TweakStreet

BIG DATA – BUSINESS INTELLIGENCE – MACHINE LEARNING

1. INTRODUCCIÓN

Tweakstreet workbench es una herramienta de ETL e integración de datos muy enfocada a usuarios no formados en ningún lenguaje de programación de uso general como Java o Python.

Está orientada a trabajar solamente a nivel de datos, lo que significa que todas sus principales funcionalidades están pensadas para leer, transformar y escribir los datos, y poder controlar en todo momento cómo se están produciendo las transformaciones, todo mediante interfaz gráfica de usuario, de tal forma que no es necesario, ni se permite, ver el código que se ejecuta.

En general, aunque Tweakstreet tiene aspectos definidos, permite personalizar la mayoría de procesamientos que se realizan para dar cabida a datos que no tengan un formato estandarizado o a transformaciones customizadas que se quieran realizar. Esto se consigue gracias al uso de su propio lenguaje de expresiones: Tweakflow, que permite programar código que se aplicará sobre los datos.

2. POSIBLES USOS DE TWEAKSTREET WORKBENCH

Tweakstreet workbench es una herramienta que permite realizar multitud de transformaciones de datos. Por ello, el principal uso que se le puede dar a esta herramienta son las tareas de ETL (Extract, Transform, Load), sobre todo aquellas en las que deseemos realizar transformaciones especiales sobre los datos, ya que gracias a la multitud de librerias que implementa, unido al potencial del lenguaje Tweakflow, cualquier transformación que sea independiente (es decir, que no realice operaciones con estado u operaciones que afecten a contenidos externos que no sean los datos) probablemente se pueda resolver.

Al contrario que otras ETL, Tweakstreet no se limita a los almacenamientos de datos en formato de tabla, como ficheros de tipo CSV o bases de datos relacionales, sino que también permite trabajar con representaciones de tipo árbol, incluyendo así bases de datos no relacionales y ficheros de tipo JSON. Tweakstreet permite también extraer datos de ficheros que ni siquiera cumplen formatos estándares, como ficheros en formato binario o cualquier otro formato que el usuario pueda después diseccionar en diferentes campos.

3. INSTALACIÓN

La herramienta es completamente gratuita, y posee versiones para Windows, MacOS y Linux. La web para descargarla es la siguiente:

https://tweakstreet.io/download/

Instalación en Windows:

El fichero que se descarga es un paquete que contiene la aplicación completa, con todos los ficheros. Sencillamente crear una carpeta en el directorio donde el usuario desee y mover el paquete ahí. Descomprimir el paquete. Se extraerán todos los ficheros. Entre ellos, se encuentra el ejecutable, con el nombre: Tweakstreet.exe. Al realizar doble click sobre el ejecutable se arrancará la aplicación.

Si se desean realizar conexiones a bases de datos, es necesario instalar el driver correspondiente en el directorio: C:\Users\<usuario>\.tweakstreet\drivers, con la aplicación cerrada.

Instalación en MacOS:

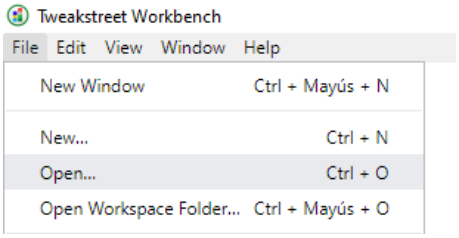
Abrir el fichero dmg y arrastrar "Tweakstreet.app" al disco duro, normalmente a la carpeta de aplicaciones. Se ejecuta haciendo doble click en "Tweakstreet.app".

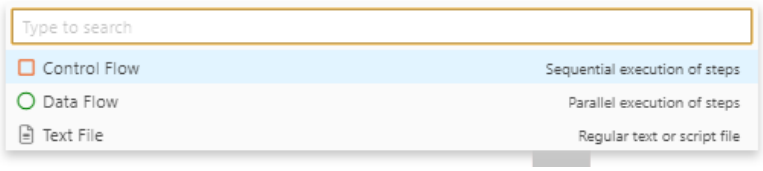
Instalación en Linux:

Extraer los ficheros del paquete a algún directorio del disco duro, y ejecutar el fichero binario "tweakstreet" desde una terminal.

4. CREACIÓN DE UN FICHERO FLOW

Para crear un nuevo fichero con el que trabajar, en el menú "File", pulsar la opción "New…" y seleccionar el tipo de Flow que se desea.





5. FLOWS Y SUS TIPOS

En Tweakstreet, un Flow es una sucesión de elementos de procesamiento (elementos que realizan las diferentes operaciones disponibles) que se ejecutan. Dependiendo del tipo de Flow que deseemos, en Tweakstreet se pueden crear dos tipos de ficheros de trabajo diferentes:

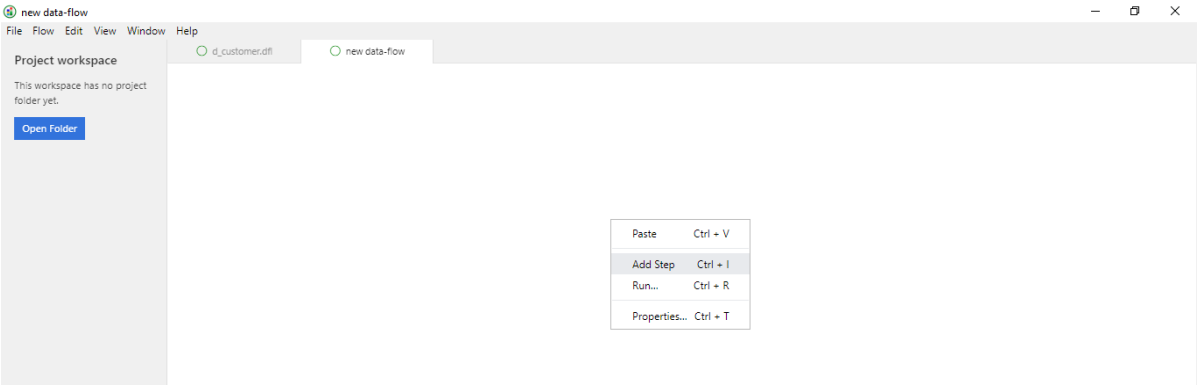
- **Ficheros de Control Flow**: en este tipo de ficheros la ejecución es secuencial, es decir, se comienza por la unidad de procesamiento "Start" y se va ejecutando una a una cada unidad de procesamiento.
- **Ficheros de Data Flow**: en este tipo de ficheros la ejecución es paralela, es decir, todas las unidades de procesamiento se ejecutan a la vez, pudiendo algunas quedar a la espera de recibir datos de otras unidades.

6. STEPS

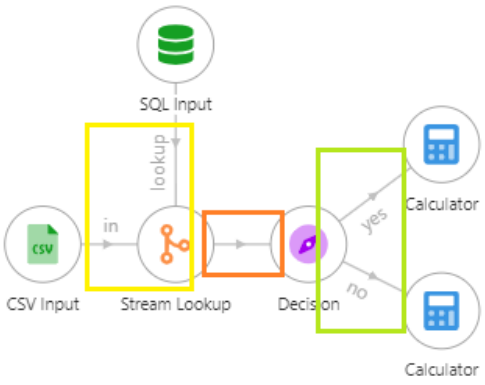
Tweakstreet posee multitud de unidades de procesamiento llamadas "Steps", que permiten multitud de operaciones como:

- Entrada y salida de ficheros.
- Transformaciones sobre los datos.
- Consultas al sistema.
- Consultas web.
- Otras utilidades como conteo de filas, sleeps, enviar mails, crear conectores customizados, etc.

Los Steps se añaden pulsando CTRL+i, o haciendo click derecho sobre la pantalla blanca y haciendo click en la opción "Add step", posteriormente seleccionando el Step que deseemos.



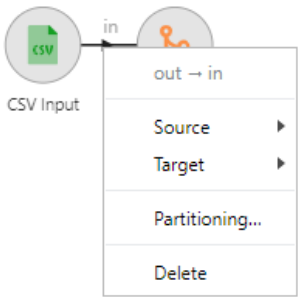
Cada Step posee una serie de puertas de entrada y de salida, dependiendo del tipo que sea, habiendo siempre como mínimo la posibilidad de establecer una de entrada, una de salida y otra de salida de error. Dependiendo del Step en específico, puede haber varias puertas de entrada, si se necesita distinguir los datos que tienen que entrar, o varias puertas de salida, si hay salidas condicionales. Cuando se conecta un Step con otro (manteniendo presionada la tecla shift y clicando y arrastrando desde un step a otro), lo que se realiza es una conexión entre una puerta de salida del Step de origen y una puerta de entrada del Step destino.



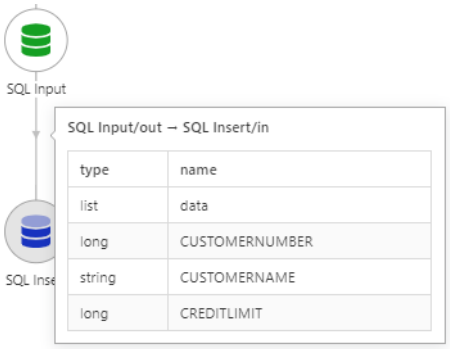
En la imagen de arriba, se puede observar:

- El Step “Stream lookup” tiene dos puertas de entradas diferentes: “in” y “lookup”, como se puede ver en el rectángulo amarillo.
- El Step “Stream lookup” y el Step “Decision” solo tienen una puerta de salida y una de entrada, respectivamente, como se observa en el rectángulo naranja.
- El Step “Decision” tiene dos puertas de salida: “yes” y “no”, como se puede ver en el rectángulo verde.

El tipo de puerta por la que entra o sale se puede editar haciendo click derecho en la conexión, y en las opciones Source y Target, respectivamente.



Si se deja el ratón encima de un enlace entre dos Steps, se pueden ver los campos que viajan por él.

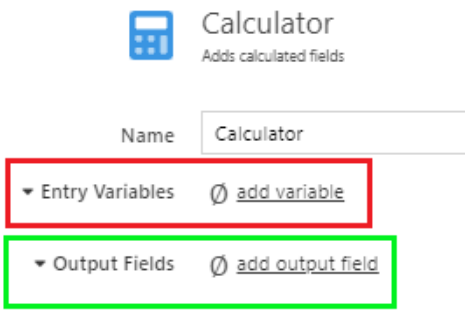


Los Step se pueden configurar haciendo doble click sobre ellos. Cada Step tiene configuraciones diferentes, dependiendo de la operación que realicen. En Tweakstreet, los Step se pueden visualizar como funciones: reciben datos, los transforman, y los devuelven. Es por esto que en cada Step se pueden asignar variables de entrada y variables de salida.

Aclaración: las variables y los campos de entrada de un Step son conceptos diferentes. Las variables se definen dentro del Step, y los campos son los que llegan desde un Step con el que se tiene una conexión. El conjunto total de datos de entrada a un Step son los campos de entrada más las variables de entrada.

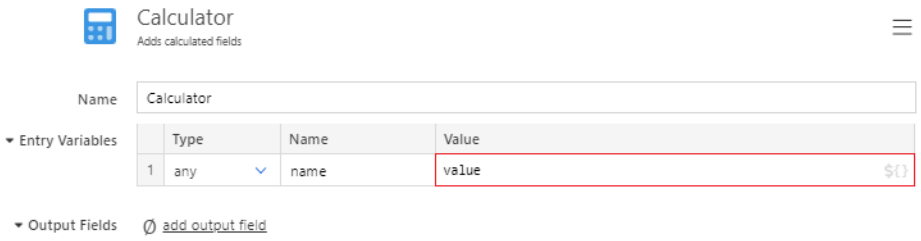
Los valores de las variables de entrada pueden ser desde valores que se introduzcan a mano, que se calculen en el propio Step, valores que se hayan guardado como parámetros de ejecución o valores que vengan de un Step anterior.

Los valores de las variables de salida son datos que se calculan o se transforman en el propio Step. Si este paso lo conectamos con otro Step, los campos no modificados de los Step conectados anteriores pasarán también.



En la imagen, las variables de entrada se definen en la sección del rectángulo rojo, y las de salida en la sección del rectángulo verde, pero esto es un ejemplo. En otros Steps puede que cambie la distribución.

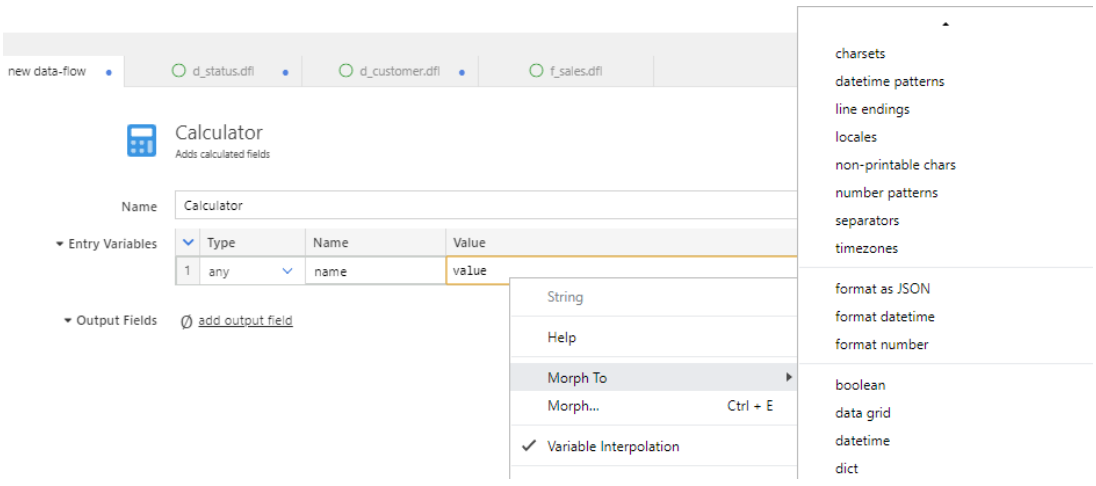
En el apartado de configuración de un Step, los campos "Value" de cada variable se llaman Widjets.



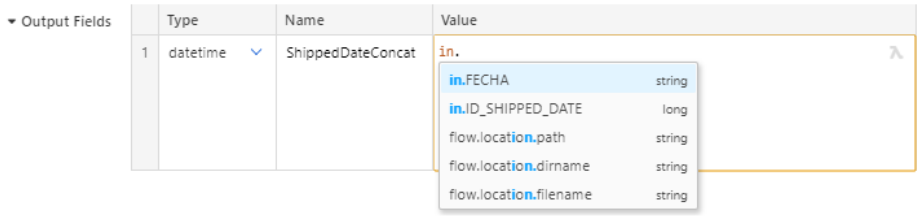
Cada Widjet puede ser del tipo que deseemos: si se hace click derecho sobre uno de estos campos y navegamos por la opción "Morph to" se podrán ver la multitud de posibles tipos de valores de los que la aplicación dispone. Entre ellos cabe destacar:

- **"Field reference"**: que permite hacer referencia a alguno de los campos de entrada del Step.
- **"Result reference"**: que permite hacer referencia a alguno de los campos de salida del Step.
- **"Formula"**: que permite escribir código Tweakflow para realizar transformaciones sobre los datos.

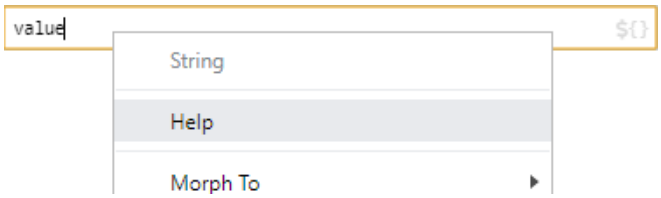
Nota: Los Widjet de tipo "String" no necesitan comillas, sencillamente se escribe la cadena de caracteres.



En los Widjets también se pueden solicitar sugerencias de ayuda a la escritura. Por ejemplo, si se quiere hacer referencia a una variable de entrada declarada en la configuración que hemos llamado “Constante”, por ejemplo, podemos escribir “Const” y pedir una sugerencia pulsando CTRL+Enter. Si desde donde hemos solicitado la sugerencia tenemos acceso a la variable, aparecerá en un desplegable y podremos seleccionarla. Al hacer esto Tweakstreet autocompletará el código.



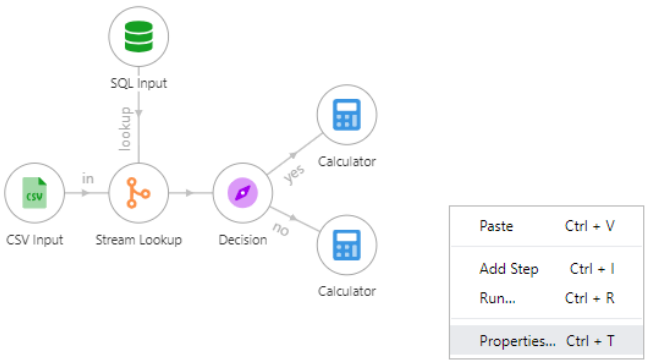
Cada Widjet tiene su página de ayuda, a la que se puede acceder haciendo click derecho sobre él, en la opción “Help”.



Por último, indicar que los Step están pensados para ser independientes a los demás Steps, es decir, ningún Step va a producir efectos sobre otros Steps. Además, ninguno posee estado, salvo el Step llamado “Stateful Calculator”, con lo que cualquier Step va a devolver siempre los mismos resultados a pesar de que se ejecute varias veces durante la ejecución del Flow.

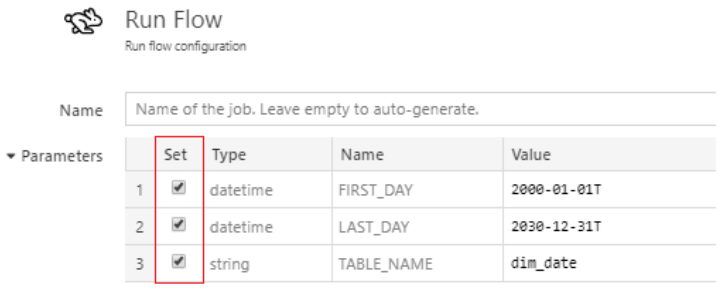
7. PARÁMETROS, VARIABLES Y SERVICIOS

En la configuración de los Flows encontramos algunos campos útiles para la ejecución. Para acceder a la configuración de un Flow, hacer click derecho sobre la pantalla blanca donde están los Steps, y pulsar en la opción “Properties…”.

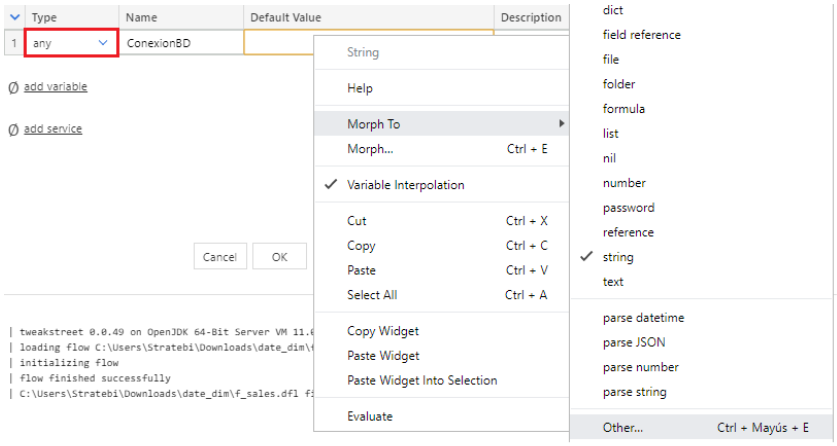


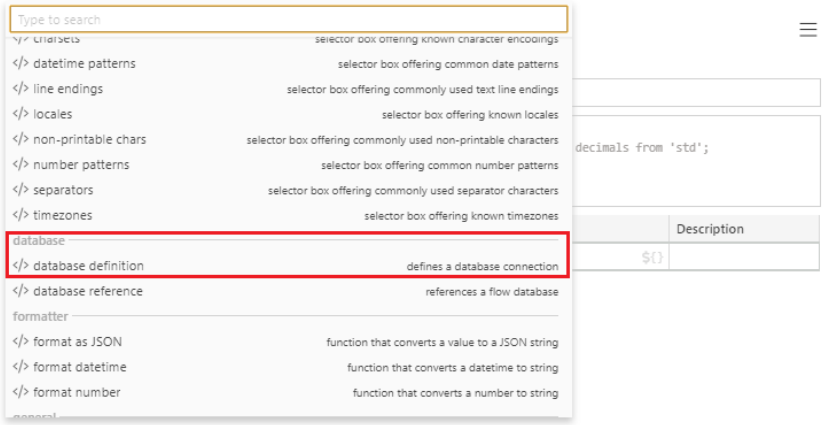
Las secciones que se encuentran son:

- **Imports**: en este cuadro de texto se pueden importar los módulos que se deseen. La manera de hacerlo se explica en la sección Módulos.
- **Parameters**: permite definir parámetros que estarán disponibles durante toda la ejecución. La definición consiste en asignar un tipo de dato y un nombre, y opcionalmente, un valor por defecto. A la hora de ejecutar el Flow, se podrán introducir los valores para esos parámetros, o dejar el valor por defecto quitando el check de la columna “Set”.



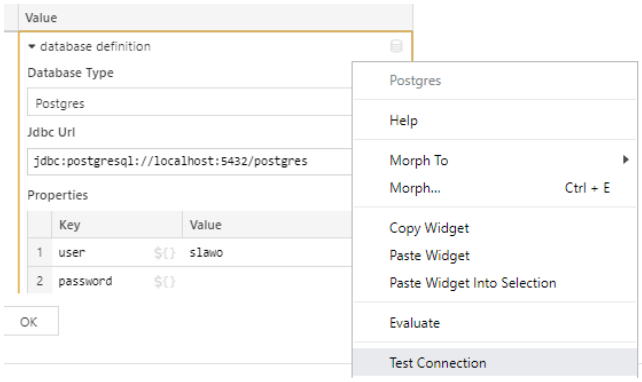
Estos parámetros funcionan igual que los Widjets de los Steps, permiten usar la opción Morph y transformarlo en cualquier expresión: desde tipos de datos simples, pasando por fórmulas, hasta conexiones con bases de datos.
Las conexiones con bases de datos se encuentran haciendo click derecho en el Widjet, accediendo a la opción de “Morph to…”, y seleccionando la opción “Other”, o haciendo click en la opción “Morph” y luego se elige de la lista que se despliega. El tipo de dato que se debe asignar a este parámetro será Any.





Con esto se podrían generar Flows genéricos e ir especificando datos antes de la ejecución sin tener que alterarlo.

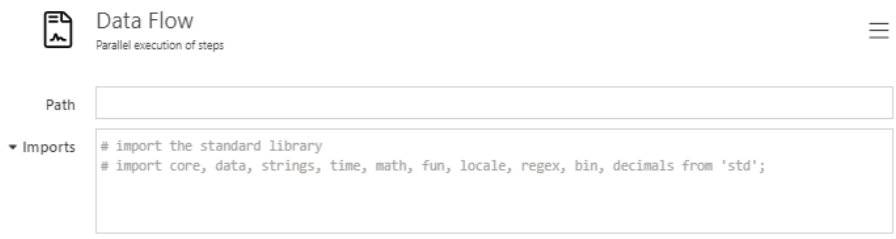
- **Variables**: permite definir variables que estarán disponibles durante toda la ejecución. Pese a su nombre, este apartado está pensado para definir constantes, o para validar o realizar cálculos sobre los parámetros introducidos.
- **Services**: permite definir valores de configuración o recursos como conexiones con bases de datos o credenciales para servidores. (La conexión a una base de datos se puede probar haciendo click derecho sobre el widjet, y pulsando en la opción “Test connection”)



8. MÓDULOS

Tweakstreet posee módulos con librerías que poseen funciones ya definidas. Existen tres tipos de módulos:

- **Módulos estándares**: Tweakstreet posee 8 módulos estándares con funciones de transformación de datos muy útiles. Estos módulos se deben importar desde la configuración del Flow. (El acceso a esta configuración se explica en la sección: Parámetros, variables y servicios) En la sección "imports", se puede observar por defecto un import de algunas librerías del módulo std comentado. Tienen el formato: import lib1, lib2, … , libN from 'modulo';



  Como se ve en la imagen, el import del módulo estándar (std) viene por defecto ya escrito, pero comentado. Para descomentarlo, sencillamente eliminar el carácter de comentario '#'.
- **Módulos customizados**: el usuario puede crear sus propios módulos añadiendo sus propias librerías. Los módulos customizados también se importan igual que los estándares, pero usando el PATH hacia la librería después del "from".
- **Módulos globales**: el usuario puede crear sus propios módulos y luego inyectarlos en el fichero de trabajo, para no tener que importarlo.

Se puede encontrar más información sobre los módulos customizados en la sección de "Modules", y sobre los módulos globales en la sección de "Global modules" de la documentación de la aplicación.

9. TWEAKFLOW

Tweakflow es el lenguaje de expresiones basado en Java creado por Twineworks y que se puede usar en Tweakstreet, ampliando su potencial de manera considerable.

Es un lenguaje que puede recordar a Python, pero está pensado para no parecerse demasiado a ningún lenguaje y así que cualquier persona pueda aprenderlo.

Aquí se describen algunos conceptos básicos para manejarlo:

- De entre los tipos de datos, se deben destacar:
  - El tipo `long`, para datos de tipo entero.
  - El tipo `datetime`, para fechas, que deben tener como mínimo el formato *yyyy-mm-ddTHH:MM:SS*. Ni Tweakstreet ni Tweakflow soportan formatos de fechas sin hora.
  - El tipo `nil`, que representa el valor nulo. (null en la mayoría de lenguajes de programación)
  - El tipo `list`, para listas. Una lista se define como `[elemento1, elemento2, … , elementoN]`. A los elementos se accede con `lista[i]`, siendo `lista` el nombre de la lista e ***i*** el índice del elemento, que empieza en 0. Se les puede aplicar un cast (un cambio de tipo de datos) a dict.
  - El tipo `dict`, para diccionarios. Un diccionario es un conjunto de pares clave-valor. Se definen como `{:clave1 valor1, :clave2 valor2, … , :claveN valorN}` A los valores del diccionario se accede con las claves de la siguiente manera `: dict[:clave]`, siendo `dict` el nombre del diccionario y `clave` la clave del elemento del que queremos el valor.
  - El tipo `any`, que actúa como tipo genérico si no sabemos qué tipo de dato necesitamos.

  Un dato se puede convertir a otro tipo de dato mediante un "casting". No todos los casting están permitidos. Se realizan de la siguiente manera:

  Valor/variable as tipoDato

  Ej:

  ```
  4 as String
  ```

  `x as Double`, siendo x una variable de tipo long.

- Las variables se definen como `tipoDato nombreVariable: valor`. El valor puede ser una función, en cuyo caso no se le asigna tipo de dato. Si no se le asigna tipoDato, Tweakflow asignará el que más concuerde con lo escrito.
  Las variables que se definen son visibles en toda la sesión de Tweakflow, y en el caso de Tweakstreet, son visibles al Step en el que se definan. Si se quiere declarar una variable que sea solo visible a una expresión que se escriba después, se puede usar let:

  ```
  let{
          (Definición de variables separadas por ; y un salto de línea)
  }expresión
  ```

  Por ejemplo:

  ```
  let{
          f: (int x) -> int x + x;
          long x: 4;
  }f(x)
  ```

  `f` y `x` son solo visibles para la expresión que sigue la llave de cierre.

- Las funciones se definen como:

```
(tipoParametroEntrada1 nombreParametroEntrada1, … , tipoParametroEntradaN
nombreParametroEntradaN) -> tipoParametroSalida expresión
```

Ej: `(long a) -> long a + 4` sería una función que recibe un entero ***a***, devuelve un entero y realiza una suma del entero que recibe más 4.

Como se observa, a las funciones solo se les permite evaluar una expresión. Si se necesita evaluar más expresiones o crear variables intermedias se puede usar let:

```
(long a) ->
let{

      long b: 4

} a + b
```

Para llamar a las funciones, se deben asignar a una variable y después:
nombreFuncion(parametroEntrada1, … , parametroEntradaN)

Ej:

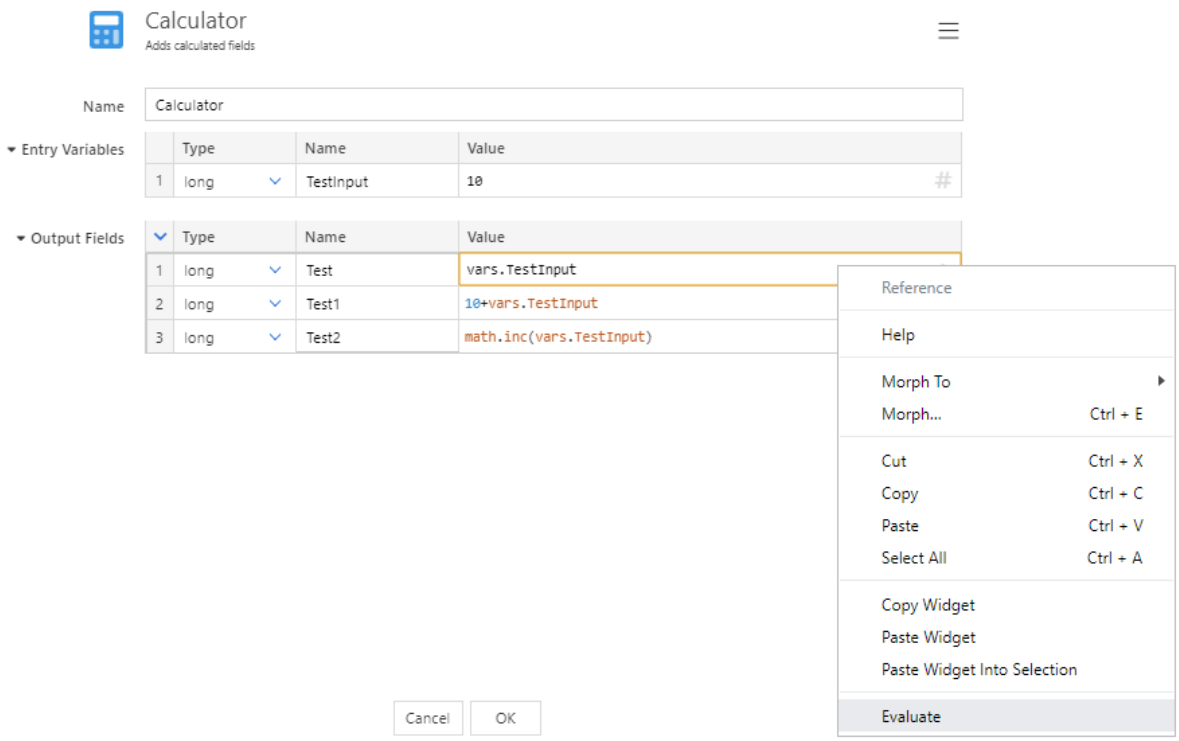
```
f: (long a) -> a + 4;
f(4)
```

>> 8

10. TESTING

En Tweakstreet no es necesario ejecutar el trabajo completo para conocer si la transformación que se ha configurado en el Widjet de un Step va a devolver lo que se desea, o si el código escrito en Tweakflow es correcto o contiene errores. En otras palabras, podemos ejecutar Widjets en un Step sin tener que ejecutar el Flow completo.

Para ello simplemente seleccionamos los Widjets a ejecutar y, haciendo click derecho, seleccionamos la opción “Evaluate”. Importante: es necesario que todo el código que se ejecuta detrás compile, con lo que al usar Evaluate podrían mostrarse errores de otros Steps o errores en la configuración del fichero.

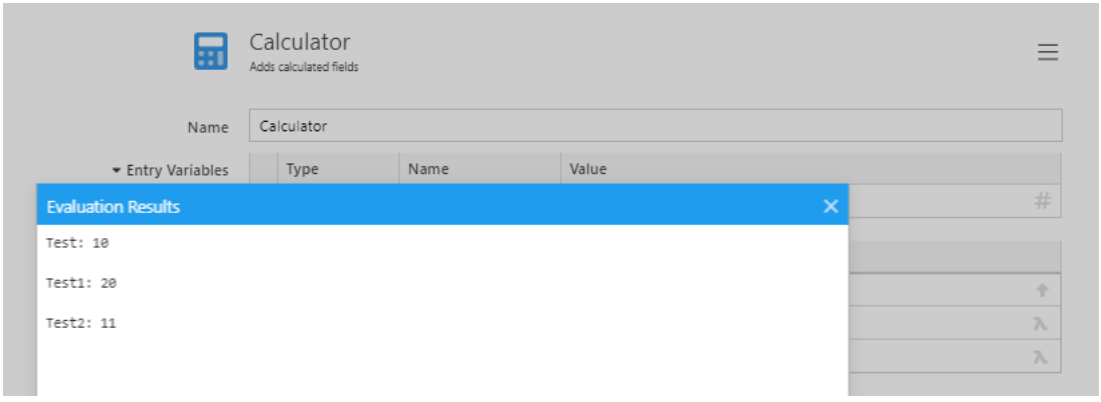


En la imagen, hemos creado una variable de entrada "TestInput", de tipo long y valor 10. Por otra parte, hemos creado tres variables de salida:

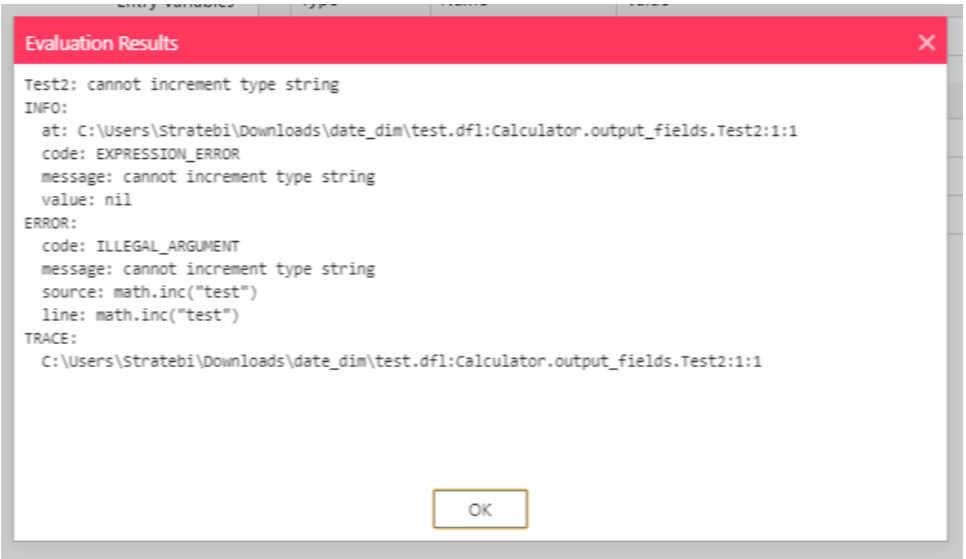
- Test: de tipo long, que sencillamente tiene como valor una referencia a la variable de entrada.
- Test1: de tipo long, que tiene como valor una fórmula que suma 10 a "TestInput".
- Test2: de tipo long, que tiene como valor una llamada a la función "inc" de la librería math del módulo std, que sencillamente incrementa en 1 el valor numérico que reciba como argumento.

Nota: en Test, el Widjet es de tipo "Numeric", y en Test1 y Test2 el Widjet es de tipo "Formula"

Si todo compila de forma correcta, se mostrará una ventana con el resultado. Obviamente para que proporcione un resultado visible, hay que asignarle datos al Widjet que se quiere ejecutar, datos que se pueden introducir manualmente o que se pueden declarar como variables. En el caso del ejemplo, hemos creado esa variable de entrada "TestInput".



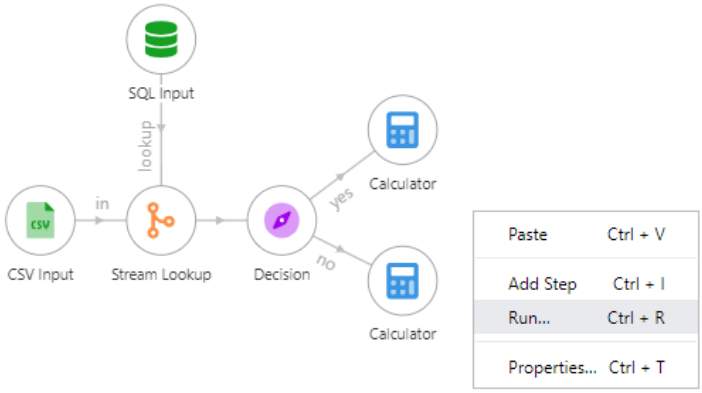
En el caso de que haya algún error en alguno de los Widjets, como por ejemplo, si se llamara a la función inc con una cadena de caracteres, la pantalla de error que aparece sería la siguiente:



11. EJECUCIÓN Y DEBUG DESDE LA APLICACIÓN

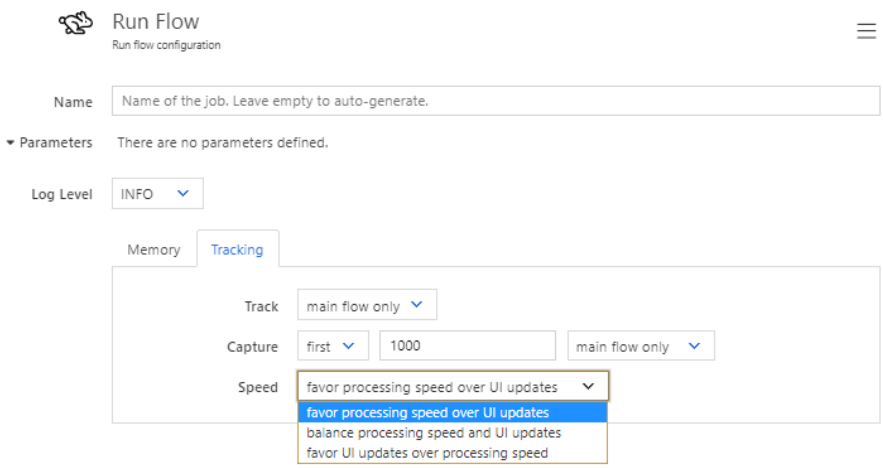
Una de las características más importantes de Tweakstreet es que permite visualizar durante la ejecución en todo momento cómo van evolucionando los datos a medida que van pasando por los Steps, aspecto que se explicará a continuación.

Para ejecutar un fichero, click derecho sobre la pantalla blanca donde están los steps, y pulsamos en la opción “Run…”.



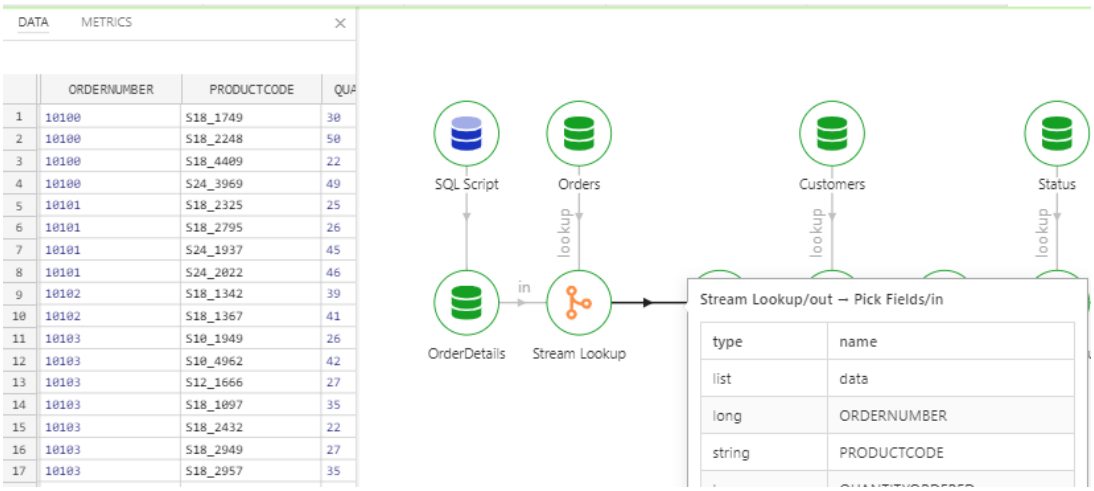
En la ventana de configuración se pueden seleccionar varias opciones:

- **Log Level**: el nivel de detalle que queremos en el log de la ejecución. La información sobre los niveles se puede encontrar en la documentación de la herramienta.
- **Memory**: la cantidad de memoria principal que se le va a asignar a la ejecución.
- **Tracking**: de qué queremos que se obtenga traza durante la ejecución y cuántas filas queremos que capture. La última opción de esta pestaña permite que se seleccione si se quiere favorecer la velocidad de procesamiento frente a las actualizaciones de la interfaz gráfica; al contrario, favorecer las actualizaciones de la interfaz gráfica frente al procesamiento (en el caso de que se quiera conocer con detalle cómo se van transformando los datos) o si se quiere un equilibrio de ambos.



Una vez se comience la ejecución, aparecerá una ventana con los flows creados y una animación para indicar qué Steps se están ejecutando (indicadas con una circunferencia de líneas discontinuas que giran), cuáles han terminado de ejecutar bien (indicadas con una circunferencia verde) y cuales han terminado de ejecutar con error (indicadas con una circunferencia en rojo).

Y aquí viene el aspecto más interesante. Si se hace click sobre una de las flechas entre dos Steps, se puede ver en tiempo real los datos que van pasando a través de ese enlace. De esa forma se puede conocer la evolución de los datos a lo largo del trabajo para asegurarse que las transformaciones que queríamos realizar se están haciendo de forma correcta.



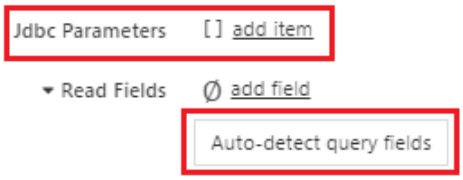
12. ALGUNOS TIPS PARA TAREAS ETL

1. **Steps para bases de datos SQL**

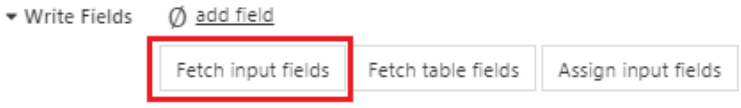
   Todos los Steps para realizar queries SQL (excepto los de inserción o actualización de datos) tienen en común que se debe escribir el código a mano, lo que en cierta medida permite tener un mejor control de lo que se ejecuta, con la desventaja de tener que manejar SQL para poder escribirlas.

   Para realizar queries SQL existen diversos pasos, dependiendo del tipo de query que queramos ejecutar:

   - **SQL Input**: para consultas de tipo SELECT o SHOW. Una vez escrita la consulta, permite detectar los campos que se van a consultar y establecerlos como campos de salida pulsando el botón "Auto-detect query fields". Se permite además añadir parámetros JDBC sobre la query y definirlos en "JDBC Parameters".



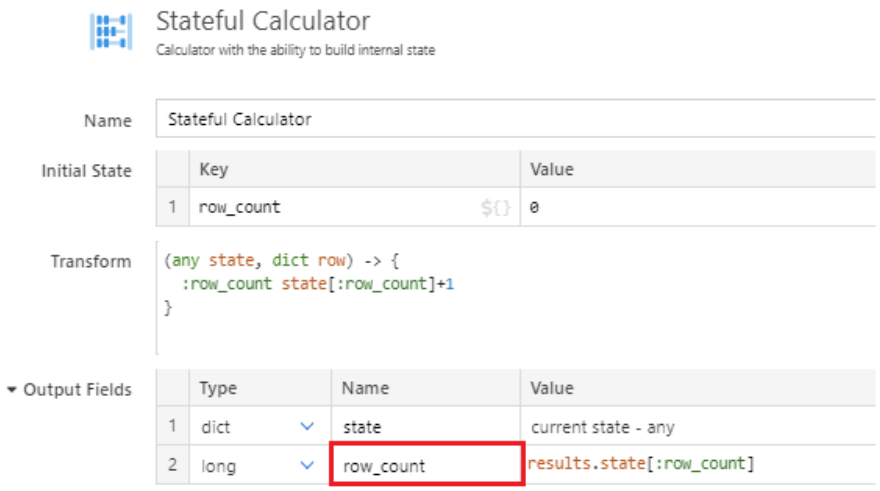
   - **SQL Insert**: permite realizar inserciones sobre la base de datos. Usando el botón "Fetch input fields" de la sección Write fields, Tweakstreet completa la sección con los campos que entran en el Step.



Es importante tener en cuenta que este Step no genera la tabla si no está creada, pero si se despliega la sección SQL, y pulsando en "Fetch SQL", Tweakstreet generará el código necesario para que la inserción se pueda ejecutar correctamente, que normalmente implica generar la tabla.
Importante: ese código no se ejecutará con el Step. O bien se pulsa en el botón "Run SQL" para ejecutarlo manualmente, o bien se copia y se pega en un Step de tipo SQL Script para ejecutarlo antes de ese Step.

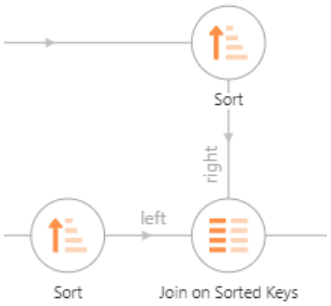
- **SQL Insert or Update**: permite realizar actualizaciones sobre una tabla de la base de datos en el caso de que el elemento ya exista, o una inserción en el caso de que no.
- **SQL Lookup**: permite realizar una consulta sobre una tabla de la base de datos usando una key.
- **SQL Script**: permite ejecutar scripts SQL sobre una base de datos.

2. **Generar claves subrogadas.**
Para generar claves subrogadas en dimensiones que no vayan a usar versionado (es decir, que no sean "Slow changing dimensions"), se puede usar el Step "Stateful calculator". Es el único Step de Tweakstreet que posee estado.
El Step por defecto viene configurado con un generador de una secuencia de números desde el 1. Con lo cual, lo único que hay que cambiar es el nombre del campo que se va a generar, que por defecto se llama: "row count".

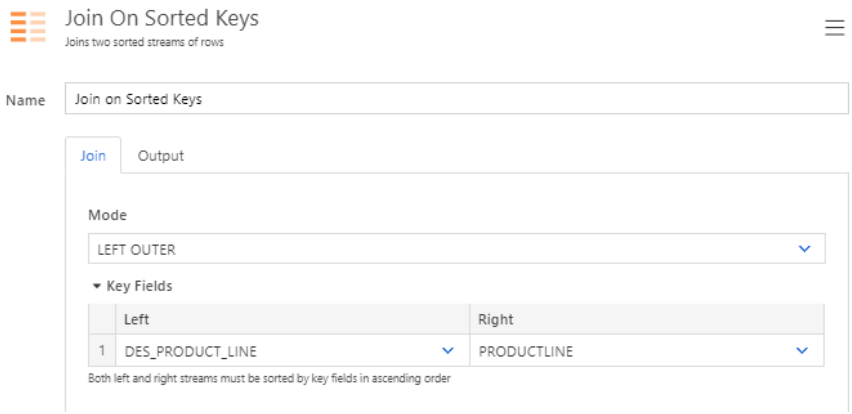


Tras el Step “Stateful calculator” se puede usar un Step de tipo “Pick fields” para eliminar el campo “State” que genera el “Stateful calculator”.

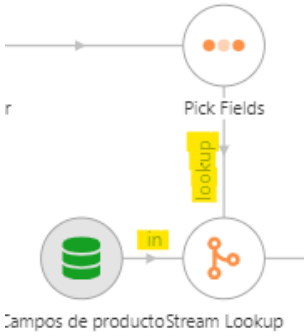
3. **Realizar joins.**
   Los joins en Tweakstreet solo se pueden realizar entre dos tablas por cada Step.
   Existen dos maneras de realizar joins en Tweakstreet.
   La primera es usando el paso diseñado para ello: “Join on Sorted Keys”. El inconveniente es que para que funcione, es necesario justo antes ordenar las claves de manera ascendente usando un par de Steps de tipo “Sort”, uno por cada tabla.



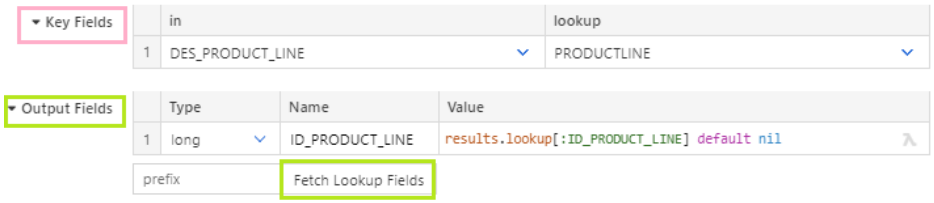
En el Step del join, sencillamente configurar la pestaña “Input” seleccionando el tipo de Join que se desea y las claves de la tabla de la izquierda y de la tabla de la derecha que se van a comparar y si se desea, configurar la pestaña “Output” para configurar aspectos de los campos de salida.



La segunda manera es usando el paso “Stream Lookup”. En este paso hay que tener cuidado con las conexiones con los Steps anteriores, ya que la primera se establecerá como la conexión para la tabla de entrada y la segunda se establecerá como la tabla en la que se va a realizar la búsqueda, pero se puede cambiar como se indica en la sección Steps.



En la configuración del Step, sencillamente indicar cuál es el campo de la tabla de entrada que se empleará como clave y cuál es el campo de la tabla de búsqueda con el que se comparará. Si se pulsa el botón “Fetch lookup fields”, aparecerán los campos de la tabla de búsqueda que se pueden incorporar, y se pueden renombrar o eliminar si se desea, opción que no permite el Step de “Join on sorted keys”.



En la imagen, las claves se establecen en la sección del rectángulo rosa, y los campos a añadir de la tabla de búsqueda aparecerán en la sección del rectángulo verde una vez se haga click en el botón “Fetch lookup keys”.

4. **Slow changing dimensions.**
   Permite insertar datos en una tabla de una dimensión del tipo “Slow changing”.
   En la pestaña Fields hay que configurar:
   - **Primary key**: aquí se debe escribir el nombre de la clave subrogada que se generará para la dimensión.
   - **Business keys**: aquí se deben poner los campos de entrada al Step que corresponden con las “Natural keys” de la dimensión.
   - **Dimension fields**: aquí se deben poner los campos de entrada al Step que hacen que esta dimensión sea Slow changing. Hay que indicar el tipo de operación que se debe aplicar ante un cambio. Existen tres tipos:
     1. **Insert**: cambios en el campo harán que la versión actual del registro pase a expirada y se genere un nuevo registro con una nueva versión.
     2. **Update**: cambios en el registro sencillamente actualizará el campo del registro de versión actual.
     3. **Punch-through**: cambios en el campo provocarán una actualización del campo en todas las versiones del registro.

     Al igual que el Step “Insert SQL”, también contiene el apartado SQL para poder generar el código SQL para crear la tabla.

Para más información sobre el resto de campos del Step, consultar la documentación de la herramienta.

5. **Tipo de datos Date.**
   El lenguaje que usa Tweakstreet, Tweakflow, no contempla el tipo de datos Date a secas, muy presente en dimensiones de tipo tiempo o en campos de tipo tiempo, sino que sólo contempla el tipo de datos DateTime.
   Si en alguna tarea ETL es necesario comparar fechas, y alguna de ellas viene en formato Date a secas, lo más probable es que Tweakstreet le haya asignado el tipo de datos String. Para transformarlo a DateTime, se pueden aplicar una transformación usando el Step "Calculator":

   - Esta transformación emplea una función de la librería time del módulo std, con lo cual, es necesario importarla en las preferencias del Flow. (puede encontrar cómo en la sección Módulos)
   - Conectar un Step de tipo "Calculator", y dentro:
     - En la sección "Entry variables", crear una variable de tipo "function", de nombre TimeParser, hacer un Morph del Widjet a "formula", y escribir la siguiente expresión:

       time.parser("uuuu-MM-dd",default_tz: 'Europe/Madrid')

       En Tweakflow, una fecha de tipo DateTime tiene el formato:

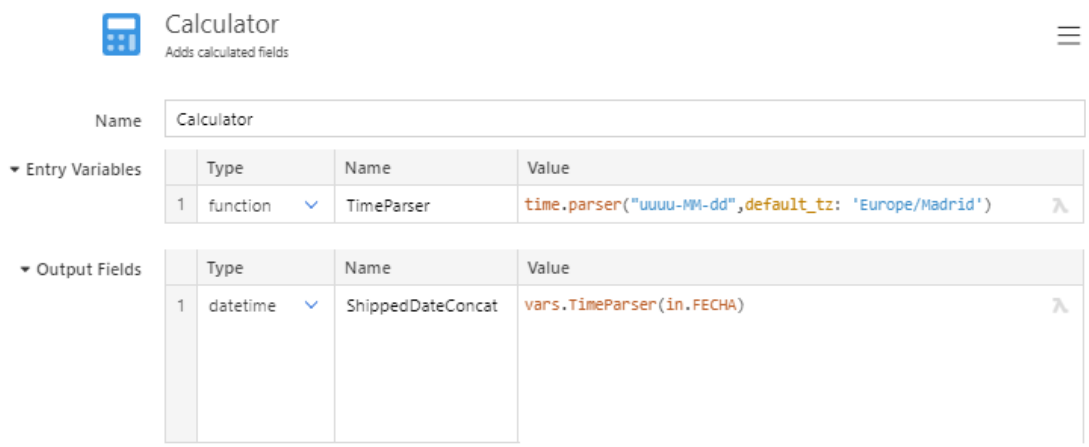
       aaaa-mm-ddTHH:mm:ss+HH:mm@TZ

       "parser" es una función de la librería time que intenta transformar fechas que cumplan el patrón pasado como primer argumento a el tipo de datos DateTime. En el caso de que los datos de tipo DateTime con los que se quiere comparar sean de una zona horaria distinta a la de Greenwich, se puede establecer una propia con el argumento "default_tz", como se indica arriba.
     - En la sección "Output variables", crear una variable de tipo "DateTime", y asignarle el nombre de campo que queremos que tenga. Realizar un Morph sobre el Widjet a "formula" y escribir lo siguiente:

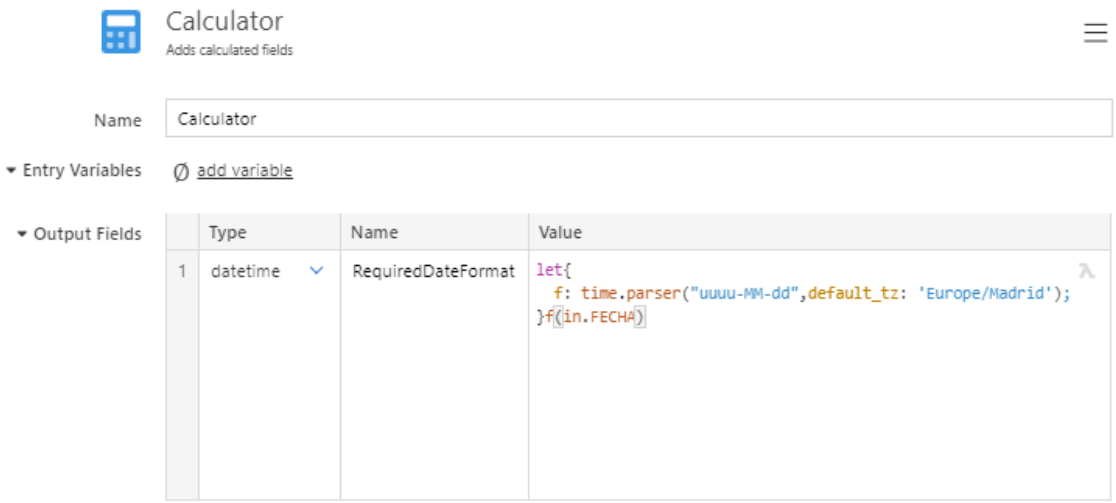
       vars.TimeParser(in.FECHA)

       vars contiene las variables de entrada definidas en el Step, e in contiene los campos de entrada al Step. FECHA es el campo que queremos transformar de String a DateTime.

   Con esto hemos conseguido transformar de Date a DateTime.
   En la imagen se puede ver la configuración del Step "Calculator":



Se puede crear la misma transformación sin crear la variable "TimeParser" creándola directamente en la variable de salida usando "let":



13. EJECUCIÓN DESDE LA CONSOLA DE COMANDOS

Tweakstreet permite ejecutar flows desde la consola de comandos usando el programa "engine" (engine.sh en el caso de Linux, engine.bat en el caso de Windows).

Para ejecutar un flow, sencillamente hay que ejecutar el programa y a continuación, escribir la ruta al trabajo. Por ejemplo:

```
C:\TweakStreet\bin\engine.bat C:\Users\Stratebi\Desktop\I+D\TweakStreet\d_status.dfl
```

Y a continuación, saldrá por la consola un log indicando el curso de la ejecución del flow. Se sabrá que se ha ejecutado correctamente si encuentra el siguiente log:

```
2020-03-02T12:59:26+01:00 | INFO  | d_status.dfl              | flow finished successfully
```

Si el flow necesita parámetros de ejecución, se le puede asignar mediante la opción `-p` en el caso en el que sean parámetros de tipo String, o `-ep` en el caso de que no lo sean.

En el caso en el que sean parámetros de tipo String, sencillamente poner `-p NOMBRE VALOR`, con NOMBRE el nombre del parámetro y VALOR el nombre del valor:

```
bin/engine.sh -p p1 value1 -p p2 value2 flow.cfl
```

En el caso en el que sean parámetros de otro tipo, se usa la opción ep, que interpretará el valor como una expresión de TweakFlow, y luego intentará hacerle un casting al tipo de datos del parámetro:

```
bin/engine.sh -ep flag true -ep date '2019-04-27T' -ep f '(x)->x*x' flow.cfl
```

En este ejemplo tenemos tres parámetros: flag de tipo booleano, date de tipo fecha y f de tipo función.

Otras opciones que permite el programa “engine” es asignar módulos globales y alocar la memoria que se desee. Puede encontrar cómo en la documentación de la herramienta.

14. MÁS INFORMACIÓN

Para más información sobre la herramienta, consulte la documentación:

https://tweakstreet.io/docs/

Para más información sobre el lenguaje Tweakflow, consulte la documentación:

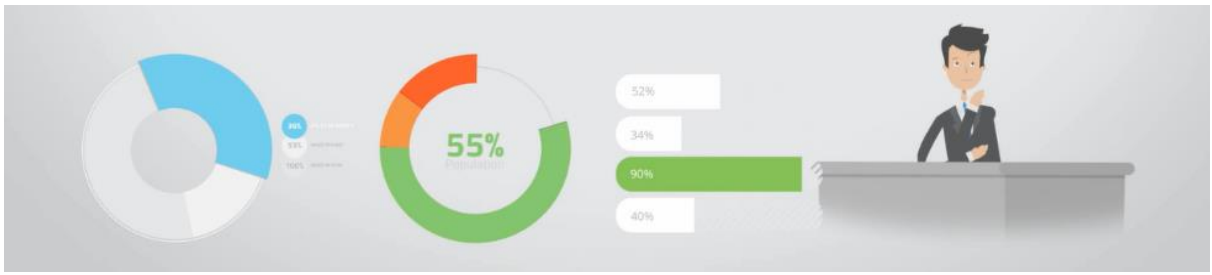
https://twineworks.github.io/tweakflow/index.html

Nota: Las secciones que pueden interesarle son la sección “Getting started” y la sección “Reference”.

15. DESCRIPCIÓN STRATEBI

En Stratebi ofrecemos **gran cantidad de soluciones analíticas** por una compañía de **rápido crecimiento,** innovando en las áreas tecnológicas de mayor desarrollo en la actualidad: **Business Intelligence, Big Data y Social Intelligence**, muchas de ellas, basadas en soluciones **Open Source**.

Además, somos **Partners Certificados en Microsoft PowerBI y Vertica**, con gran número de proyectos con ámbas tecnologías



Desarrollamos nuevas soluciones analíticas basadas en Open Source, para la generación de Cuadros de Mando en tiempo real, con tecnologías IoT para SmartCities, machine learning, etc...

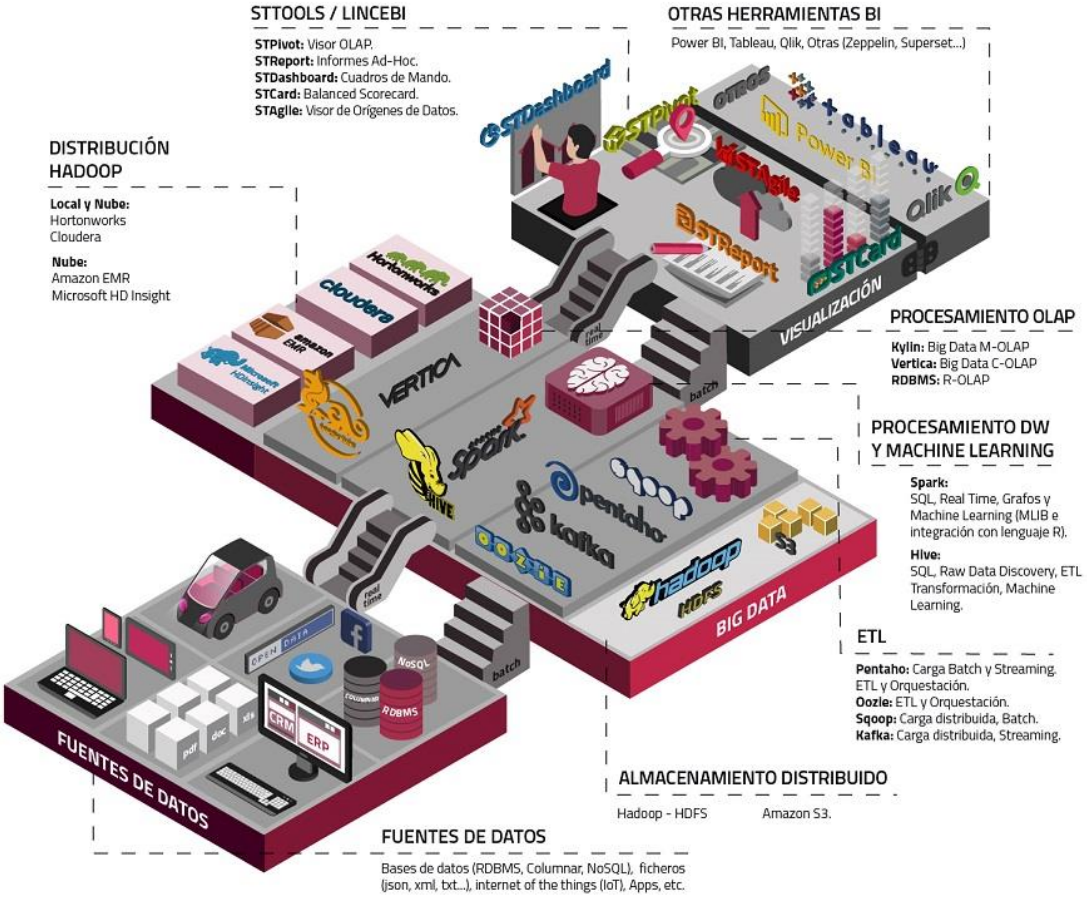
LinceBI
The best open source
BI based solution

16. TECNOLOGÍAS

Recientemente, hemos sido nombrados Partners Certificados de Vertica, Talend, Microsoft, Snowflake, Kylligence, Pentaho, etc…

17. DESCRIPCIÓN DE LAS SOLUCIONES

Stratebi son los creadores de la solución LinceBI.com sobre la que podrás desarrollar e innovar.



18. INFORMACIÓN SOBRE STRATEBI

Stratebi es una empresa española, con sede en Madrid y oficinas en Barcelona, Alicante y Sevilla, creada por un grupo de profesionales con amplia experiencia en sistemas de información, soluciones tecnológicas y procesos relacionados con soluciones de Open Source y de inteligencia de Negocio.

Esta experiencia, adquirida durante la participación en proyectos estratégicos en compañías de reconocido prestigio a nivel internacional, se ha puesto a disposición de nuestros clientes a través de Stratebi.

Stratebi es la única empresa española que ha estado presente todos los Pentaho Developers celebrados en Europa (Mainz-Alemania, Barcelona, Lisboa, Roma, Amsterdam, Sintra, Amberes (2) Londres…), habiendo organizado el de Barcelona.

En Stratebi nos planteamos como **objetivo** dotar a las compañías e instituciones, de herramientas escalables y adaptadas a sus necesidades, que conformen una estrategia Business Intelligence capaz de rentabilizar la información disponible. Para ello, nos basamos en el desarrollo de soluciones de Inteligencia de Negocio, mediante tecnología Open Source.

Stratebi **son profesores y responsables de proyectos del Master en Business Intelligence de la Universidad UOC, UCAM, EOI…**

Los profesionales de Stratebi son los creadores y autores del primer weblog en español sobre el mundo del Business Intelligence, Data Warehouse, CRM, Dashboards, Scorecard y Open Source.

Stratebi es partner de las principales soluciones Analytics: Microsoft PowerBI, Talend, Pentaho, Vertica, Snowflake, Kyligence, Cloudera…

Todo Bi, se ha convertido en una referencia para el conocimiento y divulgación del Business Intelligence en español.

19. REFERENCIAS STRATEBI

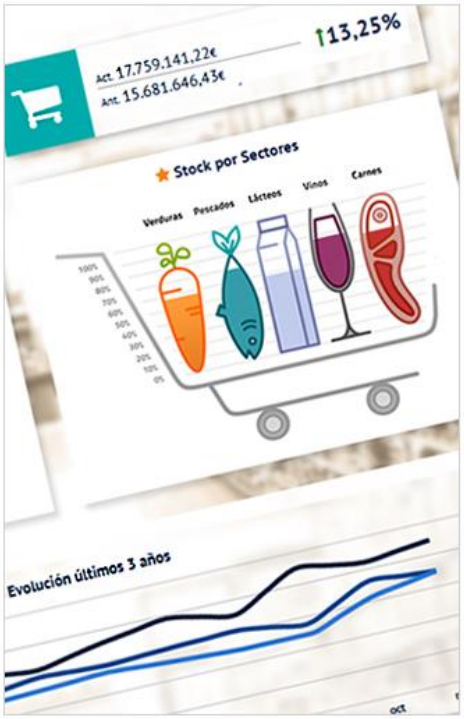
Trabajamos en los principales sectores y con algunas de las compañías y organizaciones más importantes de España.

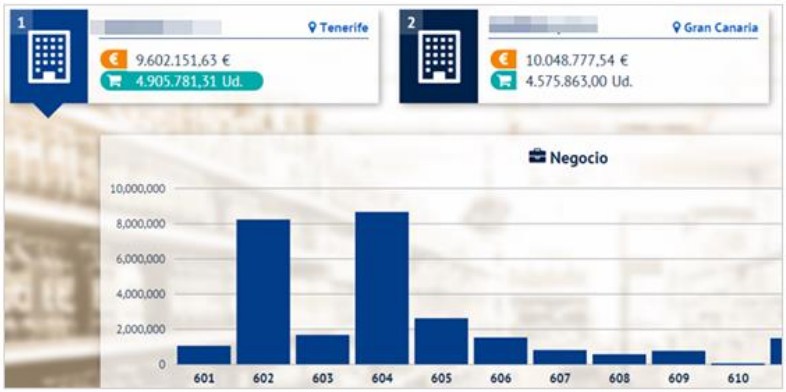
SECTOR PRIVADO

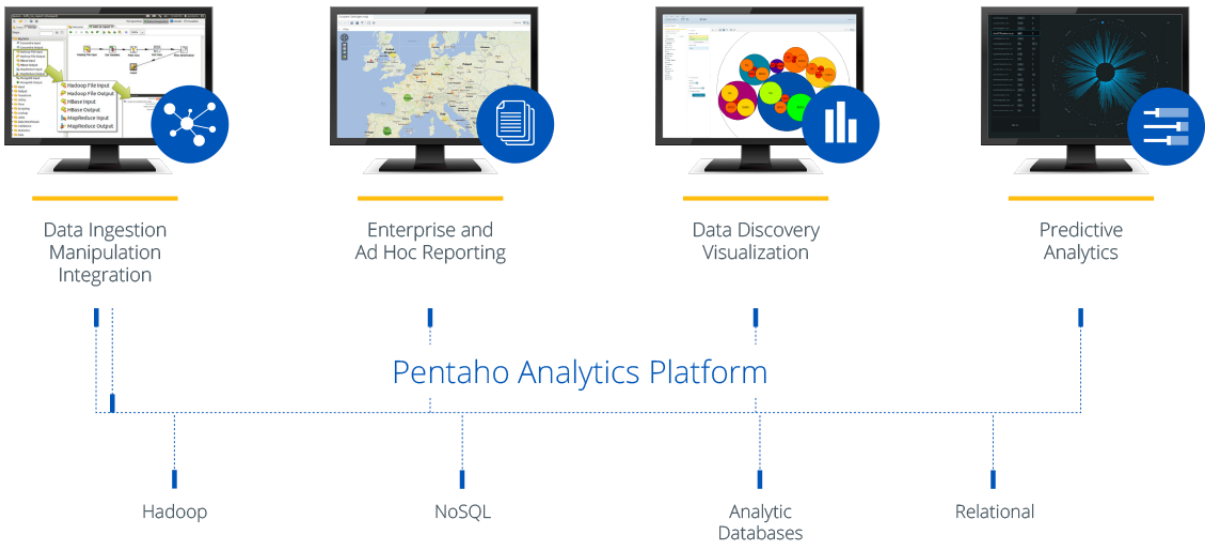
SECTOR PÚBLICO

20. EJEMPLOS DE DESARROLLOS ANALYTICS

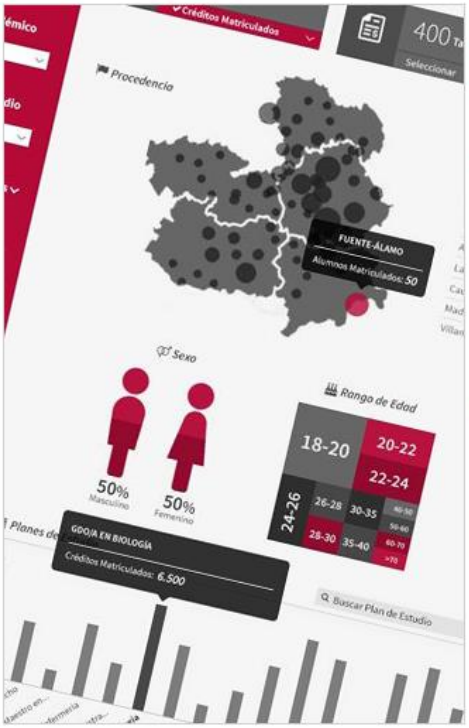
A continuación se presentan **ejemplos de algunos screenshots** de cuadros de mando diseñados por Stratebi, con el fin de dar a conocer lo que se puede llegar a obtener, así como Demos Online en la web de Stratebi:





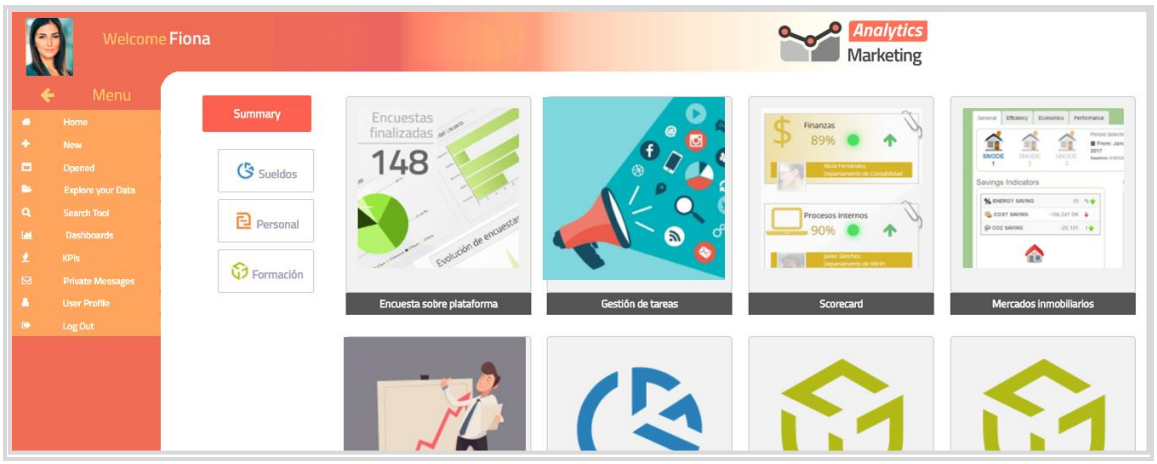



facebook DASHBOARD COMENTARIOS
Sexo
Sentimiento
Localización
Tipo de recurso
Evolución Comentarios por días
smart MADRID
Última Actualización
TOP 5 PUNTOS DE MEDIDA
PROXIMIDAD DE PUNTOS DE MEDIDA
EVOLUCIÓN INTENSIDAD DE TRÁFICO


Créditos Matriculados
Procedencia
FUENTE-ÁLAMO
Alumnos Matriculados: 50
Sexo
50% Masculino
50% Femenino
Rango de Edad
18-20
20-22
22-24
24-26
26-28
28-30
30-35
35-40
GRADO EN BIOLOGÍA
Créditos Matriculados: 6.500
Buscar Plan de Estudio


INTERÉS PLATAFORMA
181 Inscritos en Cursos
8 Desinscritos en Cursos
181 Inicios de Cursos
110 Inscritos en Eventos Virtuales
63 Inscritos en Debates
181 Inscritos en Talleres
USUARIOS ACTIVOS POR PAÍS
INSCRITOS EN CURSOS POR COMUNIDAD EDU


JORNADA COMPLETA
IDIOMAS
TIC
TIPO DE CENTRO
TIPO DE AUSENCIA
Nº de profesores por sexo
36%
64%
Nº total de profesores
1.587
Media de edad
43 años
Búsqueda


Welcome Fiona
Analytics
Marketing
Menu
Home
New
Opened
Explore your Data
Search Tool
Dashboards
KPIs
Private Messages
User Profile
Log Out
Summary
Sueldos
Personal
Formación
Encuestas finalizadas
148
Encuesta sobre plataforma
Gestión de tareas
Finanzas
89%
Procesos internos
90%
Scorecard
Mercados inmobiliarios