Sistemas Inteligentes II

Universidad de Las Palmas de Gran Canaria

Trabajo de curso – Inteligencia de Negocio.

Curso 2018/19

Christian Brito Ramos
David Alberto Medina Medina

Índice

1.-Objetivo del trabajo.

El objetivo de este trabajo es diseñar e implementar un proceso ETL que construya un cubo de datos en formato estrella para el análisis de tweets. Para ello se accederá al sitio de Twitter y se descargará la información de los tweets haciendo uso de la API que pone a disposición de los desarrolladores esta entidad. El cubo de datos se implementará siguiendo el modelo dimensional con estructura en estrella con una tabla de hechos y tres tablas de dimensiones. Usaremos Pentaho Data Integration, Weka y Java.

1.1.-Notas de Implementación.

A continuación, se describe las especificaciones e instrucciones de como llevar a cabo el objetivo planteado por la docencia de la asignatura.

Obtención y limpieza de datos

Los datos necesarios para realizar esta práctica se obtendrán accediendo mediante una aplicación Java. Esta aplicación permitirá recuperar tweets sobre diferentes temas que decida el usuario, pero filtrando solo aquellos tweets que contengan información sobre la localización. Debido a que el número de tweets que se pueden recuperar cada día está limitado por la plataforma, los datos se deberán recuperar en varios días por lo que será necesario realizar un proceso de eliminación de duplicados. El cubo creado deberá contener información al menos de 1000 tweets.

Estructura del cubo OLAP

Como se indicó anteriormente el cubo estará formado por una tabla de hechos y tres tablas de dimensiones que permitiría el análisis del número de tweets según su información temporal, el tipo de tweet y la ubicación del usuario. Por tanto, las tablas de dimensiones que se deben crear serán las siguientes:

- Dimensión temporal: Tabla que contiene la información temporal de la creación del tweet con una granularidad diaria, y con las siguientes jerarquías: Año → Trimestre → Mes → Día; y Año → Semana → Día de la semana.
- Dimensión tipo de tweet: Tabla que contendrá información sobre los tres tipos de tweets: original, respuesta o retweet.
- Dimensión ubicación: Tabla que contendrá la información relativa a la ubicación de la cuenta desde la que se escribió el mismo. La jerarquía será: Región → Subregión → País.

La tabla de hechos debe contener los siguientes campos:

- Día del acceso. Será el campo por el que se indexa la tabla de dimensión temporal.
- Código del país. Será el campo para indexar la tabla correspondiente a la dimensión localización.
- Tipo tweet. Será el campo por el que se indexa la tabla de dimensiones tipo de tweet.
- Tamaño del tweet en caracteres.
- Número de "me gusta" (likes) que ha recibido el tweet.
- Número de veces que se ha "retweeteado".

Implementación con Pentaho Data Integration

La implementación se realizará utilizando la herramienta ETL Pentaho Data Integration. El resultado será un trabajo (job) que controle el flujo y compruebe las condiciones de error que se puedan dar, e incorpore las diferentes transformaciones necesarias para realizar el trabajo.

2.-Implementación.

2.1.- Aplicación Java para la recopilación de datos.

Como ya se ha especificado anteriormente, la finalidad de esta aplicación en java es obtener datos de Twitter. El uso de la aplicación se basa en introducir un hashtag, recuperar la mayor cantidad de tweets que contengan ese hashtag y por último, tenemos la opción de seguir buscando con otros hashtags o exportar los tweets a un fichero.

El formato del fichero es CSV y los campos que lo componen son los siguientes:

1. Identificador de Tweet.
2. Fecha y hora de la creación del Tweet.
3. Código del país.
4. Tipo de tweet.
5. Tamaño del texto.
6. Número de "me gusta".
7. Número de retweets.

La interfaz de la aplicación la podemos ver en la *Ilustración 2-1*.

Se compone de un campo de texto donde se introducirá el hashtag que se quiere emplear para la búsqueda de Tweets y también se hace uso de un área de texto donde se irá mostrando el aspecto final del CSV de tal manera que a medida que se vayan realizando búsquedas, se irán añadiendo filas con los datos relevantes para el trabajo de los resultados factibles obtenidos.

Finalmente, se hace uso de dos botones para comenzar la búsqueda y para guardar los datos en un fichero por medio del explorador de archivos haciendo uso de un JFileChooser.

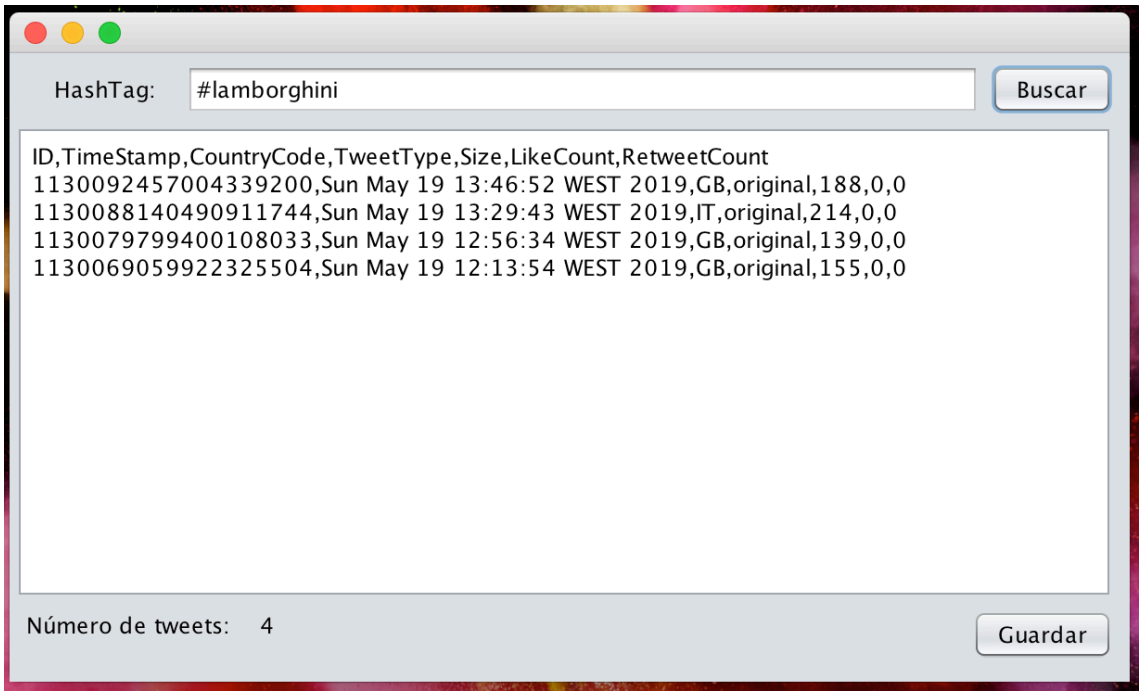

*Ilustración 2-**¡Error! No hay texto con el estilo especificado en el documento.**-1. Interfaz Java TwitterGetter.*

2.2.- Pentaho Data Integration.

Comenzamos mostrando una visión del Job General en la *Ilustración 2-2.*

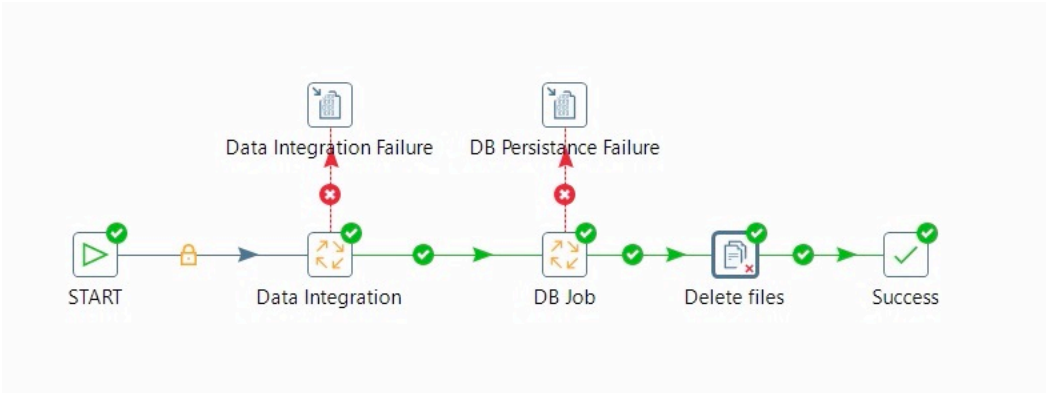

Ilustración: General_job.kjb

A continuación, mostramos el Job que realiza la integración de los datos. Ver *Ilustración 2-3.*

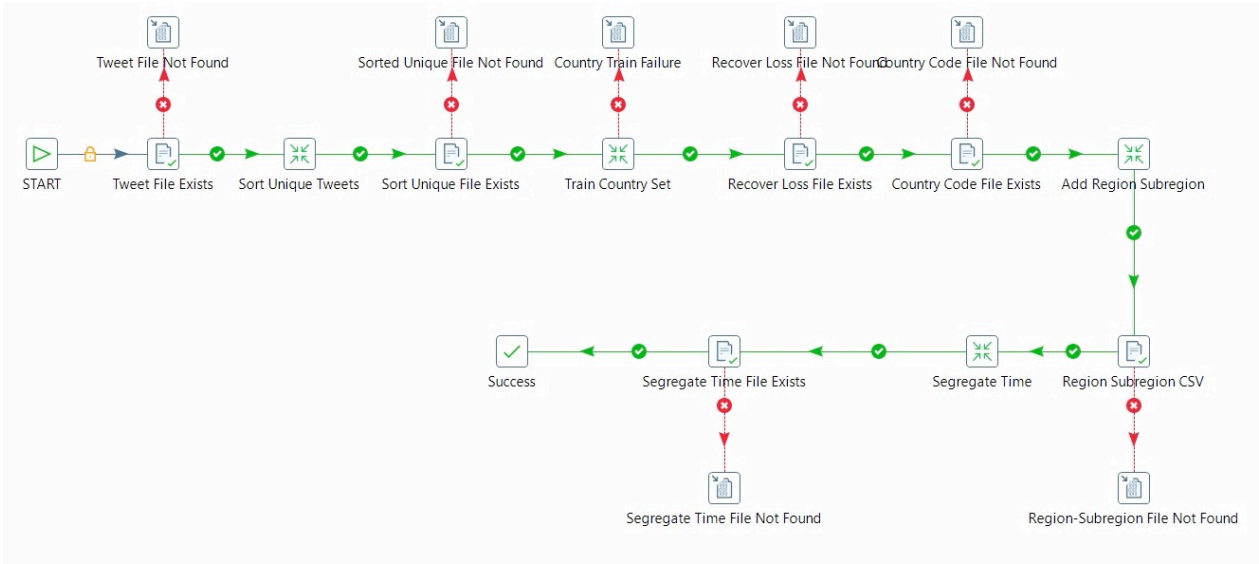

Ilustración 2-3: Data_Integration_Job.kjb

Por último, el siguiente Job es el encargado de crear la tabla de la base de datos y rellenar las tablas con los valores del CSV que obtuvimos en la aplicación java y previamente tratados del paso anterior.

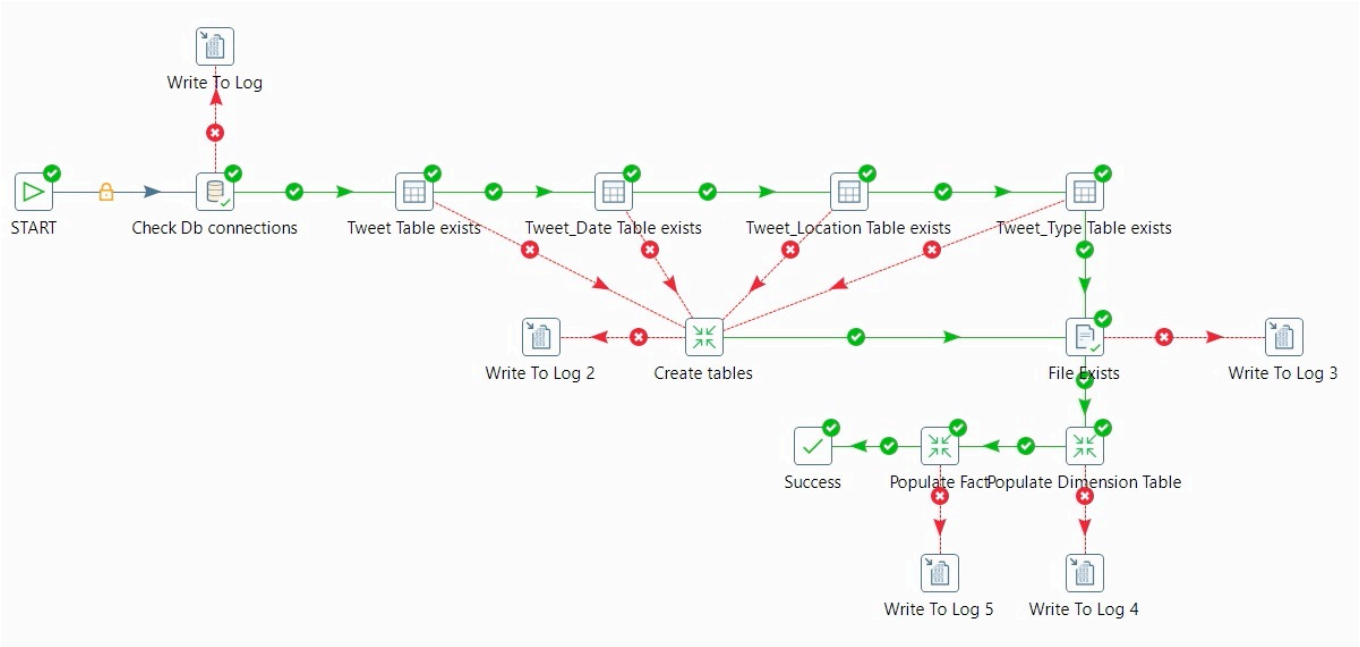

Ilustración 2-4: Db_Job.kjb

2.2.1- Data Integration Job.

En este apartado describiremos las transformaciones realizadas en el Job de la integración de datos (*Ilustración 2-3*).

Se realiza en primer lugar la lectura inicial del fichero CSV. Como puede observarse, se asigna a cada campo el tipo de dato que le corresponde. En el caso del campo TimeStamp, se le ha asignado el tipo Date atendiendo al formato del valor de su string.

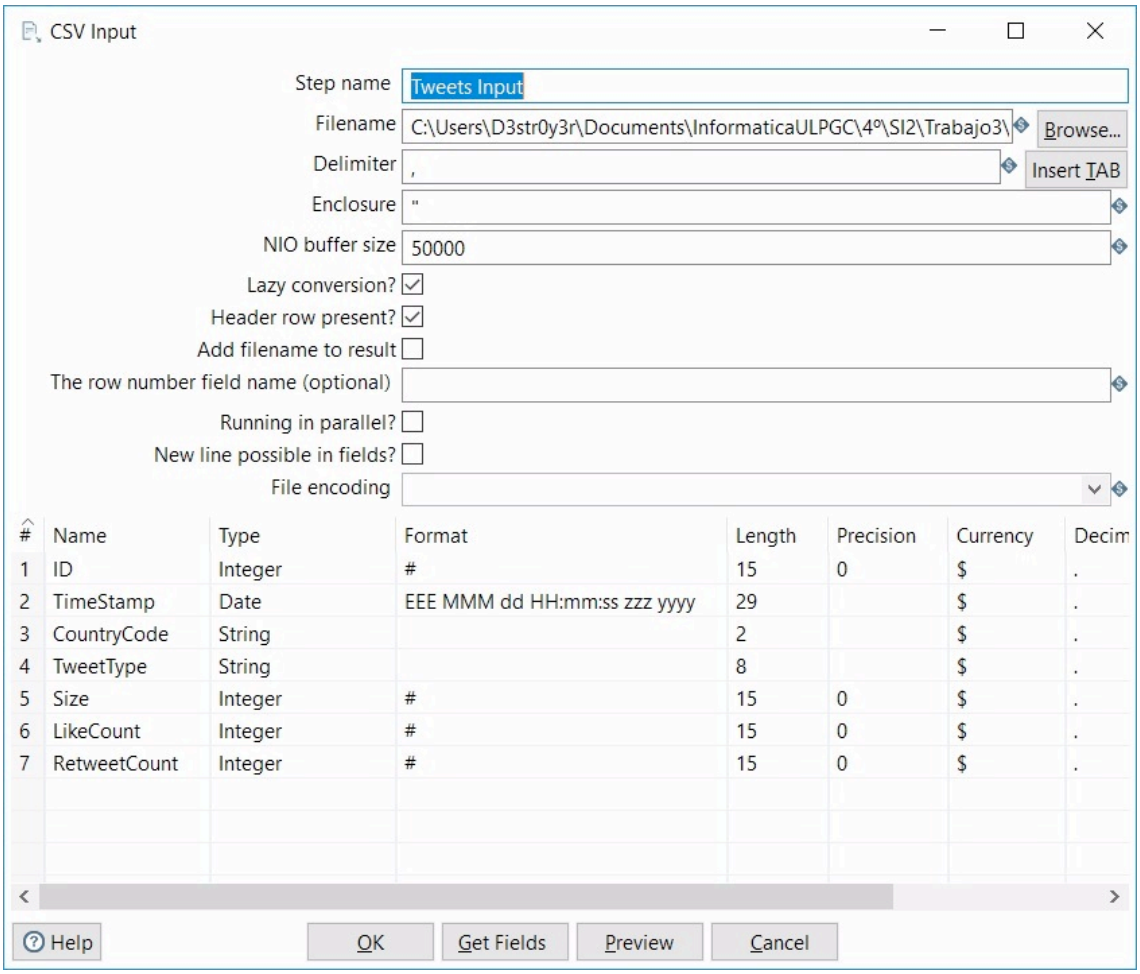

Ilustración 2-5: Tratamiento Fichero Entrada.

Ha sido necesaria la creación de una transformación que ordena los tweets por ID y elimina los Tweets duplicados. Ver *Ilustración 2-5.*

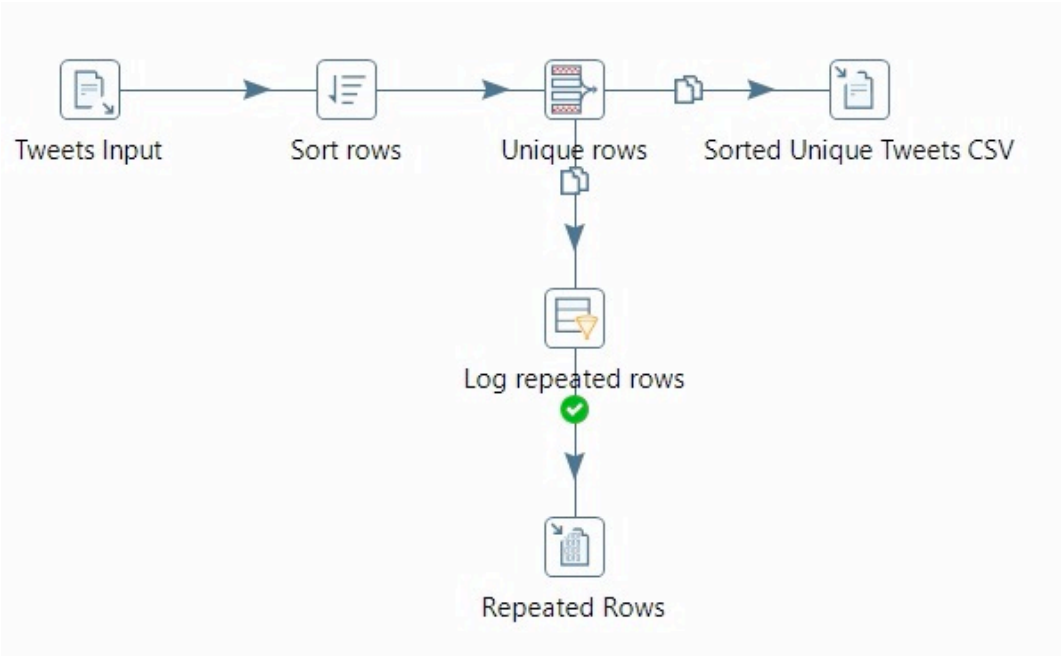

Ilustración 2-5: Sort_Unique_Trans.ktr

La siguiente transformación (*Ilustración 2-6*) carga un modelo entrenado de Weka y estima los valores perdidos de Códigos de Países utilizando una regresión logística.

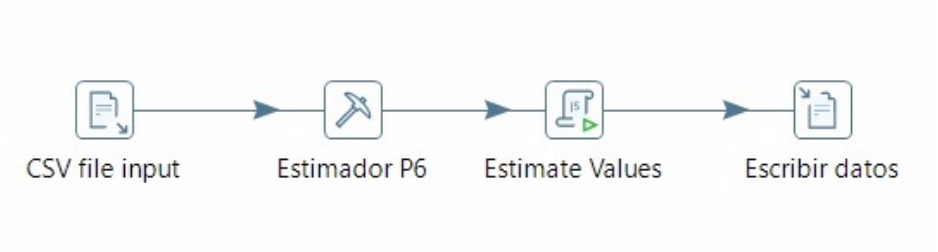

Ilustración 2-6: Train_Country_Set_Trans.ktr

A continuación, ha sido necesaria crear una transformación que busca la región y subregión del fichero CSV Country-Region-Subregion.csv a partir del código de país que la pasamos como entrada, creando un fichero con dos columnas nuevas (región y subregión). Ver *Ilustración* 2-7.

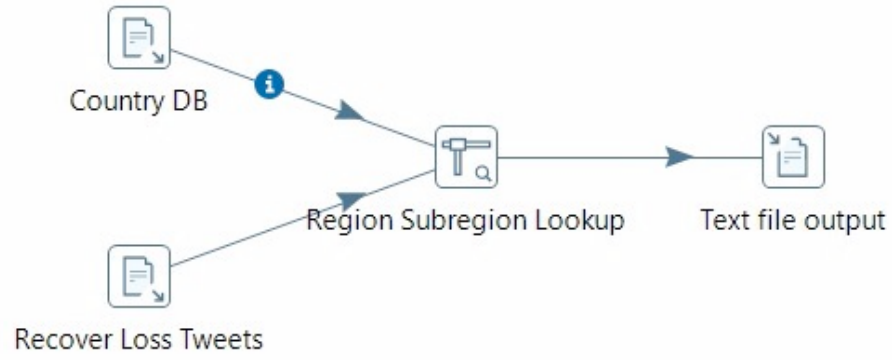

Ilustración 2-7: Region_Subregion_Trans.ktr

Esta transformación extrae el número del año, trimestre, mes, día del mes, semana del año y día de la semana a partir del campo TimeStamp de nuestro CSV generando los campos pertinentes. Ver *Ilustración 2-8.*

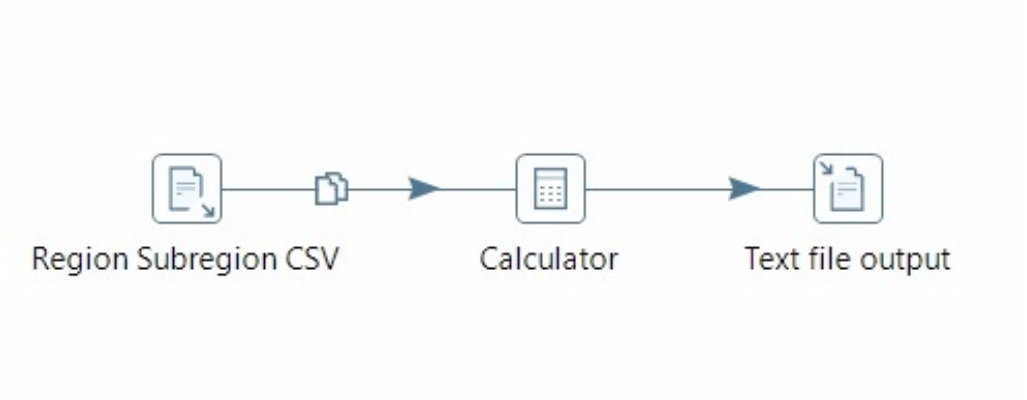

Ilustración 2-8: Segregate_TimeStamp_Trans.ktr

Con esta última transformación se termina de describir todas las transformaciones realizadas en la integración de datos.

2.2.2- Data Base Job.

A continuación, se detallarán las transformaciones realizadas en el Job de la base de datos (*Ilustración 2-4*).

La transformación del fichero Create_Table_Trans.ktr tan solo crea las tablas que serán necesarias para construir el cubo OLAP en posteriores consultas.

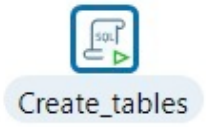

Ilustración 2-9: Create_Table_Trans.ktr

El Script SQL desarrollado para crear las tablas del cubo OLAP se puede ver en la *Ilustración 2-10*.

Execute SQL statements

Step name: Create_tables

Connection: Local DB — Edit... New... Wizard...

SQL script to execute. (statements separated by ;) Question marks will be replaced by arguments.

```
create table if not exists tweet_type(
    type                    ENUM('original', 'isRetweet', 'isReplied') primary key
    );

create table if not exists tweet_location(
    countrycode             varchar(2) primary key,
    subregion               varchar(64),
    region                  varchar(8)
    );

create table if not exists tweet_date(
    date                    varchar(255) primary key,
    year                    Int,
    trimester               Int,
    month                   Int,
    month_day               Int,
    week                    Int,
    week_day                Int
    );

create table if not exists tweet(
    id                      varchar(255) primary key,
    tweet_len               Int,
    likes                   Int,
    retweeted               Int,
    tweet_type              ENUM('original', 'isRetweet', 'isReplied'),
    tweet_location          varchar(2),
    tweet_date              varchar(255),
    foreign  key(tweet_type) references tweet_type(type),
    foreign  key(tweet_location) references tweet_location(countrycode),
    foreign  key(tweet_date) references tweet_date(date)
    );
```

Ilustración 2-11: Script SQL Create Tables.

La siguiente transformación rellena con los datos del CSV cada una de las dimensiones alojadas en las tablas: tweet_type, tweet_location y tweet_date. Ver *Ilustración 2-12.*

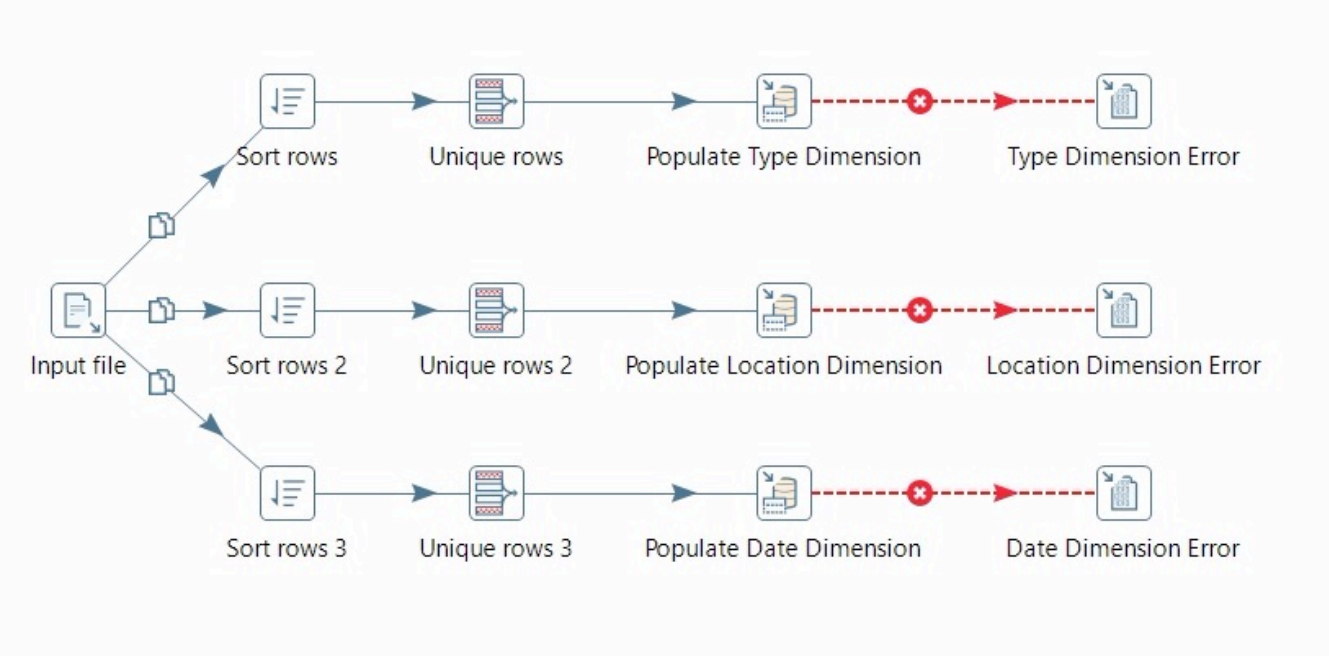

Ilustración 2-12: Populate_Dimensions_Trans.ktr

La última transformación del Job DB_Job.kjb consiste en rellenar la tabla de hechos con los datos transformados del CSV. El nombre de la tabla MySQL es: tweet. Ver *Ilustración 2-13*.

Ilustración 2-13: Populate_Fact_Trans.ktr