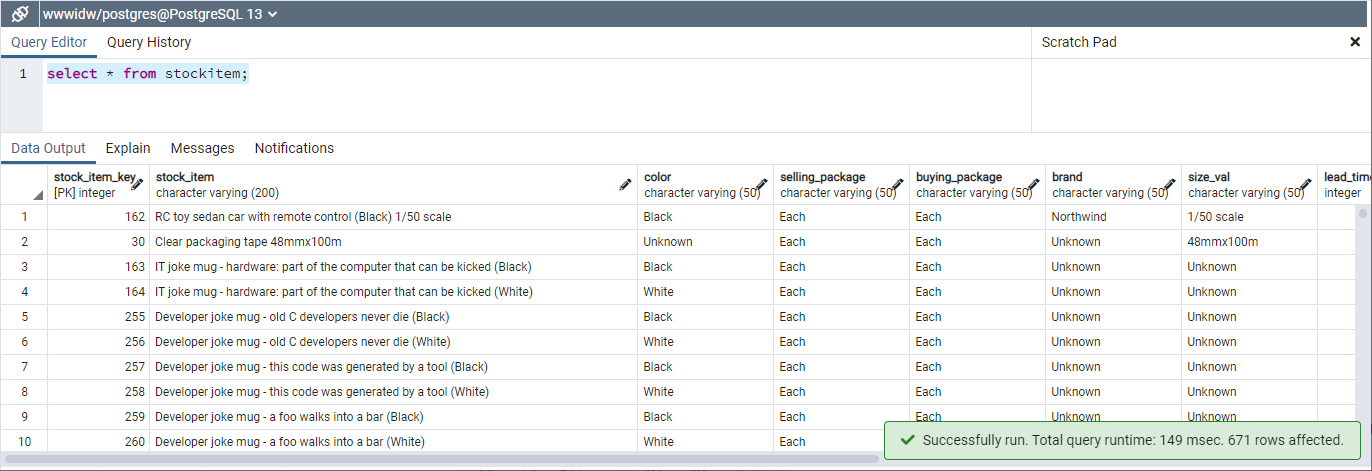
**Consultas para Customer:**

****

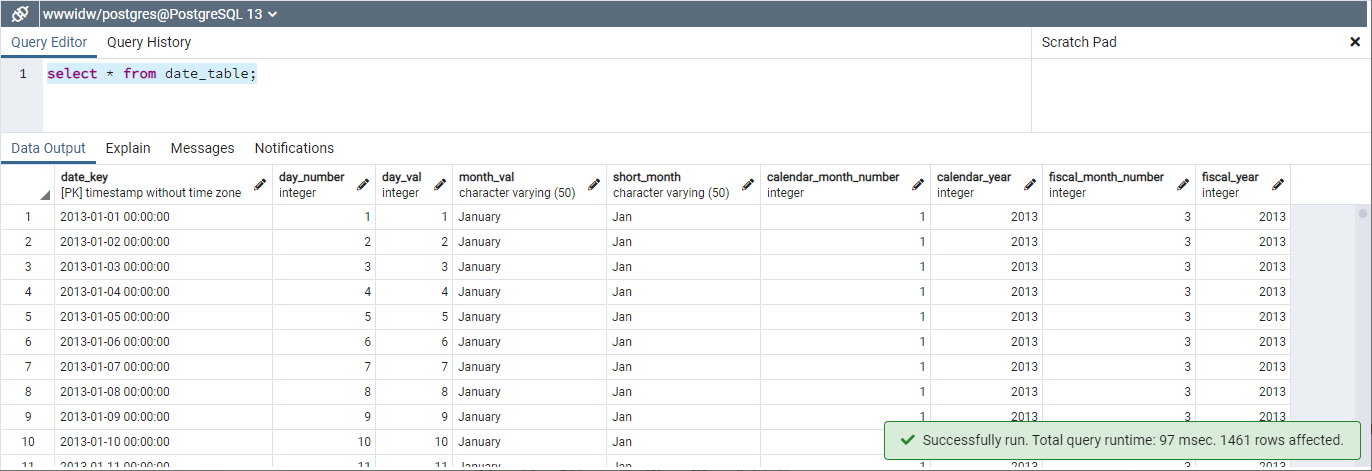
**Consultas para Employee:**

****

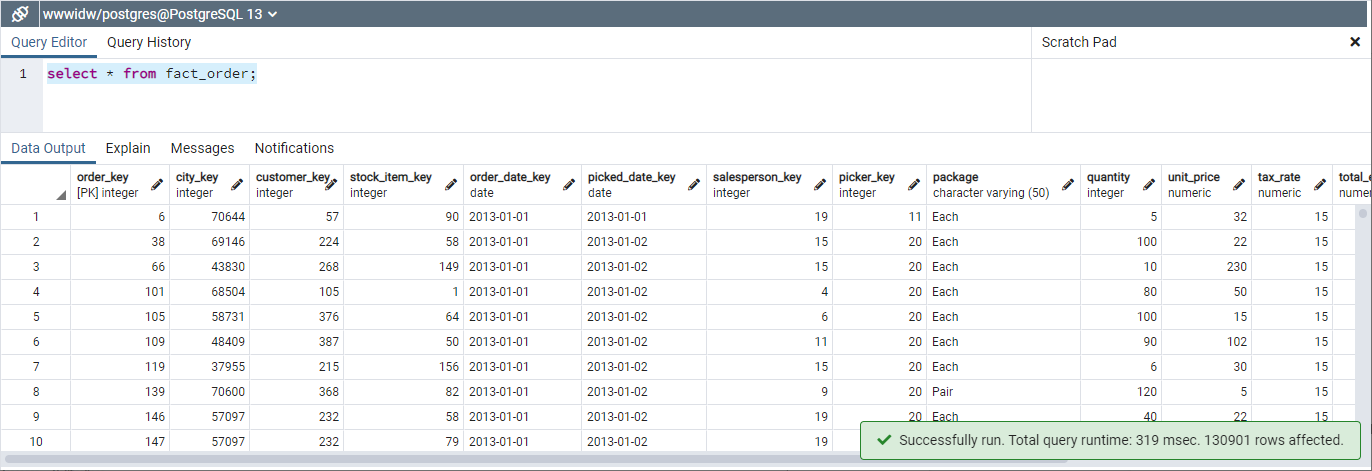
**Consultas para Stockitem:**

****

**Consultas para Date\_table:**

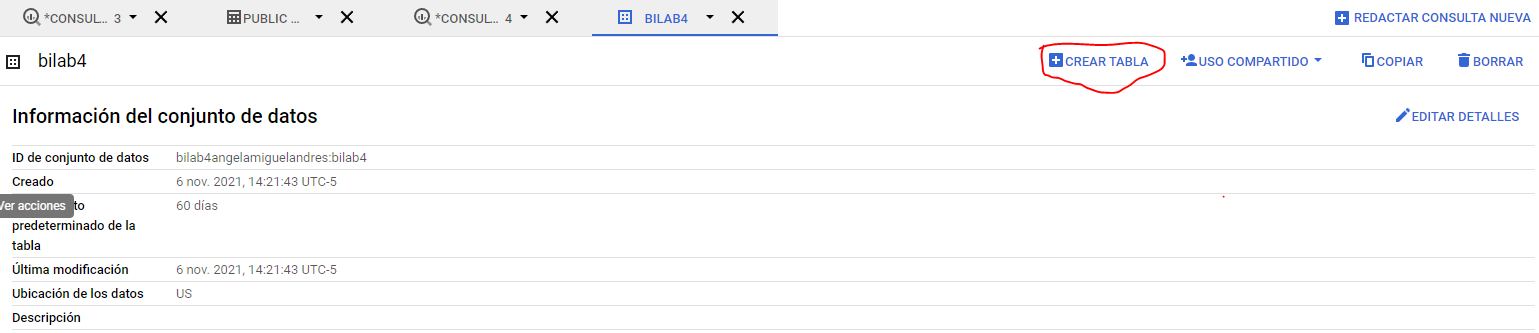
****

**Consultas para Fact\_order:**

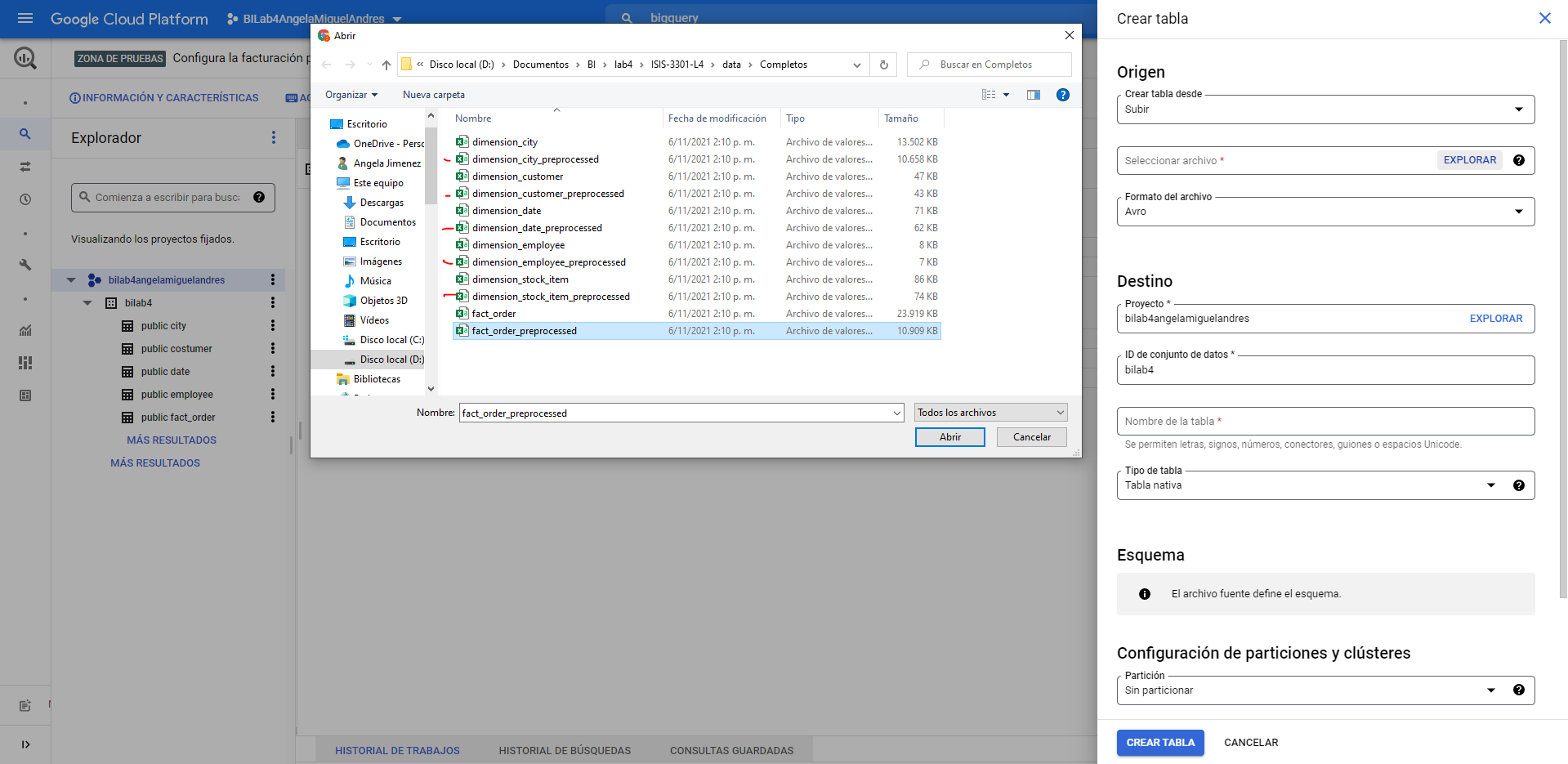
****

# Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en BigQuery

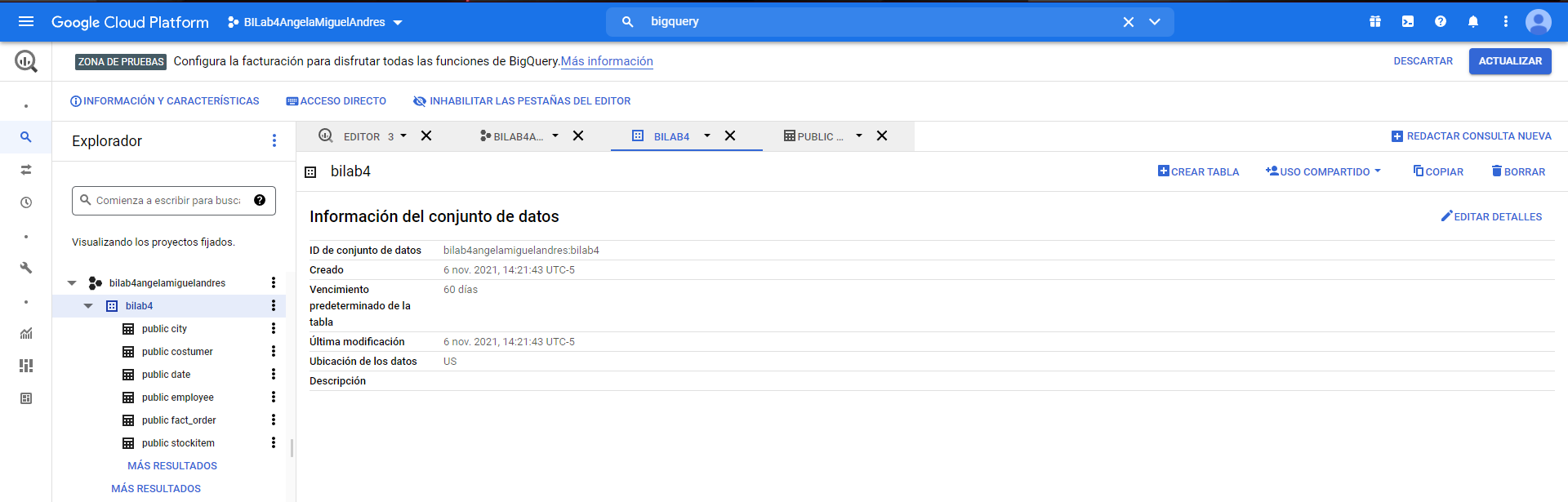
Creación de tablas:



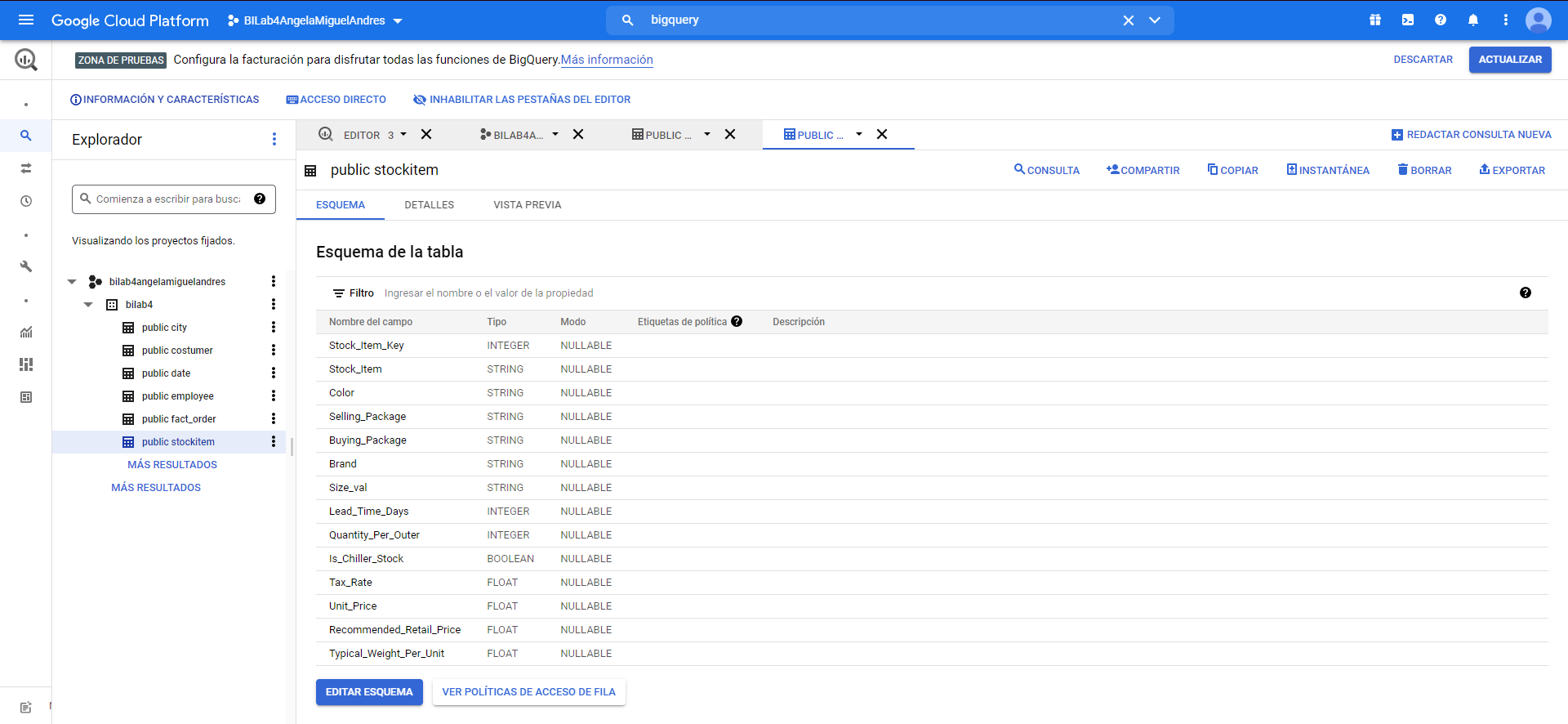
Se seleccionaron los archivos procesados de los datos



cada tabla fue creada:



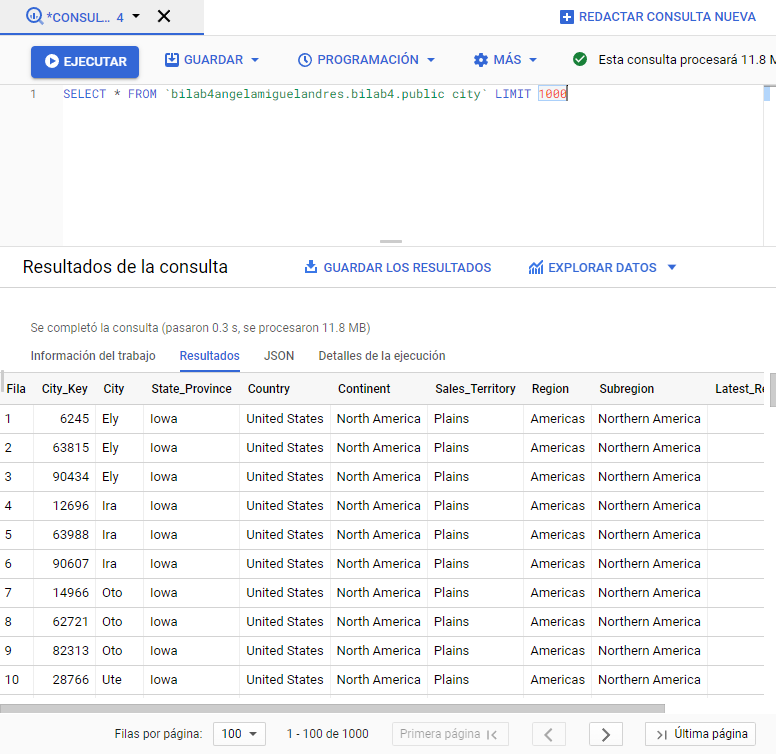
Esquema de la tabla stockitem:

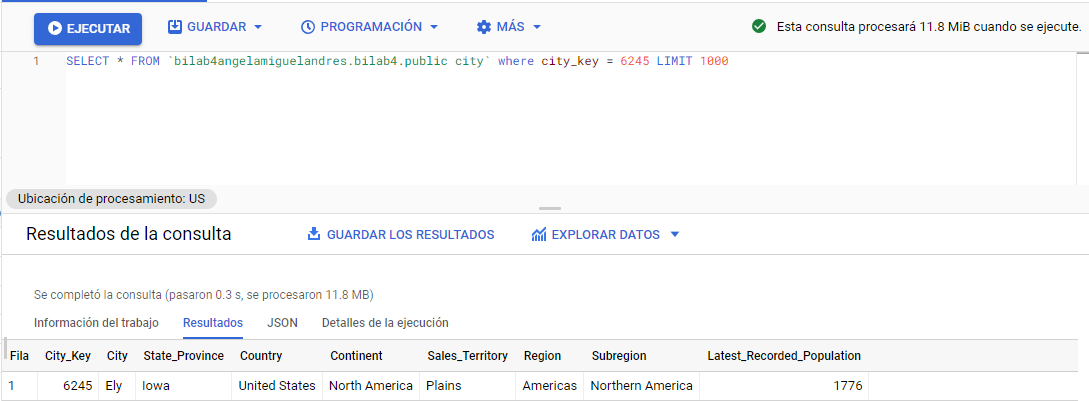




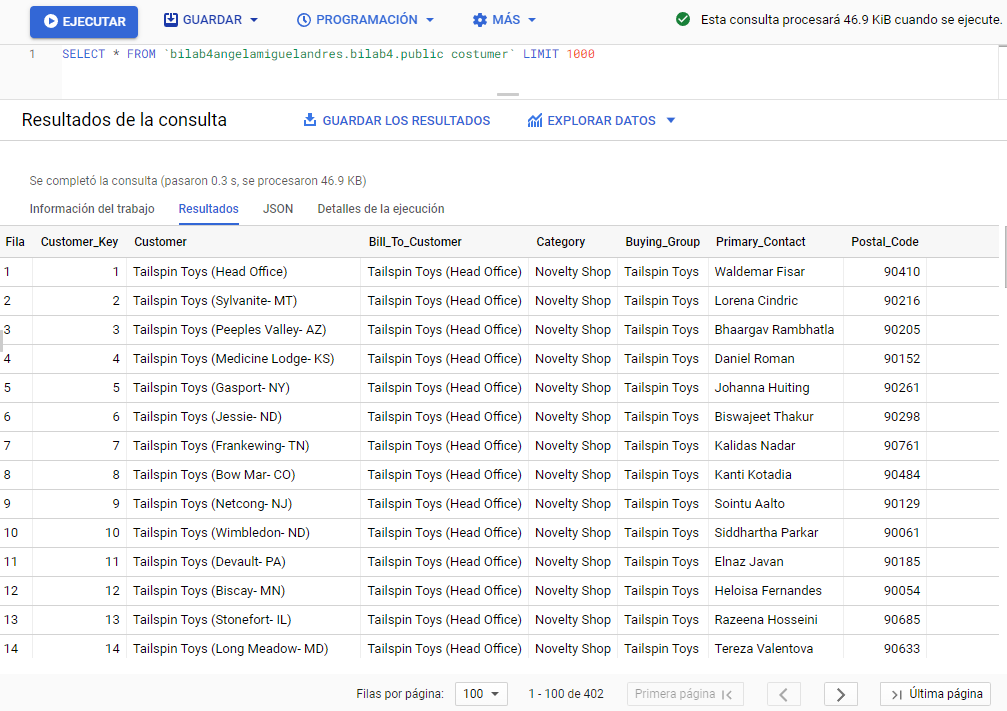
# Resultados BigQuery

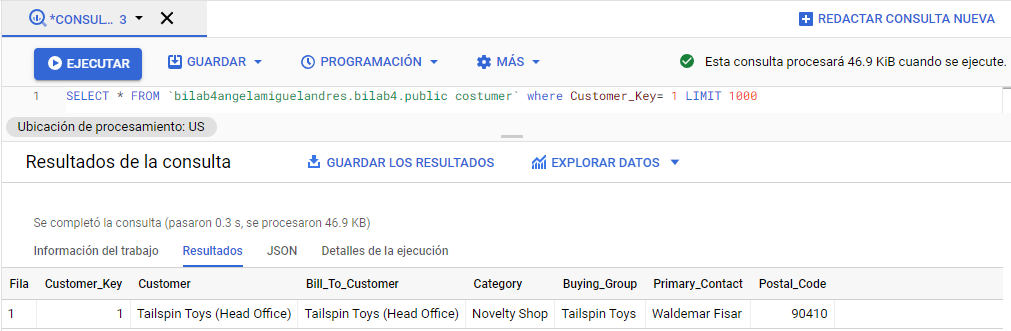
**Consultas para City:**

****

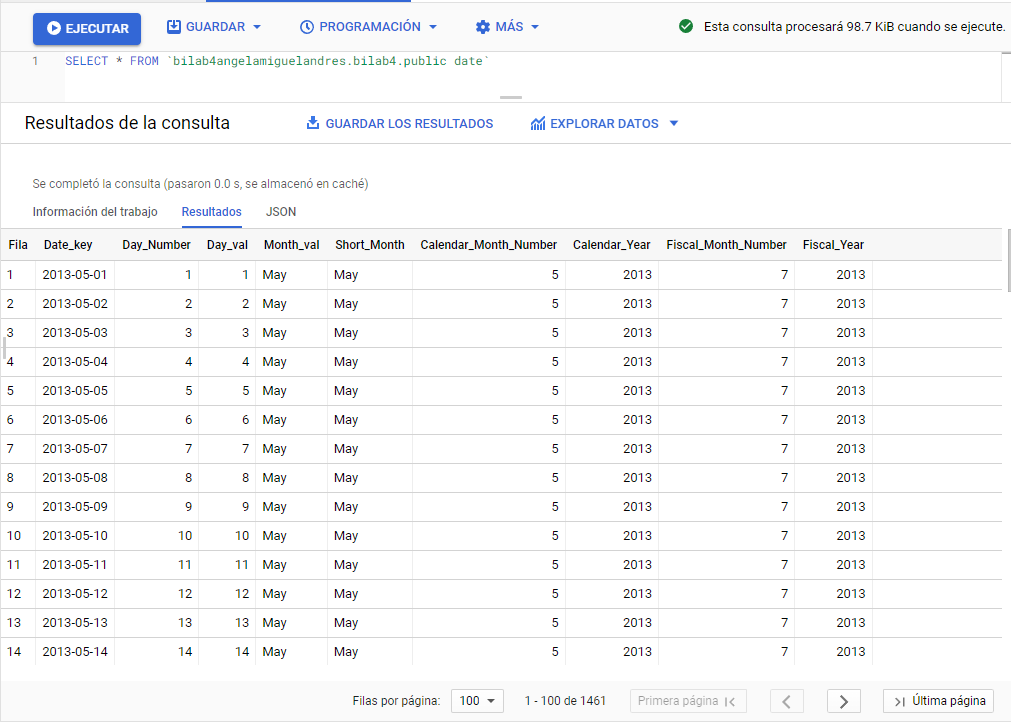
****

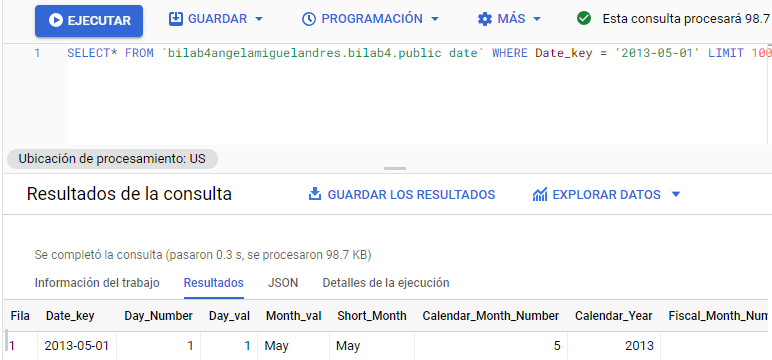
**Consultas para Customer:**

****

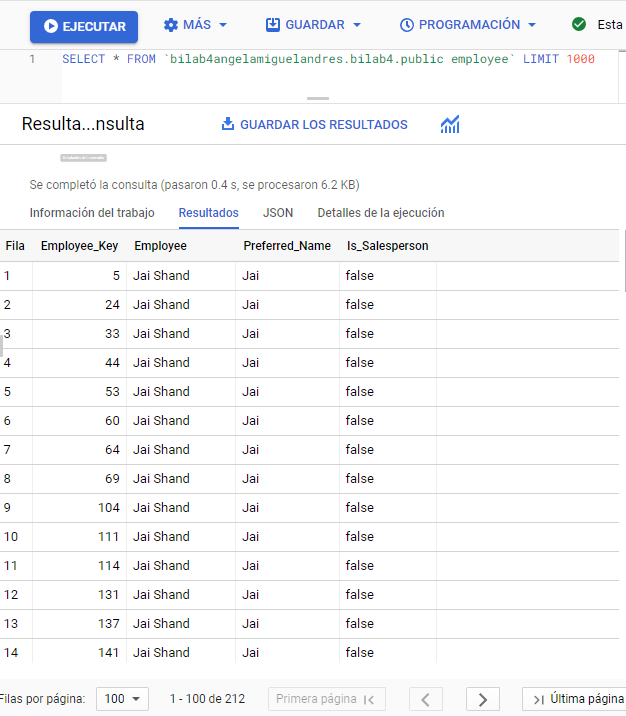
****

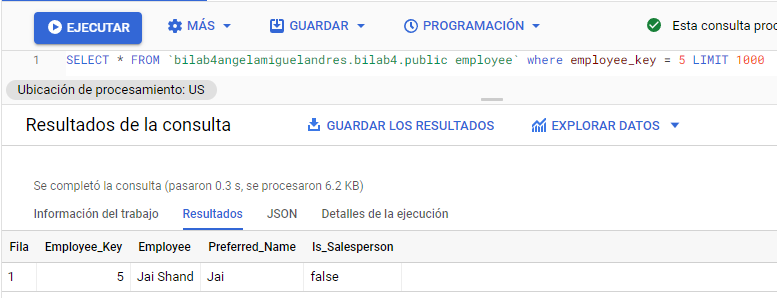
**Consultas para Date:**

****

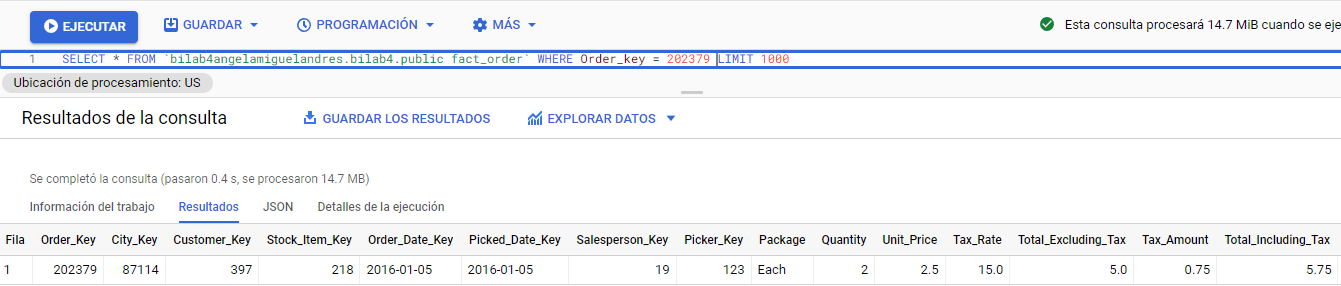
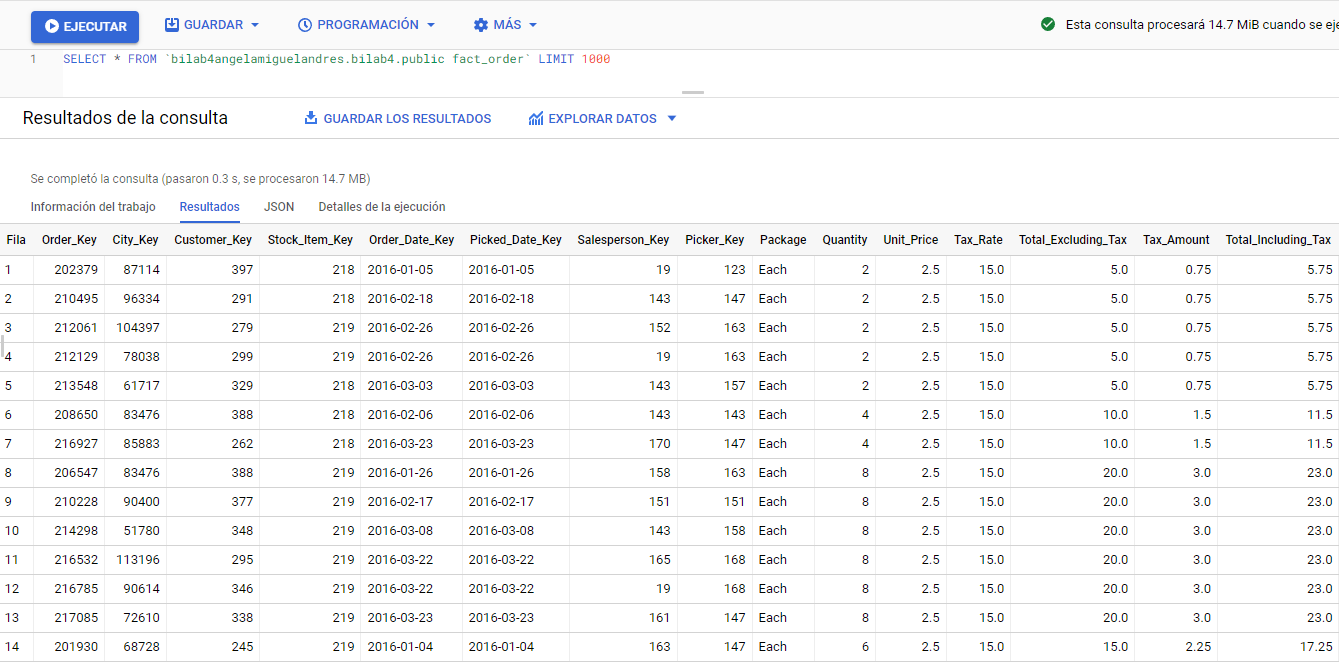


**Consultas para Employee:**

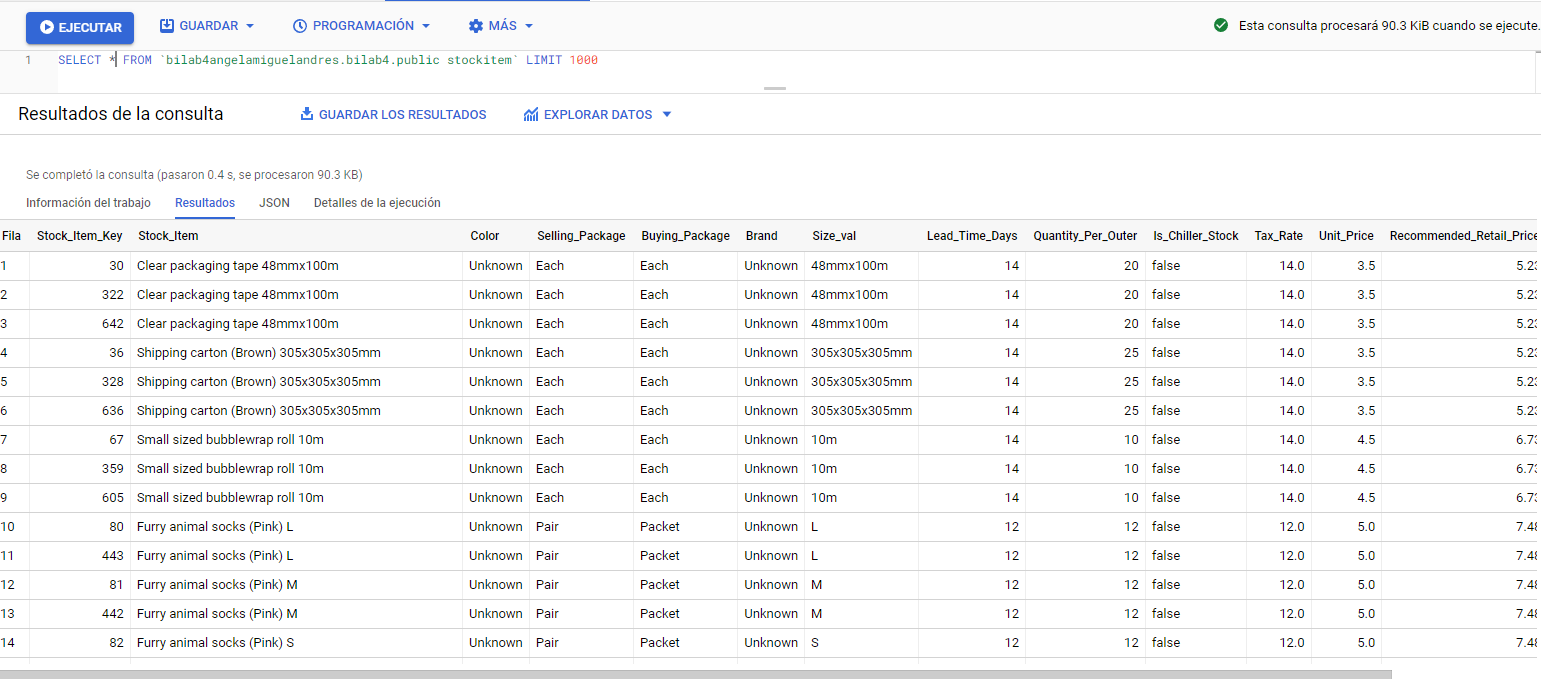


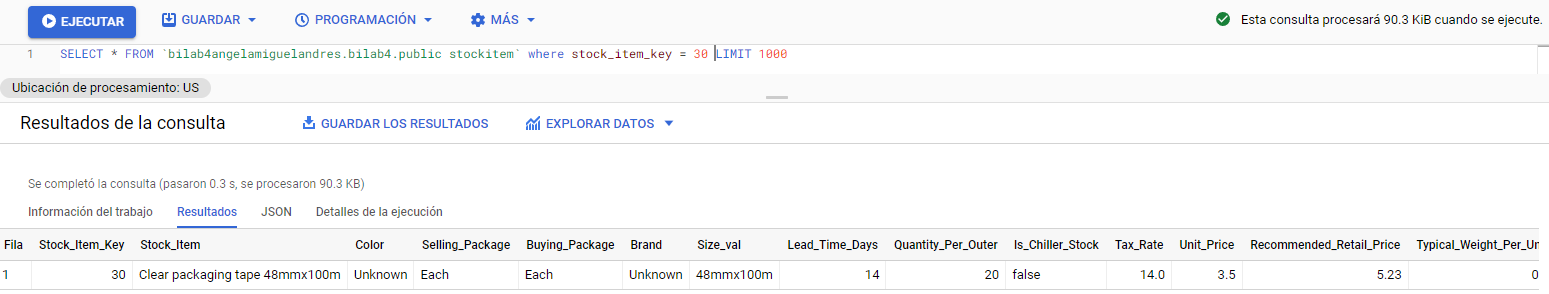
****

**Consulta para fact\_order:**

****

**Consulta para stockitem:**

****

****

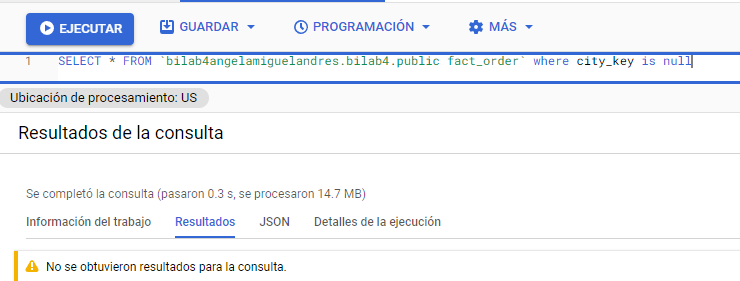
**Validación de restricciones de llave foránea en fact\_orders:**

Para este proceso se definieron 2 pasos:

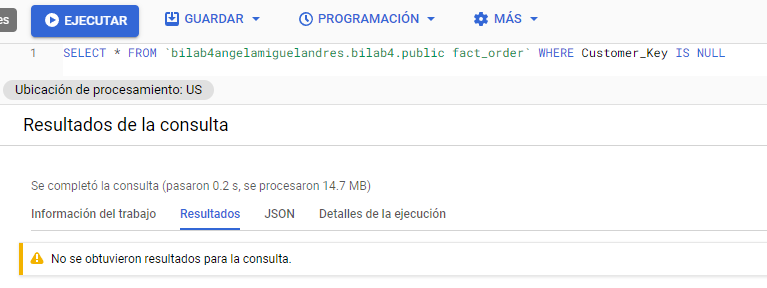
1. **La columna no tiene nulls**

Esta verificación se hará para todas las relaciones. Para el primer paso se hace una query que nos da los nulos de cada Fk, la idea es que este valor sea vacío

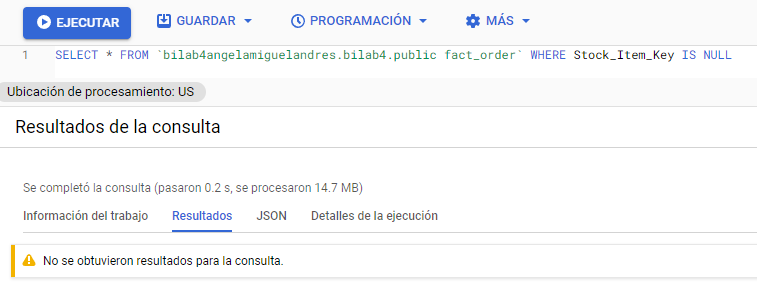
**City\_Key(public city FK)**

****

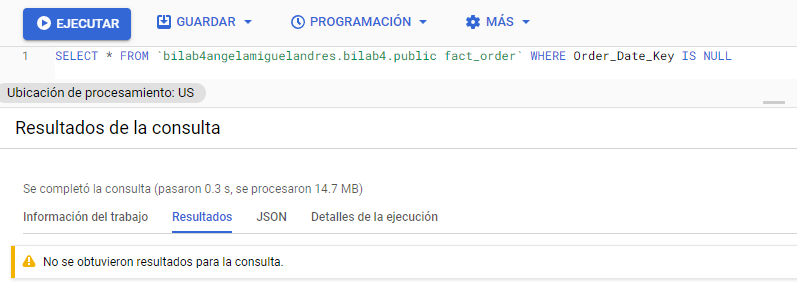
**Customer\_Key(public costumer FK)**

****

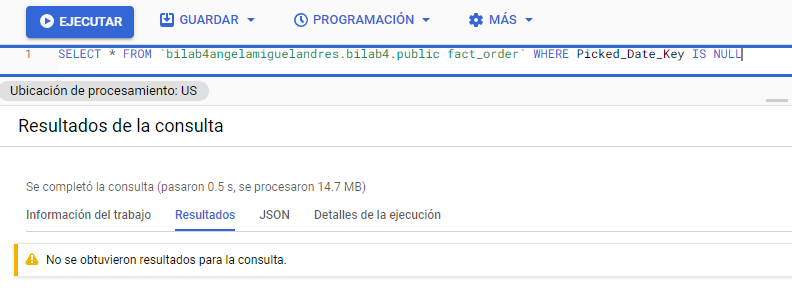
**Stock\_Item\_Key(public stockitem FK)**

****

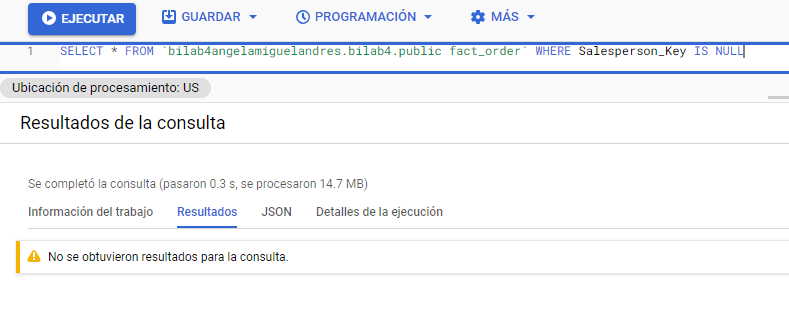
**Order\_Date\_Key(public date FK)**

****

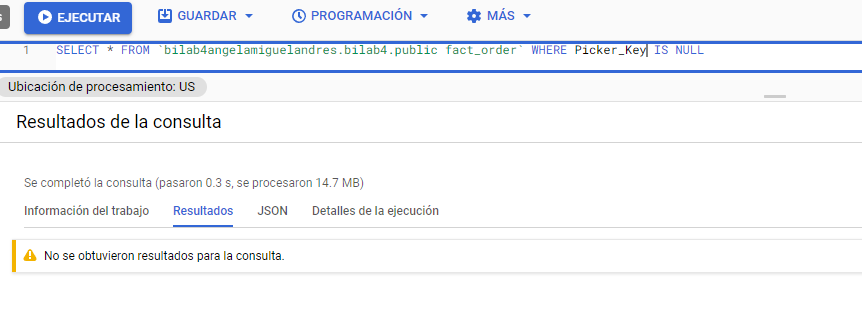
**Picked\_Date\_Key(public date FK)**

****

**Salesperson\_Key(public employee FK)**

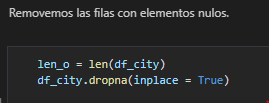
****

**Picker\_Key(public employee FK)**

****

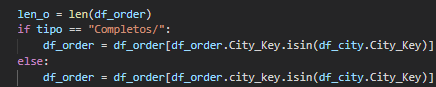
1. **Existe la FK en PK de la dimensión**

Para esta parte nos basaremos en el bono, en el cual para limpiar los datos se eliminaron los nulos:

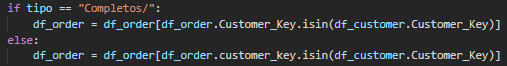


Y se usaron solo los datos que contarán con una llave foránea que exista en las diferentes dimensiones:

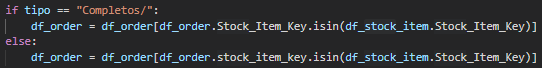
**City\_Key(public city FK)**



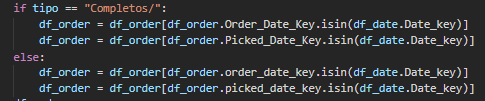
**Customer\_Key(public costumer FK)**

****

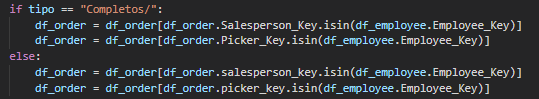
**Stock\_Item\_Key(public stockitem FK)**

****

**Order\_Date\_Key(public date FK) y Picked\_Date\_Key(public date FK)**

****

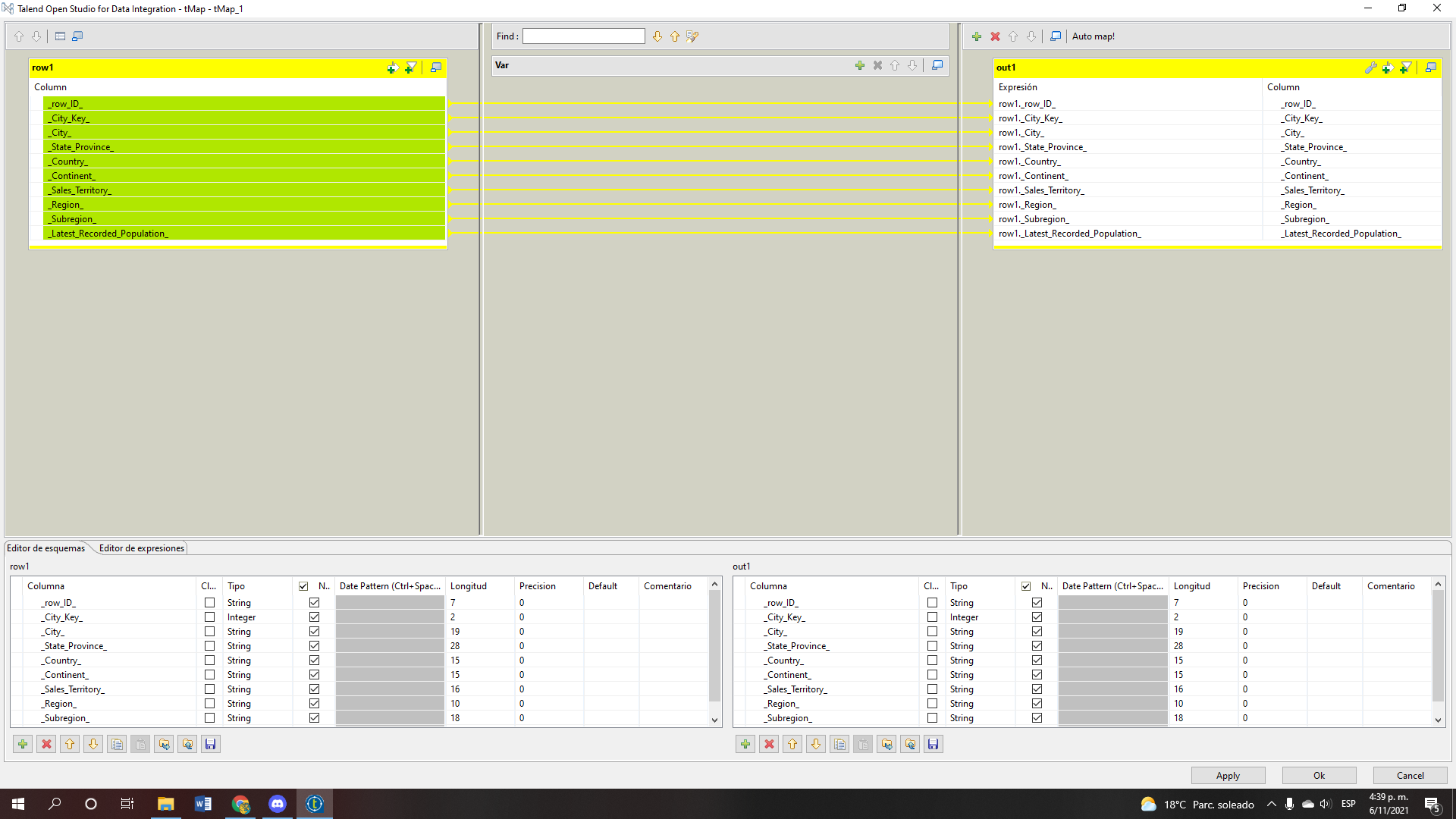
**Salesperson\_Key(public employee FK) y Picker\_Key(public employee FK)**

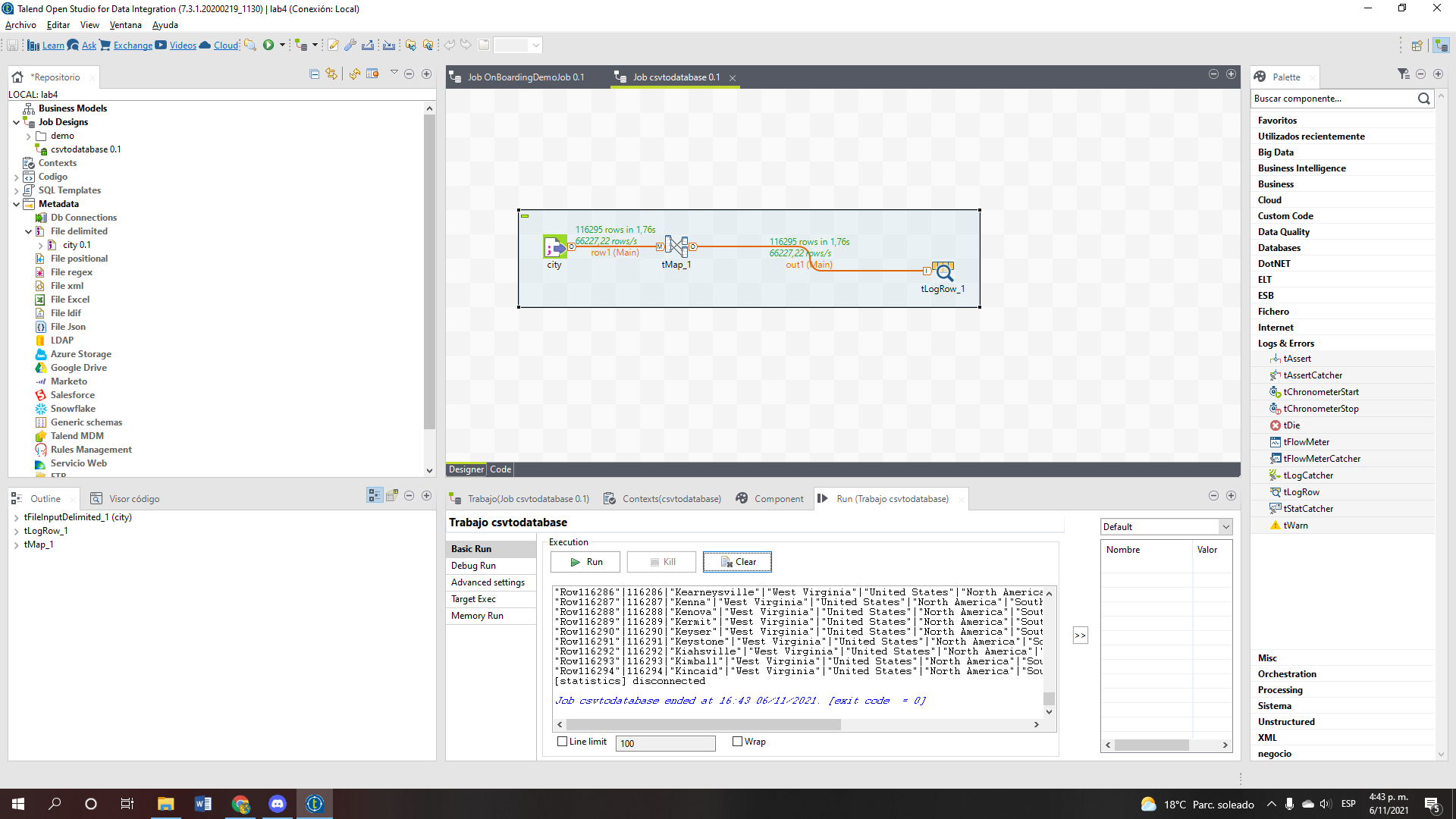


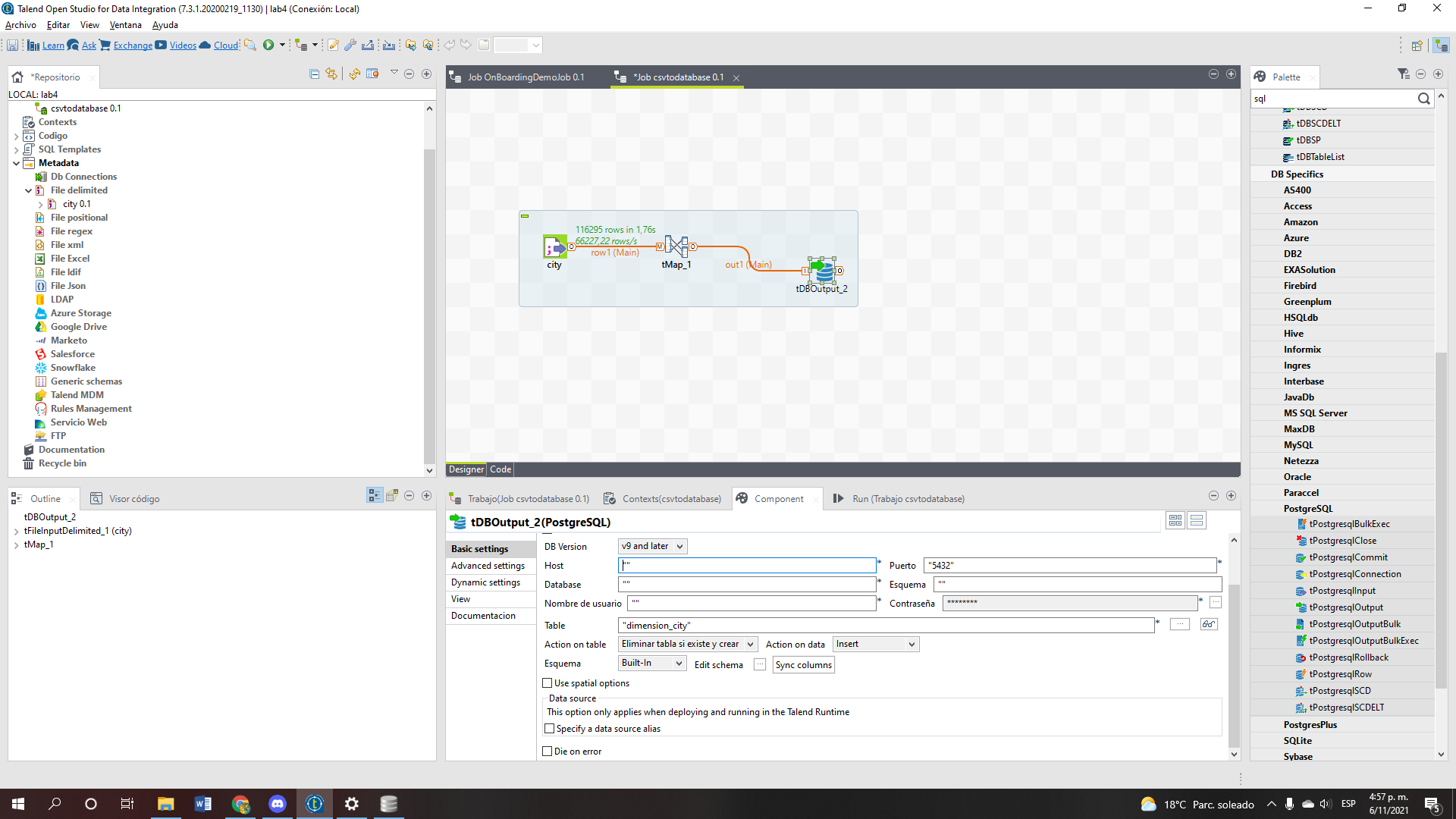
De esta forma estamos seguros de que esta condición se aplica en todas las llaves foráneas.

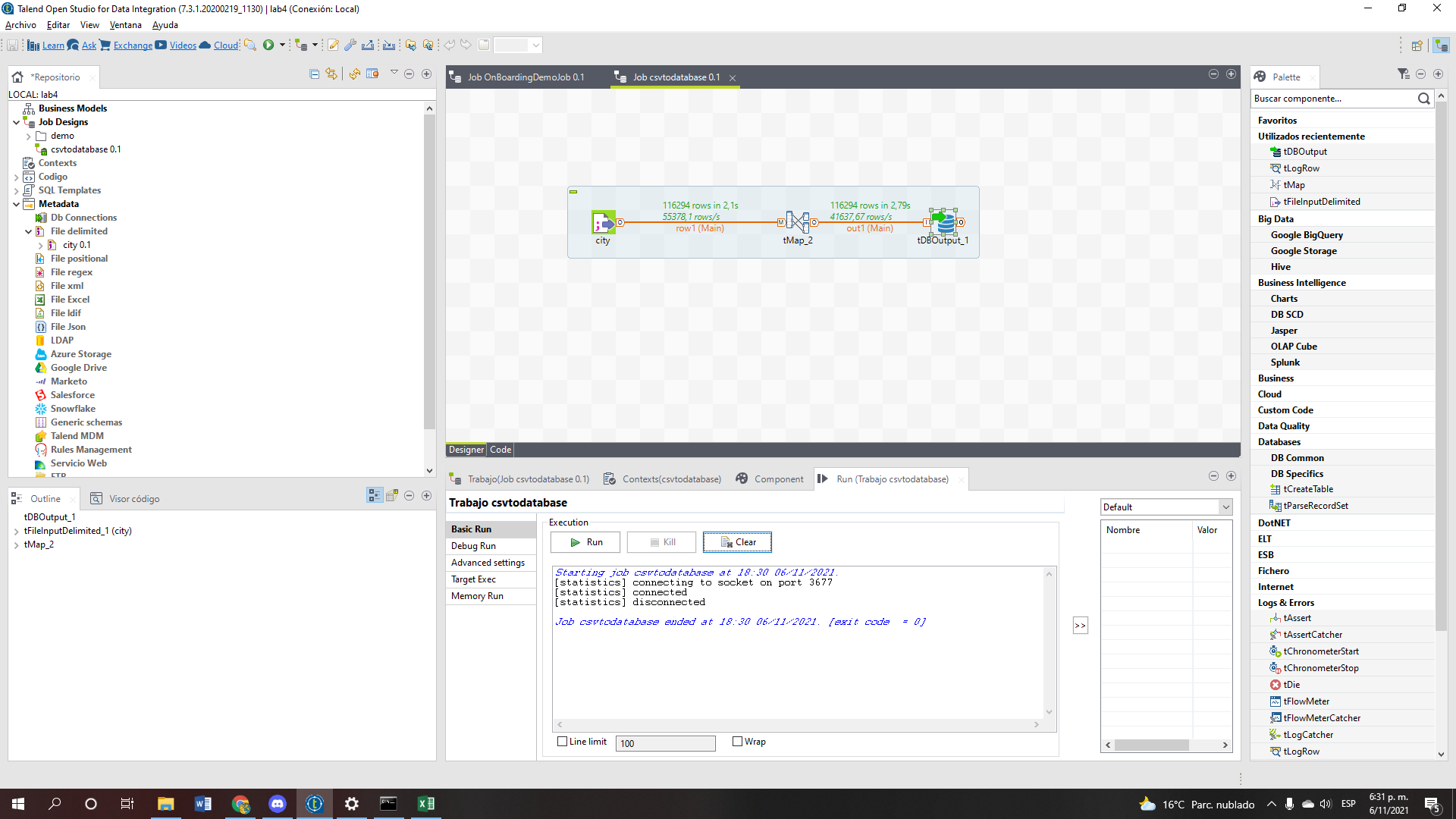


# Documentación del proceso y las transformaciones realizadas - Modificaciones a los esquemas en la 3ra herramienta





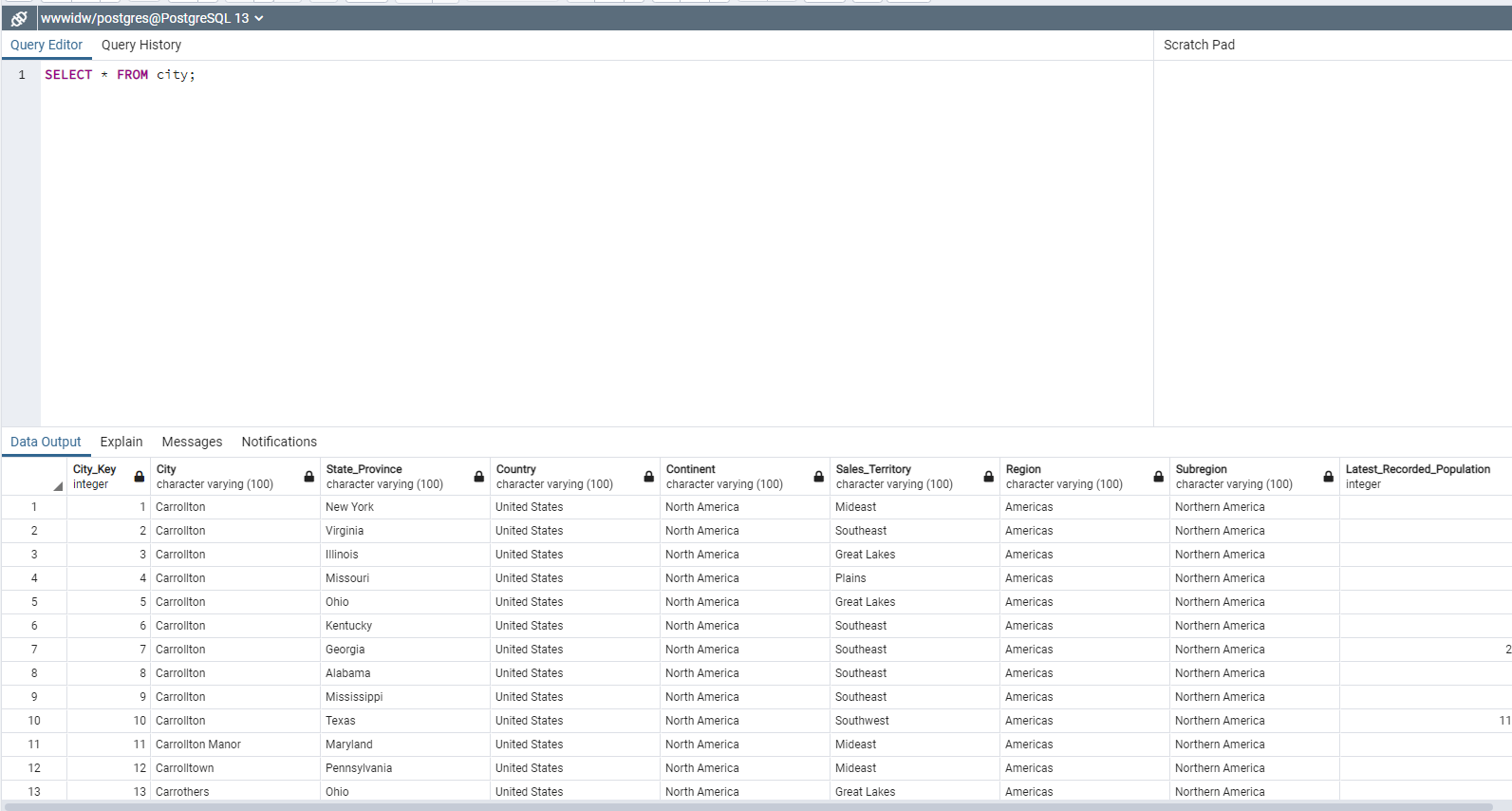




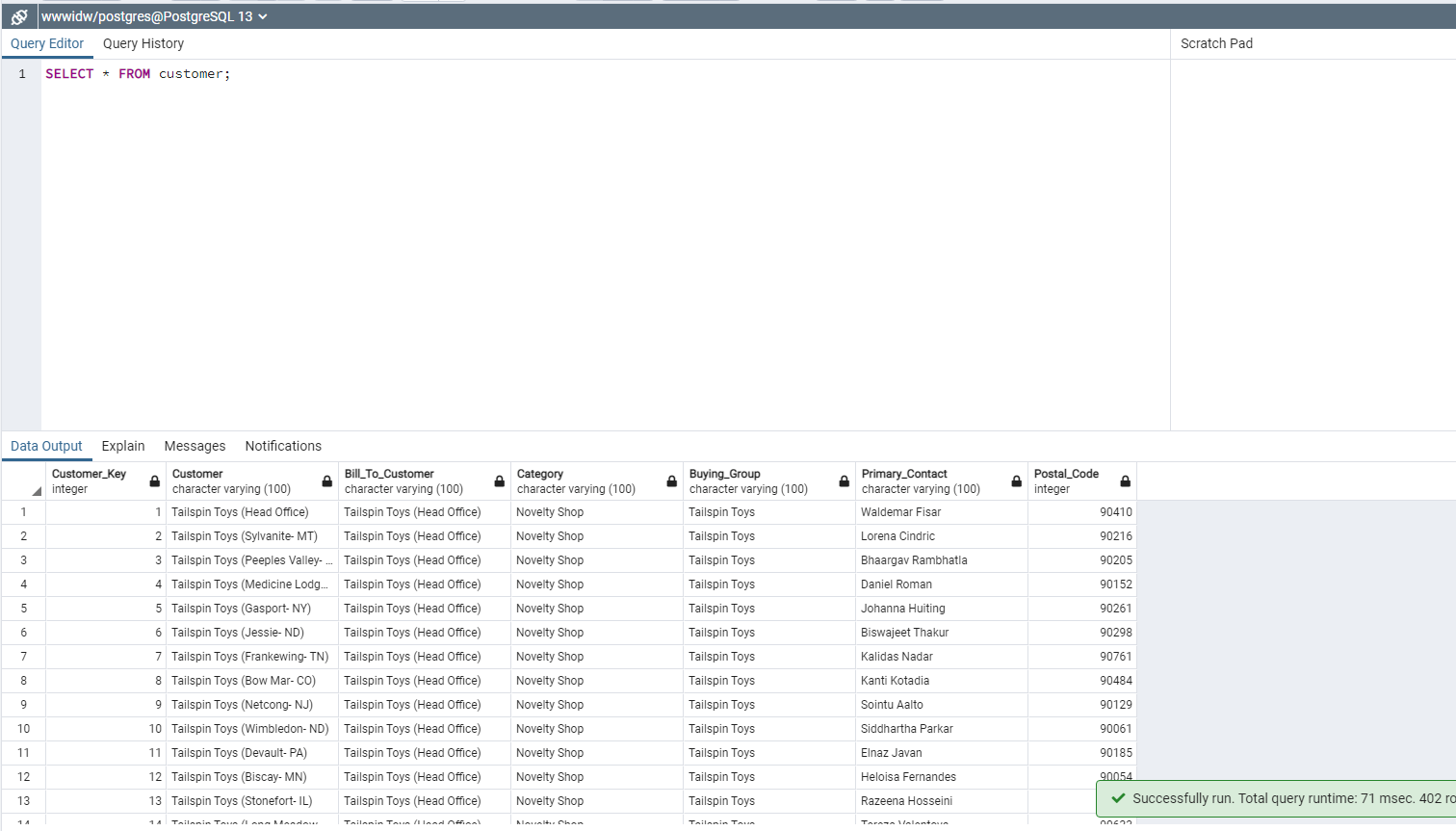


# Resultados 3ra herramienta

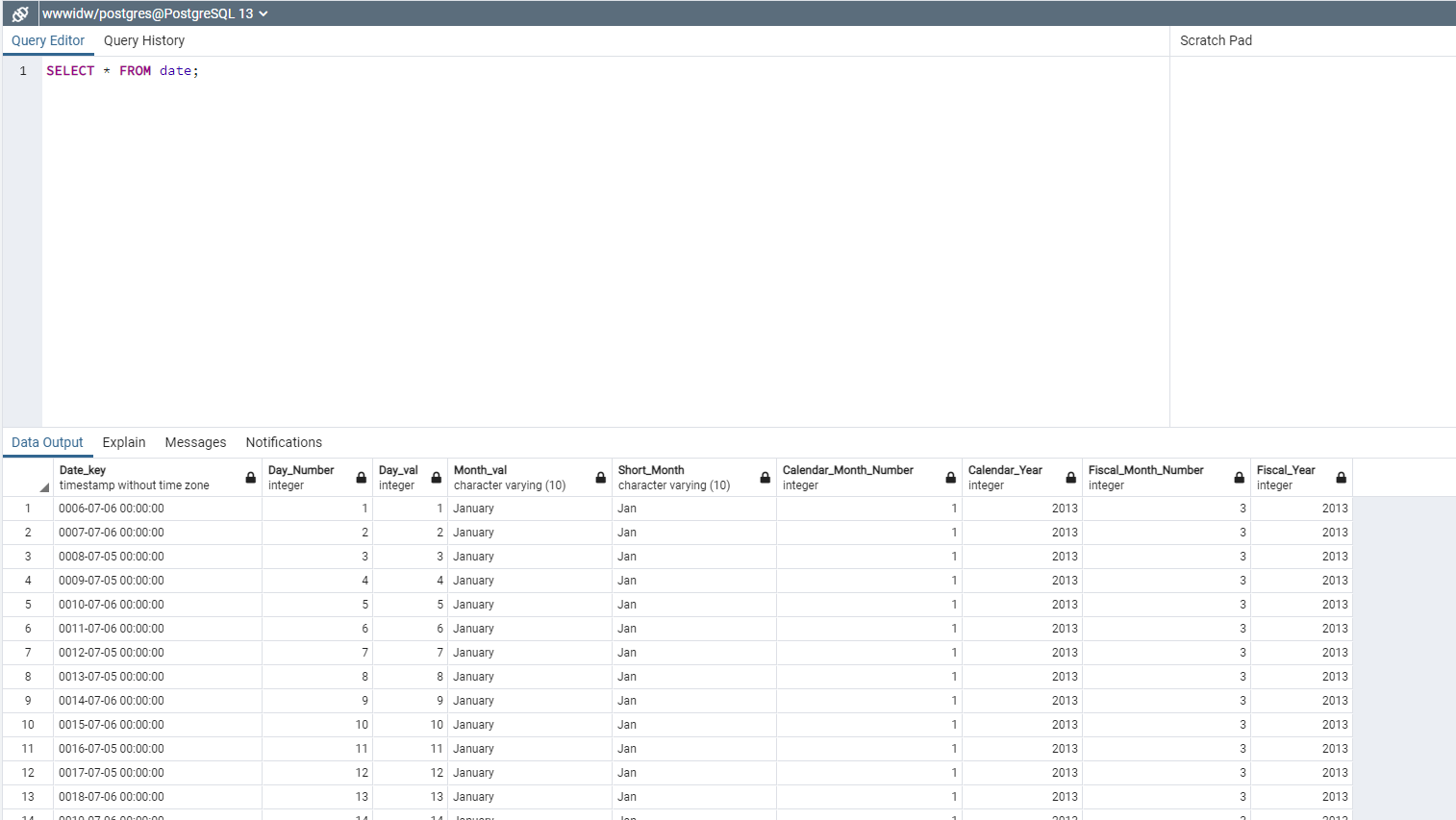
**Dimensión City:**



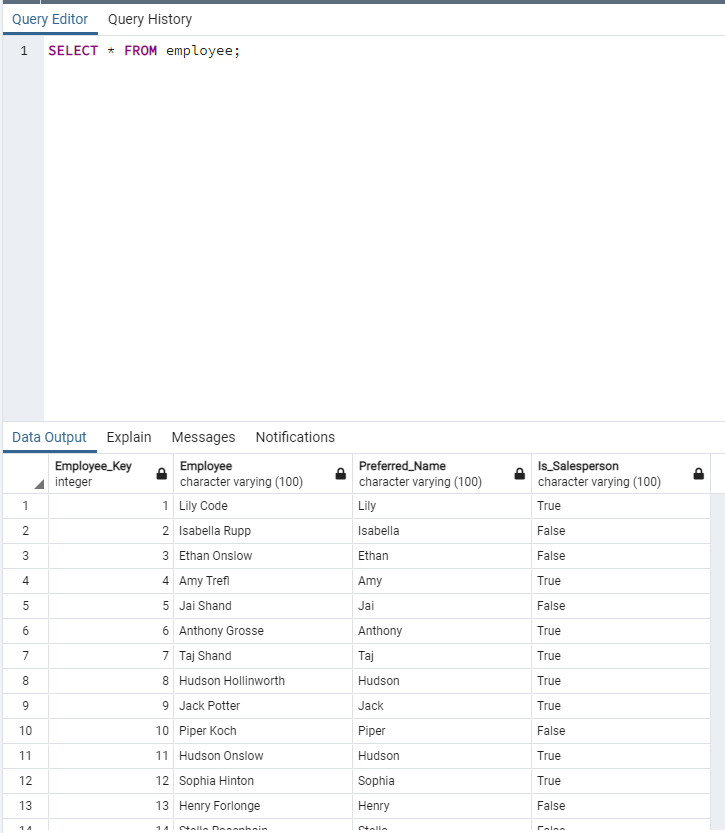
**Dimensión Customer**

****

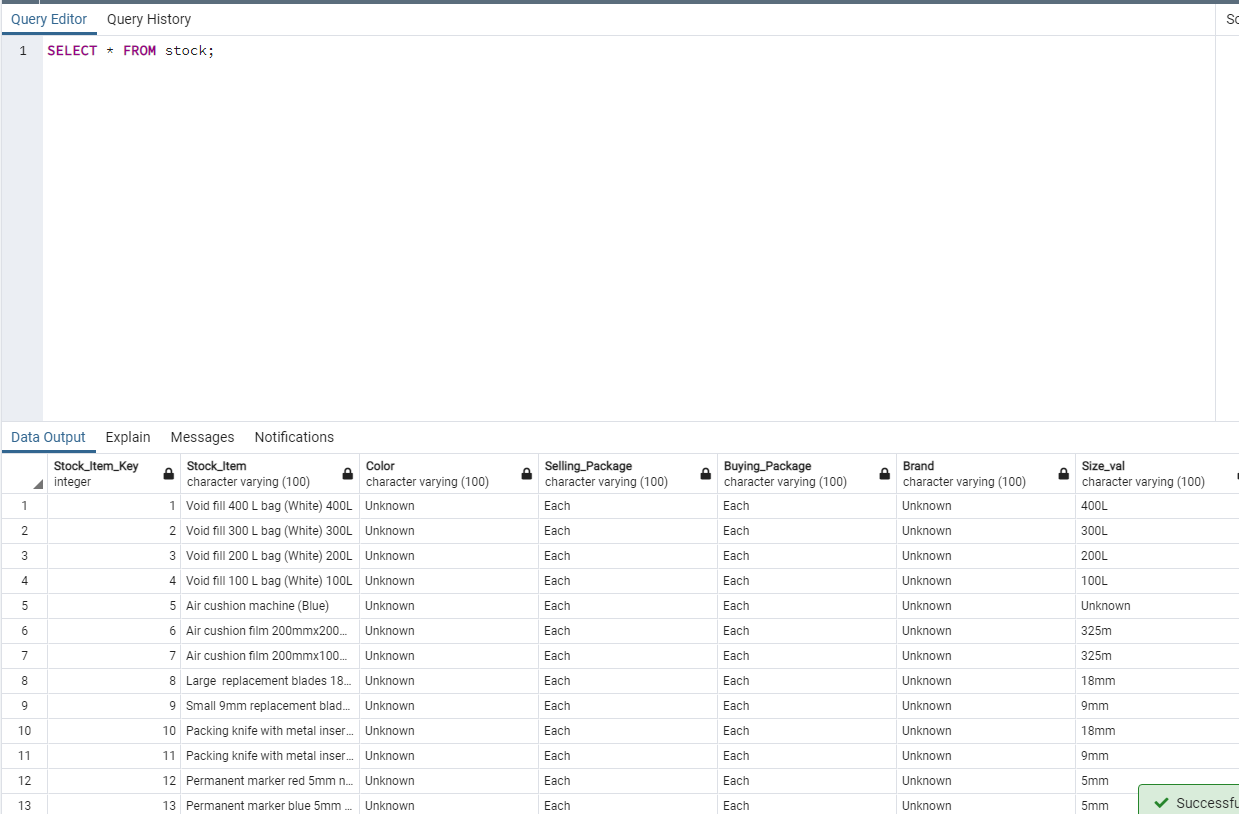
**Dimensión Date**

****

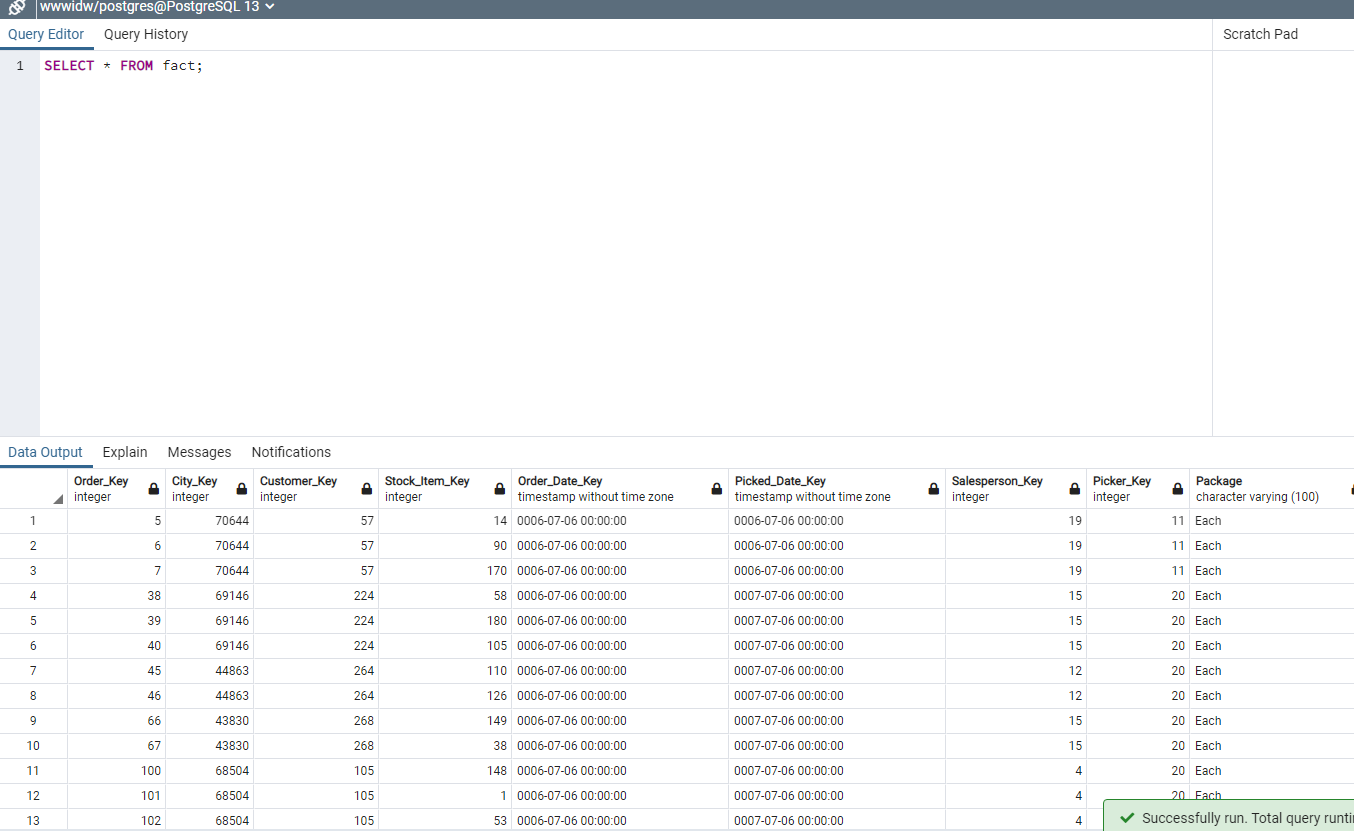
**Dimensión Employee**

****

**Dimensión Stock**

****

**Dimensión Order**

****



# Respuestas a las preguntas del laboratorio



**¿Por qué se utiliza el comando “IF NOT EXISTS” en la sentencia, en el contexto del proceso de ETL?**

R: Porque estamos desarrollando una herramienta que pueda importar datos para realizar un proceso de ETL para exportarlos a nuestro datamart, e ir acumulando y actualizando información, para que esto ocurra, la herramienta debe poder ejecutarse una y otra vez, y esto implica que cada vez que se ejecute, la herramienta debe trabajar con lo que haya en el datamart en ese momento y es muy posible que las tablas ya están creadas, a menos de que sea la primera ejecución o se haya decidido añadir una dimensión nueva.

**¿Por qué para la columna de día se utiliza el nombre day\_val y no “day”?**

Day es una palabra reservada para una función de sql que retorna el día de una fecha seleccionada.

**¿De dónde se obtiene la información sobre las columnas que hay que crear en la tabla?**

Al menos en pentaho spoon se utiliza directamente un script para creación de tablas, en BigQuery se saca del csv y en Talend se saca del csv también.

**¿Cuál es la diferencia entre un ‘job’ y una ‘transformación’?**

Un job tiene que ver con el flujo de control, es decir cómo establecer una serie de tareas y establecer un orden para su ejecución. Las transformaciones son una serie de tareas que implican la transformación de datos y pueden ejecutarse en paralelo (mientras que en los job's se ejecutan en orden).

**¿Por qué se hace uso del nodo ‘Insert/Update’ y no del ‘Table Output’?**

Porque Insert/Update como dice el nombre ingresa nuevos datos y Table Output solo inserta datos, y que solo se inserten datos es un problema (como se explicó anteriormente en la parte del bono#1 con bulkload).

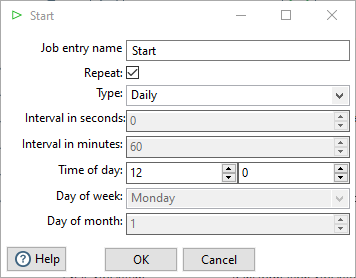
**¿Cuál es la finalidad de los campos ‘The key(s) to lookup the values:’ y ‘Update fields’ del nodo ‘Insert/Update’?**

El campo ‘The key(s) to lookup the values:’ nos sirve para que cuando estemos exportando la información a la base de datos, se revise para cada entrada que vamos a exportar y si existe una entrada en la base de datos cuyos valores correspondientes a los parámetros que le pasamos al campo antes mencionado existen, de ser así sabemos que este campo no se debe insertar, sino actualizar.

El campo ‘Update fields’ sirve para saber, para los registros que vayamos a actualizar, qué campos han de ser actualizados.

**¿Será posible programar la fecha y hora en la cual la organización puede correr de forma automática el proceso de ETL?**

La pregunta está en el contexto de la herramienta spoon, y en esta herramienta si es posible, se tendría que configurar el nodo start, como se ve a continuación:



En este ejemplo estaríamos configurando el job para ejecutarse todos los días.

# Comparación de herramientas



En principio en todas las herramientas se logró realizar el mismo proceso, sin embargo encontramos las siguientes diferencias entre los distintas herramientas

|  | Google BigQuery | Spoon Pentaho | Talend |
| --- | --- | --- | --- |
| Conexión a la DB | Se suben a un proyecto en la nube (drive) y se administra el tiempo de duración | Permite conectar a postgresql, pero a la vez a otras plataformas locales o pasandoles su dirección en red | Permite exportar los datos a diferentes tipos de bases de datos como sqlite, postgresDB, mysql, etc. |
| Creación de tablas | Se agregan las tablas, bien sea subidas o creadas desde ceros, al proyecto. | Mediante Scripts | El interfaz ofrece distintas herramientas para crear las tablas, aunque también se pueden crear utilizando el lenguaje java. |
| Job’s | No maneja el concepto de job. | Crea un job inicial de donde parte el flujo de ejecución. | Utiliza el concepto de job para organizar un set de instrucciones que van a darle forma a la información. |
| Transformaciones | No maneja el concepto de transformaciones. | Es necesario crear múltiples transformaciones para desarrollar el proyecto. | No maneja el concepto de transformaciones. |

A partir de esto podemos concluir que talend y pentaho son las que más opciones nos dan como herramientas de ETL, sin embargo Google BigQuery nos ofrece la ventaja de que podemos trabajar en la nube.