

SOLUCIÓN PRAC 2
DISEÑO Y USO DE BASES DE DATOS ANALÍTICAS



Santiago Domínguez Collado.

1. Identificación de los procesos ETL's.

Para la identificación de los procesos ETL vamos a seguir las mismas directrices que en el ejemplo facilitado en los recursos de esta práctica.

Cabe destacar que, en este caso, no se dispondrá de una base de datos que haga el papel de staging area si no que las tablas irán todas a la misma base.

En nuestro caso identificamos los procesos ETL's en dos bloques y utilizaremos un prefijo en el nombre para identificarlos:

- Bloque IN: procesos de carga de los datos desde las fuentes a tablas intermedias. Estos procesos se distinguen por el prefijo: IN_ en el nombre.
- Bloque TR: procesos de transformación para la carga de datos desde tablas intermedias a nuestro almacén según el modelo multidimensional diseñado. Se diferencian los procesos ETL de transformación para la carga de dimensiones, de los procesos de transformación para la carga de las tablas de hecho. Estos procesos se distinguen con el prefijo TR_ en el nombre.

A. Bloque IN (de las fuentes a tablas intermedias)

Nombre ETL	Descripción
IN_COUNTRY	Carga de registros de la fuente COUNTRY.json a la tabla IN_COUNTRY
IN_MORTCAUSE	Carga de registros de la fuente REGION.json a la tabla IN_REGION
IN_REGION	Carga de registros de la fuente REGION.json a la tabla IN_REGION
IN_SARAMPION	Carga de los ficheros: Sarampion1.xml, Sarampion2.xml, Sarampion3.xml y Sarampion5.xml correspondientes a los casos, muertes, hospitalizaciones y pruebas del sarampión en los países de la OMS a la tabla IN_SARAMPION
IN_WUE_DATA	Carga del fichero wuenic2018rev_data_2019-11-16.csv a la tabla IN_WUE_DATA. Contiene datos de cobertura de población infantil y objetivo.
IN_COV_SERIES	Carga del fichero con información cerca de la cobertura de vacunas en los países miembros de la OMS, coverage_estimates_series.xls a la tabla IN_COV_SERIES.

B. Bloque TR (poblar las tablas de nuestro almacén)

Nombre ETL	Descripción
TR_DIM_ANIO	Carga y transformación de la dimensión año (DIM_ANIO).
TR_DIM_PAIS	Carga y transformación de la dimensión país (DIM_PAIS).

TR_DIM_GEOGRAFÍA	Carga y transformación de la dimensión geografía (DIM_GEOGRAFÍA).
TR_DIM_VACUNAS	Carga y transformación de la dimensión vacunas (DIM_VACUNAS).
TR_DIM_CAUSAS_MORTALIDAD	Carga y transformación de la dimensión de causas de mortalidad (DIM_VACU DIM_CAUSAS_MORTALIDAD NAS).

Nombre ETL	Descripción
TR_FACT_COBERTURA	Carga y transformación de la tabla de hechos FACT_COBERTURA.
TR_FACT_COBERTURA_SARAMPION	Carga y transformación de la tabla de hechos FACT_COBERTURA_SARAMPION.

2. Diseño de los procesos ETL's.

En este apartado, y teniendo en cuenta las consideraciones anteriores, vamos a diseñar e implementar los procesos de carga mediante la herramienta de diseño proporcionada: Pentaho Data Integration (PDI). Y en particular, el programa de escritorio llamado Spoon, que corresponde al entorno gráfico (IDE) de desarrollo de ETL's.

A. Creación de tablas Intermedias.

Será necesario crear previamente tablas en las que alojar la información de las fuentes.

IN_COUNTRY

```
CREATE TABLE [dbo].[IN_COUNTRY](
    [ISO_CODE] [varchar](50) NOT NULL,
    [COUNTRY] [varchar](150) NOT NULL,
    [CATEGORY] [varchar](300) NOT NULL,
    [VALUE] [varchar](300) NOT NULL,
) ON [PRIMARY]
GO
```

IN_MORTCAUSE

```
CREATE TABLE [dbo].[IN_MORTCAUSE](
    [ID] [numeric](5) NOT NULL,
    [DISEASE] [varchar](300) NOT NULL
) ON [PRIMARY]
GO
```

IN_REGION

```
CREATE TABLE [dbo].[IN_REGION](
    [CODE] [varchar](50) NOT NULL,
    [REGION] [varchar](150) NOT NULL,
) ON [PRIMARY]
GO
```

IN_SARAMPION

```
CREATE TABLE [dbo].[IN_SARAMPION](
    [COD] [numeric](10,0) NOT NULL,
    [NOMBRE] [varchar](100) NOT NULL,
    [ANIO] [numeric](15,0) NOT NULL,
    [CASOS_CONFIRMADOS] [numeric](38,0) NULL,
    [MUERTES] [numeric](38,0) NULL,
    [HOSPITALIZACIONES] [numeric](38,0) NULL,
    [MUESTRAS_LABORATORIO] [numeric](38,0) NULL,
) ON [PRIMARY]
GO
```

IN_WUE_DATA

```
CREATE TABLE [dbo].[IN_WUE_DATA](
    [GROUP] [varchar](50) NOT NULL,
    [SUBGROUP] [varchar](50) NOT NULL,
    [NAME] [varchar](50) NOT NULL,
    [YEAR] [numeric](15) NOT NULL,
    [VACCINE] [varchar](50) NOT NULL,
    [COVERAGE] [numeric](15, 0) NOT NULL,
    [VACCINATED] [varchar](50) NOT NULL,
    [TARGET] [varchar](50) NOT NULL,
    [SOURCE] [varchar](200) NOT NULL,
) ON [PRIMARY]
GO
```

IN_COV_SERIES

```
CREATE TABLE [dbo].[IN_COV_SERIES](
    [WHO_REGION] [varchar](100) NOT NULL,
    [ISO_code] [varchar](3) NOT NULL,
    [Cname] [varchar](30) NOT NULL,
    [Continent] [varchar](30) NOT NULL,
    [Vaccine] [varchar](300) NOT NULL,
    [Year] [numeric](4, 0) NOT NULL,
    [Percent_covrage] [numeric](2, 0) NOT NULL,
    [Asterisc] [varchar](1) NULL,
) ON [PRIMARY]
GO
```

B. Creación del modelo multidimensional.

En este punto veremos los scripts de creación del modelo físico multidimensional.

En la creación, además de atributos y métricas, se crearán también las restricciones definidas y que son propias del modelo multidimensional, las claves primarias de las dimensiones y las foráneas de las tablas de hechos.

a) Dimensiones

DIM_ANIO

```
CREATE TABLE [dbo].[DIM_ANIO](
    [SK_DIM_ANIO] [numeric](4, 0) NOT NULL,
    [DESC_ANIO] [varchar](50) NOT NULL,
    CONSTRAINT [PK_DIM_ANIO] PRIMARY KEY CLUSTERED
(
    [SK_DIM_ANIO] ASC
)
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

DIM_PAIS

```
CREATE TABLE [dbo].[DIM_PAIS](
    [SK_DIM_PAIS] [numeric](3, 0) NOT NULL,
    [ISO_CODE] [varchar](3) NOT NULL,
    [NOMBRE_PAIS] [varchar](30) NOT NULL,
    [WORLD_BANK_INCOME] [numeric](3, 0) NOT NULL,
    [WHO_REGION_CODE] [varchar](4) NOT NULL,
    [WHO_REGION ] [varchar](100) NOT NULL,
    [ISO2_CODE] [varchar](3) NOT NULL,
    [SK_DIM_CAUSA] [numeric](2, 0) NOT NULL,
    [LAND_AREA] [varchar](15) NOT NULL,
    [LANGUAGES_EN_2012] [varchar](300) NOT NULL,
    CONSTRAINT [PK_DIM_PAIS] PRIMARY KEY CLUSTERED
(
    [SK_DIM_PAIS] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[DIM_PAIS] WITH CHECK ADD CONSTRAINT
[FK_DIM_PAIS_DIM_CAUSA] FOREIGN KEY([SK_DIM_CAUSA])
REFERENCES [dbo].[DIM_CAUSAS_MORTALIDAD] ([SK_DIM_CAUSA])
GO
```

```
ALTER TABLE [dbo].[DIM_PAIS] CHECK CONSTRAINT
[FK_DIM_PAIS_DIM_CAUSA]
GO
```

DIM_GEOGRAFIA

```
CREATE TABLE [dbo].[DIM_GEOGRAFIA](
    [SK_DIM_GEOGRAFIA] [numeric](3, 0) NOT NULL,
    [COD_REGION] [varchar](4) NOT NULL,
    [DESC_REGION] [varchar](30) NOT NULL,
    [SK_DIM_PAIS] [numeric](3, 0) NOT NULL,
    [DESC_PAIS] [varchar](300) NOT NULL,
    CONSTRAINT [PK_DIM_GEOGRAFIA] PRIMARY KEY CLUSTERED
(
    [SK_DIM_GEOGRAFIA] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
ALTER TABLE [dbo].[DIM_GEOGRAFIA] WITH CHECK ADD CONSTRAINT
[FK_DIM_GEOGRAFIA_DIM_PAIS] FOREIGN KEY([SK_DIM_PAIS])
REFERENCES [dbo].[DIM_PAIS] ([SK_DIM_PAIS])
GO
```

```
ALTER TABLE [dbo].[DIM_GEOGRAFIA] CHECK CONSTRAINT
[FK_DIM_GEOGRAFIA_DIM_PAIS]
GO
```

DIM_VACUNA

```
CREATE TABLE [dbo].[DIM_VACUNA](
    [SK_DIM_VACUNA] [numeric](3, 0) NOT NULL,
    [TIPO_VACUNA] [varchar](50) NOT NULL,
    [NOMBRE_VACUNA] [varchar](300) NOT NULL,
    [GVAP] [varchar](1) NOT NULL,
    CONSTRAINT [PK_DIM_VACUNA] PRIMARY KEY CLUSTERED
(
    [SK_DIM_VACUNA] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

DIM_CAUSAS_MORTALIDAD

```
CREATE TABLE [dbo].[DIM_CAUSAS_MORTALIDAD](
    [SK_DIM_CAUSA] [numeric](2, 0) NOT NULL,
    [COD_CAUSA] [varchar](4) NOT NULL,
    [DESC_CAUSA] [varchar](100) NOT NULL,
    CONSTRAINT [PK_DIM_CAUSA] PRIMARY KEY CLUSTERED
(
    [SK_DIM_CAUSA] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
ON [PRIMARY]
) ON [PRIMARY]
GO
```

b) Tablas de hechos.

FACT_COBERTURA

```
CREATE TABLE [dbo].[FACT_COBERTURA](
    [SK_DIM_ANIO] [numeric](4, 0) NOT NULL,
    [SK_DIM_GEOGRAFIA] [numeric](3, 0) NOT NULL,
    [SK_DIM_VACUNA] [numeric](3, 0) NOT NULL,
    [COBERTURA] [numeric](2, 0) NOT NULL,
    [NUM_PV] [numeric](8, 0) NULL,
    [OBJETIVO] [numeric](8, 0) NULL,
)
GO

ALTER TABLE [dbo].[FACT_COBERTURA] WITH CHECK ADD CONSTRAINT
[FK_ANIO] FOREIGN KEY([SK_DIM_ANIO])
REFERENCES [dbo].[DIM_ANIO] ([SK_DIM_ANIO])
GO
ALTER TABLE [dbo].[FACT_COBERTURA] CHECK CONSTRAINT [FK_ANIO]
GO

ALTER TABLE [dbo].[FACT_COBERTURA] WITH CHECK ADD CONSTRAINT
[FK_GEOGRAFIA] FOREIGN KEY([SK_DIM_GEOGRAFIA])
REFERENCES [dbo].[DIM_GEOGRAFIA] ([SK_DIM_GEOGRAFIA])
GO
ALTER TABLE [dbo].[FACT_COBERTURA] CHECK CONSTRAINT [FK_GEOGRAFIA]
GO

ALTER TABLE [dbo].[FACT_COBERTURA] WITH CHECK ADD CONSTRAINT
[FK_VACUNA] FOREIGN KEY([SK_DIM_VACUNA])
REFERENCES [dbo].[DIM_VACUNA] ([SK_DIM_VACUNA])
GO
ALTER TABLE [dbo].[FACT_COBERTURA] CHECK CONSTRAINT [FK_VACUNA]
```

GO

FACT_COBERTURA_SARAMPION

```
CREATE TABLE [dbo].[FACT_SARAMPION](
    [SK_DIM_ANIO] [numeric](4, 0) NOT NULL,
    [SK_DIM_GEOGRAFIA] [numeric](3, 0) NOT NULL,
    [NUM_CASOS] [numeric](8, 0) NULL,
    [NUM_MUERTES] [numeric](8, 0) NULL,
    [NUM_HOSPITALIZACIONES] [numeric](8, 0) NULL,
    [NUM_CASOS_CONFIRMADOS] [numeric](8, 0) NULL,
)
GO
```

```
ALTER TABLE [dbo].[FACT_SARAMPION] WITH CHECK ADD CONSTRAINT
[FK_ANIO_SARAMPION] FOREIGN KEY([SK_DIM_ANIO])
REFERENCES [dbo].[DIM_ANIO] ([SK_DIM_ANIO])
GO
ALTER TABLE [dbo].[FACT_SARAMPION] CHECK CONSTRAINT
[FK_ANIO_SARAMPION]
GO
```

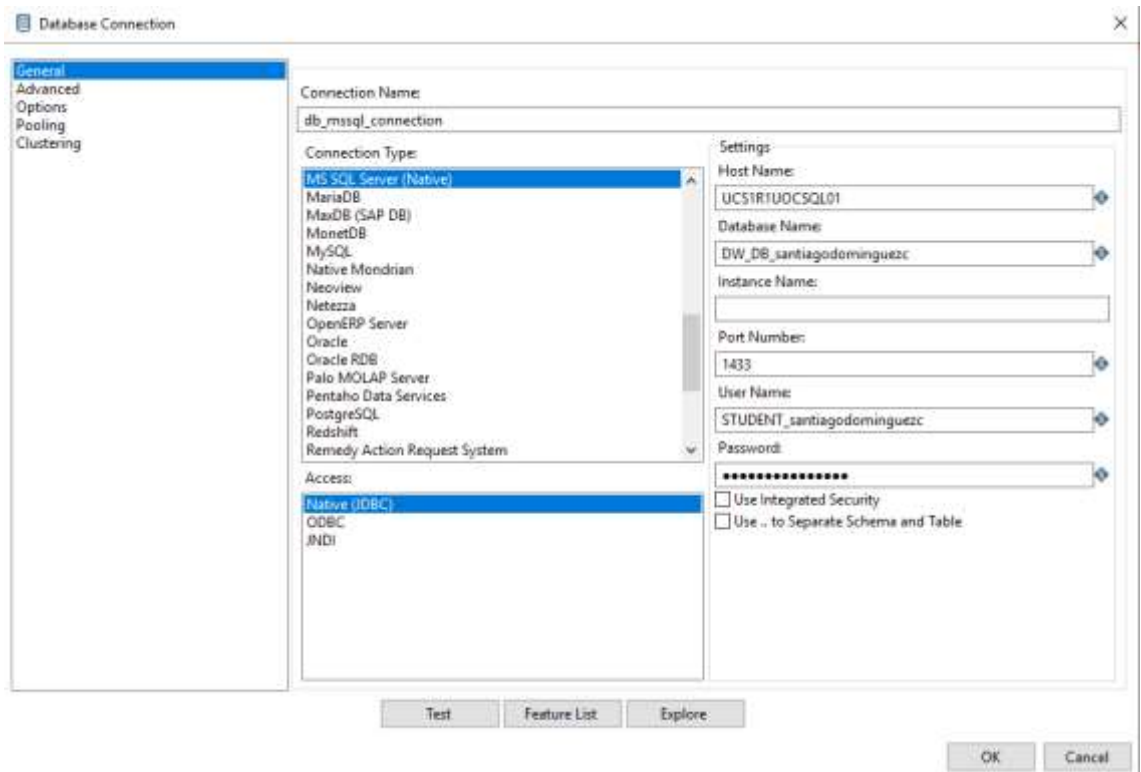
```
ALTER TABLE [dbo].[FACT_SARAMPION] WITH CHECK ADD CONSTRAINT
[FK_GEOGRAFIA_SARAMPION] FOREIGN KEY([SK_DIM_GEOGRAFIA])
REFERENCES [dbo].[DIM_GEOGRAFIA] ([SK_DIM_GEOGRAFIA])
GO
ALTER TABLE [dbo].[FACT_SARAMPION] CHECK CONSTRAINT
[FK_GEOGRAFIA_SARAMPION]
GO
```

C. Creación del proceso de extracción, transformación y carga (ETL).

Una vez que tenemos implementado el modelo físico del almacén, pasaremos a diseñar los procesos ETL que permitirán poblar las tablas intermedias, las tablas dimensiones y de hechos.

A) Conexión Base de Datos SQL Server

Durante todas las transacciones de la herramienta ETL será necesario configurar el acceso a la base de datos. Es por ello por lo que aquí mostramos el paso, para no repetirlo constantemente.



B) Bloque IN

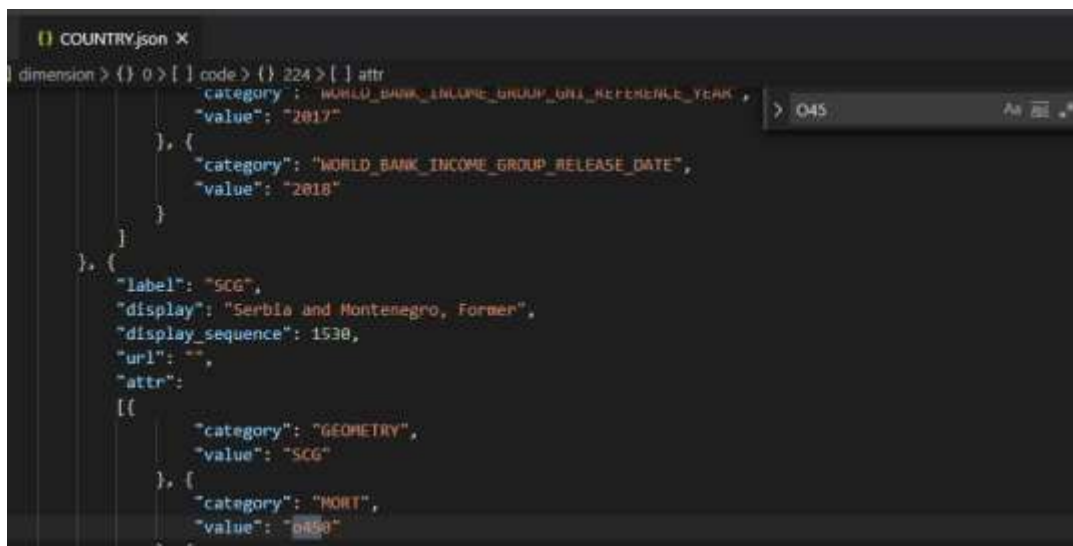
Transformación IN_COUNTRY

El primer proceso por desarrollar es la carga de la fuente información sobre los países miembro de la OMS a la tabla intermedia IN_COUNTRY. Partimos del fichero: COUNTRY.json

El proceso IN_RAMa contiene cinco transformaciones: Dos lecturas de json, Selección de valores, Operación de cadena y Carga a la tabla intermedia IN_COUNTRY.

Previo a mostrar cada uno de los pasos en la transformación, durante el desarrollo se detecto una errata en uno de los valores a cargar, donde debería ir 0450, estaba escrito o450, lo que alteraba el tipo del atributo.

Dicho esto, lo reemplazamos haciendo uso del editor de texto.



Una vez tratado el fichero procedemos a cargar el json con el step Input Json.

Step name: **JSON Input**

File | Content | Fields | Additional output fields

Source from field

Source is from a previous step: ☒

Select fields:

Use field as file names: ☐

Read source as URL: ☐

Do not pass field downstream: ☐

File or directory: Add Browse

Regular Expression:

Exclude Regular Expression:

Selected files:

#	File/Directory	Wildcard (RegExp)	Exclude wildcard	Required	Inclu
1	F:\Fuentes\COUNTRY.json			N	N

< >

Show filename(s)...

OK Preview rows Cancel

Help

Basándonos en los ejemplos que facilita la propia herramienta podemos especificar el path de cada variable para que el step extraiga los datos como queramos.

Step name: **JSON Input**

File | Content | **Fields** | Additional output fields

#	Name	Path	Type	Format	Length	Precision	Currency	Decimal	G
1	ISO_CODE	\$.dimension..code.label	String						
2	COUNTRY	\$.dimension..code.display	String						
3	ATTR	\$.dimension..code.attr	String						

OK Preview rows Cancel

Help

En este caso el json en cuestión requiere otro step del mismo tipo, esto ocurre cuando los ficheros tienen a tributos en mayo profundidad.

Json input

Step name: **JSON Input 2**

#	Name	Path	Type	Format	Length	Precision	Currency	Decimal	Group	Trim
1	CATEGORY	\$.category	string							non
2	VALUE	\$.value	string							non

OK Preview logs Cancel

Help

Ponemos todos los valores en mayúscula para evitar problemas posteriores al relacionar tablas.

String operations

Step name: **String operations**

The fields to process:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits
1	ISO_CODE		none	upper	none			N	None	none
2	COUNTRY		none	upper	none			N	None	none
3	CATEGORY		none	upper	none			N	None	none
4	VALUE		none	upper	none			N	None	none

OK Get fields Cancel

Help

Por último, cargamos los valores en la tabla previamente creada en el servidor.

Table output

Step name: **Table output**

Connection: **bd_mssql_connection** Edit... New... Wizard...

Target schema: Browse...

Target table: **IN_COUNTRY** Browse...

Commit size: **1000**

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

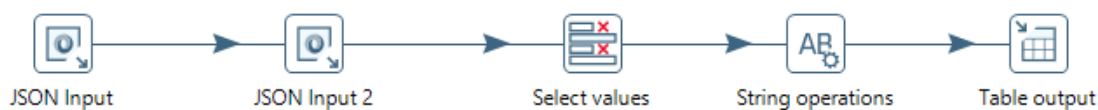
Fields to insert:

#	Table field	Stream field
1	ISO_CODE	ISO_CODE
2	COUNTRY	COUNTRY
3	CATEGORY	CATEGORY
4	VALUE	VALUE

Get fields

Enter field mapping

La transformación completa es la siguiente:



Transformación IN_MORTCAUSE

Nuevamente extraemos y cargamos la información desde un json.

La transformación consta de tres pasos: Lectura de fichero, Operación de cadea y Carga en la tabla intermedia IN_MORTCAUSE.

Previamente ha sido necesario modificar el primer registro del JSON pues no aportaba información y no era del tipo correcto. El registro con los valores a 0 será eliminado mediante una query desde el propio MSSQL.

```
{
  "label": "0",
  "display": "0",
  "isMeasure": false,
  "code":
  [
    {
      "label": "0000",
      "display": "All causes",
```

Una vez tratado el fichero procedemos a cargar el json con el step Input Json.

Captura de pantalla de la configuración del step "JSON Input".

Step name: JSON Input

File Content Fields Additional output fields

Source from field:

- Source is from a previous step: ☐
- Select field:
- Use field as file names: ☐
- Read source as URL: ☐
- Do not pass field downstream: ☐

File or directory: Add Browse

Regular Expression:

Exclude Regular Expression:

Selected files:

#	File/Directory	Wildcard (RegEx)	Exclude wildcard	Required	In
1	F:\Fuentes\MORTCAUSE.json			N	N

Show filename(s)...

Buttons: OK, Preview rows, Cancel, Help

Especificamos el path de cada variable para que el step extraiga los datos como queramos.

Json input

Step name: **JSON input**

#	Name	Path	Type	Format	Length	Precision	Currency	Decimal	Group
1	LABEL	\$.dimension_label	Number						
2	MORTCAUSE	\$.dimension_display	String						

Help OK Preview rows Cancel

Transformamos el único atributo de tipo cadena poniéndolo en mayúsculas.

String operations

Step name: **String operations**

The fields to process:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits
1	MORTCAUSE		none	upper	none			N	None	none

Help OK Get fields Cancel

Por último, cargamos los valores en la tabla previamente creada en el servidor.

Table output

Step name: **Table output**

Connection: db_mssql_connection Edit... New... Wizard...

Target schema: Browse...

Target table: IN_MORTCAUSE Browse...

Commit size: 1000

Truncate table ☒

Ignore insert errors ☐

Specify database fields ☒

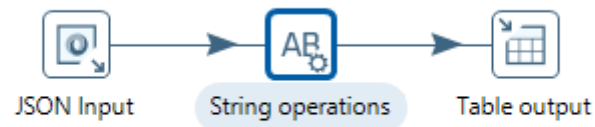
Main options Database fields

Fields to insert:

#	Table field	Stream field
1	ID	LABEL
2	DISEASE	MORTCAUSE

Get fields Enter field mapping

La transformación completa es la siguiente:



Transformación IN_REGION

Extrae la información del archivo json con información sobre las regiones de la OMS y carga los en la tabla intermedia IN_REGION.

Previamente se ha modificado el primer registro del caso anterior por las mismas razones. Se eliminará posteriormente con una query desde el propio MSSQL.

```
{ } MORTCAUSE.json    { } REGION.json ●
{ } REGION.json > [ ] dimension > { } 0 > [ ] display
1  {
2  "copyright": "(c) World Health Organization",
3  "dataset":
4  [
5  ],
6  "attribute":
7  [
8  ],
9  "dimension":
10 [
11 {
12 "label": "ELIMINAR",
13 "display": "ELIMINAR",
14 "isMeasure": false,
15 "code":
16 [
17 {
18 "label": "AFR",
19 "display": "Africa",
20 "display_sequence": 10,
21 "url": "",
```

Una vez tratado el fichero procedemos a cargar el json con el step Input Json. Especificamos el path o la ruta de sus atributos.

Json input

Step name

File	Content	Fields	Additional output fields	
#	Name	Path	Type	Format
1	LABEL	\$.dimension..label	String	
2	REGION	\$.dimension..display	String	

Por último, cargamos los valores en la tabla previamente creada en el servidor.

Table output

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

Specify database fields ☒

Main options		Database fields	
Fields to insert:			
#	Table field	Stream field	
1	CODE	LABEL	
2	REGION	REGION	

La transformación completa es la siguiente:



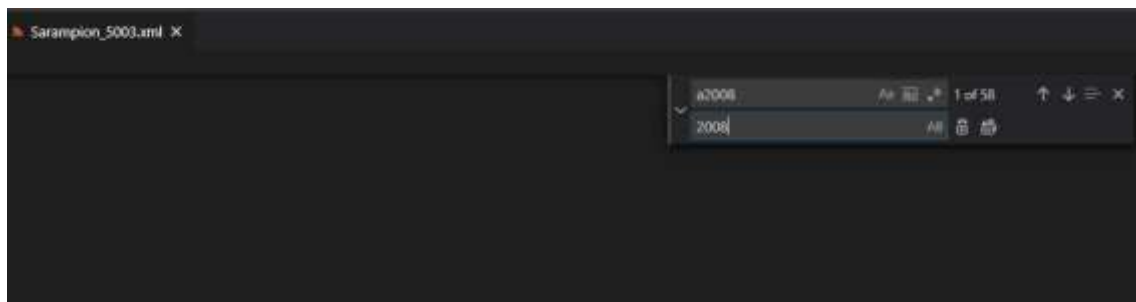
Transformación IN_SARAMPION

La siguiente transformación es la más compleja de todas. Consiste en cargar simultáneamente 4 ficheros xml en la misma tabla, de manera que cada uno de ellos quede reducido a una columna.

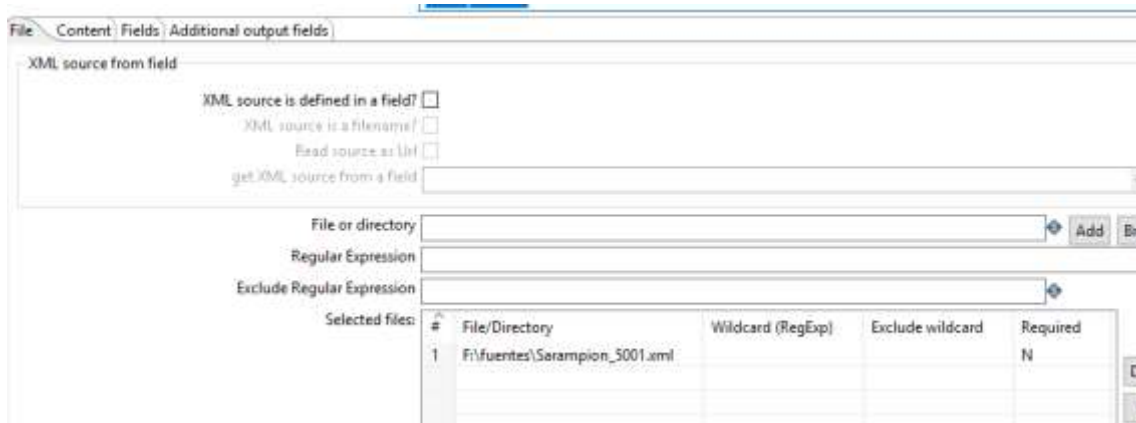
- El primer fichero se corresponde con los casos de sarampión.
- El segundo fichero se corresponde con las muertes por sarampión.
- El tercer fichero se corresponde con las hospitalizaciones por sarampión.
- El cuarto fichero se corresponde con los casos confirmados en laboratorio de sarampión.

La transformación tiene 17 pasos: 4 lecturas de ficheros, 4 operaciones de cadena, cuatro normalizadores de fila, 3 joins, 1 selección de valores y la carga en la tabla intermedia IN_Sarampion.

Previo a esto, los ficheros expresan los años con una letra delante, utilizamos el editor de texto para removerla.



Comenzamos con la lectura del fichero, las cuatro lecturas son idénticas salvo por la ruta de la fuente. De manera que este paso solo se mostrará una vez.



En la lectura del fichero es necesario especificar la ruta y el tipo de los atributos. En este caso es mucho más intuitivo que con los archivos json.

File	Content	Fields	Additional output fields			
#	Name	XPath	Element	Result type	Type	Format
1	cod	cod	Node	Value of	Integer	
2	nombre	nombre	Node	Value of	String	
3	2008	a2008	Node	Value of	Integer	
4	2009	a2009	Node	Value of	Integer	
5	2010	a2010	Node	Value of	Integer	
6	2011	a2011	Node	Value of	Integer	
7	2012	a2012	Node	Value of	Integer	
8	2013	a2013	Node	Value of	Integer	
9	2014	a2014	Node	Value of	Integer	
10	2015	a2015	Node	Value of	Integer	
11	2016	a2016	Node	Value of	Integer	
12	2017	a2017	Node	Value of	Integer	
13	2018	a2018	Node	Value of	Integer	

Ponemos en mayúscula el único valor en cada fichero que es de tipo cadena.

Step name

The fields to process:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padc
1	nombre		none	upper	none

Normalización: en este caso queremos sintetizar todos los campos de los años en uno solo para que posteriormente la inserción en la tabla FACT_SARAMPION sea mucho más sencilla. De este modo cada fichero reemplaza los campos de sus años por uno nuevo.

Row Normaliser

Step name Row Normaliser

Type field ANIO

Fields

#	Fieldname	Type	new field
1	2008	2008	CASOS_CONFIRMADOS
2	2009	2009	CASOS_CONFIRMADOS
3	2010	2010	CASOS_CONFIRMADOS
4	2011	2011	CASOS_CONFIRMADOS
5	2012	2012	CASOS_CONFIRMADOS
6	2013	2013	CASOS_CONFIRMADOS
7	2014	2014	CASOS_CONFIRMADOS
8	2015	2015	CASOS_CONFIRMADOS
9	2016	2016	CASOS_CONFIRMADOS
10	2017	2017	CASOS_CONFIRMADOS
11	2018	2018	CASOS_CONFIRMADOS

Row Normaliser

Step name Row Normaliser 2

Type field ANIO

Fields

#	Fieldname	Type	new field
1	2008	2008	MUERTES
2	2009	2009	MUERTES
3	2010	2010	MUERTES
4	2011	2011	MUERTES
5	2012	2012	MUERTES
6	2013	2013	MUERTES
7	2014	2014	MUERTES
8	2015	2015	MUERTES
9	2016	2016	MUERTES
10	2017	2017	MUERTES
11	2018	2018	MUERTES

Row Normaliser

Step name Row Normaliser 3

Type field ANIO

Fields

#	Fieldname	Type	new field
1	2008	2008	HOSPITALIZACIONES
2	2009	2009	HOSPITALIZACIONES
3	2010	2010	HOSPITALIZACIONES
4	2011	2011	HOSPITALIZACIONES
5	2012	2012	HOSPITALIZACIONES
6	2013	2013	HOSPITALIZACIONES
7	2014	2014	HOSPITALIZACIONES
8	2015	2015	HOSPITALIZACIONES
9	2016	2016	HOSPITALIZACIONES
10	2017	2017	HOSPITALIZACIONES
11	2018	2018	HOSPITALIZACIONES

Row Normaliser


Step name Row Normaliser 4

Type field ANIO

Fields

#	Fieldname	Type	new field
1	2008	2008	MUESTRAS_LABORATORIO
2	2009	2009	MUESTRAS_LABORATORIO
3	2010	2010	MUESTRAS_LABORATORIO
4	2011	2011	MUESTRAS_LABORATORIO
5	2012	2012	MUESTRAS_LABORATORIO
6	2013	2013	MUESTRAS_LABORATORIO
7	2014	2014	MUESTRAS_LABORATORIO
8	2015	2015	MUESTRAS_LABORATORIO
9	2016	2016	MUESTRAS_LABORATORIO
10	2017	2017	MUESTRAS_LABORATORIO
11	2018	2018	MUESTRAS_LABORATORIO

Utilizamos el paso “Merge Join” para unir los datos de los ficheros dos a dos en una misma entidad.

 Merge Join — □ ×

Step name:

First Step:

Second Step:


Join Type:

Keys for 1st step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

Keys for 2nd step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

 Merge Join — □ ×

Step name:

First Step:

Second Step:

Join Type:


Keys for 1st step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

Keys for 2nd step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

El último join se encarga de juntar definitivamente todos los atributos. Quedando un archivo con dos atributos comunes a los cuatro ficheros iniciales y cuatro nuevos campos, uno por cada normalización hecha

 Merge Join — □ ×

Step name:

First Step:

Second Step:

Join Type:

Keys for 1st step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

Keys for 2nd step:

#	Key field	
1	cod	
2	nombre	
3	ANIO	

Utilizamos el paso selección de valores para asegurarnos que cada atributo es del tipo deseado.

Select / Rename values

Step name

Select & Alter Remove Meta-data

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Forr
1	cod		Integer		0	N		N
2	ANIO		Integer	4		N		N
3	CASOS_CONFIRMADOS		Integer		0	N		N
4	MUERTES		Integer		0	N		N
5	HOSPITALIZACIONES		Integer		0	N		N
6	MUESTRAS_LABORATORIO		Integer		0	N		N

Por último, cargamos los valores en la tabla previamente creada en el servidor.

Table output

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

Specify database fields ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	COD	cod
2	NOMBRE	nombre
3	ANIO	ANIO
4	CASOS_CO...	CASOS_CON...
5	MUERTES	MUERTES
6	HOSPITALI...	HOSPITALIZ...
7	MUESTRAS...	MUESTRAS_L...

La transformación queda así:



Es importante señalar que los datos estaban ya ordenados en los ficheros.

También hay que mencionar que algunas de estas transformaciones podrían realizarse en la fase siguiente de los procesos ETL en lugar de en esta sin ningún inconveniente.

Transformación IN_WUE_DATA

La siguiente transformación carga el contenido de un csv en la tabla intermedia IN_WUE_DATA.

Consta de cuatro pasos: Lectura del fichero, Operación de cadena, Ordenación de filas y Carga en la tabla.

Cargamos el fichero csv, delimitador: “;”. Todos los valores son los que vienen por defecto.

CSV Input

Step name: **IN_WUE_DATA**

Filename: F:\fuentes\wuenic2011\rev_data_2019-11-16.csv

Delimiter: ;

Enclosure: "

NIO buffer size: 500000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional):

Running in parallel? ☐

New line possible in fields? ☐

File encoding: UTF-8

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim ty
1	Group	String		11		€	.	.	none
2	Subgroup	String		3		€	.	.	none
3	Name	String		11		€	.	.	none
4	Year	Integer	#	15	0	€	.	.	none
5	Vaccine	String		5		€	.	.	none
6	Coverage	Integer	#	15	0	€	.	.	none
7	Vaccinated	String	#, #	7	3	€	.	.	none
8	Target	String	#,###,###,###	9	3	€	.	.	none
9	Source	String		69		€	.	.	none

Ponemos todos los atributos de tipo cadena en mayúscula:

Step name

The fields to process:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char
1	Group		none	upper	none	
2	Subgroup		none	upper	none	
3	Name		none	upper	none	
4	Vaccine		none	upper	none	
5	Source		none	upper	none	

Ordenamos los registros de la siguiente manera:

Sort rows

Step name

Sort directory

TMP-file prefix

Sort size (rows in memory)

Free memory threshold (in %)

Compress TMP Files? ☐

Only pass unique rows? (verifies keys only) ☒

Fields:

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	Subgroup	Y	N	N	0	N
2	Name	Y	N	N	0	N
3	Year	Y	N	N	0	N
4	Vaccine	Y	N	N	0	N

Por último, cargamos los datos en la tabla previamente creada en el servidor.

Table output

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

Specify database fields ☒

Main options: Database fields

Fields to insert:

#	Table field	Stream field
1	GROUP	Group
2	SUBGROUP	Subgroup
3	NAME	Name
4	YEAR	Year
5	VACCINE	Vaccine
6	COVERAGE	Coverage
7	VACCINATED	Vaccinated
8	TARGET	Target
9	SOURCE	Source

La transacción queda finalmente:



Transformación IN_COV_SERIES

En la siguiente transformación se carga en la tabla IN_OCV_SERIES el contenido del fichero coverage_estimated_series.xls. Dicho fichero recoge los datos historificados de cobertura de vacunación de los países de la OMS.

La transformación consiste en 5 pasos: Lectura del fichero, Operación de cadena, Ordenación de filas, Filtrado de filas y carga en la Tabla.

Previo a la transformación es necesario remover algunos valores de la columna de los países. Resulta que todos los países que empiezan por THE tienen dicha palabra al final del registro y entre paréntesis, mientras que en los países ya cargados del fichero COUNTRY.json esta palabra ha sido omitida.

Es por ello que debemos eliminar dicho artículo para que no haya campos vacíos que no deberían cuando relacionemos las tablas.

A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	AMR	BHS	Bahamas (th North Ameri	DTP1	2008	95	*						
2	AMR	BHS	Bahamas (th North Ameri	DTP1	2009	99	*						
3	AMR	BHS	Bahamas (th North Ameri	DTP1	2010	99	*						
4	AMR	BHS	Bahamas (th North Ameri	DTP1	2011	99	*						
5	AMR	BHS	Bahamas (th North Ameri	DTP1	2012	99							
6	AMR	BHS	Bahamas (th North Ameri	DTP1	2013	99							
7	AMR	BHS	Bahamas (th North Ameri	DTP1	2014	94							
8	AMR	BHS	Bahamas (th North Ameri	DTP1	2015	95							
9	AMR	BHS	Bahamas (th North Ameri	DTP1	2016	94							
10	AMR	BHS	Bahamas (th North Ameri	DTP1	2017	94							
11	AMR	BHS	Bahamas (th North Ameri	DTP1	2018	94							
12	AMR	BHS	Bahamas (th North Ameri	DTP3	1979	30							
13	AMR	BHS	Bahamas (th North Ameri	DTP3	1980	34							
14	AMR	BHS	Bahamas (th North Ameri	DTP3	1981	50							
15	AMR	BHS	Bahamas (th North Ameri	DTP3	1982	62							
16	AMR	BHS	Bahamas (th North Ameri	DTP3	1983	62							
17	AMR	BHS	Bahamas (th North Ameri	DTP3	1984	05							
18	AMR	BHS	Bahamas (th North Ameri	DTP3	1985	86							
19	AMR	BHS	Bahamas (th North Ameri	DTP3	1986	95	ST80 salida(s) encontradas.						
20	AMR	BHS	Bahamas (th North Ameri	DTP3	1987	77							
21	AMR	BHS	Bahamas (th North Ameri	DTP3	1988	85							
22	AMR	BHS	Bahamas (th North Ameri	DTP3	1989	86							
23	AMR	BHS	Bahamas (th North Ameri	DTP3	1990	87							
24	AMR	BHS	Bahamas (th North Ameri	DTP3	1991	92							

A algunos de los alumnos nos ha surgido un raro error en el step para leer archivos xls que provoca que el paso no lea el contenido pese a ejecutarse correctamente. Es por ello por lo que he decido guardar el archivo como csv.

Cargamos el fichero csv, delimitador: “;”.

Step name

Filename

Delimiter

Enclosure

NIO buffer size

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional)

Running in parallel? ☐

New line possible in fields? ☐

File encoding

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim ty
1	WHO_REGION	String		3		€	,	.	none
2	ISO_code	String		3		€	,	.	none
3	Cname	String		11		€	,	.	none
4	Continent	String		4		€	,	.	none
5	Vaccine	String		7		€	,	.	none
6	Year	Integer	#	15	0	€	,	.	none
7	Percent_covrage	Integer	#	15	0	€	,	.	none
8	Asterisc	String		1		€	,	.	none

Ponemos en mayúsculas todas las cadenas.

String operations

Step name

The fields to process:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	P
1	WHO_REGION		none	upper	none	
2	ISO_code		none	upper	none	
3	Cname		none	upper	none	
4	Continent		none	upper	none	
5	Vaccine		none	upper	none	

Ordenamos los registros de la siguiente manera:

Sort rows

Step name

Sort directory

TMP file prefix

Sort size (rows in memory)

Free memory threshold (in %)


Compress TMP files? ☒

Only pass unique rows? (verifies keys only) ☐

Fields:

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	WHO_REGION	Y	N	N	0	N
2	ISO_code	Y	N	N	0	N
3	Cname	Y	N	N	0	N
4	Continent	Y	N	N	0	N
5	Vaccine	Y	N	N	0	N
6	Year	Y	N	N	0	N

Tras ordenar los registros aparecieron dos filas completamente vacías, es por ello que filtramos los registros descartando las filas cuando la columna WHO_REGION, la cual no debería ser nula nunca, lo es.

 **Filter rows**


Step name

Send 'true' data to step:

Send 'false' data to step:

The condition:

Por último, cargamos el fichero en la tabla intermedia IN_COV_SERIES

 **Table output**

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

Specify database fields ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	WHO_REGI...	WHO_REGION
2	ISO_CODE	ISO_code
3	CNAME	Cname
4	CONTINENT	Continent
5	VACCINE	Vaccine
6	YEAR	Year
7	PERCENT_...	Percent_covr...
8	ASTERISC	Asterisc

La transformación queda con este aspecto:

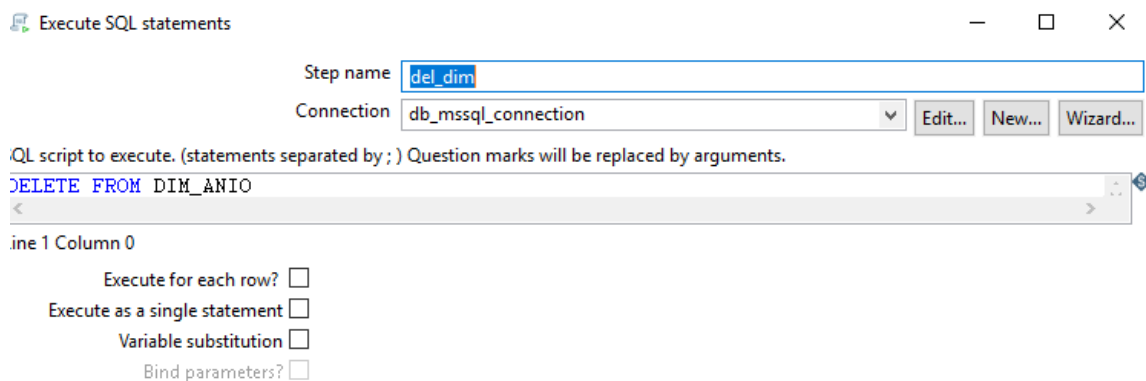


C) Bloque IN

1. Transformaciones Bloque TR_DIM

Este bloque, contiene las transformaciones para la carga inicial de las dimensiones al almacén desde las tablas intermedias IN_. Además de la carga inicial de las dimensiones del modelo multidimensional, se tendrá en cuenta que se puedan ejecutar las transformaciones las veces que sean necesarias, por esto se incluirá como primer paso de las transformaciones del bloque TR_DIM un borrado de las tablas de dimensiones. Lo mismo se hará en las TR_FACT.

A continuación, se muestra dicho paso para la dimensión DIM_ANIO, la query mostrada será la misma para todas las tablas cambiando el valor correspondiente a la tabla.

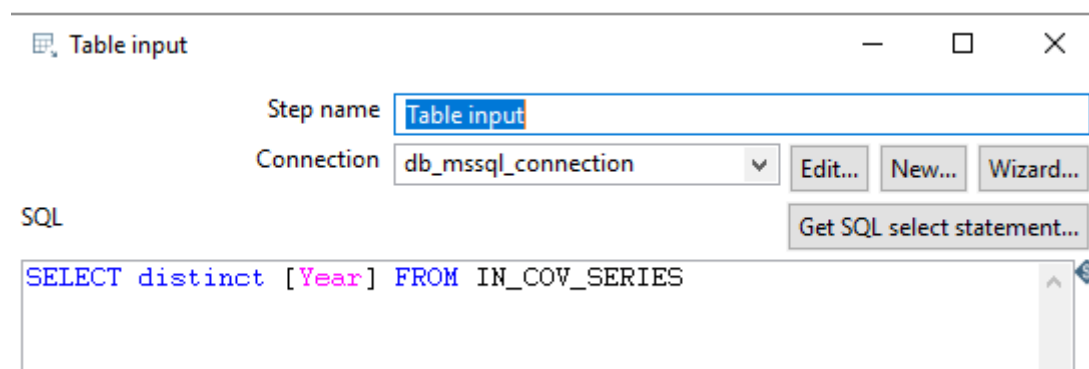


TR_DIM_ANIO

A continuación, se muestra la carga de la tabla DIM_ANIO.

La transformación consta de cinco pasos: Carga de la tabla desde nuestra bd, Selección de valores, Mapeador de valores, Ordenación de filas y carga en tabla.

Cargamos la columna año de la tabla IN_COV_SERIES:



Utilizamos el paso selección de valores para comprobar el tipo y que la carga ha sido buena en cuanto a los meta-datos se refiere.

Select / Rename values

Step name:

Select & Alter Remove Meta-data

Fields to alter the meta-data for:

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Format Lenient?
1	Year		Integer	30	0	N		N

Con el mapeador de valores generamos una nueva columna con información complementaria a cada año.

Step name:

Fieldname to use:

Target field name (empty=overwrite):

Default upon non-matching:

Field values:

#	Source value	Target value
1	1983	Calendario de vacunacion de 1983
2	1984	Calendario de vacunacion de 1984
3	1985	Calendario de vacunacion de 1985
4	1986	Calendario de vacunacion de 1986
5	1987	Calendario de vacunacion de 1987
6	1988	Calendario de vacunacion de 1988
7	1989	Calendario de vacunacion de 1989
8	1990	Calendario de vacunacion de 1990
9	1991	Calendario de vacunacion de 1991
10	1992	Calendario de vacunacion de 1992
11	1993	Calendario de vacunacion de 1993
12	1994	Calendario de vacunacion de 1994
13	1995	Calendario de vacunacion de 1995
14	1996	Calendario de vacunacion de 1996
15	1997	Calendario de vacunacion de 1997
16	1998	Calendario de vacunacion de 1998

Ordenamos los valores por año de menor a mayor y cargamos los datos en la tabla DIM_ANIO.

Table output

Step name: Table output

Connection: db_mssql_connection

Target schema: dbo

Target table: DIM_ANIO

Commit size: 1000

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

Main options | Database fields

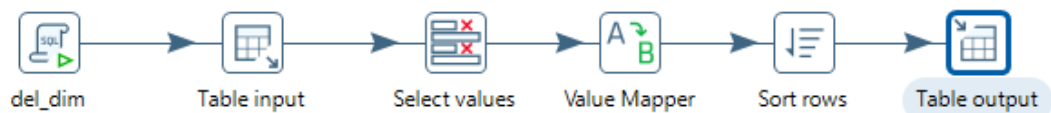
Fields to insert:

#	Table field	Stream field
1	SK_DIM_A...	Year
2	DESC_ANIO	DESC_ANIO

Get fields

Enter field mapping

La transformación es la siguiente:



El resultado de la ejecución es:

TR_DIM_CAUSAS_MORTALIDAD

En la siguiente transformación cargaremos la información deseada en la tabla DIM_CAUSAS_MORTALIDAD.

Consta de tres pasos: Lectura de tabla, Añadir secuencia y Carga de Tabla.

Extraemos los dos registros de la tabla IN_MORTCAUSE.

Table input

Step name: Table input

Connection: db_mssql_connection

SQL

Get SQL select statement...

```
SELECT ID, DISEASE FROM IN_MORTCAUSE
```

Añadimos como clave primaria una columna con una secuencia de dígitos.

Get Value From Sequence

Step name: Add sequence

Name of value: SK_DIM_CAUSA

Use a database to generate the sequence ☐

Use DB to get sequence? ☐

Connection: db_mssql_connection Edit... New... Wizard...

Schema name: Schemas...

Sequence name: Sequences...

Use a transformation counter to generate the sequence ☒

Use counter to calculate sequence? ☒

Counter name (optional):

Start at value: 1

Increment by: 1

Maximum value: 99999999

Por último, cargamos la tabla de la base de datos.

Table output

Step name: Table output

Connection: db_mssql_connection Edit... New... Wizard...

Target schema: Browse...

Target table: DIM_CAUSAS_MORTALIDAD Browse...

Commit size: 1000

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

Main options | Database fields

Fields to insert:

#	Table field	Stream field
1	SK_DIM_CA...	SK_DIM_CAU...
2	COD_CAUSA	ID
3	DESC_CAU...	DISEASE

Get fields

Enter field mapping

La transformación final es la siguiente:

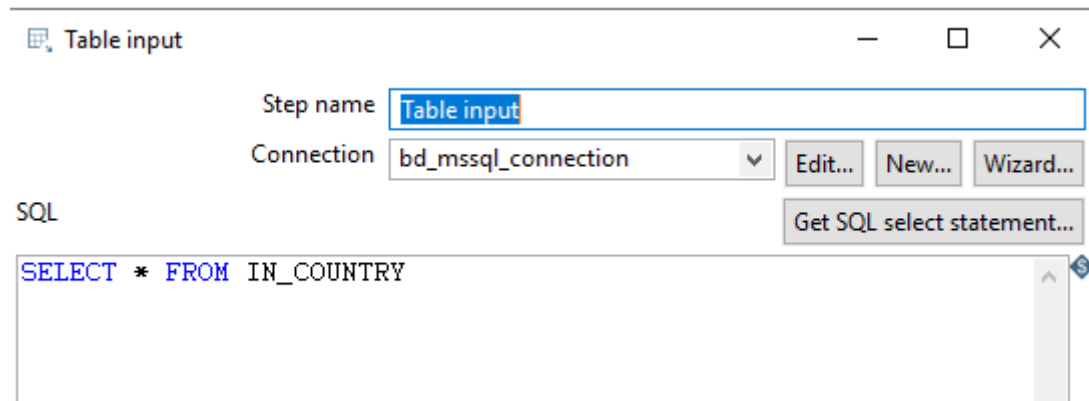


TR_DIM_PAIS

En la siguiente transformación cargaremos la información deseada en la tabla DIM_PAIS. Esta transformación tiene como peculiaridad que es necesario cargar datos que se corresponden con otros registros de otras tablas.

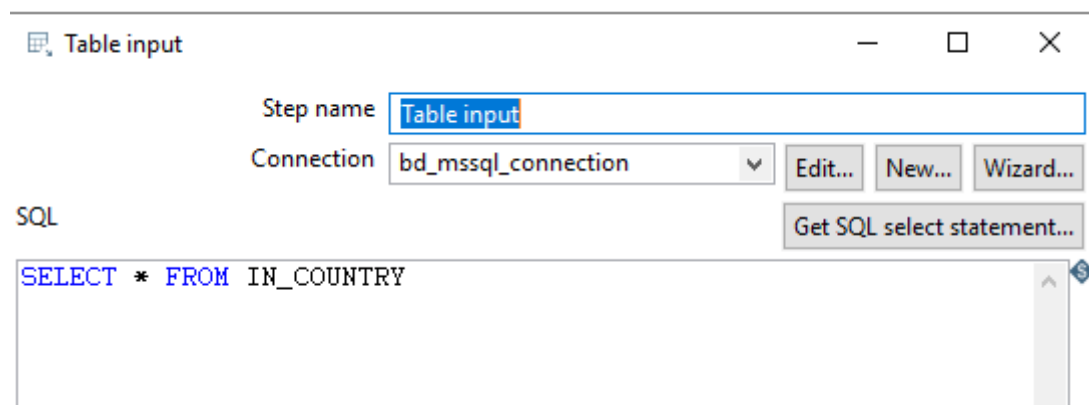
Consta de seis pasos: Lectura de tabla, Desnormalización de fila, Búsqueda en la base de datos, Ordenación de fila, Añadir secuencia y Carga de Tabla.

Cargamos todos los valores de IN_COUNTRY.



The screenshot shows a window titled 'Table input'. It has a 'Step name' field containing 'Table input'. Below it is a 'Connection' dropdown menu set to 'bd_mssql_connection', with buttons for 'Edit...', 'New...', and 'Wizard...'. To the right of the connection field is a button labeled 'Get SQL select statement...'. Below these fields is a text area labeled 'SQL' containing the query: `SELECT * FROM IN_COUNTRY`. A small blue icon with a dollar sign is visible at the bottom right of the SQL text area.

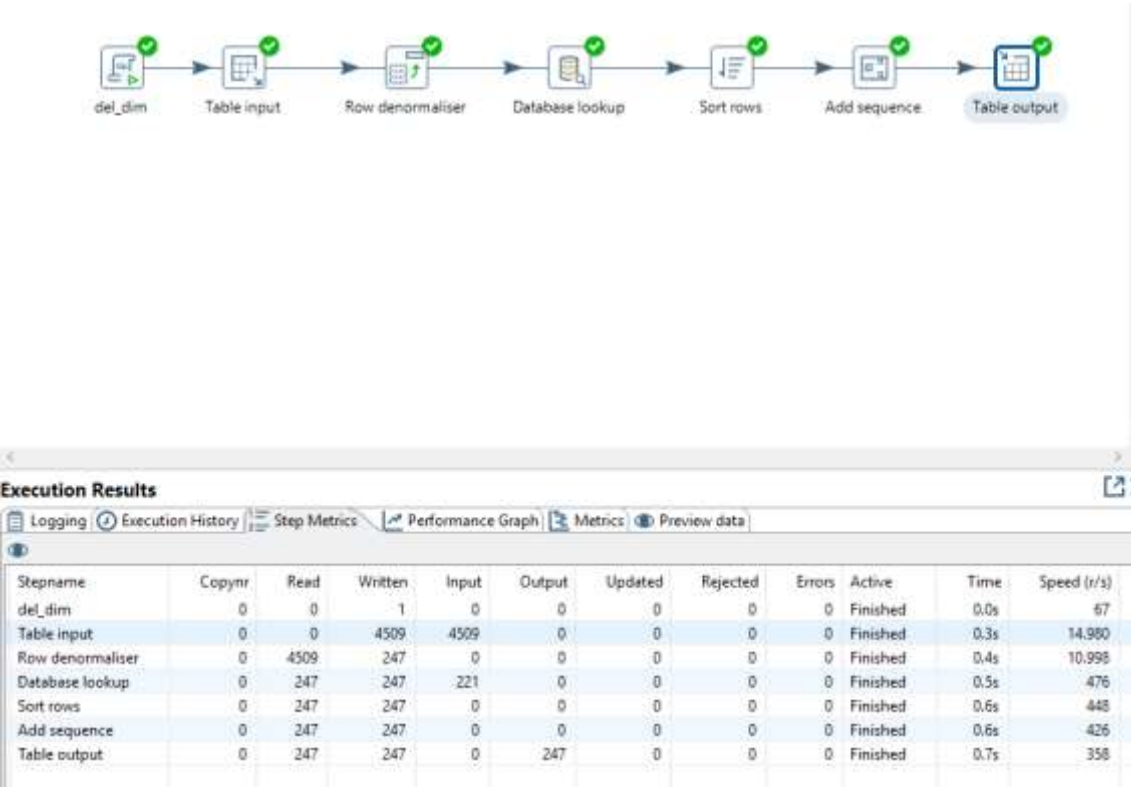
En el paso de desnormalización lo que hacemos es extraer todos los campos deseados que estaban en las columnas de category y value en la anterior tabla.



This screenshot is identical to the one above, showing the 'Table input' window with the same configuration: 'Step name' is 'Table input', 'Connection' is 'bd_mssql_connection', and the 'SQL' text area contains the query `SELECT * FROM IN_COUNTRY`.

Nos interesa la clave de la tabla de causas de mortalidad, para ello usamos la columna MORT y la comparamos con una de la tabla para que nos devuelva el valor que queremos, SK_MID_CAUSA.

La transformación final y el resultado de su ejecución es el siguiente:



TR_DIM_GEOGRAFÍA

La siguiente transformación cuenta de 6 pasos: Entrada de tabla, Orden de filas, Añadir secuencia, Búsqueda en base de datos, Selección de valores y Carga de Tabla.

Extraemos cuatro valores de la tabla IN_COV_SERIES

Table input [Close]

Step name: **Table input**

Connection: **db_mssql_connection** [Edit...] [New...] [Wizard...]

SQL: [Get SQL select statement.]

```
SELECT distinct WHO_REGION, Cname, Continent FROM IN_COV_SERIES
```

Ordenamos las filas por WHO_REGION y Cname.

Añadimos clave primaria.

Get Value From Sequence

Step name: Add sequence

Name of value: SK_DIM_GEOGRAFIA

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection: db_mssql_connection Edit... New... Wizard...

Schema name: Schemas...

Sequence name: SEQ_ Sequences...

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☒

Counter name (optional):

Start at value: 1

Increment by: 1

Maximum value: 999999999

Con el nombre del país queremos conseguir la clave primaria correspondiente a ese país de la tabla DIM_PAIS.

Database Value Lookup

Step name: Database lookup

Connection: db_mssql_connection Edit... New... Wizard...

Lookup schema: Browse...

Lookup table: DIM_PAIS Browse...

Enable cache? ☐

Cache size in rows (0=cache): 0

Load all data from table ☐

the key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	NOMBRE_PAIS	=	Cname	

values to return from the lookup table:

#	Field	New name	Default	Type
1	SK_DIM_PAIS			Integer

Do not pass the row if the lookup fails ☐

Hacemos una comprobación de los metadatos con la selección de valores:

Step name: **Select values**

Select & Alter Remove Meta-data

Fields to alter the meta-data for:

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Format Lenient?	Date Locale
1	WHO_REGION		String	100		N		N	
2	Cname		String	300		N		N	
3	Continent		String	300		N		N	
4	SK_DIM_GEOGRAFIA		Integer		0	N		N	
5	SK_DIM_PAIS		Integer	2	0	N		N	

Y cargamos en la tabla.

Table output

Step name: **Table output**

Connection: db_mssql_connection

Target schema: dbo

Target table: DIM_GEOGRAFIA

Commit size: 1000

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	SK_DIM_GE...	SK_DIM_GEO...
2	COD_REGI...	WHO_REGION
3	DESC_REGI...	Continent
4	SK_DIM_PAIS	SK_DIM_PAIS
5	DESC_PAIS	Cname

La transformación final tiene este aspecto:



TR_DIM_VACUNA

La siguiente transformación consta de trece pasos: 2 lecturas de tabla, 3 Ordenaciones de fila, fusión ordenada, 2 Mapeados de valores, Operaciones de cadena, Filtrado de filas, Ordenación de filas, Filas únicas, Añadir secuencia, Salida en tabla.

Cargamos los distintos tipos de vacuna de la tabla IN_WUE_DATA.

Table input

Step name

Connection

SQL

```
SELECT distinct VACCINE FROM IN_WUE_DATA
```

Hacemos lo mismo con la tabla IN_COV_SERIES.

Table input

Step name

Connection

SQL

```
SELECT distinct Vaccine FROM IN_COV_SERIES
```

Ordenamos ambas tablas y las unimos con el paso "sorted merge"

Sorted Merge

Step name

Fields:

#	Fieldname	Ascending
1	VACCINE	Y

Tras mucho indagar en internet ha sido posible generar un campo con el nombre de cada vacuna.

Value Mapper

Step name: Value Mapper

Fieldname to use: VACCINE

Target field name (empty=overwrite): NOMBRE_VACUNA

Default upon non-matching:

Field values:

#	Source value	Target value
12	HIB3	Third dose of Haemophilus influenzae type B vaccine
13	HPVFEM	FEMENINE Human papillomavirus
14	HPVMALE	MASCULINE Human papillomavirus
15	IPV1	First dose of inactivated polio vaccine
16	IPV1FRAC	POLIO
17	IPV2FRAC	POLIO
18	JAPENC	NOT_FOUND
19	MCV1	Measles containing vaccines
20	MCV2	Coverage estimates are for the nationally recommended age for the second dose of measles containing vaccine
21	MENA	Meningococcal Conjugate Vaccine
22	MMR	Measles, Mumps & Rubella Vaccine
23	MUMPS	Mumps
24	PAB	PROTECTION AT BIRTH
25	PAB_DTP1	PROTECTION AT BIRTH AT diphtheria
26	PCV1	first dose of pneumococcal conjugate vaccine

Con operaciones de cadena ponemos en mayúscula ambas columnas.

Filtramos por si hubiera algún valor nulo:

Filter rows

Step name: Filter rows

Send 'true' data to step:

Send 'false' data to step: Sort rows 3

The condition:

NOMBRE_VACUNA IS NULL

Reordenamos por si el filtrado hubiera desordenado los datos.

Eliminamos las filas repetidas con el paso filas únicas:

Unique rows

Step name: Unique rows

Settings

Add counter to output? ☐ Counter field

Redirect duplicate row ☐ Error description

Fields to compare on (no entries means: compare complete row)

#	Fieldname	Ignore case
1	VACCINE	N
2	NOMBRE_VACUNA	N

Generamos una columna para las vacunas del GVAP con el último mapeador de valores.

Step name:

Fieldname to use:

Target field name (empty=overwrite):

Default upon non-matching:

id values:

	Source value	Target value
	BCG	1
	DIPHCV4	0
	DIPHCV5	0
	DIPHCV6	0
	DTP	1
	DTP2	0
	DTP3	1
	DTP4	0
	HEPB3	0
0	HEPB_BD	1
1	HEPB_BDALL	0
2	HIB3	1
3	HPVFEM	0
4	HPVMALE	0
5	IPV1	1
6	IPV1FRAC	0
7	IPV2FRAC	0
8	JAPENC	0
9	MCV1	1
0	MCV2	1
1	MENA	0
2	MMR	0

Añadimos la clave primaria como en anteriores ocasiones.

☒ Get Value From Sequence

Step name:

Name of value:

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection:

Schema name:

Sequence name:

Por último, cargamos la última tabla de las dimensiones.

#	Table field	Stream field
1	SK_DIM_VA...	SK_DIM_VAC...
2	TIPO_VACU...	VACCINE
3	NOMBRE_V...	NOMBRE_VA...
4	GVAP	GVAP

2. Transformaciones Bloque TR_FACT

En este apartado, una vez cargadas en la base de datos tanto las fuentes como las dimensiones, el siguiente paso consiste en desarrollar las transacciones para poblar nuestras tablas de hechos.

Transformación TR_FACT_SARAMPION

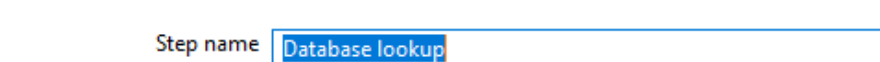
Mediante esta transformación poblaremos la tabla de hechos FACT_SARAMPION según el modelo multidimensional definido y que nos permitirá realizar un análisis evolutivo de la cobertura de inmunización de la enfermedad del sarampión por año, país.

La transformación tiene 5 pasos: Carga de tabla, 2 Búsquedas en base de datos y Salida en tabla.

Comenzamos cargando todos los atributos de la tabla IN_SRAMPION.

```
SELECT * FROM IN_SRAMPION
```

Extraemos, usando el atributo año, la clave primaria SK_DIM_ANIO.



Database Value Lookup

Step name: Database lookup

Connection: db_mssql_connection

Lookup schema:

Lookup table: DIM_ANIO

Enable cache? ☐

Cache size in rows (0=cache all rows): 0

Load all data from table ☐

Values to return from the lookup table :

#	Field	New name	Default	Type
1	SK_DIM_ANIO			Number

Repetimos el proceso, esta vez utilizando el nombre del país para extraer la clave de la tabla DIM_GOGRAFIA.

Database Value Lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache all rows):

Load all data from table ☐

the key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	DESC_PAIS	=	NOMBRE	

values to return from the lookup table:

#	Field	New name	Default	Type
1	SK_DIM_GEOGRAFIA			Number

Por, último, cargamos todas las columnas obtenidas con sus correspondientes en la tabla FACT_SARAMPION.

Table output

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

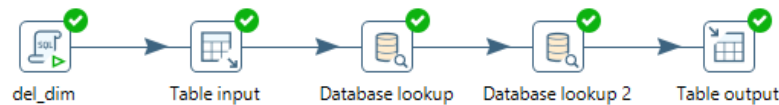
Specify database fields ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	SK_DIM_A...	SK_DIM_ANIO
2	SK_DIM_GE...	SK_DIM_GEO...
3	NUM_CASOS	CASOS_CON...
4	NUM_MUE...	MUERTES
5	NUM_HOS...	HOSPITALIZ...
6	NUM_CAS...	MUESTRAS_L...

La transformación final y el resultado de la ejecución tienen este aspecto.



Execution Results

Logging Execution History Step Metrics Performance Graph Metrics Preview data

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	del_dim	0	0	1	0	0	0	0	0	Finished
2	Table input	0	0	583	583	0	0	0	0	Finished
3	Database lookup	0	583	583	583	0	0	0	0	Finished
4	Database lookup 2	0	583	583	583	0	0	0	0	Finished
5	Table output	0	583	583	0	583	0	0	0	Finished

Transformación TR_FACT_COBERTURA

Mediante esta transformación poblaremos la tabla de hechos FACT_COBERTURA según el modelo multidimensional definido y que nos permitirá realizar un análisis evolutivo de la cobertura de inmunización por año, país y vacuna.

Esta transformación, a pesar de ser una sola, actúa con dos flujos de trabajo de forma paralela.

Flujo 1:

La transformación tiene 5 pasos: Carga de tabla, 3 Búsquedas en base de datos y Salida en tabla.

Cargamos los atributos necesarios de la tabla IN_WUE_DATA.

The screenshot shows a configuration window for a data transformation step. The 'Step name' field is set to 'IN_WUE_DATA'. The 'Connection' dropdown is set to 'db_mssql_connection'. There are buttons for 'Edit...', 'New...', and 'Wizard...'. Below this, there is a section labeled 'SQL' with a button 'Get SQL select statement...'. The SQL query displayed is:

```
SELECT [NAME]
      , [YEAR]
      , [VACCINE]
      , [COVERAGE]
      , [VACCINATED]
      , [TARGET]
FROM IN_WUE_DATA
```

Flujo2:

Hacemos lo mismo con la tabla IN_COV_SERIES-

The screenshot shows a configuration window for a data transformation step. The 'Step name' field is set to 'IN_COV_SERIES'. The 'Connection' dropdown is set to 'db_mssql_connection'. There are buttons for 'Edit...', 'New...', and 'Wizard...'. Below this, there is a section labeled 'SQL' with a button 'Get SQL select statement...'. The SQL query displayed is:

```
SELECT [Cname]
      , [Vaccine]
      , [Year]
      , [Percent_coverage]
FROM IN_COV_SERIES
```

Con el atributo año, name y vacuna cargamos las calves principales de las tablas DIM_ANIO, DIM_GEOGRAFIA y DIM_VACUNA, respectivamente.

El proceso es el mismo en ambos flujos.

Obtenemos SK_DIM_ANIO:

Database Value Lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache):

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	SK_DIM_ANIO	=	YEAR	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	SK_DIM_ANIO			Integer

Obtenemos SK_DIM_GEOGRAFIA:

Database Value Lookup

Step name:

Connection:

Lookup schema:

Lookup table:

Enable cache? ☐

Cache size in rows (0=cache):

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	DESC_PAIS	=	NAME	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	SK_DIM_GEOGRAFIA			None

Obtenemos SK_DIM_VACUNA.

Database Value Lookup

Step name: SK_DIM_VACUNA

Connection: db_mssql_connection

Lookup schema: dbo

Lookup table: DIM_VACUNA

Enable cache? ☐

Cache size in rows (Default): 1000

Load all data from table ☐

The key(s) to look up the value(s):

#	Table field	Comparator	Field1	Field2
1	TIPO_VACUNA	=	VACCINE	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	SK_DIM_VACUNA			None

Do not pass the row if the lookup fails ☐

Cargamos los datos en la tabla FACT_COBERTURA.

Flujo 1:

Table output

Step name: Table output

Connection: db_mssql_connection

Target schema: dbo

Target table: FACT_COBERTURA

Commit size: 1000

Truncate table ☐

Ignore insert errors ☐

Specify database fields ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	SK_DIM_A...	SK_DIM_ANIO
2	SK_DIM_GE...	SK_DIM_GEO...
3	SK_DIM_VA...	SK_DIM_VAC...
4	COBERTURA	COVERAGE
5	NUM_PV	VACCINATED
6	OBJETIVO	TARGET

Flujo 2:

Table output

Step name

Connection

Target schema

Target table

Commit size

Truncate table ☐

Ignore insert errors ☐

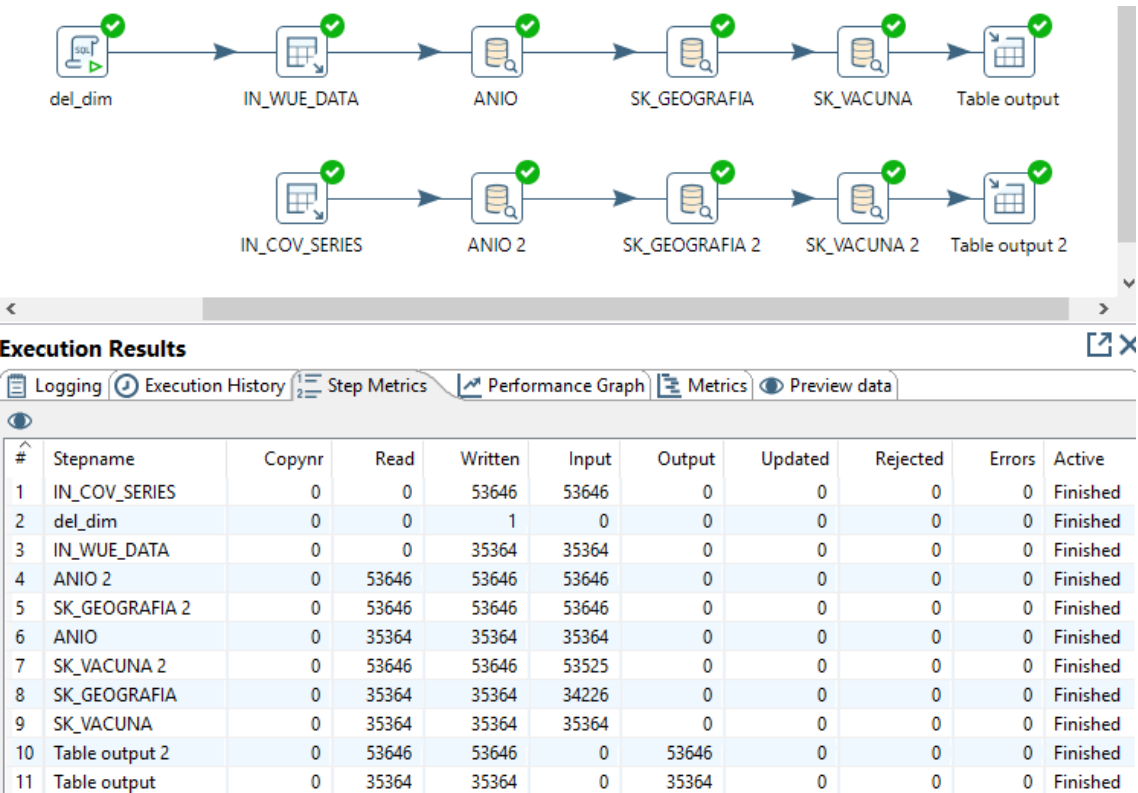
Specify database fields ☒

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	SK_DIM_A...	SK_DIM_ANIO
2	SK_DIM_GE...	SK_DIM_GEO...
3	SK_DIM_VA...	SK_DIM_VAC...
4	COBERTURA	Percent_covr...

El resultado final de la ejecución es el siguiente:

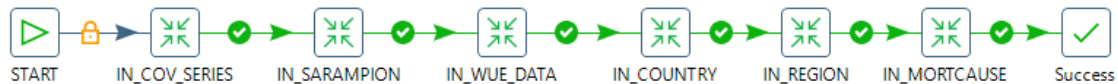


La razón por la que se ha implementado este diseño de dos flujos es porque hay dos atributos provenientes de la fuente IN_COV_SERIES que no existen y esta es la manera más sencilla de cargarlos todos los datos en la tabla FACT_COBERTURA.

3. Implementación de trabajos con procesos ETL's.

Vamos a diseñar los trabajos (jobs) mediante PDI que van a permitir la ejecución secuencial de todos los procesos ETL's incluidos en cada bloque definido. Cada trabajo contiene como pasos cada una de las transformaciones implementadas en el apartado anterior de Diseño de ETL's.

JOB_IN



- Start: Componente General de diseño de trabajos, que marca el Inicio del mismo.
- IN_COV_SERIES: Ejecución de transformación IN_COV_SERIES.
- IN_SARAMPION: Ejecución de transformación IN_SEGR_N.
- IN_EGR_C16_17: Ejecución de transformación IN_SARAMPION.
- IN_WUE_DATA: Ejecución de transformación IN_WUE_DATA.
- IN_COUNTRY: Ejecución de transformación IN_COUNTRY.
- IN_REGION: Ejecución de transformación IN_REGION.
- IN_MORTCAUSE: Ejecución de transformación IN_MORTCAUSE.
- Success: Componente General de diseño de trabajos, que marca la Finalización del trabajo.

Resultados:

Job / Job Entry	Comment	Result	Reason	Filename	Nr	Log date
Job: JOB_IN						
Job: JOB_IN	Start of job execution		start			2020/05/28 23:29:24
START	Start of job execution		start			2020/05/28 23:29:24
START	Job execution finished	Success			0	2020/05/28 23:29:24
IN_COV_SERIES	Start of job execution		Followed unconditional link	F:\Desktop\PRAC2\IN_COV_SERIES...	1	2020/05/28 23:29:31
IN_COV_SERIES	Job execution finished	Success				
IN_SARAMPION	Start of job execution		Followed link after success	F:\Desktop\PRAC2\IN_SARAMPIO...	2	2020/05/28 23:29:32
IN_SARAMPION	Job execution finished	Success				
IN_WUE_DATA	Start of job execution		Followed link after success	F:\Desktop\PRAC2\IN_WUE_DATA...	3	2020/05/28 23:29:33
IN_WUE_DATA	Job execution finished	Success				
IN_COUNTRY	Start of job execution		Followed link after success	F:\Desktop\PRAC2\IN_COUNTRY.ktr	4	2020/05/28 23:29:34
IN_COUNTRY	Job execution finished	Success				
IN_REGION	Start of job execution		Followed link after success	F:\Desktop\PRAC2\IN_REGION.ktr	5	2020/05/28 23:29:34
IN_REGION	Job execution finished	Success				
IN_MORTCAUSE	Start of job execution		Followed link after success	F:\Desktop\PRAC2\IN_MORTCAUS...	6	2020/05/28 23:29:34
IN_MORTCAUSE	Job execution finished	Success				
Success	Start of job execution		Followed link after success		6	2020/05/28 23:29:34
Success	Job execution finished	Success				
Job: JOB_IN	Job execution finished	Success	finished		6	2020/05/28 23:29:34

JOB_TR_DIM



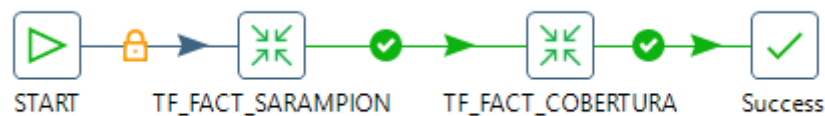
- Start: Componente General de diseño de trabajos, que marca el Inicio del mismo.
- TF_DIM_ANIO: Ejecución de transformación TF_DIM_ANIO.

- TF_DIM_CAUSAS_MORTALIDAD Ejecución de transformación TF_DIM_CAUSAS_MORTALIDAD.
- TF_DIM_VACUNA: Ejecución de transformación TF_DIM_VACUNA.
- TF_DIM_PAIS: Ejecución de transformación TF_DIM_PAIS.
- TF_DIM_GEOGRAFIA: Ejecución de transformación TF_DIM_GEOGRAFIA.
- Success: Componente General de diseño de trabajos, que marca la Finalización del trabajo.

No es posible re-ejecutar el trabajo para plasmar los resultados en esta memoria debido a que las dimensiones no se pueden vaciar a causa de las dependencias que hay entre ellas y con los hechos.

No obstante, en los siguientes apartados se verán los datos como prueba de que han sido cargados correctamente.

JOM_TF_FACT



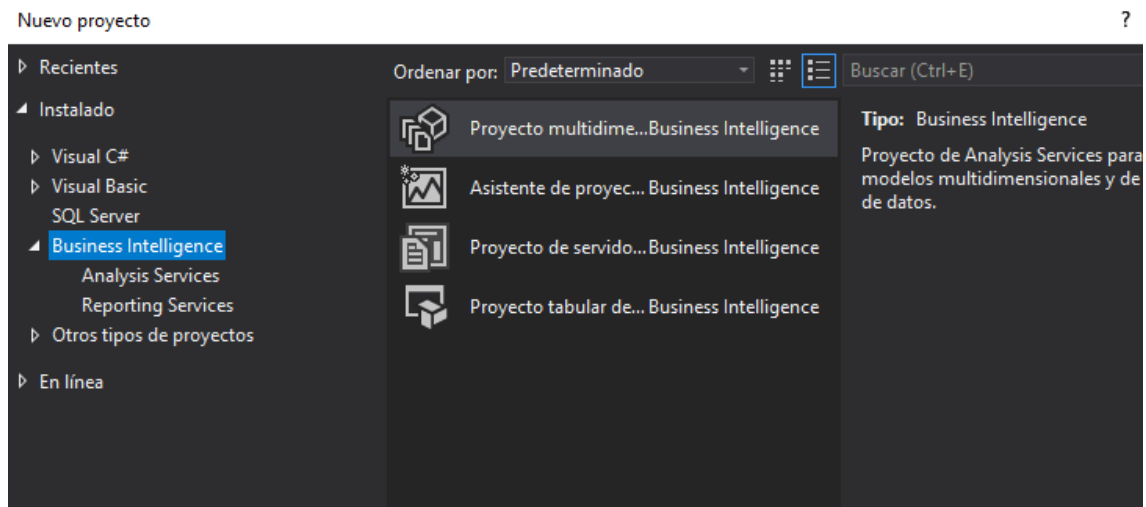
- Start: Componente General de diseño de trabajos, que marca el Inicio del mismo.
- TF_FACT_SARAMPION: Ejecución de transformación TF_FACT_SARAMPION.
- TF_FACT_COBERTURA Ejecución de transformación TF_FACT_COBERTURA.
- Success: Componente General de diseño de trabajos, que marca la Finalización del trabajo.

No es posible re-ejecutar el trabajo para plasmar los resultados en esta memoria debido a que los hechos no se pueden vaciar a causa de las dependencias que hay entre ellos y con las dimensiones.

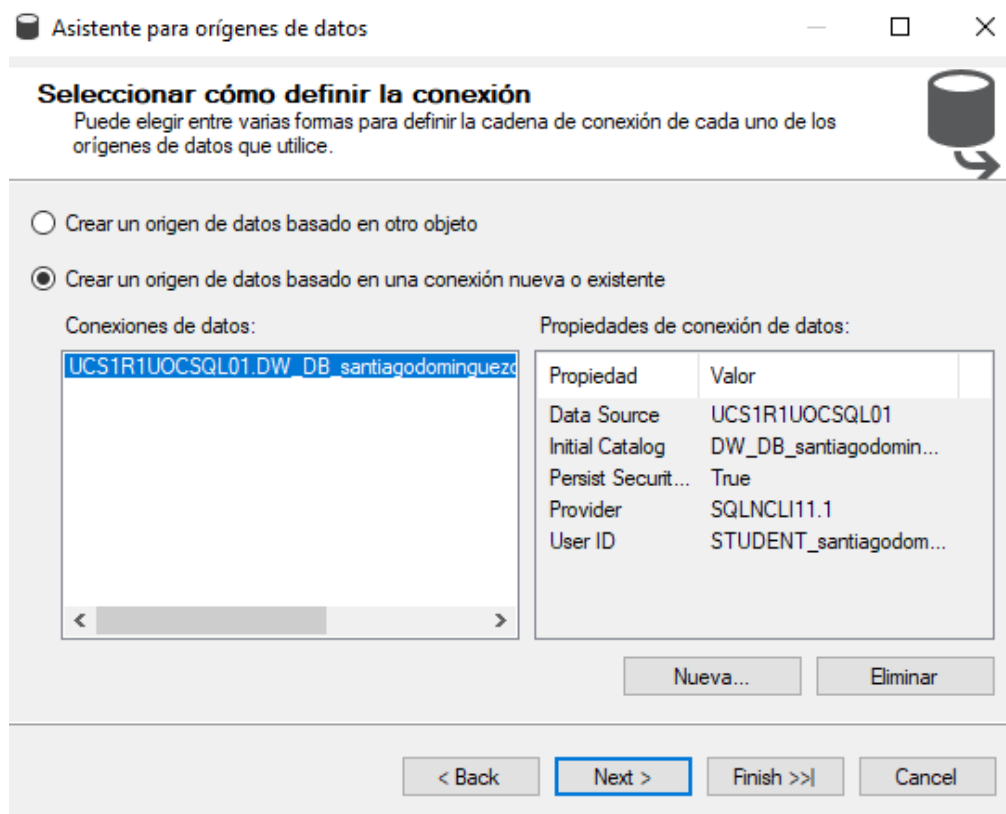
No obstante, en los siguientes apartados se verán los datos como prueba de que han sido cargados correctamente.

4. Creación de modelo OLAP

Creamos el proyecto del tipo “proyecto multidimensional y de minería de datos” que tanto permite crear cubos como proyectos de minería de datos.



Conectamos nuestro proyecto a la base de datos creando un nuevo origen de datos.



Como no tenemos información de qué usuario va a acceder a cada parte de nuestro cubo, seleccionamos la opción “Heredar”

Información de suplantación

Puede definir las credenciales de Windows que utilizará Analysis Services para conectarse con el origen de datos.



☐ Utilizar un nombre de usuario y una contraseña de Windows específicos

Nombre de usuario:

Contraseña:

☐ Utilizar la cuenta de servicio

☐ Utilizar las credenciales del usuario actual

☒ Heredar

A. VISTAS DEL ORIGEN DE DATOS.

Crearemos dos vistas del origen de datos, una por cada diseño lógico diseñado en la primera parte de la práctica. En el explorador de soluciones >vistas> click derecho > nueva vista.

VISTA 1.

Esta vista recoge el diseño lógico de la implementación de FACT_SARAMPION.

Asistente para vistas del origen de datos

Seleccionar tablas y vistas

Seleccione los objetos de la base de datos relacional que deben incluirse en la vista del origen de datos.



Objetos disponibles:

Nombre	Tipo
DIM_VACUNA (dbo)	Tabla
FACT_COBERTURA (dbo)	Tabla
IN_COUNTRY (dbo)	Tabla
IN_COV_SERIES (dbo)	Tabla
IN_MORTCAUSE (dbo)	Tabla
IN_REGION (dbo)	Tabla
IN_SARAMPION (dbo)	Tabla
IN_WUE_DATA (dbo)	Tabla



Objetos incluidos:

Nombre	Tipo
DIM_ANIO (dbo)	Tabla
DIM_CAUSAS_MORTA...	Tabla
DIM_GEOGRAFIA (dbo)	Tabla
DIM_PAIS (dbo)	Tabla
FACT_SARAMPION (dbo)	Tabla

Filtro:



☐ Mostrar objetos del sistema

Agregar tablas relacionadas

Es frecuente que no se carguen las relaciones entre tablas correctamente, de modo que creamos nosotros las que faltan.

Por ejemplo, a continuación, se muestra para el caso entre las tablas DIM_PAIS Y DIM_GEOGRAFIA.

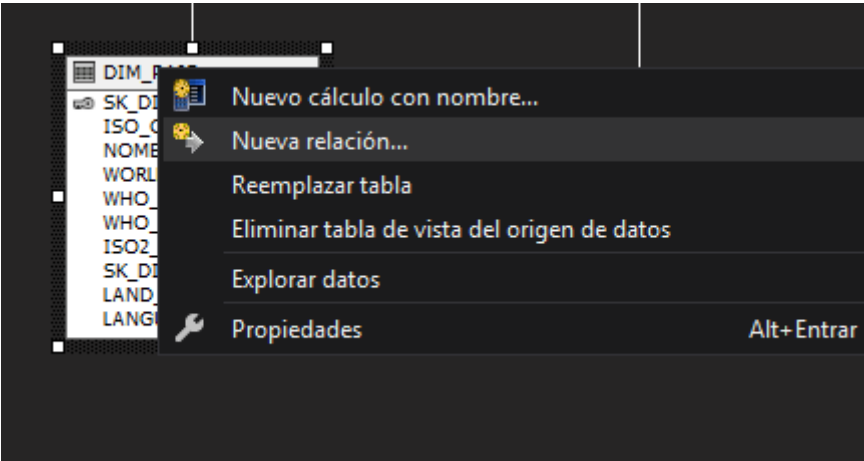
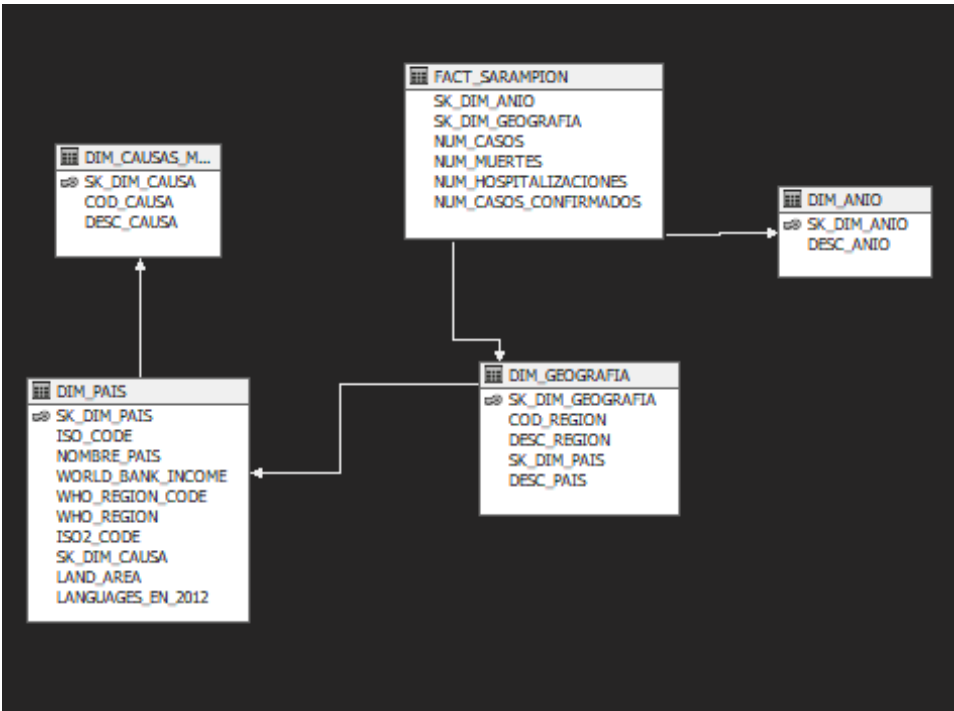


Tabla de origen (de clave externa):	Tabla de destino (de clave principal):
DIM_PAIS	DIM_GEOGRAFIA
Columnas de origen	Columnas de destino
SK_DIM_PAIS	SK_DIM_PAIS

El resultado de la vista de datos es el siguiente:



VISTA 2.

Esta vista recoge el diseño lógico de la implementación de FACT_COBERTURA.

Vista del origen de datos:
VISTA_COBERTURA

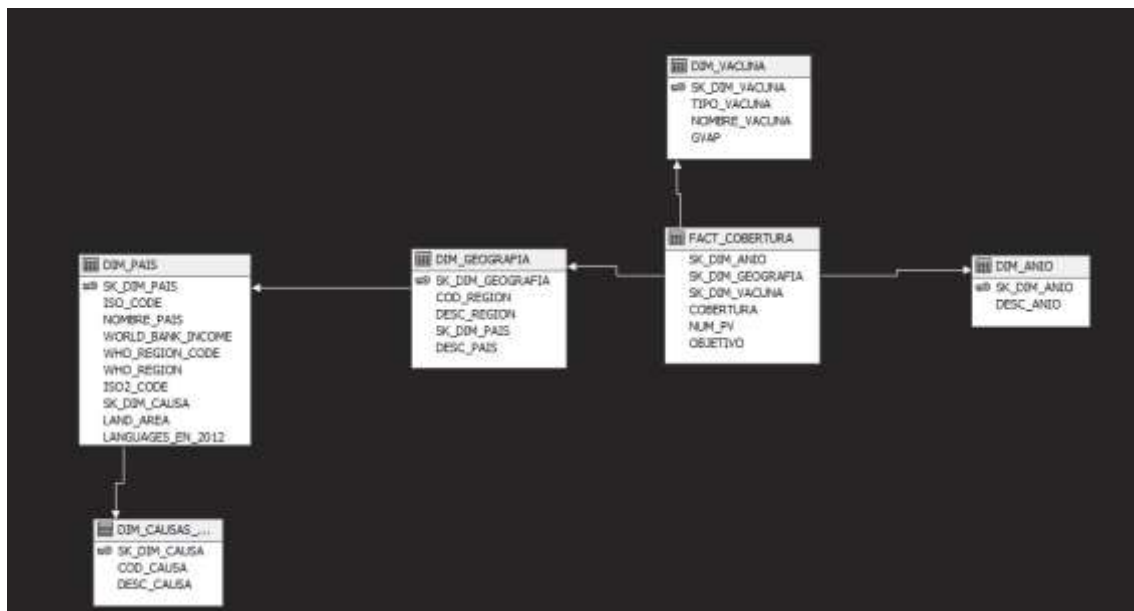
Tablas de grupo de medida:

Sugerir

<input checked="" type="checkbox"/>	<input type="checkbox"/>	FACT_COBERTURA
<input checked="" type="checkbox"/>	<input type="checkbox"/>	DIM_ANIO
<input checked="" type="checkbox"/>	<input type="checkbox"/>	DIM_CAUSAS_MORTALIDAD
<input checked="" type="checkbox"/>	<input type="checkbox"/>	DIM_GEOGRAFIA
<input checked="" type="checkbox"/>	<input type="checkbox"/>	DIM_PAIS
<input checked="" type="checkbox"/>	<input type="checkbox"/>	DIM_VACUNA

< Back Next > Finish >>| Cancel

Arreglamos las relaciones que no se hayan cargado, posteriormente este el diseño resultante:



B. CUBOS.

Ahora ya estamos preparados para crear los cubos con los modelos multidimensionales correspondientes. Vamos a crear 2 cubos, uno para cada vista de origen de datos.

CUBO 1

Empezamos con el primer cubo, creamos un nuevo cubo desde el menú contextual de cubos para que aparezca el asistente para cubos.

Seleccionamos la opción de “Usar tablas existentes” para generar el cubo.

Asistente para cubos

Seleccionar método de creación

Se pueden crear cubos usando tablas existentes, creando un cubo vacío o generando tablas en el origen de datos.

¿Cómo desea crear el cubo?

☒ Usar tablas existentes

☐ Crear un cubo vacío

☐ Generar tablas en el origen de datos

Plantilla:

(Ninguno)

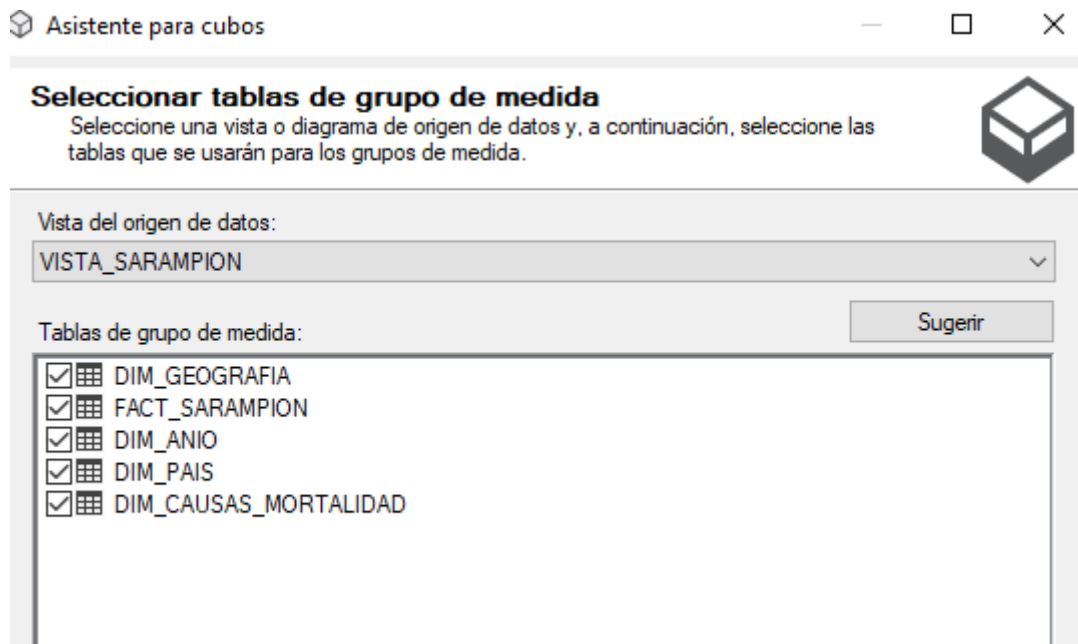
Descripción:

Cree un cubo basado en una o varias tablas de un origen de datos.

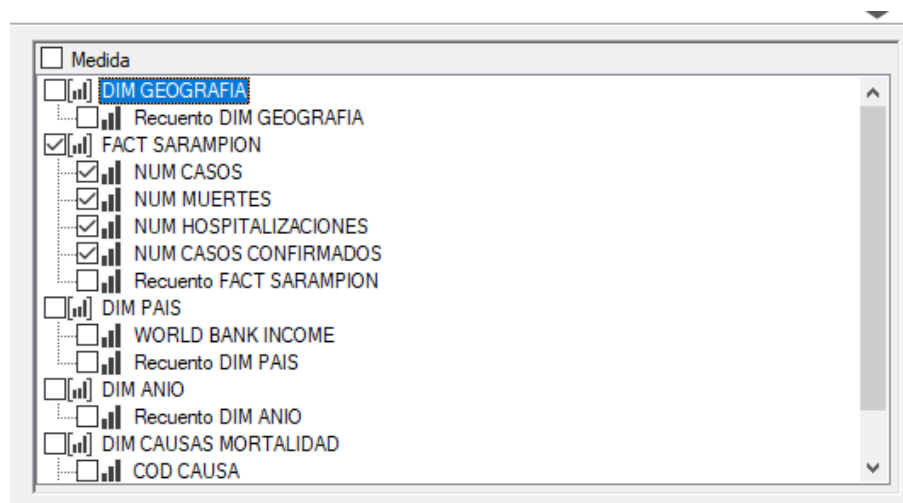
< Back Next > Finish >>| Cancel

Moves to the next wizard page

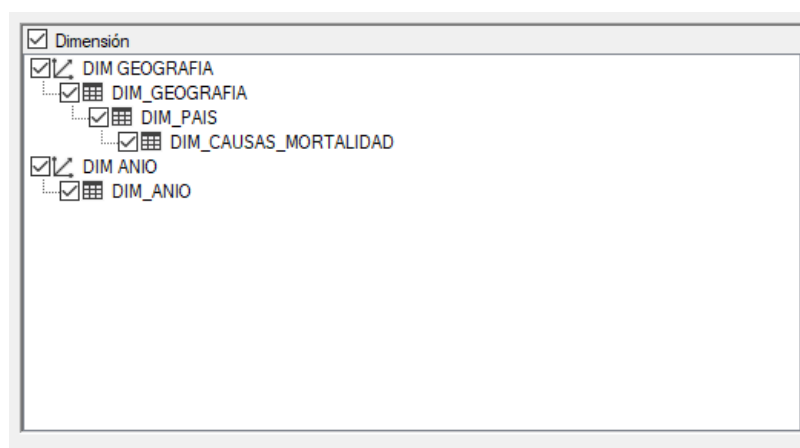
El primer cubo utilizará las tablas de la primera vista generada.



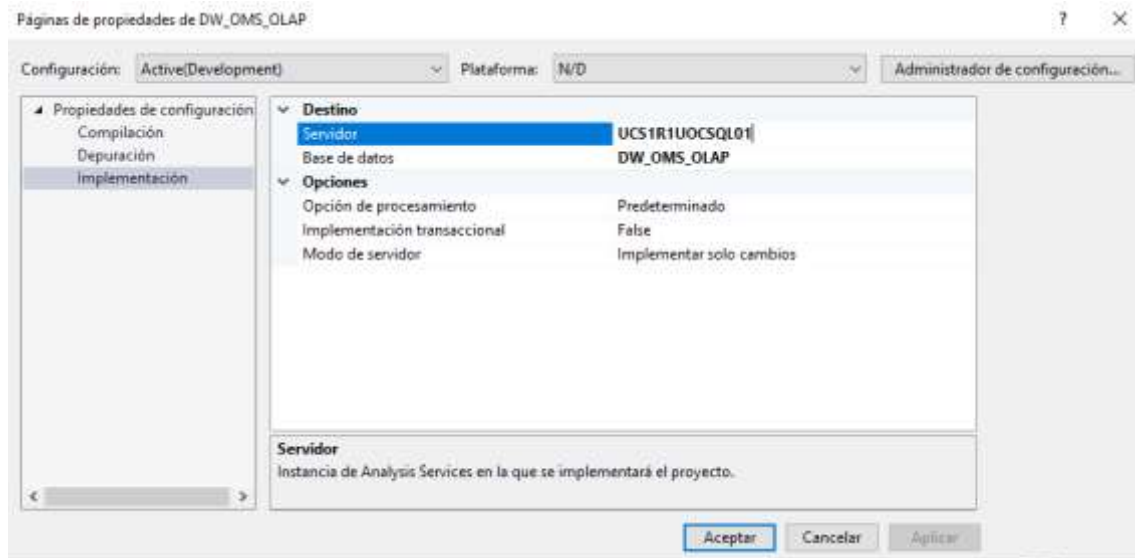
Establecemos los atributos de la tabla FACT_SARAMPION como únicas medidas.



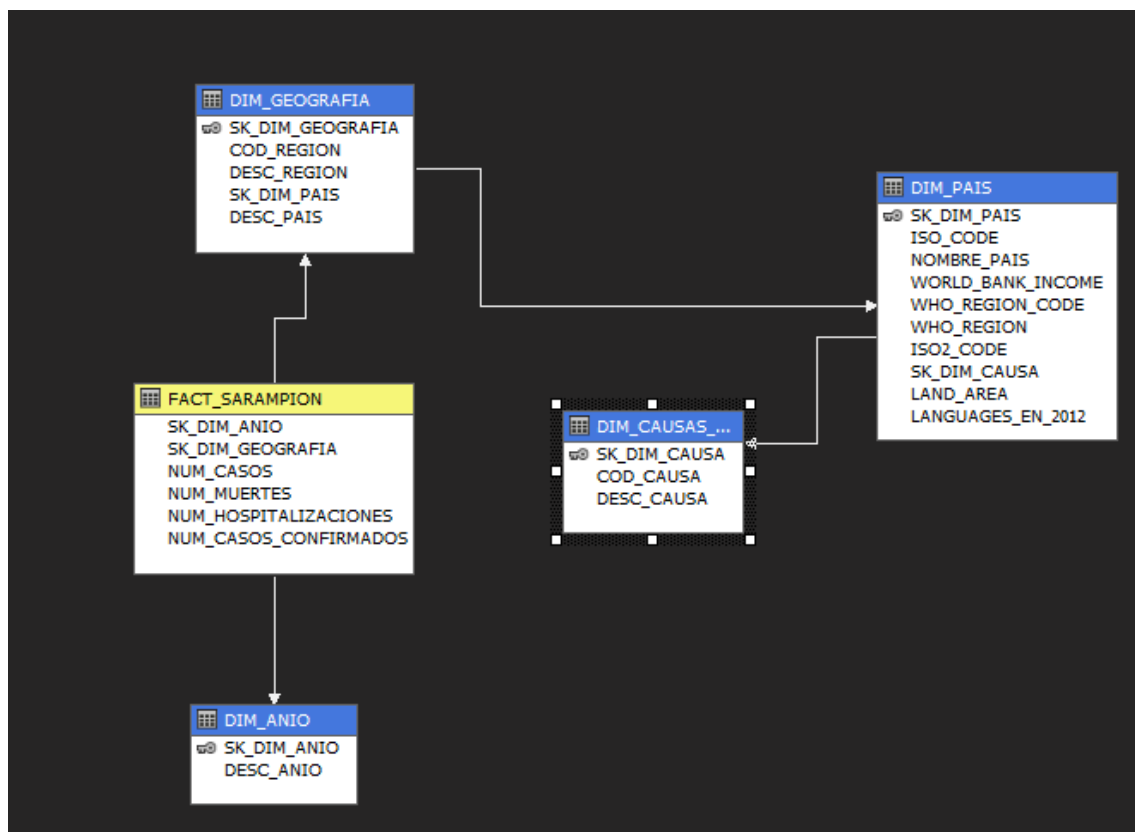
Y seleccionamos todas las dimensiones en el paso para elegir las mismas.



Para poder compilar el cubo es necesario cambiar el servidor en el que se implementará, por defecto es localhost. Se cambia en propiedades del proyecto > implementación.

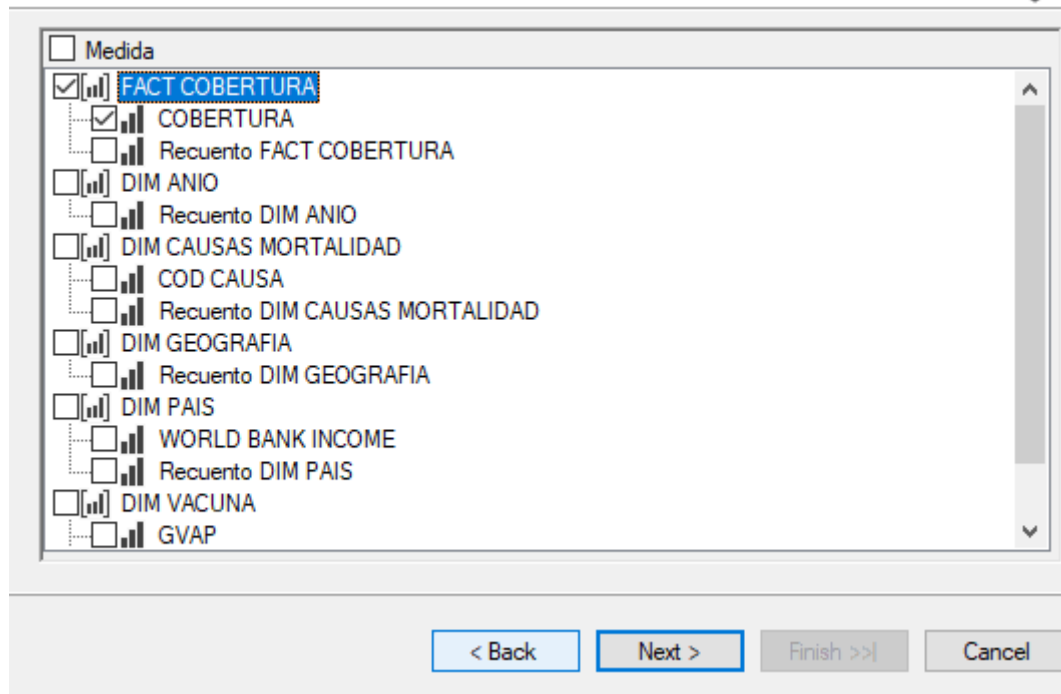


Finalmente, el diagrama del cubo es el siguiente.

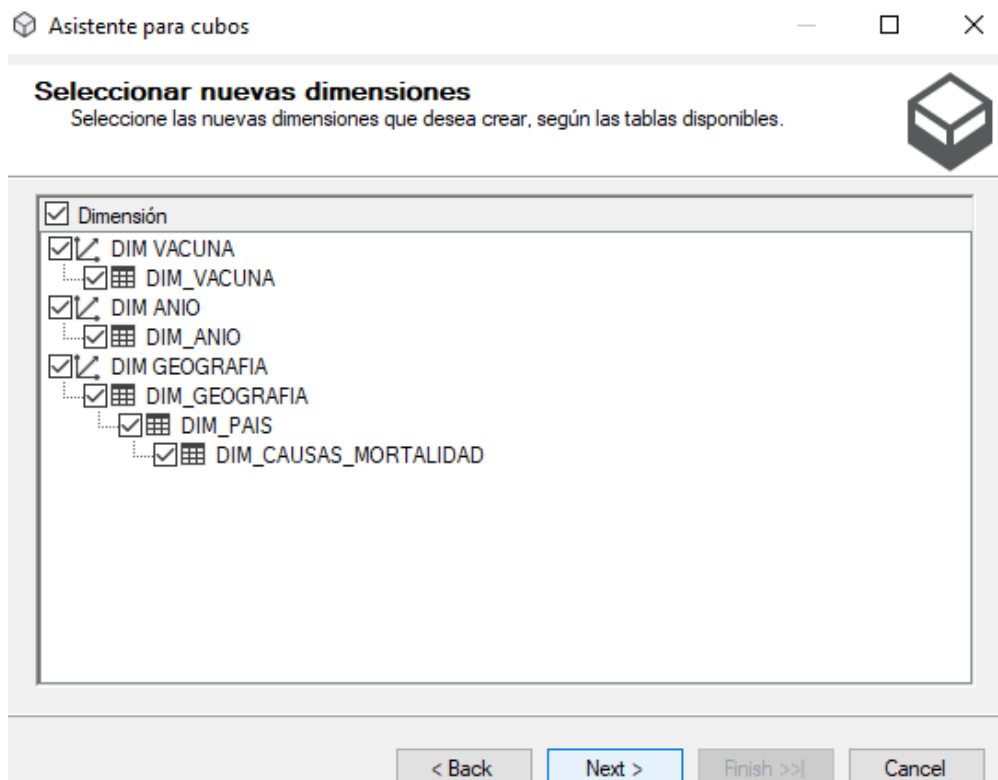


CUBO 2

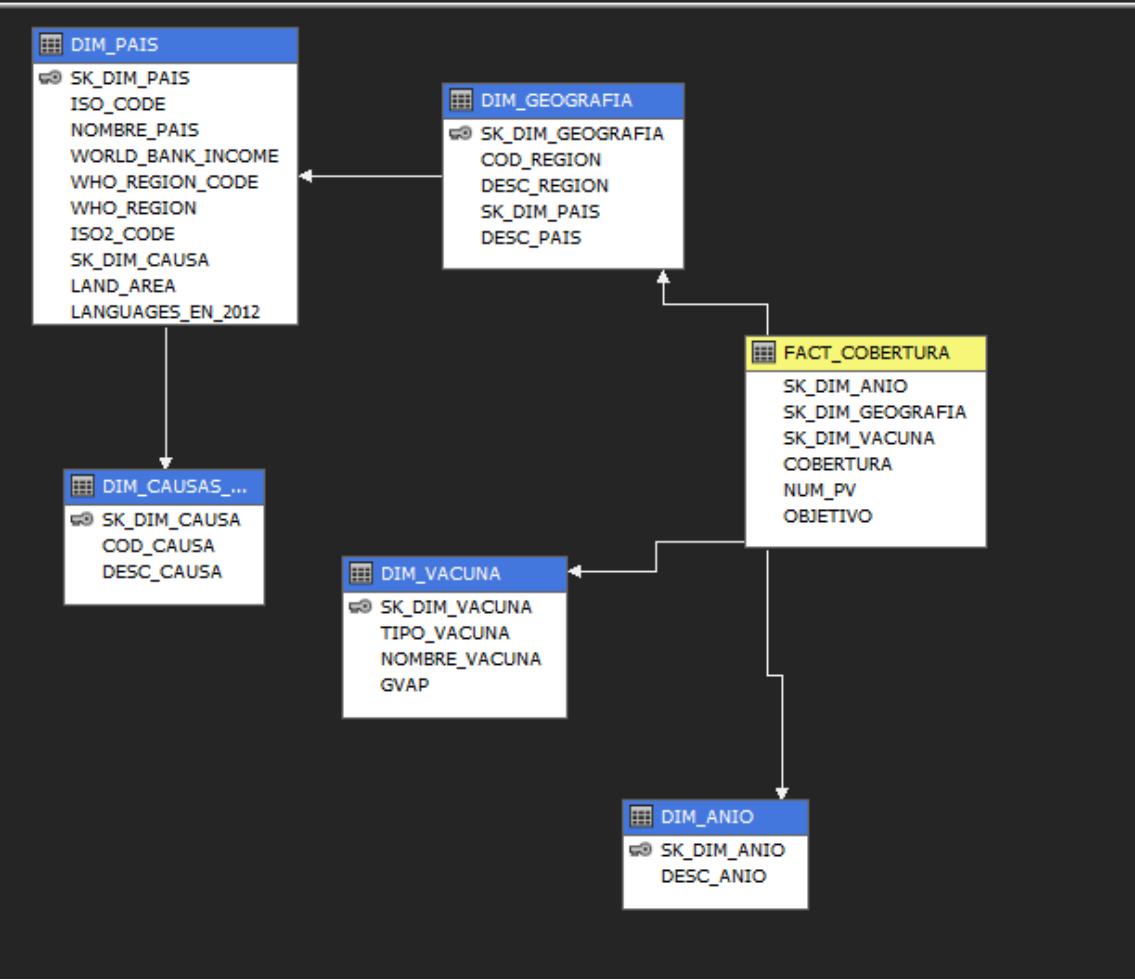
Repetimos el proceso para crear el segundo cubo, el cual está basado en la segunda vista. Seleccionamos cobertura como única métrica para el mismo.



De nuevo, seleccionamos todas las dimensiones disponibles.



Finalmente, el diagrama del cubo es el siguiente.

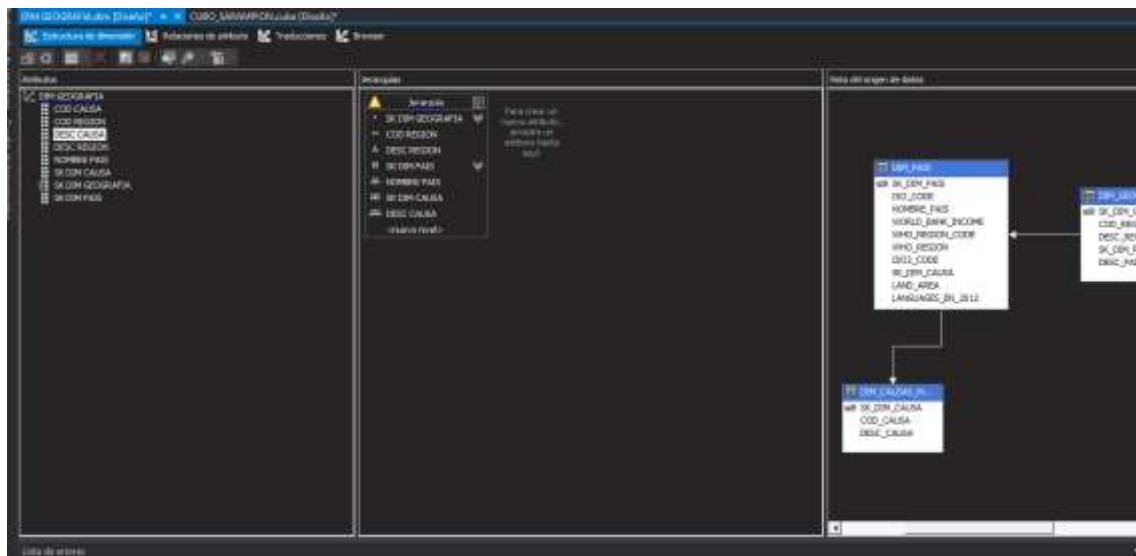


C. JERARQUÍAS Y CUBOS.

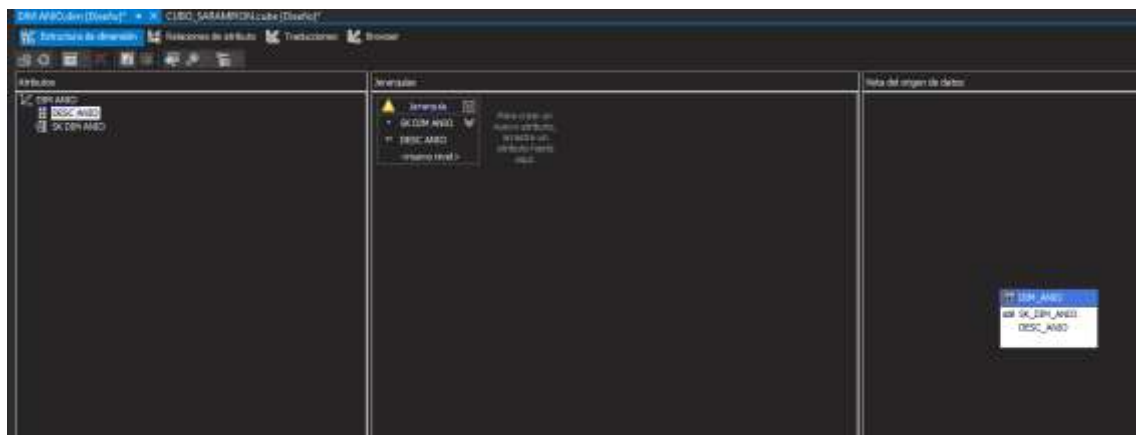
A fin de poder utilizar todos los atributos que deseamos de nuestras dimensiones en las consultas, es necesario implementarlos y generar una jerarquía.

El proceso es sencillo. Pulsando en las dimensiones se abrirán tres paneles contiguos, en el de la izquierda deberemos arrastrar los atributos de las dimensiones que queremos usar en las consultas. Y luego, arrastrando de nuevo estos atributos al panel del centro se definen las jerarquías.

DIM_GEOGRAFÍA



DIM_ANIO



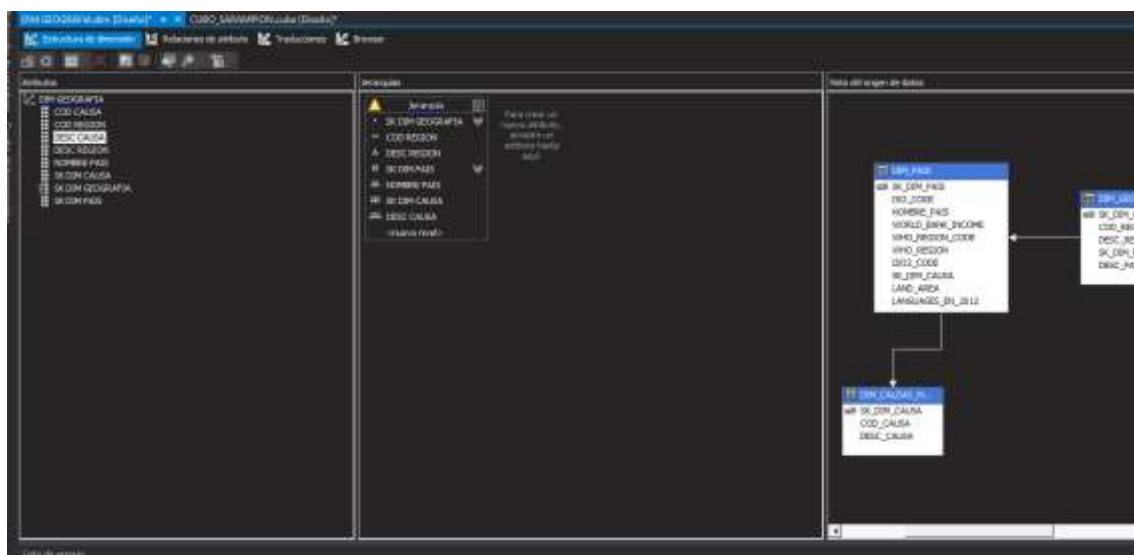
DIM VACUNA



DIM ANIO 1



DIM GEOGRAFÍA 1



Tras este paso nuestros cubos ya están listos para la fase de explotación de estos, solo es necesario pulsar en la opción de “implementar”, disponible en el menú desplegable de “Compilar”.

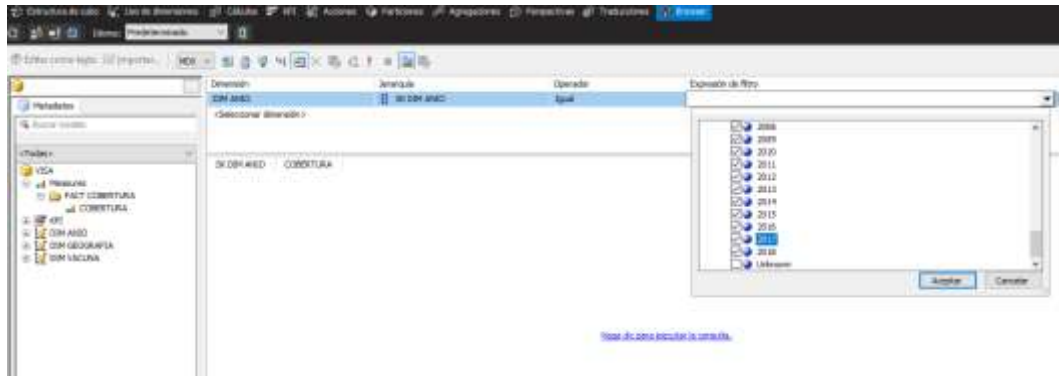
D. EXPLOTACIÓN DE LOS CUBOS.

En este apartado se muestran los resultados de las consultas que dan respuesta a las preguntas del enunciado de la práctica.

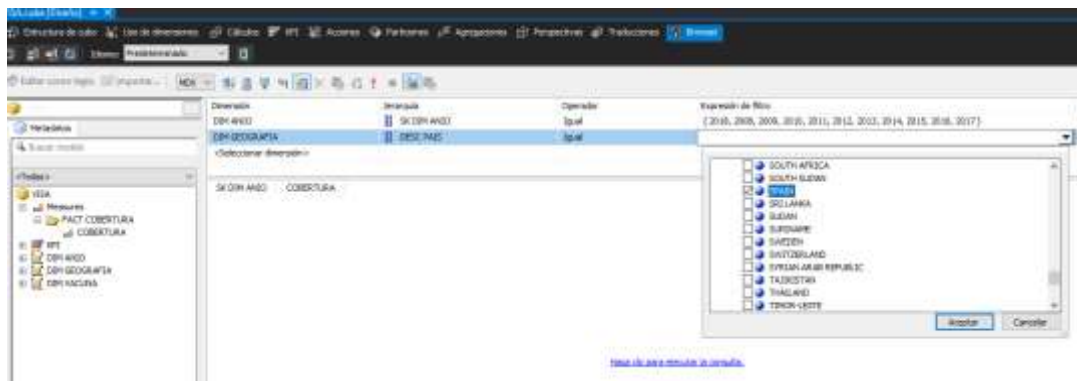
Las consultas se hacen desde el apartado Browser de los cubos.

- **Análisis de la tendencia de los diez últimos años de la cobertura de inmunización de España.**

Lo primero es establecer un filtro con solo los diez últimos años.



Y otro con España como único país admitido



El resultado es el siguiente:

Se muestra la interfaz de SQL Server Enterprise Manager. En el árbol de la izquierda, se expande el cubo 'CUBO' y se selecciona la tabla 'FactCobertura'. En el panel central, se define una consulta con la siguiente expresión de filtro: `SK_DIN_AÑO = 2008`. En el panel derecho, se muestra la lista de países disponibles para filtrar, con España seleccionada.

SK_DIN_AÑO	Cobertura
2008	1736
2009	1406
2010	1408
2011	1444
2012	1343
2013	1408
2014	1536
2015	1942
2016	1833
2017	2181
2018	2476

Si queremos ver el resultado por vacuna solo hay que añadir dicho campo a la consulta.

Como en este caso tenemos que mostrar solo los casos en los que la cobertura sea mayor a 90%, es necesario acceder al modo diseño y añadir la cláusula "HAVING". Indicando que solo deseamos seleccionar las coberturas superiores o iguales a 90.

- **Ranking de países de la OMS (Organización Mundial de la Salud) que llegan al cumplimiento del objetivo 1 para la meta 2020 propuesta por el GVAP.**
Lo primero es filtrar por vacunas del GVAP.

SK DIM ANIO	SK DIM VACUNA	DESC PAIS	COBERTURA
1966	6	CROATIA	68
1966	31	CROATIA	71
1967	6	CROATIA	71
1967	31	CROATIA	70
1968	6	CROATIA	82
1968	19	CROATIA	78
1968	31	CROATIA	82
1969	6	CROATIA	75

De acuerdo con el objetivo del 2020, dichas vacunas deben superar el 90% de cobertura. Añadimos la cláusula “HAVING”.

```

SELECT NON EMPTY { [Measures].[COBERTURA] }
ON COLUMNS,
NON EMPTY { ([DIM ANIO].[SK DIM ANIO].[SK DIM ANIO].ALLMEMBERS * [DIM VACUNA].[SK DIM VACUNA].[SK DIM VACUNA].ALLMEMBERS * [DIM GEOGRAFIA].[DESC PAIS].[DESC PAIS].ALLMEMBERS) }
HAVING [Measures].[COBERTURA] >=99

```

DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS FROM (SELECT ({ [DIM VACUNA].[GWAF].[I.] }) ON COLUMNS FROM (NTSA)) WHERE ({ [DIM VACUNA].[GWAF].[I.] }) CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS

SK DIM ANIO	SK DIM VACUNA	DESC PAIS	COBERTURA
1974	8	NETHERL	95
1974	31	NETHERL	95
1975	6	NETHERL	94
1975	8	POLAND	97
1975	31	NETHERL	94
1976	1	ESTONIA	98
1976	1	FIJI	99
1976	1	POLAND	93
1976	6	NETHERL	95
1976	6	POLAND	98
1976	31	NETHERL	95
1977	1	ESTONIA	99
1977	1	FIJI	99
1977	1	POLAND	93
1977	8	NETHERL	96
1977	8	POLAND	94
1977	8	SINGAPORE	98
1977	31	NETHERL	96
1977	31	SINGAPORE	96
1978	1	ESTONIA	97
1978	1	FIJI	99
1978	1	POLAND	94
1978	1	TURKEY	91
1978	6	NETHERL	96
1978	6	POLAND	96

Si se desea ver los países “top”, basta con modificar la cláusula “HAVING” y ponerla a 99%:

```

SELECT NON EMPTY { [Measures].[COBERTURA] }
ON COLUMNS,
NON EMPTY { ([DIM ANIO].[SK DIM ANIO].[SK DIM ANIO].ALLMEMBERS * [DIM VACUNA].[SK DIM VACUNA].[SK DIM VACUNA].ALLMEMBERS * [DIM GEOGRAFIA].[DESC PAIS].[DESC PAIS].ALLMEMBERS) }
HAVING [Measures].[COBERTURA] >=99

```

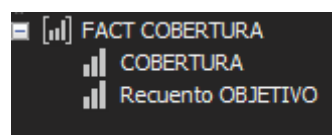
DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS FROM (SELECT ({ [DIM VACUNA].[GWAF].[I.] }) ON COLUMNS FROM (NTSA)) WHERE ({ [DIM VACUNA].[GWAF].[I.] }) CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS

SK DIM ANIO	SK DIM VACUNA	DESC PAIS	COBERTURA
1976	1	FIJI	99
1977	1	ESTONIA	99
1977	1	FIJI	99
1978	1	FIJI	99
1978	31	CUBA	99
1979	6	CHILE	99
1980	1	BULGARIA	99
1980	1	CUBA	99

- Ranking de países con mayor porcentaje de población objetivo vacunada.

Esta consulta es igual que las anteriores, solo requiere una ligera modificación, si recordamos el apartado de ETL, los datos de la población objetivo tenían el atributo “NUM_PV” como null. Utilizaremos esto en los filtros.

En el diseño del cubo, pulsamos “Nueva medida de columna” haciendo click derecho en el campo.



Realizamos la consulta. Cuando este valor no es nulo, su valor es 1.

SK DIM ANIO	DESC PAIS	NOMBRE VACUNA	COBERTURA	Recuento NUM PV
1966	CROATIA	DIPHTHERIA, TE...	68	0
1966	CROATIA	DIPHTHERIA, TE...	90	0
1966	CROATIA	THIRD DOSE OF ...	71	0
1967	CROATIA	DIPHTHERIA, TE...	71	0
1967	CROATIA	DIPHTHERIA, TE...	85	0
1967	CROATIA	THIRD DOSE OF ...	70	0
1968	CROATIA	DIPHTHERIA, TE...	82	0
1968	CROATIA	DIPHTHERIA, TE...	93	0
1968	CROATIA	MEASLES CONTA...	78	0
1968	CROATIA	THIRD DOSE OF ...	82	0
1969	CROATIA	DIPHTHERIA, TE...	75	0
1969	CROATIA	DIPHTHERIA, TE...	89	0
1969	CROATIA	MEASLES CONTA...	48	0

Filtramos con la cláusula HAVING.

```

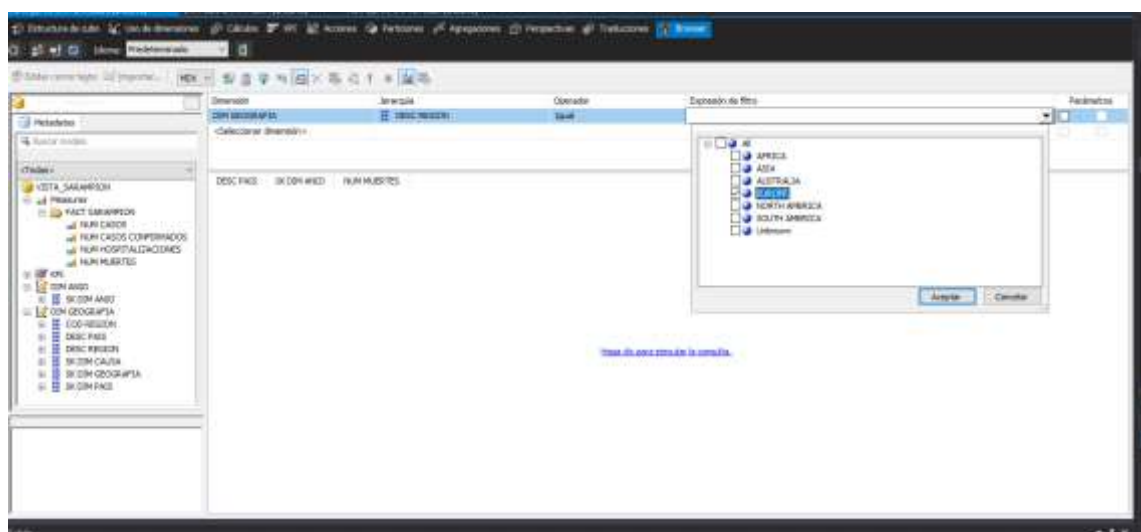
SELECT NON EMPTY ( [Measure].[COBERTURA], [Measure].[Recuento NUM PV] )
ON COLUMNS,
NON EMPTY ( ( [DIM ANDO].[SK DIM ANDO].[SK DIM ANDO].ALLMEMBERS * [DIM GEOGRAFIA].[DESC PAIS].[DESC PAIS].ALLMEMBERS * [DIM VACUNA].[NOMBRE VACUNA].[NOMBRE VACUNA].ALLMEMBERS ) )
HAVING [Measure].[Recuento NUM PV] = 1 AND [Measure].[COBERTURA] >= 0

DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_UNIQUE_NAME ON ROWS FROM [VISA] CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS

```

SK DIM ANDO	DESC PAIS	NOMBRE VACUNA	COBERTURA	Recuento NUM PV
1997	ALBANIA	DIPHTHERIA	99	1
1997	ANTIGUA	DIPHTHERIA	99	1
1997	BRUNEI D...	DIPHTHERIA	99	1
1997	COSTA RI...	FIRST DOSE OF ...	99	1
1997	CUBA	DIPHTHERIA	99	1
1997	DOMINICA	FIRST DOSE OF	99	1

- Evolución de los casos de muerte por sarampión en los países de Europa.



En este caso, basta con implementar un filtro de la dimensión geografía, en concreto el campo DESC_REGION que solo acepte información de la región de Europa.

Para poder mostrar una evolución de los casos de muerte, es necesario arrastrar a la consulta los campos muerte, año y país.

DESC PAIS	SK DIM ANIO	NUM MUERTES
ALBANIA	2008	0
ALBANIA	2009	0
ALBANIA	2010	0
ALBANIA	2011	0
ALBANIA	2012	0
ALBANIA	2013	0
ALBANIA	2014	0
ALBANIA	2015	0
ALBANIA	2016	0
ALBANIA	2017	0
ALBANIA	2018	3
ANDORRA	2008	0
ANDORRA	2009	0
ANDORRA	2010	0
ANDORRA	2011	0
ANDORRA	2012	0
ANDORRA	2013	0
ANDORRA	2014	0
ANDORRA	2015	0

- Planteamiento de otros análisis que puedan enriquecer el sistema con otras herramientas o visualizaciones que se estimen oportunas.

Un último análisis interesante podría ser una comparación entre países de distintos continentes respecto a la cobertura de la enfermedad del sarampión.

Por ejemplo, una comparativa entre Uzbekistán y España en 2016:

Dimensión	Jerarquía	Operador	Expresión de filtro
DIM ANIO	SK DIM ANIO	Igual	{ 2016 }
DIM GEOGRAFIA	DESC PAIS	Igual	{ SPAIN, UZBEKISTAN }
<Seleccionar dimensión>			
DESC PAIS	NUM CASOS		
SPAIN	97		
UZBEKISTAN	7		