

PRACTICA 3

ALMACÉNS E MINARÍA DE DATOS

Alex Baquero Domínguez

alex.baquero@rai.usc.es

Tabla de contenido

| | |
|---|-----------|
| 1. Talend. Añade saída un novo campo de inciencia semanal por millón de habitantes, agrupando por ano e país ou continente | 3 |
| 2. Apache Hop. | 7 |
| <u>a.</u> Número de muertes por país | 7 |
| <u>b.</u> Número de muertes por continente. | 9 |
| <u>c.</u> País con el mayor número de casos por semana. | 10 |
| <u>d.</u> Muertes anuales por países con más de 10 millones de población. | 12 |

1. *Talend. Añade saída un novo campo de inciencia semanal por millón de habitantes, agrupando por ano e país ou continente*

Creamos a conexión da base de datos: COVID.

Database Connection

New Database Connection on repository - Step 2/2

You must press the Check Button to check the Database Setting

DB Type: PostgreSQL

Db Version: v9 and later

String of Connection: jdbc:postgresql://localhost:5433/COVID?

Login: postgres

Password: ****

Server: localhost

Port: 5433

DataBase: COVID

Schema: public

Additional parameters:

Test connection

Export as context Revert Context

[How to install a driver](#)

< Back Next > Finish Cancel

Creamos a table covid coas suas correspondientes columnas e as tablas onde almacenaremos a saída.

```
CREATE TABLE covid(  
    country varchar(40),  
    country_code varchar(5),  
    continent varchar(10),  
    population bigint,  
    indicator_ varchar(5),  
    weekly_count int,  
    year smallint,  
    week smallint,  
    rate_14_day float,  
    cumulative_count int,  
    source_ varchar(60),  
    note varchar(40)
```

);

```
CREATE TABLE covid_IS_AnoPais(
```

```
    country varchar(35),
```

```
    year smallint,
```

```
    weekly_count int
```

);

```
CREATE TABLE covid_IS_Continente(
```

```
    continent varchar(9),
```

```
    year smallint,
```

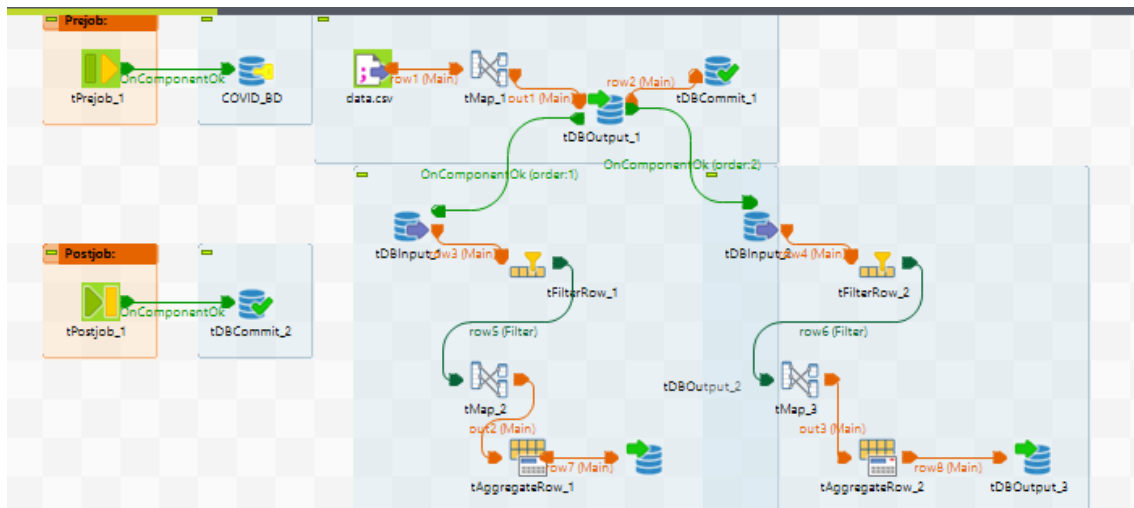
```
    week smallint,
```

```
    weekly_count int,
```

```
    incidencia float
```

);

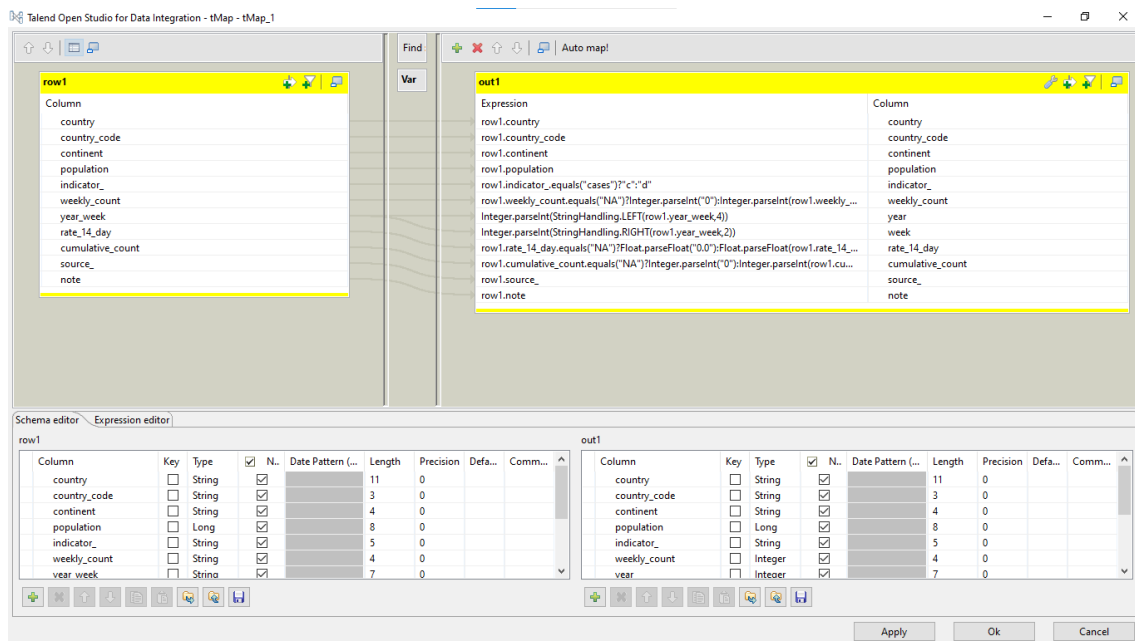
O job de talend quedaría desta maneira:



Primeiramente, conectamos a base de dados que creamos en PostgreSQL. Procedemos a copiar os datos do .csv a través do talend debido a que este na columna de year_week ten un formato extraño. A mellor solución é mapear a táboa e dividir esta columna en dúas: year e week.

```
Integer.parseInt(StringHandling.LEFT(row1.year_week,4))
```

```
Integer.parseInt(StringHandling.RIGHT(row1.year_week,2))
```



Na inserción de datos encontramos un problema. Algunhas columnas de tipo numérico conteñen en algunha fila o dato “NA”. Para iso, poñemos o seguinte:

row1.weekly_count.equals("NA")?Integer.parseInt("0"):Integer.parseInt(row1.weekly_count)

row1.rate_14_day.equals("NA")?Float.parseFloat("0.0"):Float.parseFloat(row1.rate_14_day)

row1.cumulative_count.equals("NA")?Integer.parseInt("0"):Integer.parseInt(row1.cumulative_count)

Almacenamos os datos na base de datos a través do tDBOutput_1 e utilizamos o mesmo para redireccionar o contido as entradas tDBInput_1 e tDBInput_2.

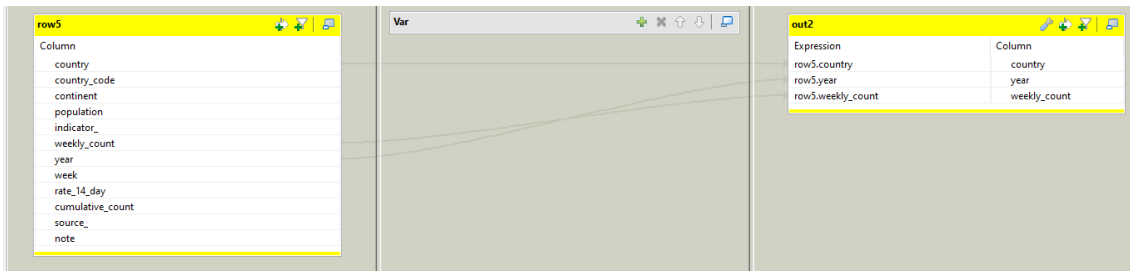
Aquí os camiños da tarefa separanse depende de se queremos agrupalos por ano, país ou por continente.

- Agrupación por ano e país:
 - tFilterRow: Existen datos que son cases e deaths. Interesanos os que son cases.

Conditions

| InputColumn | Function | Operator | Value |
|-------------|----------|----------|-------|
| indicator_ | Empty | Equals | "c" |

- tMap: Quedamos coas columnas que son país, ano e casos semanais. Estas son as columnas que creamos na tabla covid_is_anopais



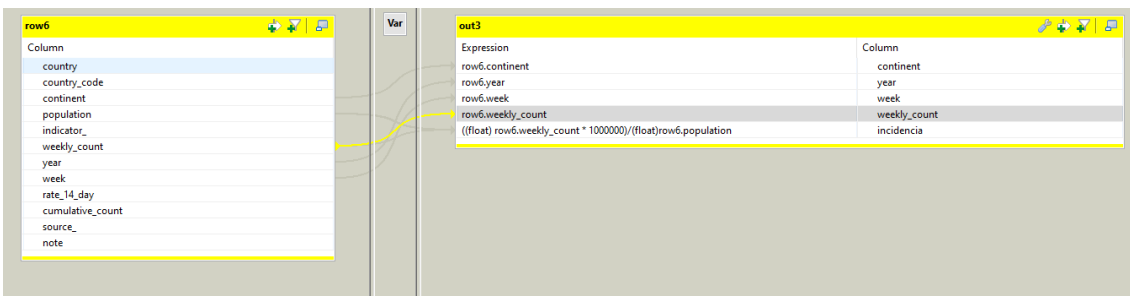
- tAggregateRow: Facemos un group by por ano e pais
- tDBOutput: Almacenamos todos os datos na táboa creada anteriormente "covid_is_anopais"
- Agrupación por continente.
 - tFilterRow: Quitamos os datos que conteñen o dato "NA" pois non nos interesan.

Conditions

| InputColumn | Function | Operator | Value |
|--------------|----------|--------------|-------|
| country_code | Empty | Not equal to | "NA" |

- tMap: Quedamonos cos columnas continentes,ano,semana,casos semanais e añadimos unha nova saída: incidencias. Para calculalo, implementamos o seguinte código:

*$((float) row6.weekly_count * 1000000)/(float)row6.population$*



- tAggregateRow: Facemos un group by por continente
- tDBOutput: Almacenamos todos os datos na tabla creada anteriormente "covid_is_continente"

Toda esta información está gardada en PostgreSQL, entonces facemos un commit na base de datos.

2. Apache Hop.

a. Número de muertes por país

The screenshot shows the Apache Hop interface with a data pipeline consisting of five steps: Table input, Filter rows, Sort rows, Group by, and muertes. Below the pipeline, the 'Metrics' tab is active, displaying a table of performance metrics for each step.

| # | Transform Name | Copy | Input | Read | Written | Output | Updated | Rejected | Errors | Buffers Input | Buffers Output | Duration | Speed | Status |
|---|----------------|------|--------|--------|---------|--------|---------|----------|--------|---------------|----------------|----------|---------|----------|
| 1 | Table input | 0 | 53,073 | 0 | 53,073 | 0 | 0 | 0 | 0 | 0 | 0 | 0.218* | 202,569 | Finished |
| 2 | Filter rows | 0 | 0 | 53,073 | 26,537 | 0 | 0 | 0 | 0 | 0 | 0 | 0.222* | 202,569 | Finished |
| 3 | Sort rows | 0 | 0 | 26,537 | 26,537 | 0 | 0 | 0 | 0 | 0 | 0 | 0.360* | 60,038 | Finished |
| 4 | Group by | 0 | 0 | 26,537 | 228 | 0 | 0 | 0 | 0 | 0 | 0 | 0.090* | 59,903 | Finished |
| 5 | muertes | 0 | 0 | 228 | 228 | 229 | 0 | 0 | 0 | 0 | 0 | 0.110* | 507 | Finished |

Importamos os datos da base de datos do postgresSQL, seleccionando as columnas que vamos a utilizar a través do table input. Despois, collemos os datos que corresponden a mortes, é dicir, os que teñen un “death” (concretamente “d” na nosa base de datos):

The 'Filter rows' dialog box is shown, allowing configuration of the filter condition. The 'Transform name' is 'Filter rows'. The 'Send 'true' data to transform' dropdown is set to 'Sort rows'. The 'Send 'false' data to transform' dropdown is empty. The 'The condition:' section shows a field with 'indicator_' followed by an equals sign and a field with 'd' (String).

Transform name: Filter rows

Send 'true' data to transform: Sort rows

Send 'false' data to transform:

The condition:

indicator_ = d (String)

Buttons: ? Help, OK, Cancel

Finalmente, agrupamos a consulta por países e sumamos as mortes semanais a través “group by”:

The fields that make up the group:

| # | Group field |
|---|-------------|
| 1 | country |

Get

Aggregates:

| # | Name | Subject | Type | Value |
|---|--------|--------------|------|-------|
| 1 | mortes | weekly_count | Sum | |

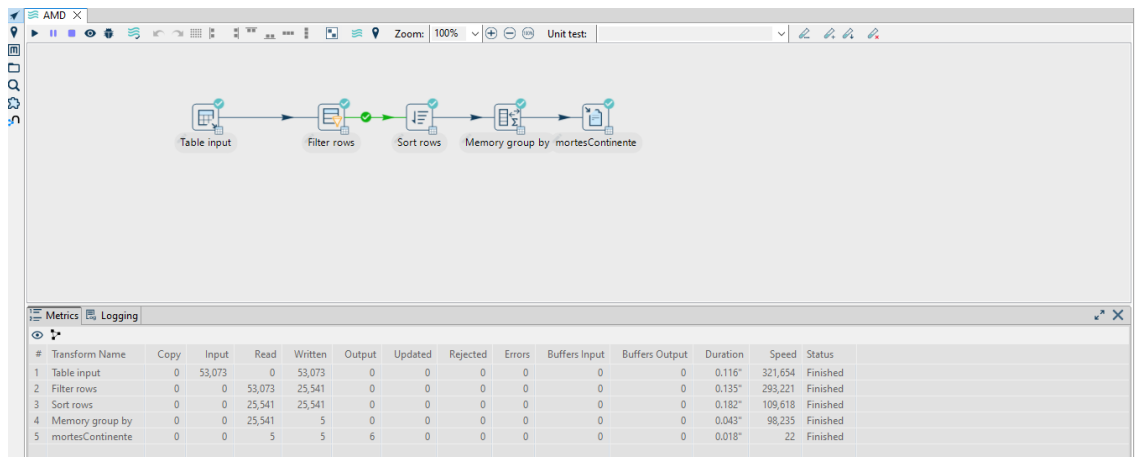
Get lookup

Help OK Cancel

O resultado é o seguinte:

| | A | B | C | D |
|----|----------------|---------|---|---|
| 1 | country | mortes | | |
| 2 | Afghanistan | 7684 | | |
| 3 | Africa (total) | 251517 | | |
| 4 | Albania | 3443 | | |
| 5 | Algeria | 6801 | | |
| 6 | America (tot | 2733254 | | |
| 7 | American Sa | 31 | | |
| 8 | Andorra | 150 | | |
| 9 | Angola | 1866 | | |
| 10 | Anguilla | 8 | | |
| 11 | Antigua And | 139 | | |
| 12 | Argentina | 128679 | | |
| 13 | Armenia | 8545 | | |
| 14 | Aruba | 218 | | |
| 15 | Asia (total) | 1292728 | | |
| 16 | Australia | 8837 | | |
| 17 | Austria | 16245 | | |
| 18 | Azerbaijan | 9539 | | |
| 19 | Bahamas | 812 | | |
| 20 | Bahrain | 1488 | | |
| 21 | Bangladesh | 29117 | | |
| 22 | Barbados | 462 | | |
| 23 | Belarus | 6753 | | |

b. Número de muertes por continente.



Importamos os datos da base de datos do postgresSQL seleccionando as columnas que vamos a utilizar a través do table input. Despois, collemos os datos que corresponden a mortes, é dicir, os que teñen un “death” (concretamente “d” na nosa base de datos) e os que non teñen un código que é igual a NA (eliminamos os países que inclúen a todo un continente):

The condition:

```
indicator_ = [d]  
  
AND  
  
NOT ( country_code CONTAINS [NA] )
```

Usamos un “group by” para agrupar a consulta por continente e sumar as mortes semanais e utilizamos un “text file output” para gardar todo nun .csv de saída.

Memory group by

Transform name

Always give back a result row ☐

The fields that make up the group:

| # | Group field |
|---|-------------|
| 1 | continent |

Get Fields

Aggregates :

| # | Name | Subject | Type | Value |
|---|--------|--------------|------|-------|
| 1 | mortes | weekly_count | Sum | |

Get lookup fields

Help OK Cancel

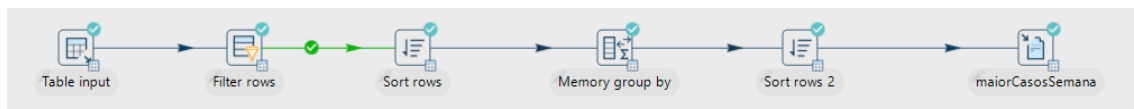
O resultado seria o seguinte:

| | | |
|----------------------|------------------|------|
| mortes..csv | 30/10/2022 18:02 | Arcl |
| mortesContinente.csv | 30/10/2022 18:23 | Arcl |

mortesContinente.csv: caderno de notas

| Ficheiro | Editar | Formato | Ver | Axuda |
|------------------|--------|---------|-----|----------|
| continent;mortes | | | | |
| Asia | | | | ;1292728 |
| Africa | | | | ;247541 |
| Oceania | | | | ;13233 |
| America | | | | ;2733211 |
| Europe | | | | ;1974218 |

c. País con el mayor número de casos por semana.



Collese a información do postgresSQL a través da Table Input e filtramos os datos de tal maneira que non haxa deaths e os a columna country non conteña un país (total).

```

indicator_ = [d]


AND

NOT ( country CONTAINS [(total)] )

```

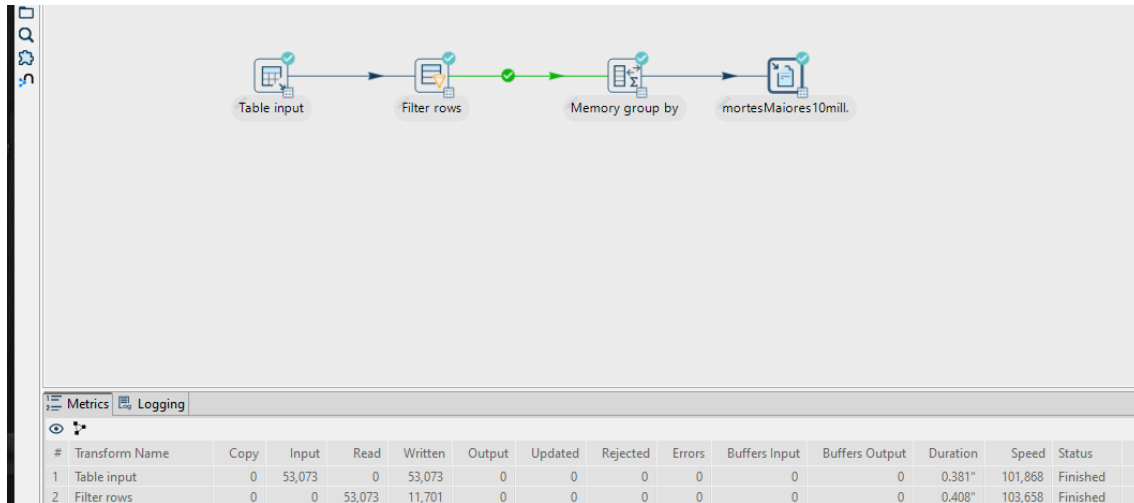
Organizamos a saída para que tenhamos os primeiros países com maior casos em cada semana poñendo a year e week de forma ascendente e a weekly_count de forma descendente. Colocamos un memory group by para agrupar cada fila pola sua data. Mandamos ao seguinte sort o country e weekly_count xa que a primeira fila é o máximo grazas ao sort anterior.

Finalmente, imprímese nun arquivo .csv:

 maiorCasosSemana.csv: caderno de notas

| Ficheiro | Editar | Formato | Ver | Axuda |
|---|--------|---------|-----|-------|
| year;week;weekly_count;country | | | | |
| 2020;01;59;China | | | | |
| 2020;02;12;Denmark | | | | |
| 2020;03;176;China | | | | |
| 2020;04;2540;China | | | | |
| 2020;05;14436;China | | | | |
| 2020;06;22995;China | | | | |
| 2020;07;30412;China | | | | |
| 2020;08;6616;China | | | | |
| 2020;09;3450;South Korea | | | | |
| 2020;10;8241;Italy | | | | |
| 2020;11;21095;Italy | | | | |
| 2020;12;36642;Italy | | | | |
| 2020;13;107819;United States Of America | | | | |
| 2020;14;194610;United States Of America | | | | |
| 2020;15;219936;United States Of America | | | | |
| 2020;16;202116;United States Of America | | | | |

d. Muertes anuales por países con más de 10 millones de población.



O esquema do pipeline é moi similar aos todos os anteriores. Primeiro seleccionamos as columnas das tablas que necesitamos a través do table input:

SELECT country, population, indicator_, weekly_count, year FROM "public".covid_talend

No filter rows, poñemos da mesma maneira que se mostra nos anteriores apartados, que o indicator_ sexa igual a d e que population sexa maior que 10 millions.

No group by, agrupamos por país e ano realizando unha operación que sume todos os datos presentes na columna weekly_count. Finalmente almacenase nun arquivo .csv

