



**Universitat Oberta  
de Catalunya**

**Máster universitario de Ciencia de Datos**

## **Práctica 2**

**Diseño y uso de bases de datos analíticas –  
identificación, diseño y desarrollo de los procesos ETL.**

**Autor:**

**Mario Ubierna San Mamés**



---

# Índice de Contenido

---

Índice de Contenido .....	3
Índice de tablas .....	5
Índice de ilustraciones .....	6
1. Introducción .....	11
1.1. Presentación .....	11
1.2. Descripción .....	11
2. Identificación de los procesos ETL .....	13
2.1. Bloque IN .....	13
2.2. Bloque TR.....	15
2.2.1. Dimensiones .....	15
2.2.2. Hechos .....	15
3. Diseño y desarrollo de los procesos ETL .....	17
3.1. Creación de tablas .....	17
3.1.1. Tablas del área intermedia ( <i>staging area</i> ) .....	17
3.1.2. Tablas de las dimensiones.....	21
3.1.3. Tablas de hechos .....	23
3.2. Bloque IN .....	24
3.2.1. Definición de variables de entorno .....	24
3.2.2. Conexión base de datos SQL Server .....	25
3.2.3. Transformación IN_DENUNCIAS_INFRACCIONES .....	26
3.2.4. Transformación IN_POBLACION.....	31
3.2.5. Transformación IN_MOVILIDAD.....	34
3.2.6. Transformación IN_AGLOMERACION .....	39

3.2.7.	Transformación IN_LLAMADAS112.....	46
3.2.8.	Transformación IN_FECHAS .....	52
3.3.	Bloque TR Dimensiones .....	61
3.3.1.	Transformación TR_DIM_GRUPO_EDAD .....	61
3.3.2.	Transformación TR_DIM_Medicion .....	64
3.3.3.	Transformación TR_DIM_TIPOLOGIA.....	68
3.3.4.	Transformación TR_DIM_AMBITO_GEOGRAFICO .....	72
3.3.5.	Transformación TR_DIM_FECHA.....	77
4.	Bibliografía .....	84

---

## Índice de tablas

---

Tabla 1 - Procesos ETL Bloque IN. ....	14
Tabla 2 - Procesos ETL Bloque TR Dimensiones.....	15
Tabla 3 - Procesos ETL Bloque TR Hechos.....	16

---

## Índice de ilustraciones

---

Ilustración 1 - STG_Denuncias_Infracciones.....	18
Ilustración 2 - STG_Poblacion. ....	18
Ilustración 3 - STG_Llamadas112.....	19
Ilustración 4 - STG_Movilidad.....	19
Ilustración 5 - STG_Evitar_Aglomeracion. ....	20
Ilustración 6 - STG_Fechas. ....	20
Ilustración 7 - Tablas de staging area.....	20
Ilustración 8 - DIM_Ambito_Geografico.....	21
Ilustración 9 - DIM_Fecha. ....	21
Ilustración 10 - DIM_Grupo_Edad. ....	22
Ilustración 11 - DIM_Medicion. ....	22
Ilustración 12 - DIM_Tipologia.....	22
Ilustración 13 - Tablas de dimensiones.....	23
Ilustración 14 - FACT_Llamadas112.....	23
Ilustración 15 - FACT_Mediciones. ....	23
Ilustración 16 - Alter table hechos.....	24
Ilustración 17 - Tablas de hechos.....	24
Ilustración 18 - Variables de entorno. ....	25
Ilustración 19 - Conexión a la base de datos. ....	26
Ilustración 20 - IN_DENUNCIAS_INFRACCIONES. ....	26
Ilustración 21 - Lectura IN_DENUNCIAS_INFRACCIONES. ....	27
Ilustración 22 - Lectura IN_DENUNCIAS_INFRACCIONES. ....	27
Ilustración 23 - Lectura IN_DENUNCIAS_INFRACCIONES. ....	28
Ilustración 24 - Mapeo Valores IN_DENUNCIAS_INFRACCIONES.....	28
Ilustración 25 - Normalización Strings IN_DENUNCIAS_INFRACCIONES. ....	29

Ilustración 26 - Ordenación IN_DENUNCIAS_INFRACCIONES. ....	29
Ilustración 27 - Guardado IN_DENUNCIAS_INFRACCIONES. ....	30
Ilustración 28 - Métricas IN_DENUNCIAS_INFRACCIONES. ....	30
Ilustración 29 - IN_POBLACION.....	31
Ilustración 30 - Lectura IN_POBLACION.....	31
Ilustración 31 - Separación Campos IN_POBLACION.....	32
Ilustración 32 - Mapeo Valores IN_POBLACION. ....	32
Ilustración 33 - Normalización Strings IN_POBLACION. ....	33
Ilustración 34 - Guardado IN_POBLACION.....	33
Ilustración 35 - Guardado IN_POBLACION.....	34
Ilustración 36 - Métricas IN_POBLACION.....	34
Ilustración 37 - IN_MOVILIDAD.....	35
Ilustración 38 - Lectura IN_MOVILIDAD.....	35
Ilustración 39 - Mapeo Valores IN_MOVILIDAD. ....	36
Ilustración 40 - Normalización IN_MOVILIDAD. ....	36
Ilustración 41 - Replace IN_MOVILIDAD. ....	37
Ilustración 42 - Select Values IN_MOVILIDAD. ....	37
Ilustración 43 - Guardado IN_MOVILIDAD.....	38
Ilustración 44 - Guardado IN_MOVILIDAD.....	39
Ilustración 45 - Métricas IN_MOVILIDAD. ....	39
Ilustración 46 - IN_AGLOMERACION.....	40
Ilustración 47 - Lectura IN_AGLOMERACION. ....	40
Ilustración 48 - Lectura IN_AGLOMERACION. ....	41
Ilustración 49 - Lectura IN_AGLOMERACIONES.....	41
Ilustración 50 - Mapeo Valores IN_AGLOMERACION. ....	42
Ilustración 51 - Replace IN_AGLOMERACION.....	42
Ilustración 52 - Split IN_AGLOMERACION.....	43
Ilustración 53 - Normalización Strings IN_AGLOMERACION. ....	43
Ilustración 54 - Replace IN_AGLOMERACION.....	44
Ilustración 55 - Normalización Filas IN_AGLOMERACION. ....	44
Ilustración 56 - Guardado IN_AGLOMERACIONES.....	45
Ilustración 57 - Guardado IN_AGLOMERACIONES.....	46

Ilustración 58 - Métricas IN_AGLOMERACION. ....	46
Ilustración 59 - IN_LLAMADAS112. ....	47
Ilustración 60 - Lectura IN_LLAMADAS112. ....	47
Ilustración 61 - Lectura IN_LLAMADAS112. ....	48
Ilustración 62 - Lectura IN_LLAMADAS112. ....	48
Ilustración 63 - Mapeo Valores IN_LLAMADAS112. ....	49
Ilustración 64 - Mapeo Valores IN_LLAMADAS112. ....	50
Ilustración 65 - Normalización IN_LLAMADAS112. ....	50
Ilustración 66 - Guardado IN_LLAMADAS112. ....	51
Ilustración 67 - Guardado IN_LLAMADAS112. ....	52
Ilustración 68 - Métricas IN_LLAMADAS112. ....	52
Ilustración 69 - IN_FECHAS. ....	53
Ilustración 70 - Borrado IN_FECHAS. ....	53
Ilustración 71 - Lectura IN_FECHAS. ....	54
Ilustración 72 - Añadimos Constante IN_FECHAS. ....	54
Ilustración 73 - Concatenación IN_FECHAS. ....	55
Ilustración 74 - Conversión IN_FECHAS. ....	55
Ilustración 75 - Guardado IN_FECHAS. ....	56
Ilustración 76 - Lectura IN_FECHAS. ....	57
Ilustración 77 - Guardado IN_FECHAS. ....	58
Ilustración 78 - Lectura IN_FECHAS. ....	59
Ilustración 79 - Guardado IN_FECHAS. ....	60
Ilustración 80 - Métricas IN_FECHAS. ....	60
Ilustración 81 - TR_DIM_GRUPO_EDAD. ....	61
Ilustración 82 - Borrado TR_DIM_GRUPO_EDAD. ....	61
Ilustración 83 - Grid TR_DIM_GRUPO_EDAD. ....	62
Ilustración 84 - Grid TR_DIM_GRUPO_EDAD. ....	62
Ilustración 85 - Normalización TR_DIM_GRUPO_EDAD. ....	62
Ilustración 86 - Secuenciación TR_DIM_GRUPO_EDAD. ....	63
Ilustración 87 - Guardado TR_DIM_GRUPO_EDAD. ....	63
Ilustración 88 - Guardado TR_DIM_GRUPO_EDAD. ....	64
Ilustración 89 - Métricas TR_DIM_GRUPO_EDAD. ....	64



Ilustración 90 - TR_DIM_MEDICION. ....	65
Ilustración 91 - Borrado TR_DIM_MEDICION. ....	65
Ilustración 92 - Grid TR_DIM_DIM_MEDICION.....	65
Ilustración 93 - Grid TR_DIM_MEDICION. ....	66
Ilustración 94 - Normalización TR_DIM_MEDICION. ....	66
Ilustración 95 - Secuenciación TR_DIM_MEDICION. ....	67
Ilustración 96 - Guardado TR_DIM_MEDICION. ....	67
Ilustración 97 - Guardado TR_DIM_MEDICION. ....	68
Ilustración 98 - Métricas TR_DIM_MEDICIONES. ....	68
Ilustración 99 - TR_DIM_TIPOLOGIA.....	69
Ilustración 100 - Borrado TR_DIM_TIPOLOGIA.....	69
Ilustración 101 - Lectura TR_DIM_TIPOLOGIA.....	70
Ilustración 102 - Secuenciación TR_DIM_TIPOLOGIA.....	71
Ilustración 103 - Guardado TR_DIM_TIPOLOGIA.....	71
Ilustración 104 - Guardado TR_DIM_TIPOLOGIA.....	72
Ilustración 105 - Métricas TR_DIM_TIPOLOGIA. ....	72
Ilustración 106 - TR_DIM_AMBITO_GEOGRAFICO. ....	73
Ilustración 107 - Borrado TR_DIM_AMBITO_GEOGRAFICO. ....	73
Ilustración 108 - Lectura TR_DIM_AMBITO_GEOGRAFICO. ....	74
Ilustración 109 - Lectura TR_DIM_AMBITO_GEOGRAFICO. ....	74
Ilustración 110 - Nulos TR_DIM_AMBITO_GEOGRAFICO. ....	75
Ilustración 111 - Secuenciación TR_DIM_AMBITO_GEOGRAFICO. ....	76
Ilustración 112 - Guardado TR_DIM_AMBITO_GEOGRAFICO. ....	76
Ilustración 113 - Guardado TR_DIM_AMBITO_GEOGRAFICO. ....	77
Ilustración 114 - Métricas TR_DIM_AMBITO_GEOGRAFICO. ....	77
Ilustración 115 - TR_DIM_FECHA.....	78
Ilustración 116 - Borrado TR_DIM_FECHA.....	78
Ilustración 117 - Lectura TR_DIM_FECHA.....	79
Ilustración 118 - Conversión String TR_DIM_FECHA. ....	79
Ilustración 119 - Split TR_DIM_FECHA.....	80
Ilustración 120 - Split TR_DIM_FECHA.....	80
Ilustración 121 - Concatenación TR_DIM_FECHA.....	81

Ilustración 122 - Conversión TR\_DIM\_FECHA..... 81

Ilustración 123 - Secuenciación TR\_DIM\_FECHA..... 82

Ilustración 124 - Guardado TR\_DIM\_FECHA..... 82

Ilustración 125 - Métricas TR\_DIM\_FECHA..... 83

---

# 1. Introducción

---

## 1.1. Presentación

A partir de la solución oficial de la primera práctica (PRA1), el estudiante debe diseñar, implementar y ejecutar los procesos de extracción, transformación y carga de los datos de las fuentes de datos proporcionadas.

Así pues, esta actividad tiene como objetivo identificar y desarrollar los procesos de carga del almacén de datos y que esta sea efectiva.

## 1.2. Descripción

Si nos centramos en los subobjetivos, esta segunda parte del caso práctico consiste en lo siguiente:

- Identificar los procesos de extracción, transformación y carga de datos (ETL) hacia el almacén de datos.
- Diseñar y desarrollar los procesos ETL mediante las herramientas de diseño proporcionadas.
- Implementar con los trabajos (*jobs*) los procesos ETL para que su carga planificada sea efectiva.

Además del documento con la solución de la PRA2 que se debe entregar, también se tendrá en consideración la implementación sobre la máquina virtual proporcionada en el curso.

En resumen, el documento de la solución de la PRA2 debe incluir los siguientes aspectos:

- Descripción de todas las acciones que se han realizado.

- Capturas de pantalla que muestren todas las partes significativas del ETL, sus características y su correspondiente explicación.
- Capturas de pantalla que demuestren la correcta ejecución de la ETL y el tiempo de ejecución.
- Capturas de pantalla que demuestren la correcta carga de los datos (cargados en la base de datos).

---

## 2. Identificación de los procesos ETL

---

A la hora de diseñar los procesos de carga de una base de datos analítica no hay una única estrategia. Es habitual estructurar los procesos ETL sobre la base de las entidades de datos que se deben actualizar, ya que existen diferencias conceptuales en la actualización de una dimensión con respecto a la de una tabla de hechos. La división del proceso de carga inicial en diferentes bloques de actualización facilitará el diseño de un orden de ejecución y la gestión de las dependencias. Cada uno de estos bloques de actualización se dividirá en las correspondientes etapas de extracción, transformación y carga.

Se identifican los dos bloques siguientes:

- **Bloque IN:** procesos de carga de los datos desde las fuentes a las tablas intermedias en el área de maniobras (*staging area*). Estos procesos se distinguen por el prefijo «IN\_» en el nombre.
- **Bloque TR:** procesos de transformación para cargar los datos desde las tablas intermedias hasta nuestro almacén, según el modelo multidimensional diseñado. Así pues, son diferentes los procesos ETL de transformación para cargar las dimensiones de aquellos que se realizan para cargar las tablas de hechos. Estos procesos se distinguen con el prefijo «TR\_» en el nombre.

### 2.1. Bloque IN

Respecto al bloque In, el cual nos va a permitir almacenar la información en el staging area, tenemos los siguientes procesos:

Nombre ETL	Descripción	Orígenes de los datos	Tabla de destino (stage)
------------	-------------	-----------------------	--------------------------

IN_ DENUNCIAS_ INFRACCIONES	Carga de los datos correspondientes a las estadísticas sobre los expedientes incoados por el artículo 36.6 LOPSC de desobediencia durante el estado de emergencia sanitaria COVID-19 en la comunidad de Euskadi.	ACUMULADO-DENUNCIAS-INFRACCIONES.xlsx	STG_Denuncias_Infracciones
IN_POBLACION	Carga los datos respectivos a las cifras de la población española.	población_9687bsc.csv	STG_Poblacion
IN_MOVILIDAD	Movilidad de la población durante el estado de alarma.	35167bsc.csv	STG_Movilidad
IN_AGLOMERACION	Porcentaje de la población que evitaba las aglomeraciones con motivo del coronavirus, por grupo de edad y provincia.	statistic_id1104235_covid19_-poblacion-que-evitabalas-aglomeraciones-seguridad-en-espana-2020.xlsx	STG_Evitar_Aglomeracion
IN_LLAMADAS_112	Llamadas al 112 por ámbito geográfico y tipología (accidentes de tráfico, civismo, incendios, asistencia sanitaria, seguridad...)	rows.xml	STG_Llamadas112
IN_FECHAS	Almacenamos todas las fechas de todos los ficheros de datos proporcionados.	STG_Llamadas112 STG_Denuncias_Infracciones STG_Movilidad	STG_Fechas

Tabla 1 - Procesos ETL Bloque IN.

## 2.2. Bloque TR

Respecto al bloque TR tenemos tanto los procesos para dotar de datos a las dimensiones como a los hechos.

### 2.2.1. Dimensiones

Los procesos ETL que se encargan de añadir la información a las dimensiones son los siguientes:

Nombre del ETL	Descripción	Tabla de origen	Tabla de destino (dimensión)
TR_DIM_FECHA	Carga y transformación de la dimensión temporal.	STG_Fechas	DIM_Fecha
TR_DIM_AMBITO_GEOGRAFICO	Carga y transformación de la dimensión con los datos de los ámbitos geográficos.	STG_Poblacion STG_Llamadas112 STG_Evitar_Aglomeration	DIM_Ambito_Geografico
TR_DIM_GRUPO_EDAD	Carga y transformación de la dimensión con los datos de los grupos de edad.	Manual, a partir de un grid.	DIM_grupo_Edad
TR_DIM_MEDICION	Carga y transformación de la dimensión con los datos de las mediciones.	Manual, a partir de un grid.	DIM_Medicion
TR_DIM_TIPOLOGIA	Carga y transformación de la dimensión con los datos de la tipología.	STG_Llamadas112	DIM_Tipologia

*Tabla 2 - Procesos ETL Bloque TR Dimensiones.*

### 2.2.2. Hechos

Respecto a los hechos tenemos los siguientes procesos de carga:

Nombre del ETL	Descripción	Tabla de origen
TR_FACT_LLAMADAS112	Carga y transformación de la tabla de hechos Fact_Llamadas112.	STG_Llamadas112
TR_FACT_MEDICIONES	Carga y transformación de la tabla de hechos Fact_Mediciones	STG_Denuncias_infracciones STG_Evitar_Aglomeracion STG_Movilidad STG_Poblacion

*Tabla 3 - Procesos ETL Bloque TR Hechos.*



---

## 3. Diseño y desarrollo de los procesos ETL

---

En este apartado, se deben diseñar los procesos de carga identificados en el punto anterior con la herramienta de diseño proporcionada. En este caso es Pentho Data Integration (PDI).

### 3.1. Creación de tablas

El primer paso para la implementación de los procesos ETL consiste en la creación de las tablas. Esto se llevará a cabo una única vez, mediante *scripts*, sobre la base de datos proporcionada (en nuestro caso: SQL Server). Se deberán crear las tablas intermedias y las tablas del modelo dimensional de la solución oficial, es decir, las dimensiones y las tablas de hechos. Para hacerlo, deben utilizarse los *scripts* facilitados junto a la solución de la PRA1.

#### 3.1.1. Tablas del área intermedia (*staging area*)

Lo primero que vamos a hacer es la creación de las tablas intermedias:

**Tabla intermedia STG\_Denuncias\_Infracciones**

```

USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Denuncias_Infracciones](
    [provincia] [varchar](100) NULL,
    [identificados_ertzaintza] [float] NULL,
    [detenidos_ertzaintza] [float] NULL,
    [denuncias_ertzaintza] [float] NULL,
    [vehic_intercept_ertzaintza] [float] NULL,
    [identificados_ppll] [float] NULL,
    [detenidos_ppll] [float] NULL,
    [denuncias_ppll] [float] NULL,
    [vehic_intercept_ppll] [float] NULL,
    [fecha] [datetime] NULL
) ON [PRIMARY]
GO

```

*Ilustración 1 - STG\_Denuncias\_Infracciones.*

**Tabla intermedia STG\_Poblacion**

```

USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Poblacion](
    [provincia_codigo] [varchar](2) NULL,
    [provincia_nombre] [varchar](100) NULL,
    [poblacion] [bigint] NULL,
    [periodo] [varchar](25) NULL
) ON [PRIMARY]
GO

```

*Ilustración 2 - STG\_Poblacion.*

**Tabla intermedia STG\_Llamadas112**

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Llamadas112](
    [anio] [int] NULL,
    [mes] [int] NULL,
    [provincia] [varchar](100) NULL,
    [comarca] [varchar](100) NULL,
    [municipio] [varchar](100) NULL,
    [tipo] [varchar](100) NULL,
    [llamadas] [int] NULL
) ON [PRIMARY]
```

*Ilustración 3 - STG\_Llamadas112.***Tabla intermedia STG\_Movilidad**

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Movilidad](
    [zonas_movilidad] [varchar](27) NULL,
    [periodo] [datetime] NULL,
    [total] [decimal](5, 2) NULL
) ON [PRIMARY]
GO
```

*Ilustración 4 - STG\_Movilidad.*

**Tabla intermedia STG\_Evitar\_Aglomeracion**

```

USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Evitar_Aglomeracion](
    [provincia] [varchar](100) NULL,
    [comunidad_autonoma] [varchar](100) NULL,
    [grupo_edad] [varchar](7) NULL,
    [porc_poblacion] [float] NULL
) ON [PRIMARY]
GO

```

*Ilustración 5 - STG\_Evitar\_Aglomeracion.***Tabla intermedia STG\_Fechas**

```

USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Fechas]
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[STG_Fechas](
    [fecha] [date] NOT NULL
) ON [PRIMARY]
GO

```

*Ilustración 6 - STG\_Fechas.*

Comprobamos que todas las tabla intermedias se han creado correctamente:

```

+  [dbo.STG_Denuncias_Infracciones]
+  [dbo.STG_Evitar_Aglomeracion]
+  [dbo.STG_Fechas]
+  [dbo.STG_Llamadas112]
+  [dbo.STG_Movilidad]
+  [dbo.STG_Poblacion]

```

*Ilustración 7 - Tablas de staging area.*

### 3.1.2. Tablas de las dimensiones

Lo segundo que debemos de hacer es la creación de las tablas de dimensiones:

#### Tabla dimensión DIM\_Ambito\_Geografico

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[DIM_Ambito_Geografico](
    [pk_ambito_geografico] [int] NOT NULL,
    [provincia_codigo] [varchar](2) NOT NULL,
    [provincia_nombre] [varchar](100) NOT NULL,
    [comunidad_autonoma] [varchar](100) NULL,
    [comarca] [varchar](100) NULL,
    [municipio] [varchar](100) NULL,
    CONSTRAINT [PK_DIM_Ambito_Geografico] PRIMARY KEY CLUSTERED
(
    [pk_ambito_geografico] 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
```

*Ilustración 8 - DIM\_Ambito\_Geografico.*

#### Tabla dimensión DIM\_Fecha

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[DIM_Fecha](
    [pk_fecha] [int] NOT NULL,
    [anyo] [int] NOT NULL,
    [mes] [int] NOT NULL,
    [dia] [int] NOT NULL,
    [fecha] [date] NOT NULL,
    CONSTRAINT [PK_DIM_Fecha] PRIMARY KEY CLUSTERED
(
    [pk_fecha] 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
```

*Ilustración 9 - DIM\_Fecha.*

### Tabla dimensión DIM\_Grupo\_Edad

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[DIM_Grupo_Edad](
    [pk_grupo_edad] [int] NOT NULL,
    [nombre] [varchar](20) NOT NULL,
    [intervalo] [varchar](20) NOT NULL,
    CONSTRAINT [PK_DIM_Grupo_Edad] PRIMARY KEY CLUSTERED
(
    [pk_grupo_edad] 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
```

*Ilustración 10 - DIM\_Grupo\_Edad.*

### Tabla dimensión DIM\_Medicion

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[DIM_Medicion](
    [pk_medicion] [int] NOT NULL,
    [nombre] [varchar](100) NOT NULL,
    [unidad_medida] [varchar](20) NOT NULL,
    CONSTRAINT [PK_DIM_Medicion] PRIMARY KEY CLUSTERED
(
    [pk_medicion] 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
```

*Ilustración 11 - DIM\_Medicion.*

### Tabla dimensión DIM\_Tipologia

```
USE [DB_marioum]
GO

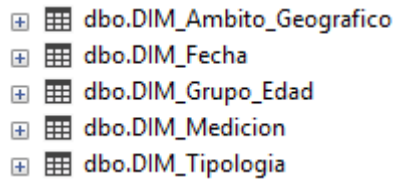
/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[DIM_Tipologia](
    [pk_tipologia] [int] NOT NULL,
    [nombre] [varchar](100) NOT NULL,
    CONSTRAINT [PK_DIM_Tipologia] PRIMARY KEY CLUSTERED
(
    [pk_tipologia] 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
```

*Ilustración 12 - DIM\_Tipologia.*

Comprobamos que todas las tablas de dimensiones se han creado correctamente:



*Ilustración 13 - Tablas de dimensiones.*

### 3.1.3. Tablas de hechos

Finalmente creamos las diferentes tablas de los hechos:

#### Tabla hecho FACT\_Llamadas112

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[FACT_Llamadas112](
    [pk_fk_fecha] [int] NOT NULL,
    [pk_fk_ambito_geografico] [int] NOT NULL,
    [pk_fk_tipologia] [int] NOT NULL,
    [llamadas] [int] NULL,
    CONSTRAINT [PK_FACT_Llamadas112] PRIMARY KEY CLUSTERED
    (
        [pk_fk_fecha] ASC,
        [pk_fk_ambito_geografico] ASC,
        [pk_fk_tipologia] 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
```

*Ilustración 14 - FACT\_Llamadas112.*

#### Tabla hecho FACT\_Mediciones

```
USE [DB_marioum]
GO

/***** Object: Table [dbo].[STG_Denuncias_Infracciones] *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[FACT_Mediciones](
    [pk_id] [int] NOT NULL,
    [fk_fecha] [int] NOT NULL,
    [fk_ambito_geografico] [int] NOT NULL,
    [fk_grupo_edad] [int] NOT NULL,
    [fk_medicion] [int] NOT NULL,
    [valor] [decimal](17, 2) NULL,
    CONSTRAINT [PK_FACT_Mediciones] PRIMARY KEY CLUSTERED
    (
        [pk_id] 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
```

*Ilustración 15 - FACT\_Mediciones.*

Realizamos los alter table de las tablas de hechos:

```

ALTER TABLE [dbo].[FACT_Llamadas112] WITH CHECK ADD CONSTRAINT [FK_FACT_Llamadas112_DIM_Ambito_Geografico] FOREIGN KEY([pk_fk_ambito_geografico])
REFERENCES [dbo].[DIM_Ambito_Geografico] ([pk_ambito_geografico])
GO

ALTER TABLE [dbo].[FACT_Llamadas112] CHECK CONSTRAINT [FK_FACT_Llamadas112_DIM_Ambito_Geografico]
GO

ALTER TABLE [dbo].[FACT_Llamadas112] WITH CHECK ADD CONSTRAINT [FK_FACT_Llamadas112_DIM_Fecha] FOREIGN KEY([pk_fk_fecha])
REFERENCES [dbo].[DIM_Fecha] ([pk_fecha])
GO

ALTER TABLE [dbo].[FACT_Llamadas112] CHECK CONSTRAINT [FK_FACT_Llamadas112_DIM_Fecha]
GO

ALTER TABLE [dbo].[FACT_Llamadas112] WITH CHECK ADD CONSTRAINT [FK_FACT_Llamadas112_DIM_Tipologia] FOREIGN KEY([pk_fk_tipologia])
REFERENCES [dbo].[DIM_Tipologia] ([pk_tipologia])
GO

ALTER TABLE [dbo].[FACT_Llamadas112] CHECK CONSTRAINT [FK_FACT_Llamadas112_DIM_Tipologia]
GO

ALTER TABLE [dbo].[FACT_Mediciones] WITH CHECK ADD CONSTRAINT [FK_FACT_Mediciones_DIM_Ambito_Geografico] FOREIGN KEY([fk_ambito_geografico])
REFERENCES [dbo].[DIM_Ambito_Geografico] ([pk_ambito_geografico])
GO

ALTER TABLE [dbo].[FACT_Mediciones] CHECK CONSTRAINT [FK_FACT_Mediciones_DIM_Ambito_Geografico]
GO

ALTER TABLE [dbo].[FACT_Mediciones] WITH CHECK ADD CONSTRAINT [FK_FACT_Mediciones_DIM_Fecha] FOREIGN KEY([fk_fecha])
REFERENCES [dbo].[DIM_Fecha] ([pk_fecha])
GO

ALTER TABLE [dbo].[FACT_Mediciones] CHECK CONSTRAINT [FK_FACT_Mediciones_DIM_Fecha]
GO

ALTER TABLE [dbo].[FACT_Mediciones] WITH CHECK ADD CONSTRAINT [FK_FACT_Mediciones_DIM_Grupo_Edad] FOREIGN KEY([fk_grupo_edad])
REFERENCES [dbo].[DIM_Grupo_Edad] ([pk_grupo_edad])
GO

ALTER TABLE [dbo].[FACT_Mediciones] CHECK CONSTRAINT [FK_FACT_Mediciones_DIM_Grupo_Edad]
GO

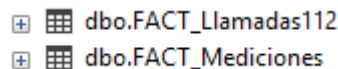
ALTER TABLE [dbo].[FACT_Mediciones] WITH CHECK ADD CONSTRAINT [FK_FACT_Mediciones_DIM_Medicion] FOREIGN KEY([fk_medicion])
REFERENCES [dbo].[DIM_Medicion] ([pk_medicion])
GO

ALTER TABLE [dbo].[FACT_Mediciones] CHECK CONSTRAINT [FK_FACT_Mediciones_DIM_Medicion]
GO

```

*Ilustración 16 - Alter table hechos.*

Comprobamos que se han creado todas las tablas correspondientes:



*Ilustración 17 - Tablas de hechos.*

## 3.2. Bloque IN

En este bloque se van a realizar las transformaciones para que la información en forma bruta se pase a las tablas intermedias, y luego haremos uso de éstas para crear las transformaciones de dimensiones y hechos.

### 3.2.1. Definición de variables de entorno

Es una buena práctica utilizar variables de entorno para así poder evitar errores en el definiciones futuras. Para ello accedemos a *kettle.properties* y definimos las siguientes variables:

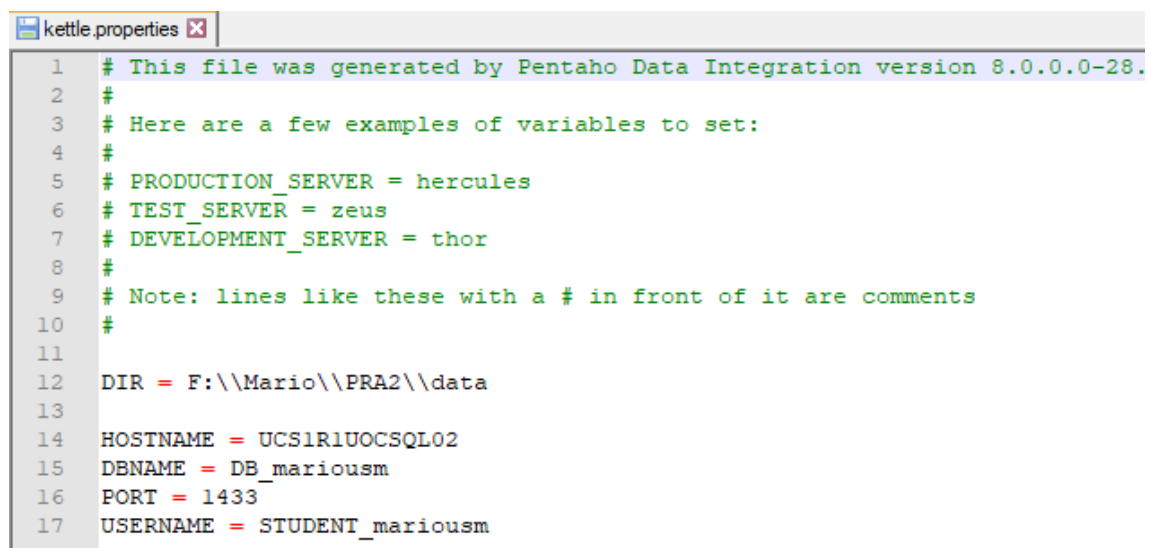
Para el origen en el que se encuentran todos los archivos definimos la variables DIR\_ENT:



- Nombre: DIR
- Valor: F:\Mario\PRA2\data

Para la cadena de conexión a la base de datos vamos a usar:

- Nombre: HOSTNAME
- Valor: UCS1R1UOCSQL02
- Nombre: DBNAME
- Valor: DB\_mariouism
- Nombre: PORT
- Valor: 1433
- Nombre: USERNAME
- Valor: STUDENT\_mariouism

A screenshot of the 'kettle.properties' window in Pentaho Data Integration. The window has a title bar with the text 'kettle.properties' and a close button. The main area contains a text editor with the following content:

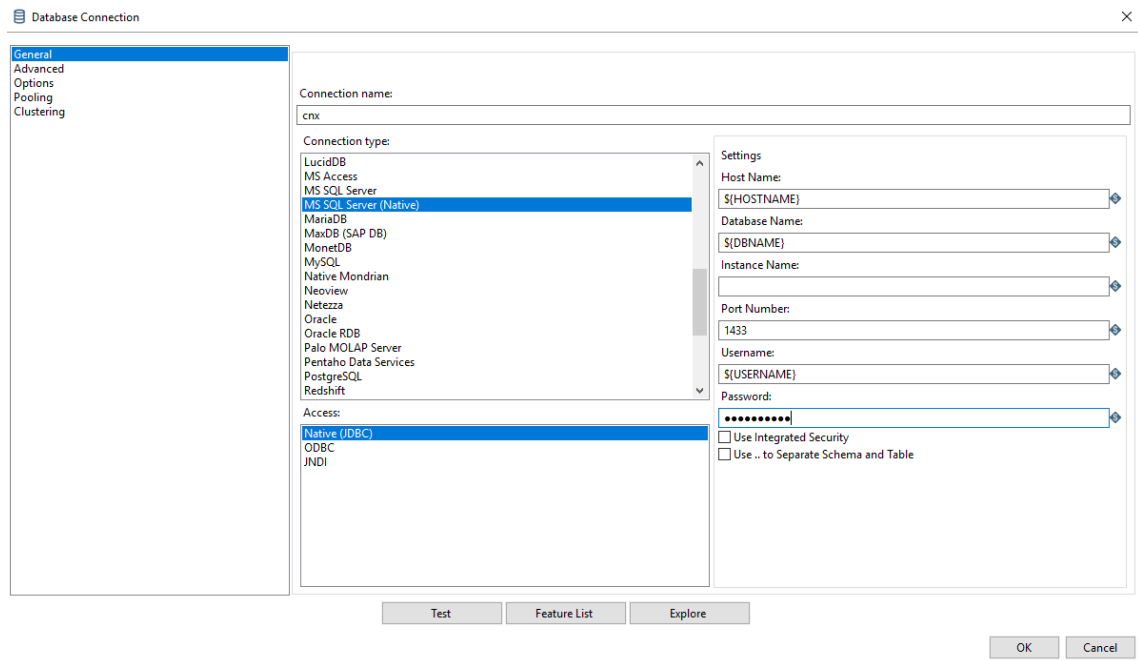
```
1  # This file was generated by Pentaho Data Integration version 8.0.0.0-28.
2  #
3  # Here are a few examples of variables to set:
4  #
5  # PRODUCTION_SERVER = hercules
6  # TEST_SERVER = zeus
7  # DEVELOPMENT_SERVER = thor
8  #
9  # Note: lines like these with a # in front of it are comments
10 #
11
12 DIR = F:\\Mario\\PRA2\\data
13
14 HOSTNAME = UCS1R1UOCSQL02
15 DBNAME = DB_mariouism
16 PORT = 1433
17 USERNAME = STUDENT_mariouism
```

*Ilustración 18 - Variables de entorno.*

### 3.2.2. Conexión base de datos SQL Server

El siguiente paso es crear la conexión a la base de datos que va a ser usada tanto por las transformaciones como por los jobs que se realicen en esta práctica.

Para ello creamos la nueva conexión y establecemos los valores definidos en las variables de entorno:



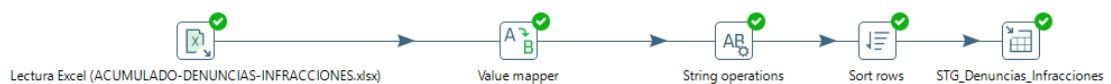
*Ilustración 19 - Conexión a la base de datos.*

### 3.2.3. Transformación IN\_DENUNCIAS\_INFRACCIONES

Una vez que ya hemos definido las variables de entorno y la conexión podemos proceder a realizar todas las transformaciones y trabajos.

La primera transformación que vamos a realizar se llama “IN\_DENUNCIAS\_INFRACCIONES”, su objetivo es leer todos los datos del archivo “ACUMULADO-DENUNCIAS-INFRACCIONES.xlsx” en la tabla intermedia “STG\_Denuncias\_Infracciones”.

En este caso no hemos hecho ninguna modificación en el Excel original, por lo que la transformación nos queda de la siguiente forma:

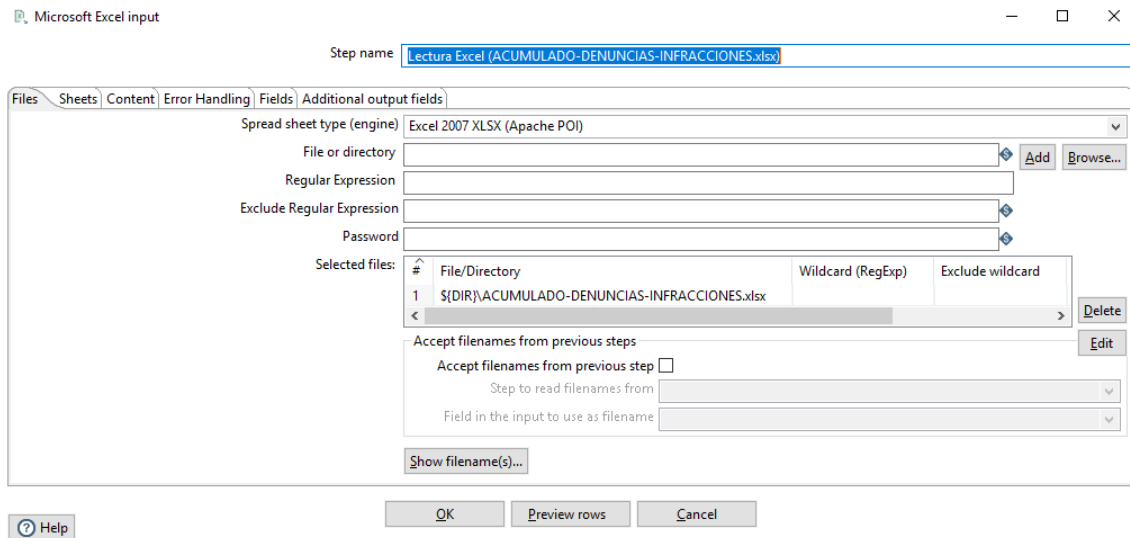


*Ilustración 20 - IN\_DENUNCIAS\_INFRACCIONES.*

Ahora vamos a explicar paso a paso lo que hemos hecho:

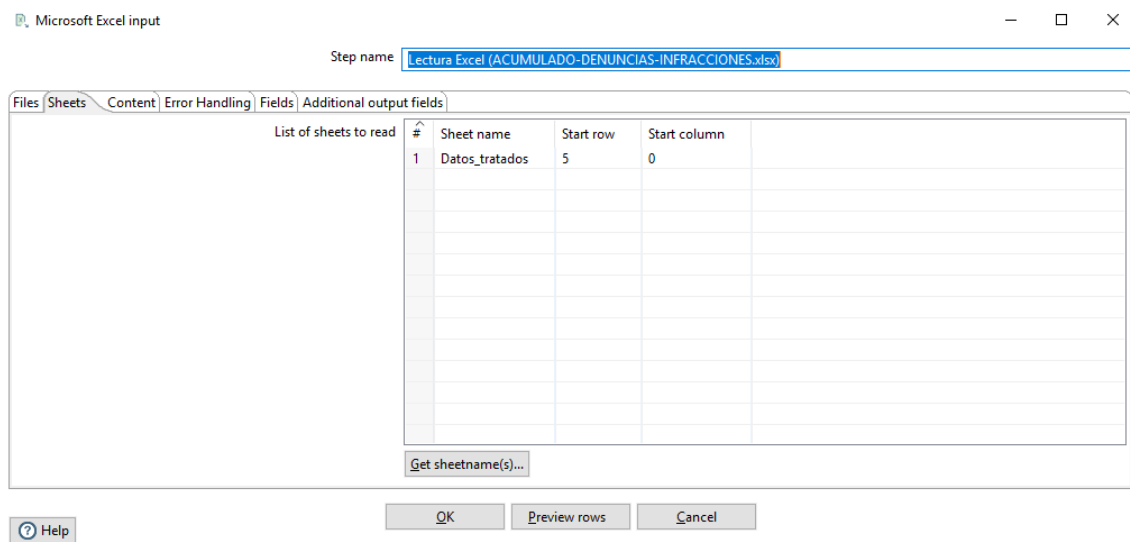
## Lectura del Excel

Lo primero de todo es leer el fichero Excel que se nos proporciona, y para ello usamos el componente “Microsoft Excel Input”, una vez hecho eso escribimos el nombre del paso, le indicamos el fichero que va a utilizar, y le indicamos que el formato del fichero Excel es XLSX:



*Ilustración 21 - Lectura IN\_DENUNCIAS\_INFRACCIONES.*

Una vez hecho eso, le indicamos qué hoja tiene que leer y desde qué fila y columna, en nuestro caso la hoja “Datos\_tratados” y la fila 5 columna 0:



*Ilustración 22 - Lectura IN\_DENUNCIAS\_INFRACCIONES.*

Posteriormente obtenemos los campos leídos en la pestaña “Field”:

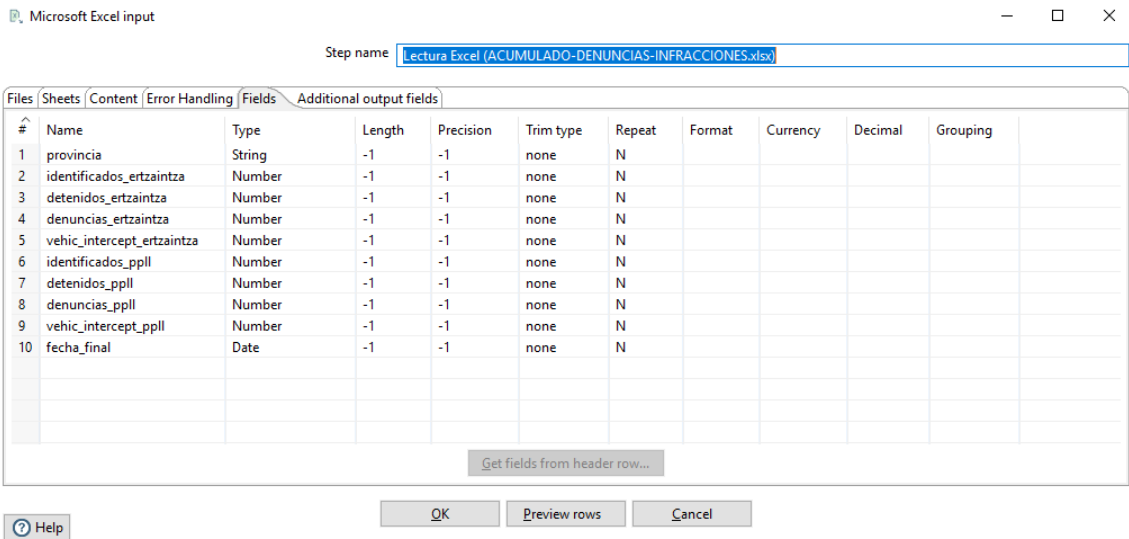


Ilustración 23 - Lectura IN\_DENUNCIAS\_INFRACCIONES.

**Mapeo**

Una vez leídos los datos vemos que las provincias están escritas en euskera, por lo que para homogeneizar los datos hemos decidido convertirlas al castellano. Por lo tanto, hacemos la traducción tal y como vemos en la siguiente captura:

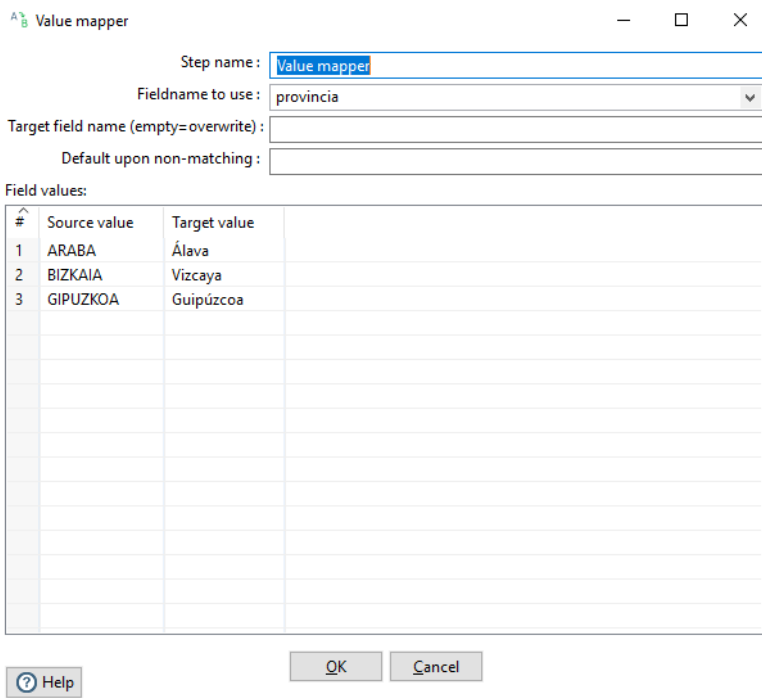


Ilustración 24 - Mapeo Valores IN\_DENUNCIAS\_INFRACCIONES.

## Normalización

Posteriormente hacemos una normalización de los campos que son de tipo “string”, ya que en éstos vamos a convertir los valores a mayúsculas y sin espacios, tal y como vemos en la siguiente ilustración:

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	Remove Special character
1	provincia		both	upper	none			N	None	none	none

Help OK Get fields Cancel

Ilustración 25 - Normalización Strings IN\_DENUNCIAS\_INFRACCIONES.

## Ordenación

Posteriormente, ordenamos todos los campos de forma ascendente:

Sort rows

Step name: Sort rows

Sort directory: %java.io.tmpdir% Browse...

TMP-file prefix: out

Sort size (rows in memory): 1000000

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	provincia	Y	N	N	0	N
2	identificados_ertzaintza	Y	N	N	0	N
3	detenidos_ertzaintza	Y	N	N	0	N
4	denuncias_ertzaintza	Y	N	N	0	N
5	vehic_intercept_ertzaintza	Y	N	N	0	N
6	identificados_ppll	Y	N	N	0	N
7	detenidos_ppll	Y	N	N	0	N
8	denuncias_ppll	Y	N	N	0	N
9	vehic_intercept_ppll	Y	N	N	0	N
10	fecha_final	Y	N	N	0	N

Help OK Cancel Get Fields

Ilustración 26 - Ordenación IN\_DENUNCIAS\_INFRACCIONES.

Guardado

Finalmente, introducimos todos los valores en la base de datos, es decir, en la tabla intermedia STG\_Denuncias\_Infracciones, indicamos que haga un truncate de la tabla y con la conexión definida guardamos los valores en la tabla correspondiente:

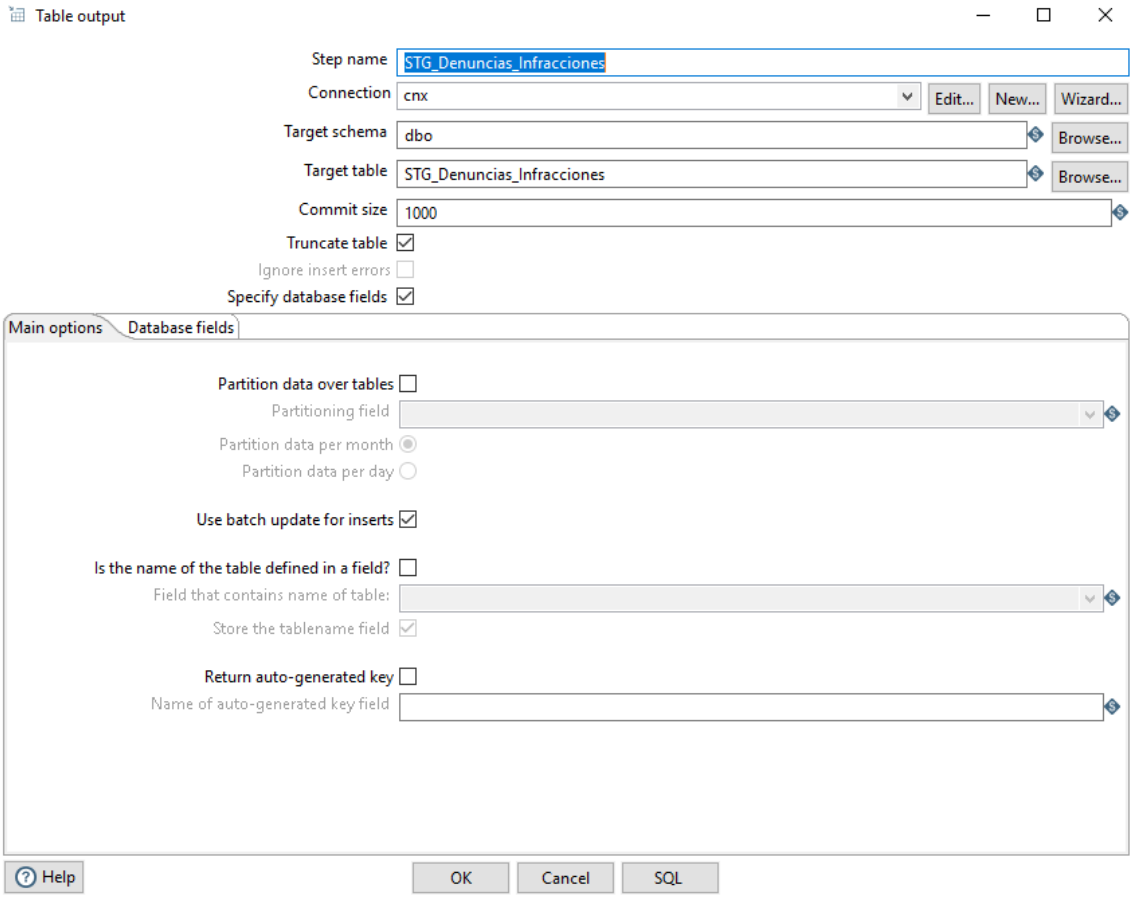


Ilustración 27 - Guardado IN\_DENUNCIAS\_INFRACCIONES.

Al ejecutar la anterior transformación obtenemos las siguiente métricas:

Execution Results

Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1 Lectura Excel (ACUMULADO-DENUNCIAS-INFRACCIONES.xlsx)	0	0	219	219	0	0	0	0	Finished	0.2s	894	-
2 Value mapper	0	219	219	0	0	0	0	0	Finished	0.3s	826	-
3 String operations	0	219	219	0	0	0	0	0	Finished	0.3s	782	-
4 Sort rows	0	219	219	0	0	0	0	0	Finished	0.3s	702	-
5 STG_Denuncias_Infracciones	0	219	219	0	219	0	0	0	Finished	0.3s	637	-

Ilustración 28 - Métricas IN\_DENUNCIAS\_INFRACCIONES.

Observamos que tenemos 219 registros leídos y en nuestra base de datos se han almacena también 219 registros, por lo que la información es correcta.

### 3.2.4. Transformación IN\_POBLACION

La segunda transformación que vamos a realizar se llama “IN\_POBLACION”, su objetivo es leer todos los datos del archivo “poblacion\_9687bsc.csv” y almacenarlos en la taba intermedia “STG\_Poblacion”.

En este caso no hemos hecho ninguna modificación al fichero original, por lo que la transformación nos queda de la siguiente forma:



Ilustración 29 - IN\_POBLACION.

#### Lectura CSV

Lo primero que debemos de hacer es cargar la información que se nos proporciona a partir del fichero CSV correspondiente. Por lo tanto, lo primero escribimos el nombre del paso, indicamos el fichero y el delimitador del CSV, en nuestro caso “;”:

CSV file input

Step name: Lectura CSV (poblacion\_9687bsc.csv)

Filename: \${DIR}\poblacion\_9687bsc.csv Browse...

Delimiter: ; Insert TAB

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional):

Running in parallel? ☐

New line possible in fields? ☐

Format: mixed

File encoding:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Edad Simple	String		5		\$	,	.	none
2	Provincias	String		25		\$	,	.	none
3	Sexo	String		11		\$	,	.	none
4	Periodo	String		18		\$	,	.	none
5	Total	Integer	#,###,###,##	9	3	\$	,	.	none

Ilustración 30 - Lectura IN\_POBLACION.

Cabe destacar que hemos tenido que modificar el tipo del campo “Total” ya que lo reconocía como decimal cuando realmente es un entero.

## Split

Luego separamos el campo “Provincias”, para así obtener tanto el código como la provincia correspondiente, para ello indicamos que el campo que queremos separar es “Provincias”, y luego en el grid establecemos los nuevos campos:

Split fields

Step name: Split fields

Field to split: Provincias

Delimiter:

Enclosure:

#	New field	ID	Remove ID?	Type	Length	Precision	Format	Group	Decimal	Currency	Nullif	Default	Trim typ
1	provincia_codigo		N	String	2						99		both
2	provincia_nombre		N	String	2						NA		both

Help OK Cancel

Ilustración 31 - Separación Campos IN\_POBLACION.

## Mapeo

Una vez hecho el paso anterior tenemos que mapear valores, esto se debe a que los nombres de las provincias no son del todo correctos (aparecen en gallego, euskera, catalán y valenciano). Además, al hacer la separación algunos nombres de provincias compuestas han desaparecido, es por ello que necesitamos de este paso para solventar los problemas:

Value mapper

Step name: Value mapper

Fieldname to use: provincia\_nombre

Target field name (empty=overwrite):

Default upon non-matching:

#	Source value	Target value	
1	Alicante/Alacant	Alicante	
2	Araba/Álava	Álava	
3	Balears,	Baleares	
4	Bizkaia	Vizcaya	
5	Castellón/Castelló	Castellón	
6	Coruña,	La Coruña	
7	Gipuzkoa	Guipúzcoa	
8	Girona	Gerona	
9	Lleida	Lérida	
10	Palmas,	Las Palmas	
11	Rioja,	La Rioja	
12	Santa	Santa Cruz de Tenerife	
13	Valencia/València	Valencia	
14	Ciudad	Ciudad Real	
15	Ourense	Orense	

Help OK Cancel

Ilustración 32 - Mapeo Valores IN\_POBLACION.



Normalización

Antes de almacenar los datos en la base de datos, vamos a normalizar los strings para que todos estén en mayúsculas y no tengan espacios ni al principio ni al final:

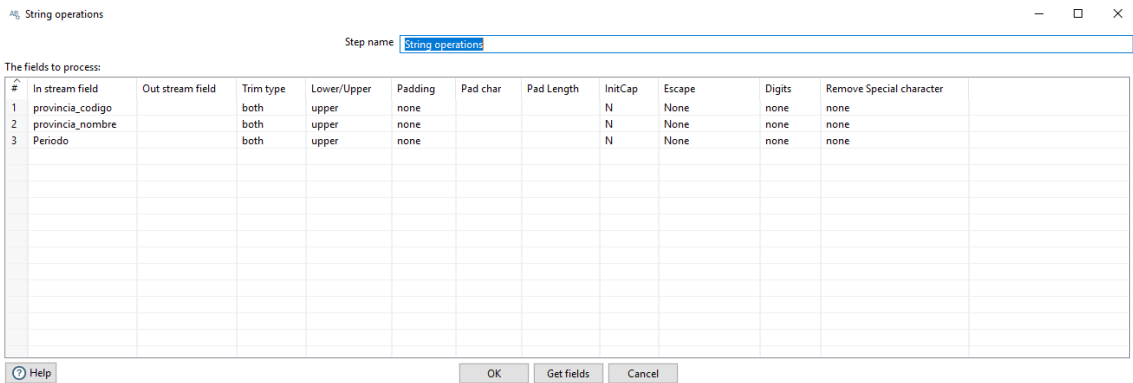


Ilustración 33 - Normalización Strings IN\_POBLACION.

Guardado

Finalmente, guardamos los datos en la tabla “STG\_Poblacion”, indicando que haga un truncate de la tabla y asociamos los campos:

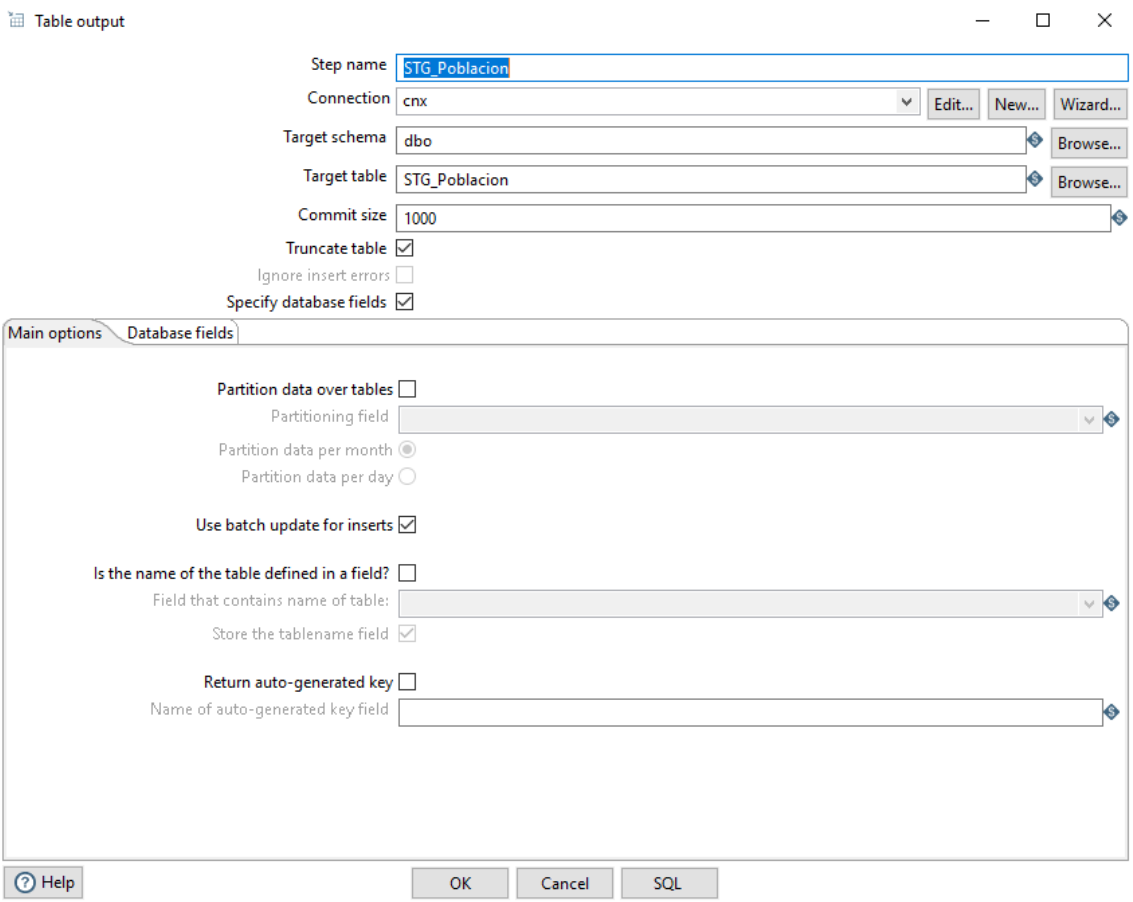


Ilustración 34 - Guardado IN\_POBLACION.

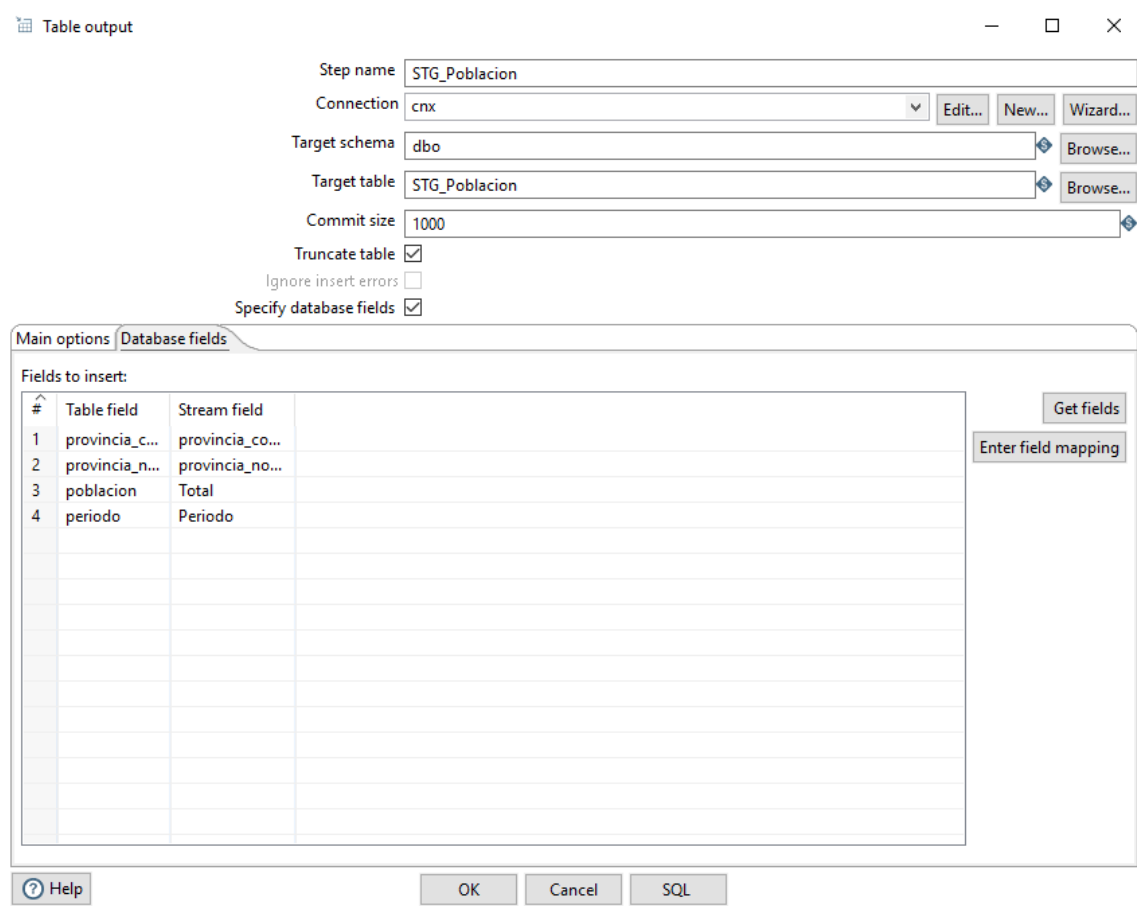


Ilustración 35 - Guardado IN\_POBLACION.

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

Execution Results

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Lectura CSV (poblacion_9687bsc.csv)	0	0	52	53	0	0	0	0	Finished	0.0s	1,325	-
2	Split fields	0	52	52	0	0	0	0	0	Finished	0.1s	426	-
3	Value mapper	0	52	52	0	0	0	0	0	Finished	0.1s	406	-
4	String operations	0	52	52	0	0	0	0	0	Finished	0.1s	397	-
5	STG_Poblacion	0	52	52	0	52	0	0	0	Finished	0.2s	292	-

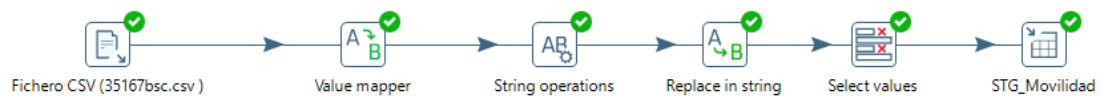
Ilustración 36 - Métricas IN\_POBLACION.

Observamos que tenemos 53 registros (52 registros + 1 cabecera) y se han almacenado 52 registros, por lo que la información es correcta.

### 3.2.5. Transformación IN\_MOVILIDAD

La tercera transformación que vamos a realizar se llama “IN\_MOVILIDAD”, su objetivo es leer todos los datos del archivo “35167bsc.csv” y guardarlos en la tabla intermedia “STG\_Movilidad”.

En este caso no hemos hecho ninguna modificación en el fichero CSV original, por lo que la transformación nos queda de la siguiente manera:



*Ilustración 37 - IN\_MOVILIDAD.*

Ahora vamos a explicar paso a paso lo que hemos hecho:

### Lectura CSV

Lo primero que tenemos que hacer es leer el fichero CSV que se nos proporciona, luego escribimos el nombre del paso, indicamos el fichero y el delimitador del CSV, en nuestro caso “,”:

CSV file input

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? ☐

Format:

File encoding:

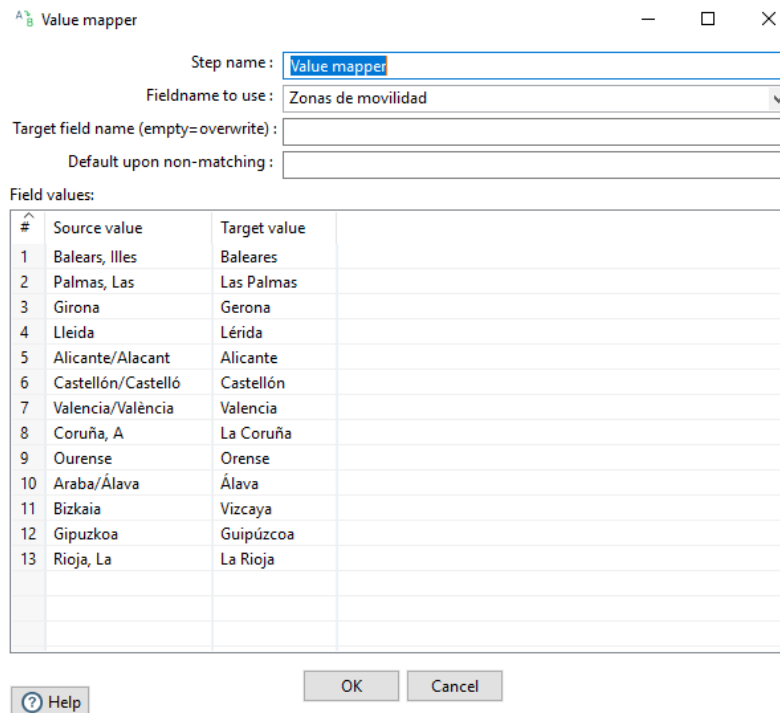
#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Zonas de movilidad	String		27		\$	,	.	both
2	Periodo	Timestamp	dd/MM/yyyy			\$	,	.	both
3	Total	String	#,##	15	0	\$	,	.	both

*Ilustración 38 - Lectura IN\_MOVILIDAD.*

Cabe destacar que el atributo “Total” hemos indicado que sea de tipo string, ya que si considerábamos que fuera numérico a la hora de introducirlo en la base de datos no guardaba los decimales.

## Mapeo

Una vez leídos todos los datos, tenemos que realizar un mapeo del campo “Zonas de movilidad”, esto se debe a que los nombres de las provincias vienen en (euskera, gallego, catalán, valenciano y balear), sin embargo para homogeneizar todas las provincias las traducimos al castellano.



Value mapper

Step name: Value mapper

Fieldname to use: Zonas de movilidad

Target field name (empty=overwrite):

Default upon non-matching:

Field values:

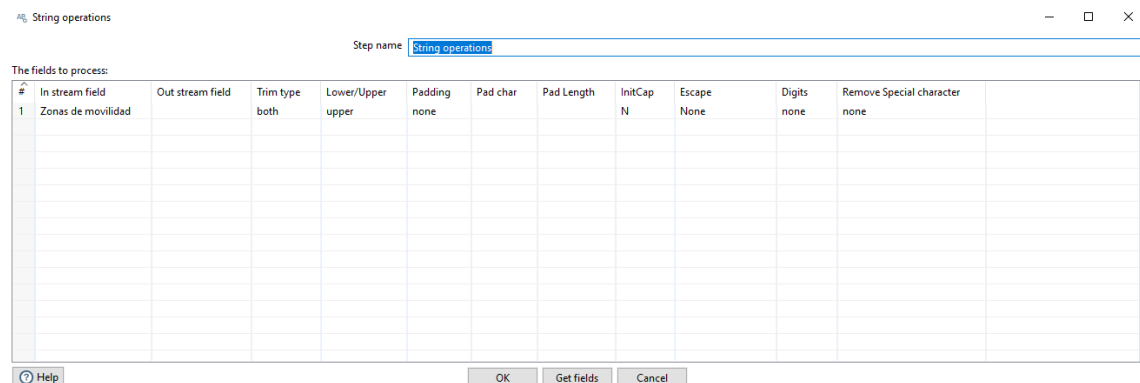
#	Source value	Target value
1	Balears, Illes	Baleares
2	Palmas, Las	Las Palmas
3	Girona	Gerona
4	Lleida	Lérida
5	Alicante/Alacant	Alicante
6	Castellón/Castelló	Castellón
7	Valencia/València	Valencia
8	Coruña, A	La Coruña
9	Ourense	Orense
10	Araba/Álava	Álava
11	Bizkaia	Vizcaya
12	Gipuzkoa	Guipúzcoa
13	Rioja, La	La Rioja

Help OK Cancel

*Ilustración 39 - Mapeo Valores IN\_MOVILIDAD.*

## Normalización

Una vez que tenemos ya los datos de forma correcta, normalizamos las provincias para que no haya un espacio al principio o al final de la cadena, y establecemos que todas las cadenas estén 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	Remove Special character
1	Zonas de movilidad		both	upper	none			N	None	none	none

Help OK Get fields Cancel

*Ilustración 40 - Normalización IN\_MOVILIDAD.*

## Replace

Posteriormente reemplazamos en el string de “Total” la coma por el punto, ya que de esta manera cuando luego introduzcamos el valor en la base de datos sí que no va a aparecer con decimales:

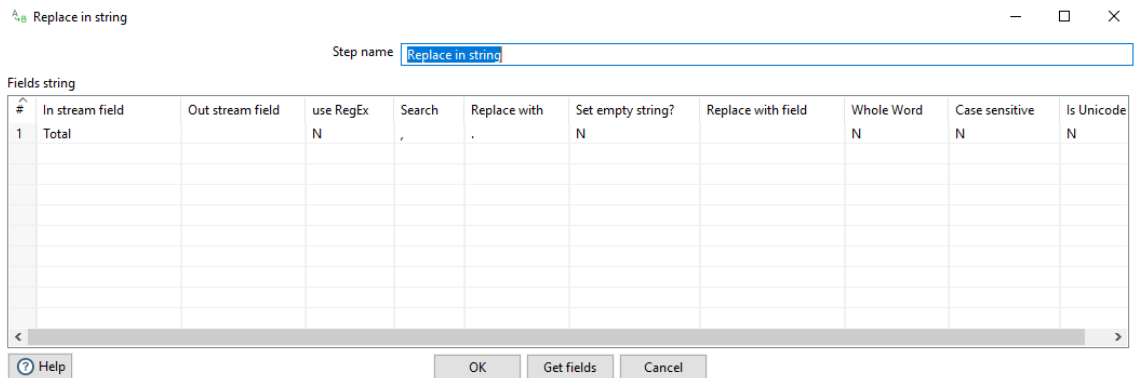


Ilustración 41 - Replace IN\_MOVILIDAD.

## Select Values

Cuando ya tenemos el string modificado, hay que convertirlo de decimal, ya que en la tabla de la base de datos el “Total” lo almacenamos como un número. Para ello, hacemos uso del componente “Select\_Values” y en “Meta-data” establecemos que cree un nuevo campo que sea de tipo numérico con el campo original “Total”, el resultado de esta operación lo guardamos en “totalSQL”:

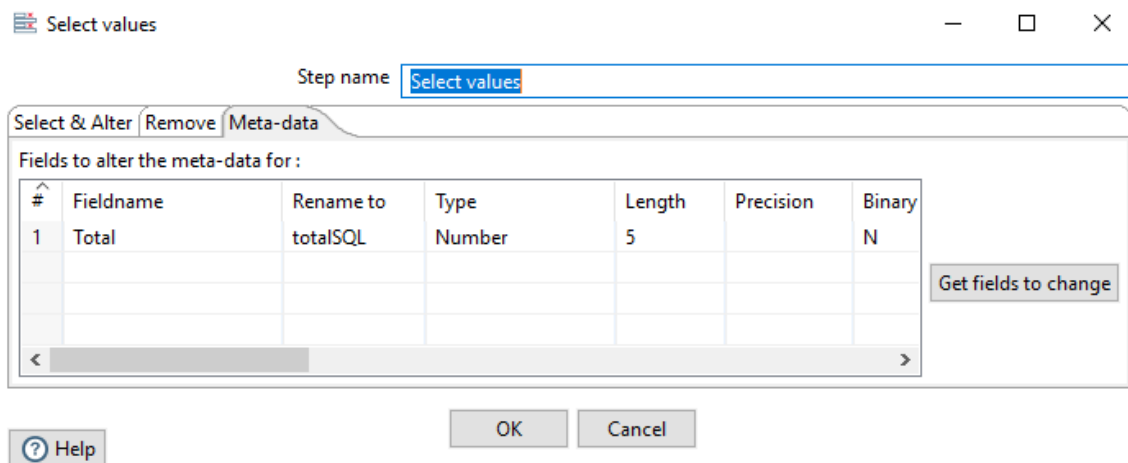


Ilustración 42 - Select Values IN\_MOVILIDAD.

## Guardado

Finalmente, guardamos todo el proceso realizado en la tabla “STG\_Movilidad”, indicando que hay un truncate de la tabla y asociamos los campos obtenidos con los de la tabla:

Table output

Step name

STG\_Movilidad

Connection

cnx

Edit...

New...

Wizard...

Target schema

dbo

Browse...

Target table

STG\_Movilidad

Browse...

Commit size

1000

Truncate table

☒

Ignore insert errors

☐

Specify database fields

☒

Main options

Database fields

Partition data over tables

☐

Partitioning field

Partition data per month

☒

Partition data per day

☐

Use batch update for inserts

☒

Is the name of the table defined in a field?

☐

Field that contains name of table:

Store the tablename field

☒

Return auto-generated key

☐

Name of auto-generated key field

Help

OK

Cancel

SQL

Ilustración 43 - Guardado IN\_MOVILIDAD.

Table output

Step name: **STG\_Movilidad**

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

Target schema: **dbo** [Browse...]

Target table: **STG\_Movilidad** [Browse...]

Commit size: **1000**

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options | Database fields

Fields to insert:

#	Table field	Stream field
1	zonas_mov...	Zonas de mo...
2	periodo	Periodo
3	total	totalSQL

[Get fields] [Enter field mapping]

[Help] [OK] [Cancel] [SQL]

*Ilustración 44 - Guardado IN\_MOVILIDAD.*

Para terminar con esta transformación obtenemos las métricas de su ejecución:

Execution Results

Logging

Execution History

Step Metrics

Performance Graph

Metrics

Preview data

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Fichero CSV (35167bsc.csv)	0	0	4732	4733	0	0	0	0	Finished	0.0s	189,320	-
2	Value mapper	0	4732	4732	0	0	0	0	0	Finished	0.1s	58,420	-
3	String operations	0	4732	4732	0	0	0	0	0	Finished	0.1s	48,286	-
4	Replace in string	0	4732	4732	0	0	0	0	0	Finished	0.1s	43,413	-
5	Select values	0	4732	4732	0	0	0	0	0	Finished	0.1s	33,560	-
6	STG_Movilidad	0	4732	4732	0	4732	0	0	0	Finished	0.3s	15,933	-

*Ilustración 45 - Métricas IN\_MOVILIDAD.*

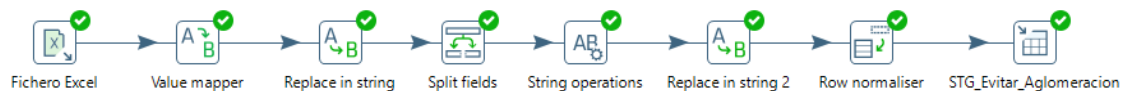
Como podemos observar, leemos 4733 registros (4732 observaciones + 1 cabecera) y almacenamos 4732, por lo que la información es correcta.

### 3.2.6. Transformación IN\_AGLOMERACION

La cuarta transformación que vamos a realizar se llama "IN\_AGLOMERACION", su objetivo es leer todo los datos del Excel "statistic\_id1104235\_covid-19\_-poblacion-que-evitaba-las-aglomeraciones-segun-edad-en-espana-2020.xlsx" y guardarlos en la tabla intermedia "STG\_AGLOMERACION".

En este caso sí hemos hecho una modificación en el fichero Excel, no sabemos por qué motivo determinadas provincias tenían un espacio o carácter especial que no era visible y al hacer la lectura, independientemente de si hacíamos un “trim” o usábamos un “string operations” no eliminaba ese “espacio/carácter especial”, es por ello que hemos eliminado de forma manual dicho espacio en el campo “provincia” de los registros afectados.

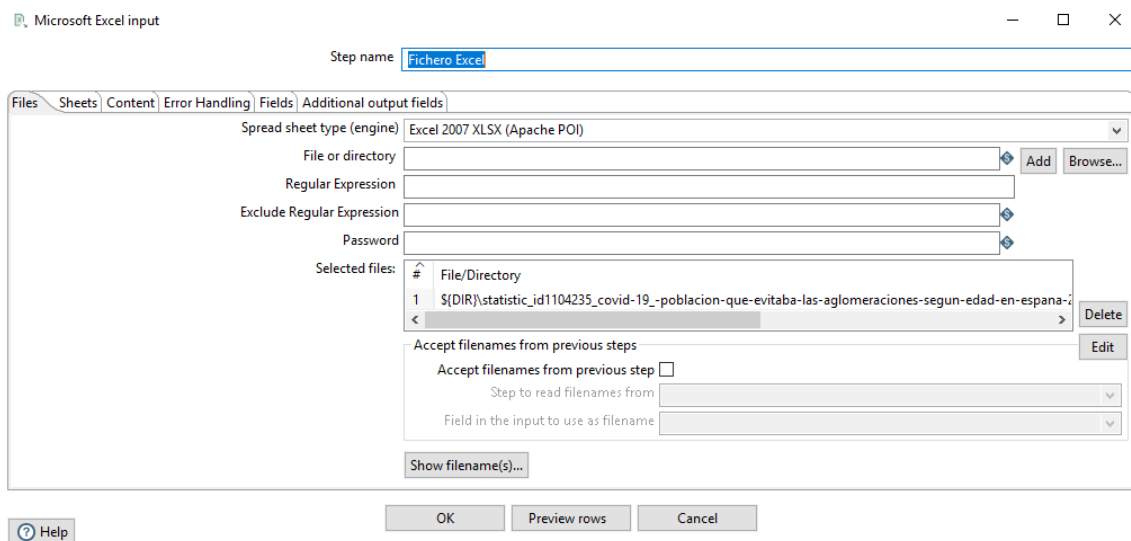
La transformación nos queda de la siguiente manera:



*Ilustración 46 - IN\_AGLOMERACION.*

### Lectura

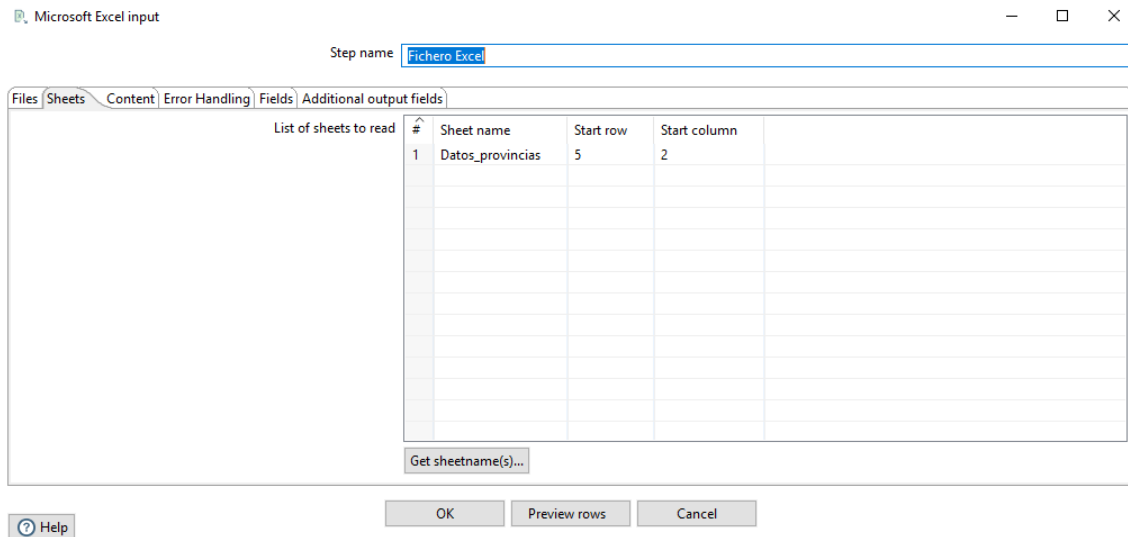
Lo primero que tenemos que hacer es leer el fichero Excel que se nos ha proporcionado, para ello escribimos el nombre del paso, indicamos el fichero y su formato correspondiente a XLSX:



*Ilustración 47 - Lectura IN\_AGLOMERACION.*

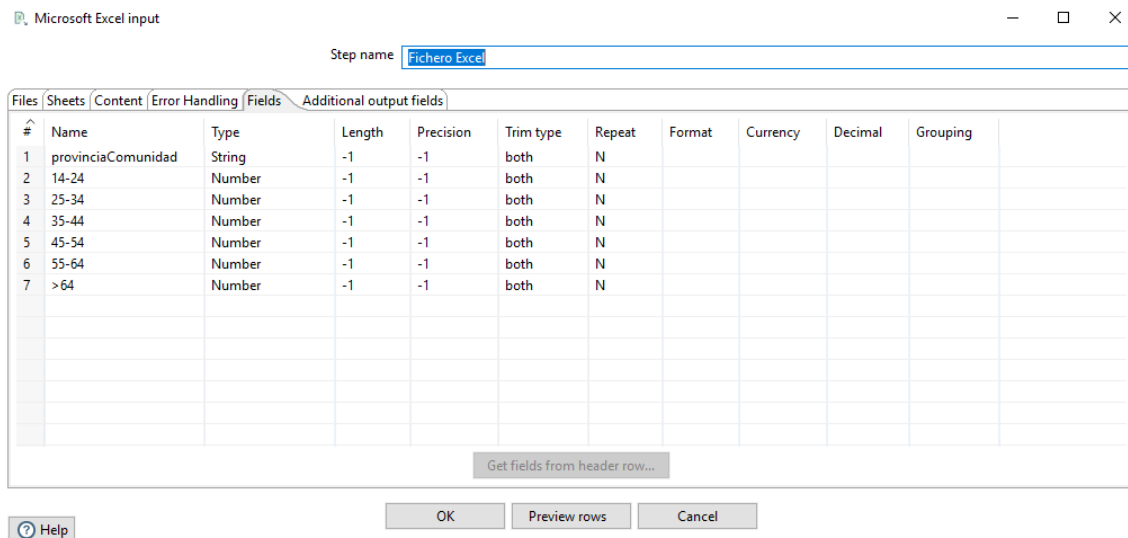
Una vez hecho eso, le indicamos qué hoja tiene que leer y desde qué fila y columna, en nuestro caso la hoja “Datos\_provincias” y la fila 5 columna 2:





*Ilustración 48 - Lectura IN\_AGLOMERACION.*

Posteriormente obtenemos los campos leídos en la pestaña “Fields”, los nombres de los campos han sido definidos de forma manual:



*Ilustración 49 - Lectura IN\_AGLOMERACIONES.*

## Mapeo

Al igual que ha sucedido con transformaciones anteriores, muchas provincias vienen también con su nombre en catalán/gallego/euskera/valenciano... Es por ello que hemos decido mantener el nombre en castellano, mapeando así los valores de las provincias que venían en otro idioma:

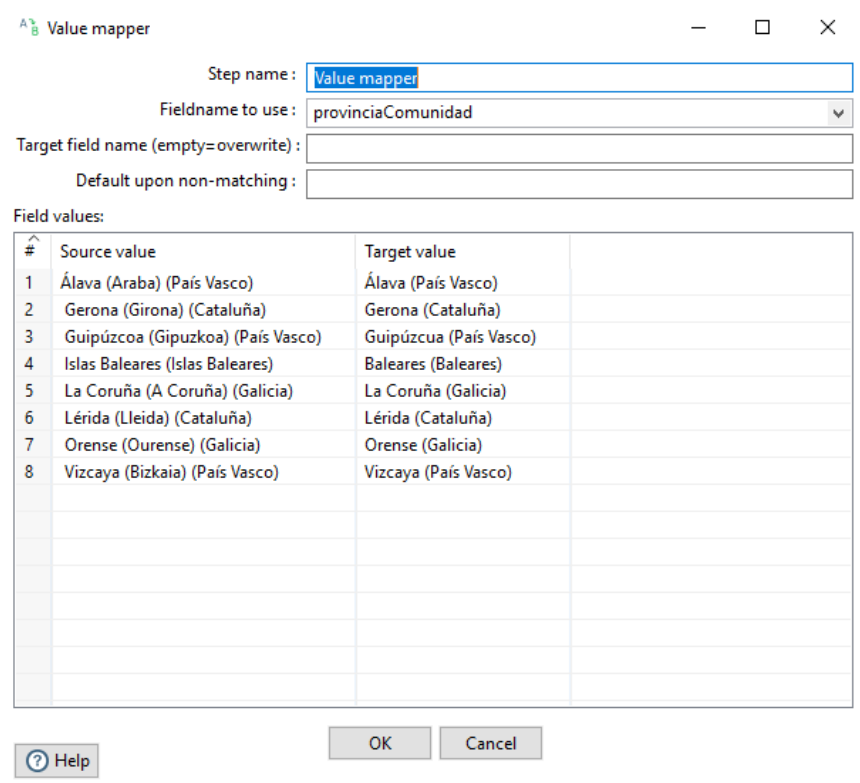


Ilustración 50 - Mapeo Valores IN\_AGLOMERACION.

Replace

Posteriormente tenemos que hacer un replace del símbolo “)” por nada, de esta forma luego podemos dividir el campo en dos, para así obtener el nombre de la provincia y su comunidad autónoma:

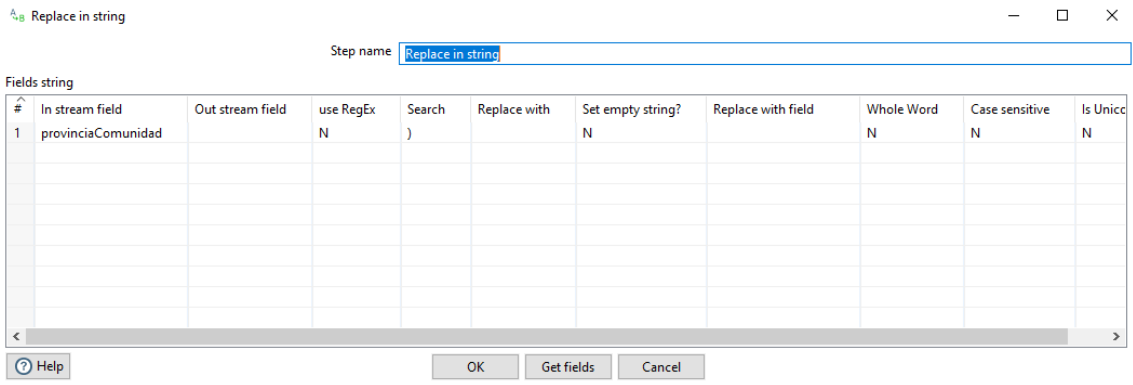


Ilustración 51 - Replace IN\_AGLOMERACION.

Split

Una vez que ya tenemos la información que queremos, la podemos separar estableciendo como separado “[espacio](“, de esta forma creamos dos nuevos campos: uno para la provincia y otro para la comunidad.

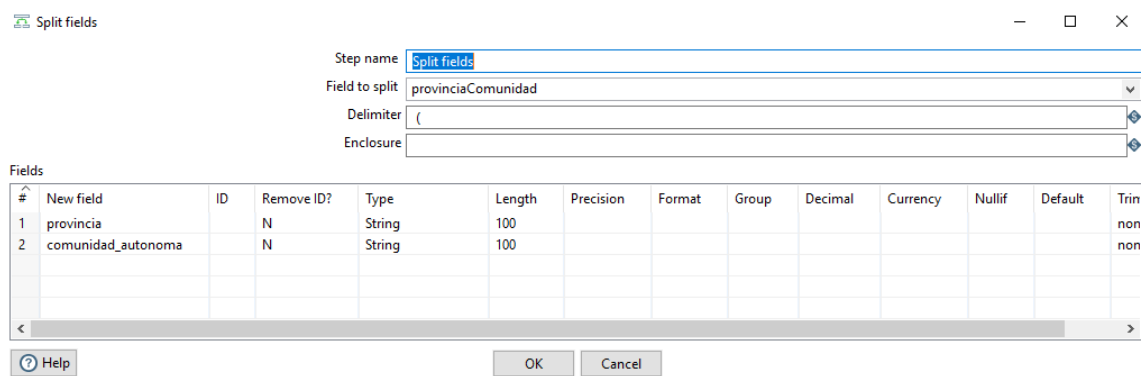


Ilustración 52 - Split IN\_AGLOMERACION.

Normalización

Cuando ya tenemos la información separada, podemos hacer uso de un “string operations” para normalizar todos los strings, es decir, establecer mayúsculas y eliminar espacios:

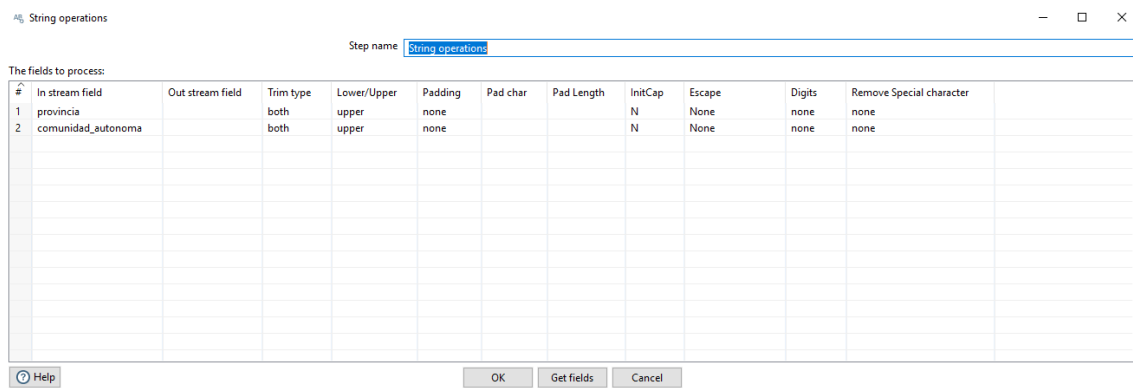


Ilustración 53 - Normalización Strings IN\_AGLOMERACION.

Replace

Analizando los datos hemos visto que todas las comunidades y provincias cumplían las reglas ortográficas, sin embargo, la comunidad Aragón la escribían sin tilde. Es por ello que para mantener la misma lógica en todas las transformaciones hemos corregido dicho problema:

Replace in string

Step name: Replace in string 2

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is
1	comunidad_autonoma		N	ARAGON	ARAGÓN	N		N	N	N

Buttons: Help, OK, Get fields, Cancel

Ilustración 54 - Replace IN\_AGLOMERACION.

### Normalización filas

Posteriormente, hemos tenido que normalizar filas para que las columnas respectivas al grupo de edad fueran filas y no columnas. Para ello establecemos el nuevo campo que vamos a crear y los valores que va a tener dicho campo:

Row normaliser

Step name: Row normaliser

Type field: grupo\_edad

#	Fieldname	Type	new field	
1	14-24	14-24	porc_poblacion	
2	25-34	25-34	porc_poblacion	
3	35-44	35-44	porc_poblacion	
4	45-54	45-54	porc_poblacion	
5	55-64	55-64	porc_poblacion	
6	>64	>64	porc_poblacion	

Buttons: Help, OK, Cancel, Get Fields

Ilustración 55 - Normalización Filas IN\_AGLOMERACION.

### Guardado

Finalmente, una vez que tenemos ya todos los datos normalizados podemos proceder al guardado de los mismo en la tabla intermedia "STG\_AGLOMERACION". Tenemos que marcar el truncate table y asociar los campos:

Table output

Step name: **STG\_Evitar\_Aglomeracion**

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

Target schema: **dbo** [Browse...]

Target table: **STG\_Evitar\_Aglomeracion** [Browse...]

Commit size: **1000**

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options | Database fields

Partition data over tables: ☐  
Partitioning field: [ ]

Partition data per month: ☒  
Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field? ☐  
Field that contains name of table: [ ]  
Store the tablename field: ☒

Return auto-generated key: ☐  
Name of auto-generated key field: [ ]

[?] Help [OK] [Cancel] [SQL]

*Ilustración 56 - Guardado IN\_AGLOMERACIONES.*

Table output

Step name: **STG\_Evitar\_Aglomeracion**

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

Target schema: **dbo** [Browse...]

Target table: **STG\_Evitar\_Aglomeracion** [Browse...]

Commit size: **1000**

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options | Database fields

Fields to insert:

#	Table field	Stream field
1	provincia	provincia
2	comunidad...	comunidad...
3	grupo_edad	grupo_edad
4	porc_pobla...	porc_poblaci...

[Get fields] [Enter field mapping]

[?] Help [OK] [Cancel] [SQL]

*Ilustración 57 - Guardado IN\_AGLOMERACIONES.*

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

**Execution Results**

Logging | Execution History | Step Metrics | Performance Graph | Metrics | Preview data

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Fichero Excel	0	0	50	50	0	0	0	0	Finished	0.5s	102	-
2	Value mapper	0	50	50	0	0	0	0	0	Finished	0.5s	101	-
3	Replace in string	0	50	50	0	0	0	0	0	Finished	0.5s	100	-
4	Split fields	0	50	50	0	0	0	0	0	Finished	0.5s	99	-
5	String operations	0	50	50	0	0	0	0	0	Finished	0.5s	98	-
6	Replace in string 2	0	50	50	0	0	0	0	0	Finished	0.5s	96	-
7	Row normaliser	0	50	300	0	0	0	0	0	Finished	0.5s	574	-
8	STG_Evitar_Aglomeracion	0	300	300	0	300	0	0	0	Finished	0.6s	540	-

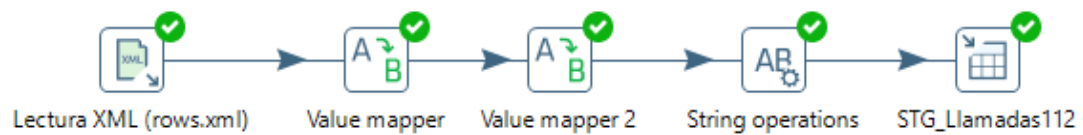
*Ilustración 58 - Métricas IN\_AGLOMERACION.*

Como podemos observar leemos 50 registros y almacenamos 300, esto se debe a la normalización de las filas para el atributo "grupo\_edad".

### 3.2.7. Transformación IN\_LLAMADAS112

La penúltima transformación respecto al bloque IN es "IN\_LLAMADAS112", ésta se encarga de hacer la lectura del archivo "rows.xml" el cual contiene todas las llamadas, y las vamos a guardar en la tabla intermedia "STG\_Llamadas112".

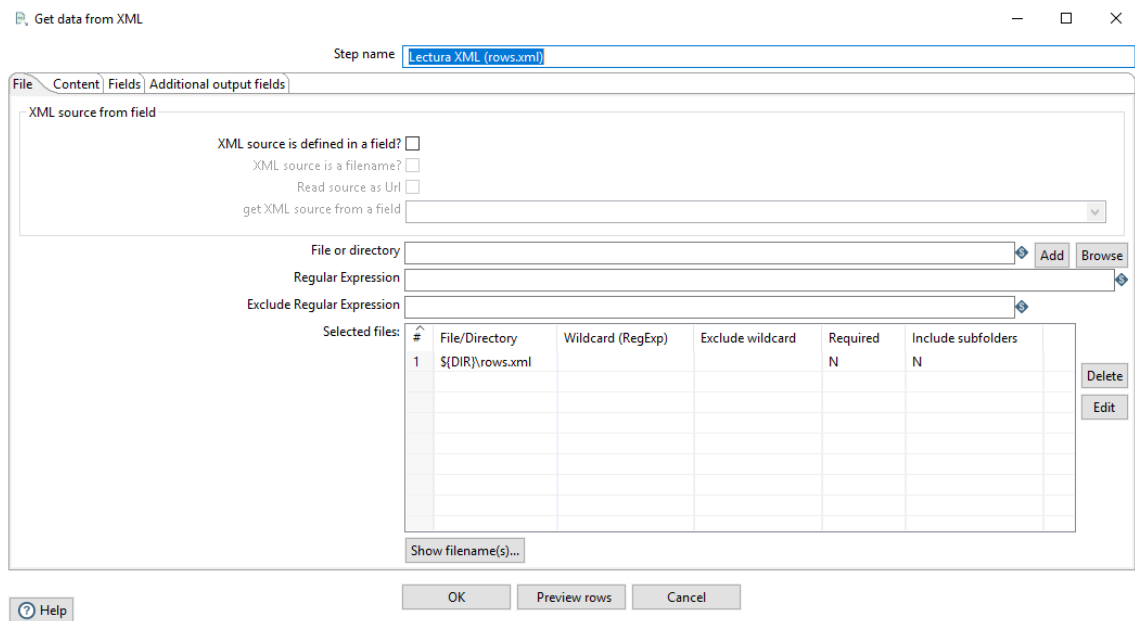
En este caso no hemos hecho ninguna modificación en el fichero XML original, por lo que la transformación nos queda de la siguiente forma:



*Ilustración 59 - IN\_LLAMADAS112.*

## Lectura XML

Lo primero que tenemos que hacer es leer la información que se nos proporciona en el fichero XML. Para ello escribimos el nombre del paso e indicamos el fichero:



*Ilustración 60 - Lectura IN\_LLAMADAS112.*

Luego nos dirigimos a la pestaña "Content" y definimos desde qué loop tiene que empezar a leer nuestro fichero XML:

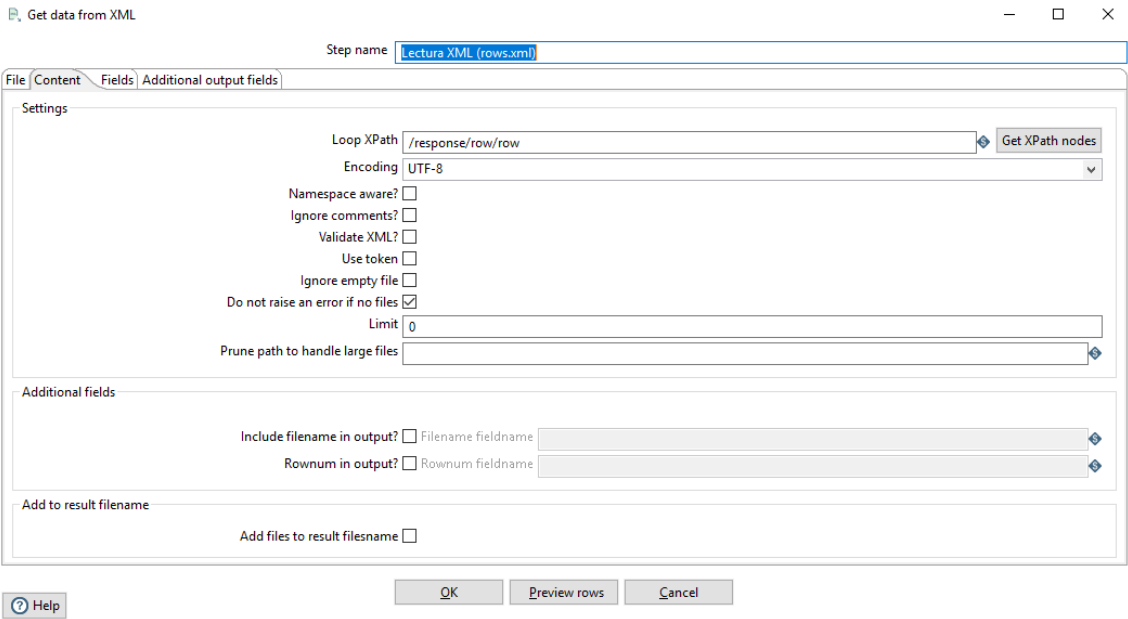


Ilustración 61 - Lectura IN\_LLAMADAS112.

Finalmente, obtenemos los campos y los definimos nosotros de forma manual:

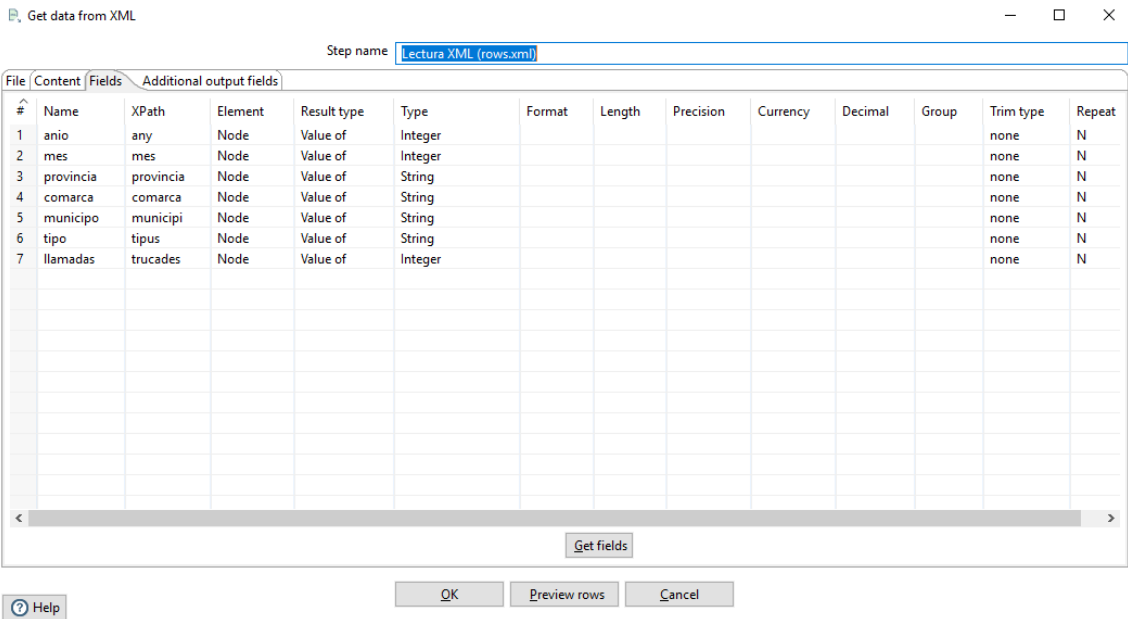


Ilustración 62 - Lectura IN\_LLAMADAS112.

**Mapeo**

Una vez leído los registros tenemos que hacer un cambio de valor de algunas provincias, ya que éstas aparecen en catalán y vamos a mantener en la base de datos solamente la traducción al castellano:





Value mapper

Step name :

Value mapper 2

Fieldname to use :

tipo

Target field name (empty=overwrite) :

Default upon non-matching :

Field values:

#	Source value	Target value
1	Medi ambient	MEDIO AMBIENTE
2	Assistència sanitària	ASISTENCIA SANITARIA
3	Seguretat	SEGURIDAD
4	Civisme	CIVISMO
5	Altres incidències	OTRAS INCIDENCIAS
6	Incendi	INCENDIO
7	Fuita (aigua, gas, altres)	FUGA
8	Trànsit	TRÁFICO
9	Accident	ACCIDENTE
10	Meteorologia	METEOROLOGÍA

Help

OK

Cancel

Ilustración 64 - Mapeo Valores IN\_LLAMADAS112.

Normalización

Antes de introducir todos los datos a la base de datos, tenemos que normalizar las cadenas de valores, es decir, establecer los campos string a mayúscula y sin espacios al comienzo ni al final:

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	Remove Special character
1	provincia		both	upper	none			N	None	none	none
2	comarca		both	upper	none			N	None	none	none
3	municipio		both	upper	none			N	None	none	none
4	tipo		both	upper	none			N	None	none	none

Help

OK

Get fields

Cancel

Ilustración 65 - Normalización IN\_LLAMADAS112.

## Guardado

Finalmente, guardamos los datos en la tabla “STG\_Llamadas112”, indicando que haga un truncate de la tabla y asociamos los campos:

Table output

Step name: STG\_Llamadas112

Connection: cnx [Edit... New... Wizard...]

Target schema: dbo [Browse...]

Target table: STG\_Llamadas112 [Browse...]

Commit size: 1000

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

Partition data over tables: ☐

Partitioning field: [ ]

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

Field that contains name of table: [ ]

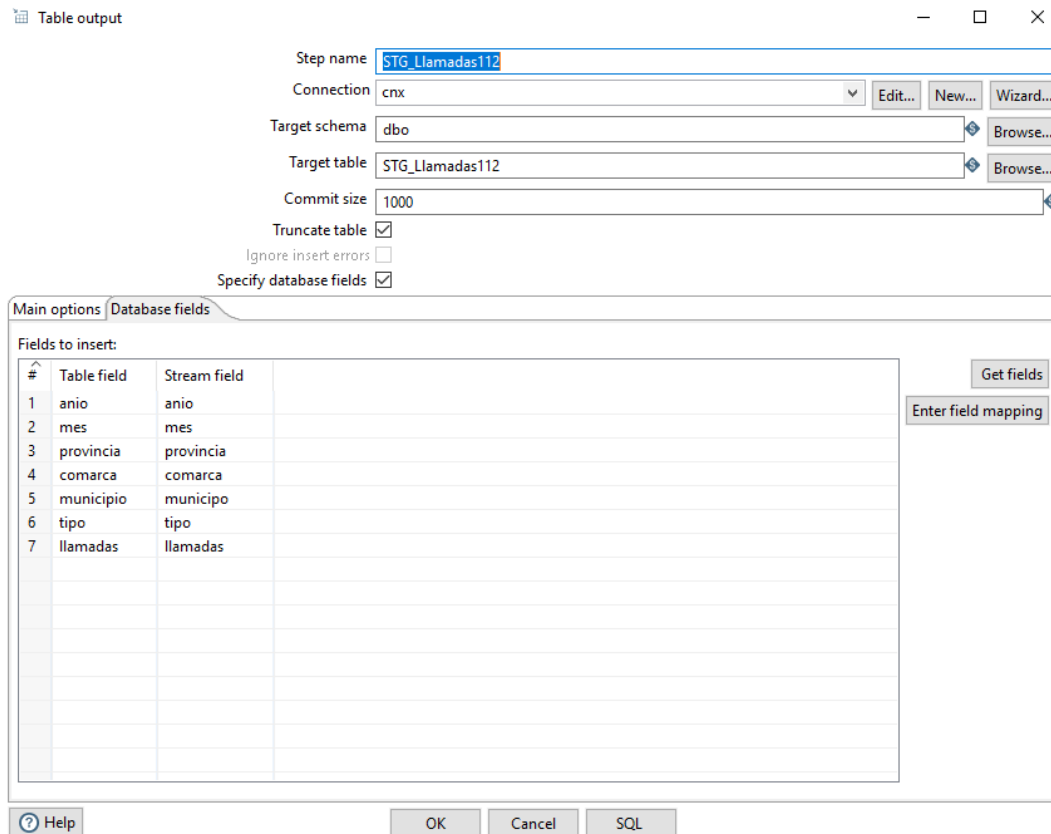
Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field: [ ]

Help OK Cancel SQL

Ilustración 66 - Guardado IN\_LLAMADAS112.



*Ilustración 67 - Guardado IN\_LLAMADAS112.*

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

**Execution Results**

Logging Execution History Step Metrics Performance Graph Metrics Preview data

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Lectura XML (rows.xml)	0	0	340307	340307	0	0	0	0	Finished	32mn 44s	173	-
2	Value mapper	0	340307	340307	0	0	0	0	0	Finished	32mn 44s	173	-
3	Value mapper 2	0	340307	340307	0	0	0	0	0	Finished	32mn 45s	173	-
4	String operations	0	340307	340307	0	0	0	0	0	Finished	32mn 45s	173	-
5	STG_Llamadas112	0	340307	340307	0	340307	0	0	0	Finished	32mn 45s	173	-

*Ilustración 68 - Métricas IN\_LLAMADAS112.*

Observamos que tenemos 340307 registros leídos y almacenamos el mismo número de registros, por lo que la información es correcta.

### 3.2.8. Transformación IN\_FECHAS

La última transformación que vamos a realizar respecto a este bloque es “IN\_FECHAS”, su objetivo es leer todas las fechas que hay en todos los ficheros fuente y almacenarlas en una tabla intermedia llamada “STG\_Fechas”.

La transformación nos ha quedado de la siguiente forma:

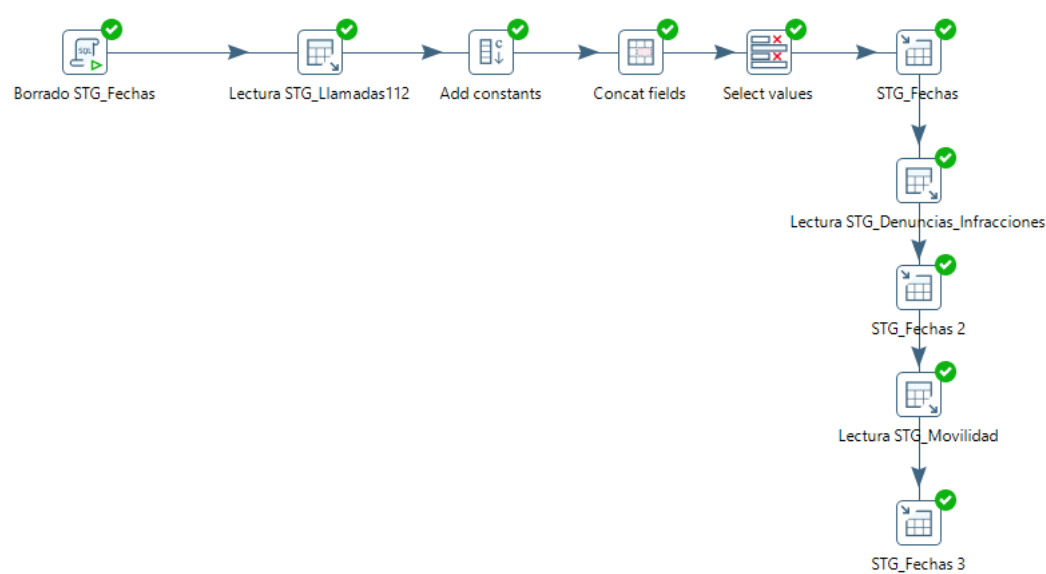


Ilustración 69 - IN\_FECHAS.

**Borrado**

Lo primero de todo es hacer un borrado de la tabla, ya que al no obtener la información directamente de los ficheros puede darse el caso de que ya tengamos información en dicha tabla, por lo tanto borramos todos los registros de forma manual a partir de una sentencia SQL:

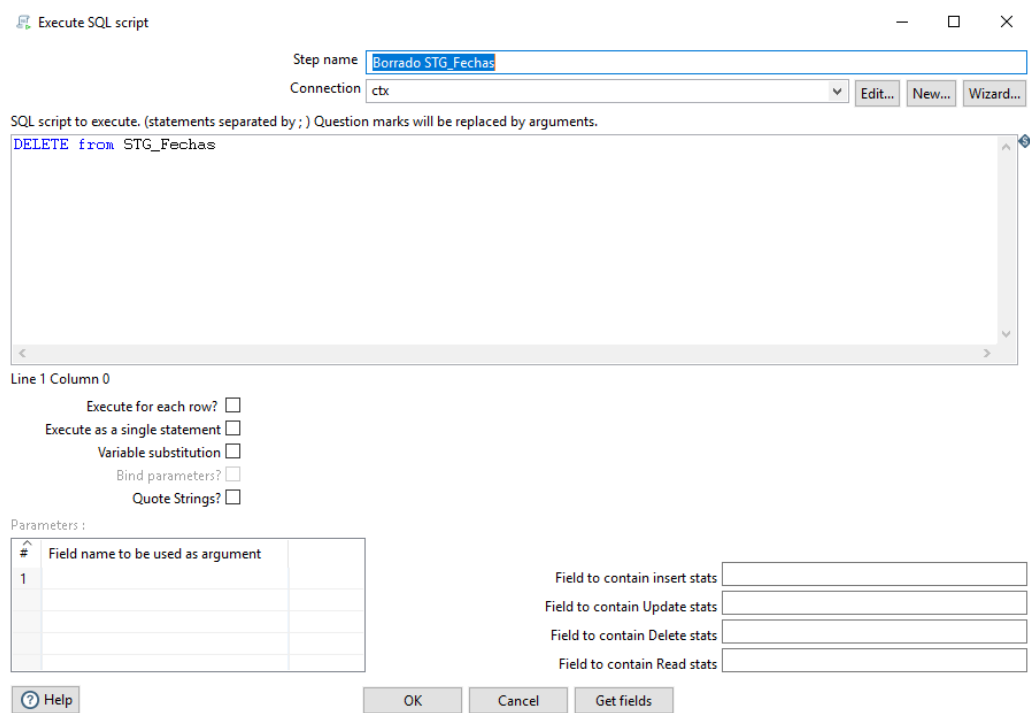


Ilustración 70 - Borrado IN\_FECHAS.

# Lectura

Una vez eliminados todos los registros leemos todas las fechas que se encuentran en la tabla intermedia “STG\_Llamadas112”, para ello cargamos la información del mes y año tal y como se muestra en la siguiente ilustración:

Table input

Step name:

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

SQL

```
SELECT anio, mes
FROM STG_Llamadas112

GROUP BY anio, mes
ORDER BY anio, mes
```

Get SQL select statement...

Line 1 Column 0

Store column info in step meta data ☐

Enable lazy conversion ☐

Replace variables in script? ☐

Insert data from step

Execute for each row? ☐

Limit size:

Help OK Preview Cancel

*Ilustración 71 - Lectura IN\_FECHAS.*

## Añadimos el día

Una característica de la información de “STG\_Llamadas112” es que sí que se nos proporciona el año y mes pero no el día, es por ello que creamos un nuevo campo para el día cuyo valor va a ser siempre 1:

[illegible]

*Ilustración 72 - Añadimos Constante IN FECHAS.*

## Concatenación

Un aspecto a tener en cuenta es que respecto a “STG\_Llamadas112” no tenemos una fecha como tal, sino que tenemos tres campos de tipo entero que nos indican el año, mes y día. Por lo tanto, lo primero que debemos de hacer es concatenar estos campos en un string:

Step name: Concat fields

Target Field Name: fecha\_llamadas112

Length of Target Field: 0

Separator: /

Enclosure: "

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	anio	String		4	0				both	
2	mes	String		2	0				both	
3	dia	String		2	0				both	

Buttons: Get Fields, Minimal width, OK, Cancel, Help

Ilustración 73 - Concatenación IN\_FECHAS.

## Conversión

Una vez que tenemos el string con el formato de la fecha, tenemos que convertir dicho campo a tipo date, para así poder almacenarlo en la base de datos:

Step name: Select values

Fields to alter the meta-data for:

#	Fieldname	Rename to	Type	Length	Precision	Binary t
1	fecha_llamadas112		Date			N

Buttons: OK, Cancel, Help

Ilustración 74 - Conversión IN\_FECHAS.

## Guardado

Finalmente, guardamos todas las fechas de la tabla “STG\_Llamadas112” en la tabla intermedia “STG\_Fechas”, para ello asociamos el campo de la transformación con el de la tabla de la base de datos:

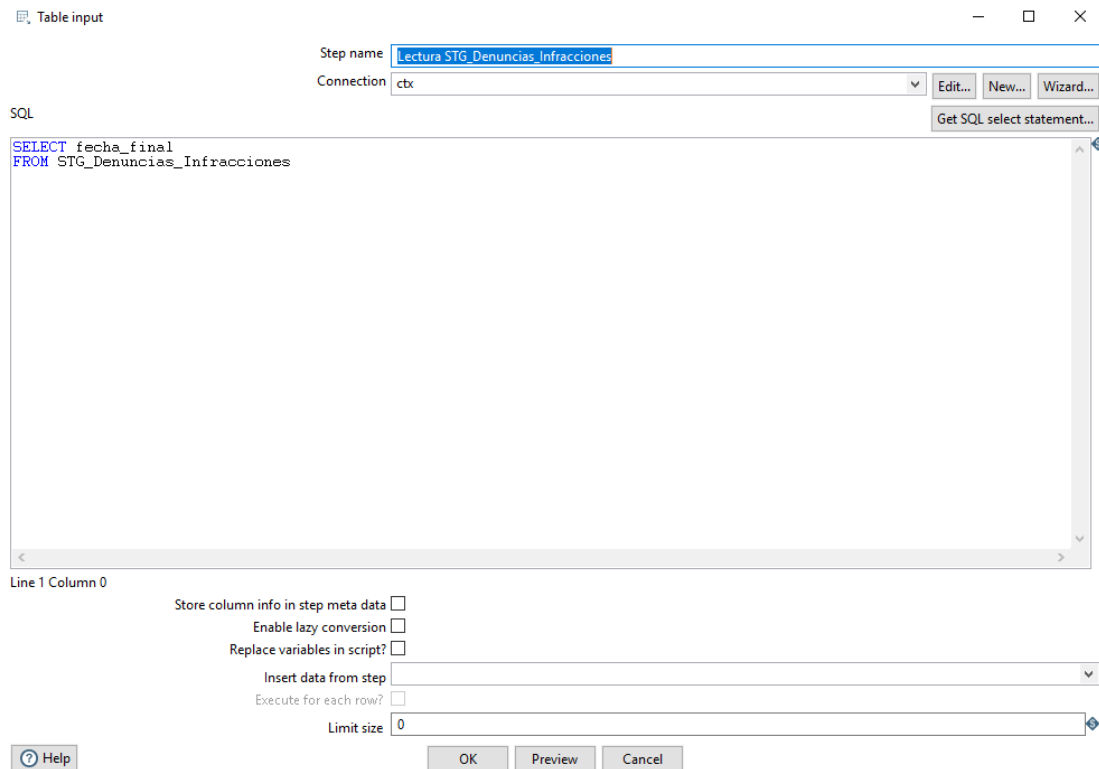
[illegible]

*Ilustración 75 - Guardado IN\_FECHAS.*

## Lectura

Al igual que hicimos con las llamadas al 112, tenemos que leer todas las fechas que hay en la tabla “STG\_Denuncias\_Infracciones”, para ello cargamos los datos a partir de la siguiente secuencia SQL:





*Ilustración 76 - Lectura IN\_FECHAS.*

## Guardado

Como en este caso la tabla “STG\_Denuncias\_Infracciones” ya contiene todas las fechas de forma correcta las podemos guardar directamente en la base de datos, para ello asociamos los campos:

[illegible]

*Ilustración 77 - Guardado IN\_FECHAS.*

## Lectura

Por último, tenemos las fechas que se encuentran en “STG\_Movilidad”, éstas las tenemos que leer y lo hacemos al igual que en los casos anterior con una sentencia SQL:

Table input

Step name:

Connection:

SQL:

Get SQL select statement...

Line 1 Column 0

☐ Store column info in step meta data

☐ Enable lazy conversion

☐ Replace variables in script?

Insert data from step

☐ Execute for each row?

Limit size

*Ilustración 78 - Lectura IN\_FECHAS.*

## Guardado

Al igual que en “STG\_Denuncias\_Infracciones” en “STG\_Movilidad” no necesitamos realizar ninguna transformación, ya que todos los datos están de forma correcta. Debido a esto podemos almacenarlos directamente en la base de datos, y para ello asociamos los campos:

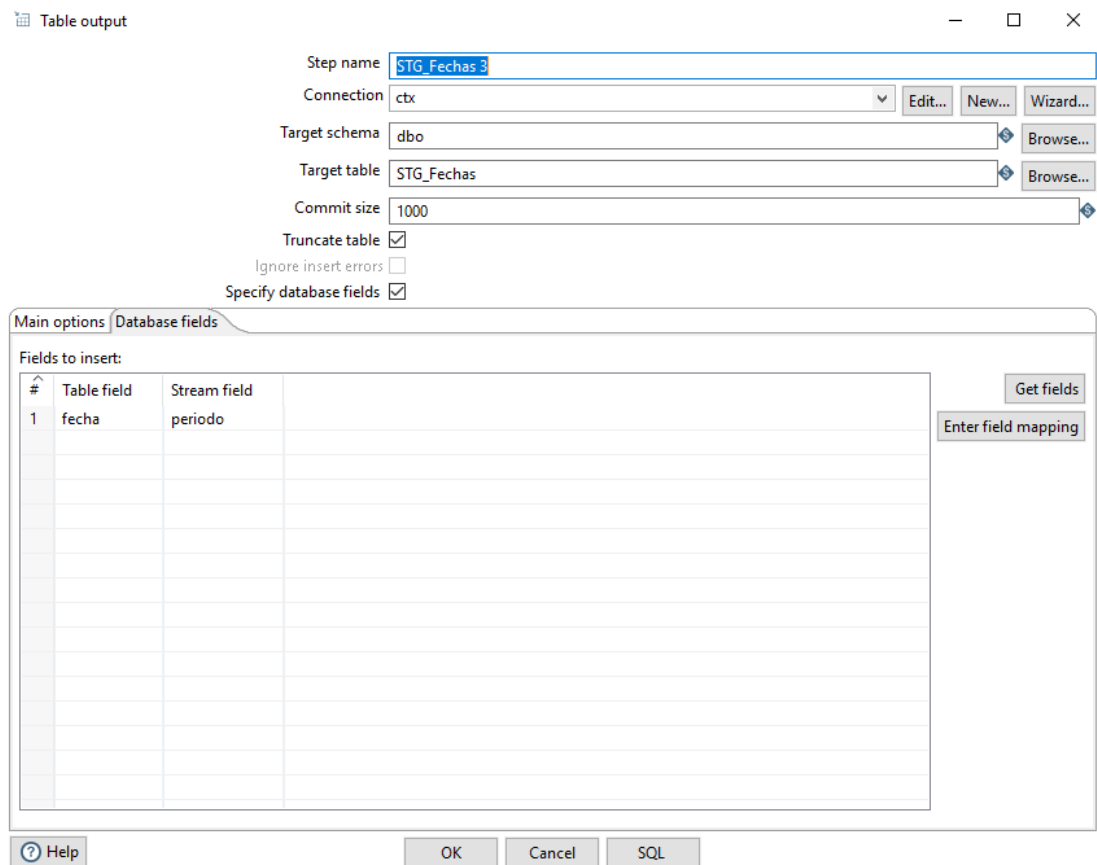


Ilustración 79 - Guardado IN\_FECHAS.

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

Execution Results													
Logging Execution History Step Metrics Performance Graph Metrics Preview data													
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado STG_Fechas	0	0	1	0	0	0	0	0	Finished	0.0s	-	-
2	Lectura STG_Llamadas112	0	0	82	82	0	0	0	0	Finished	0.1s	651	-
3	Add constants	0	82	82	0	0	0	0	0	Finished	0.1s	550	-
4	Concat fields	0	82	82	0	0	0	0	0	Finished	0.2s	366	-
5	Select values	0	82	82	0	0	0	0	0	Finished	0.2s	332	-
6	STG_Fechas	0	82	82	0	82	0	0	0	Finished	0.3s	293	-
7	Lectura STG_Denuncias_Infracciones	0	0	219	219	0	0	0	0	Finished	0.0s	16,846	-
8	STG_Fechas 2	0	219	219	0	219	0	0	0	Finished	0.2s	978	-
9	Lectura STG_Movilidad	0	0	4732	4732	0	0	0	0	Finished	0.0s	100,681	-
10	STG_Fechas 3	0	4732	4732	0	4732	0	0	0	Finished	0.4s	13,218	-

Ilustración 80 - Métricas IN\_FECHAS.

De la anterior ejecución vemos que en “STG\_Llamadas112” tenemos 82 fechas, en “STG\_Denuncias\_Infracciones” hay 219 fechas y en “STG\_Movilidad” 4732. Estas fechas no significan que sean únicas, de hecho todo lo contrario, como veremos más adelante solo 170 fechas son diferentes.

### 3.3. Bloque TR Dimensiones

Una vez que hemos almacenado toda la información en la base de datos gracias a las tablas intermedias, ahora vamos a hacer uso de estos datos para crear las diferentes dimensiones de nuestro modelo.

#### 3.3.1. Transformación TR\_DIM\_GRUPO\_EDAD

La primera transformación que vamos a realizar se llama “TR\_DIM\_GRUPO\_EDAD”, su objetivo es almacenar los diferentes grupos de edad para así hacer uso de ellos en el hecho de mediciones, el resultado de esta transformación va a ser los datos almacenados en “DIM\_Grupo\_Edad”.

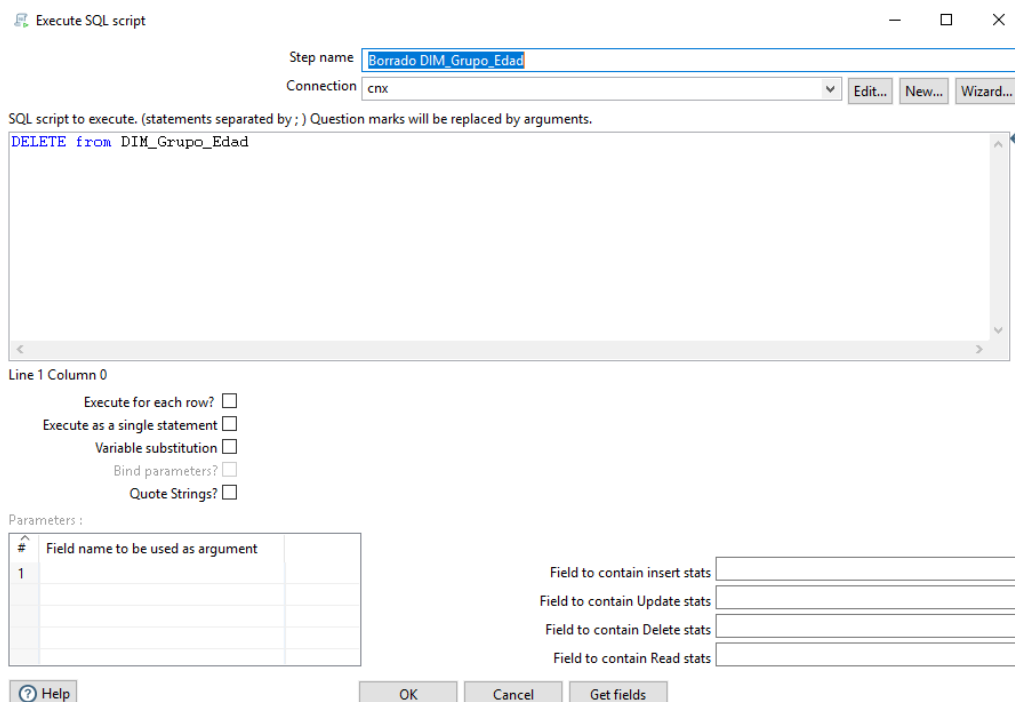
La transformación nos ha quedado de la siguiente forma:



*Ilustración 81 - TR\_DIM\_GRUPO\_EDAD.*

#### Borrado

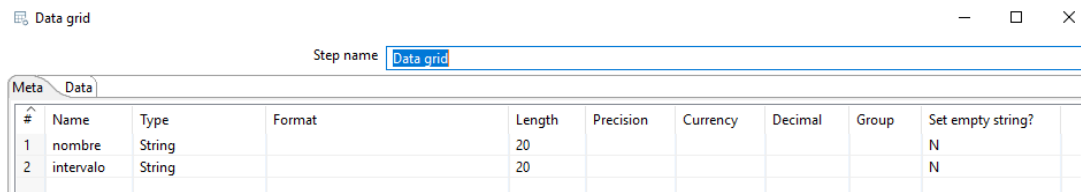
Lo primero que debemos de hacer es el borrado de los registros que contenía la dimensión, para ello escribimos directamente la sentencia SQL y la ejecutamos:



*Ilustración 82 - Borrado TR\_DIM\_GRUPO\_EDAD.*

## Grid

Puesto que la información de esta dimensión es fija y tiene tan solo 7 registros, nos resulta más fácil almacenar la información a partir de un grid (ya que en el enunciado de la práctica no se indica que no se pueda hacer uso de ellos), es por ello que hemos definido el siguiente grid:

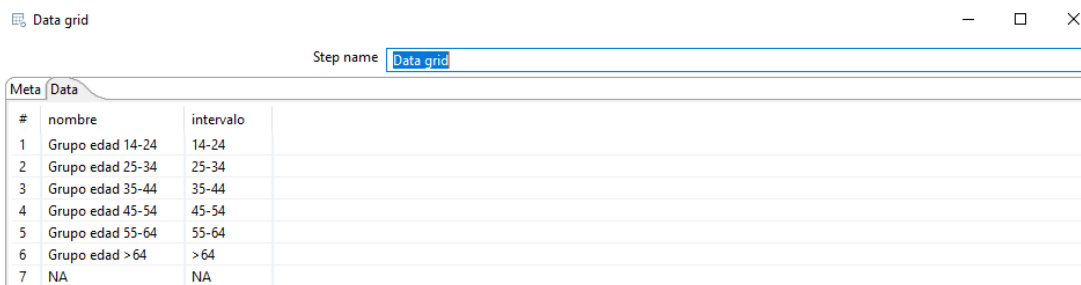


The screenshot shows a window titled 'Data grid' with a 'Step name' field set to 'Data grid'. Below the window title is a tabbed interface with 'Meta' and 'Data' tabs. The 'Data' tab is active, displaying a table with the following columns: #, Name, Type, Format, Length, Precision, Currency, Decimal, Group, and Set empty string?. The table contains two rows:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Set empty string?
1	nombre	String		20					N
2	intervalo	String		20					N

*Ilustración 83 - Grid TR\_DIM\_GRUPO\_EDAD.*

Una vez definidos los campos, introducimos los registros de forma manual. Cabe destacar que vamos a tener un registro con valores “NA”, esto significa que esta dimensión no va a aplicar para calcular ciertas medidas:



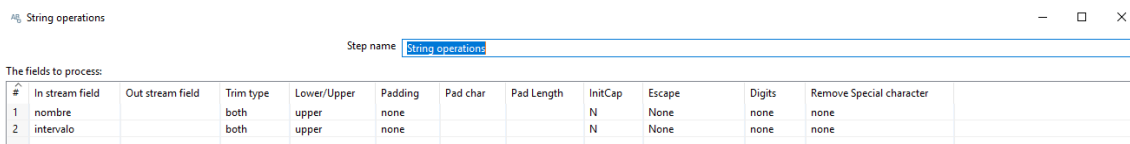
The screenshot shows the same 'Data grid' window, but now with data records entered. The table has two columns: 'nombre' and 'intervalo'. The records are as follows:

#	nombre	intervalo
1	Grupo edad 14-24	14-24
2	Grupo edad 25-34	25-34
3	Grupo edad 35-44	35-44
4	Grupo edad 45-54	45-54
5	Grupo edad 55-64	55-64
6	Grupo edad >64	>64
7	NA	NA

*Ilustración 84 - Grid TR\_DIM\_GRUPO\_EDAD.*

## Normalización

Normalizamos tanto el nombre como el intervalo para que estén en mayúsculas y no tengan espacios ni al principio ni al final:



The screenshot shows a window titled 'String operations' with a 'Step name' field set to 'String operations'. Below the window title is a tabbed interface with 'Meta' and 'Data' tabs. The 'Data' tab is active, displaying a table with the following columns: #, In stream field, Out stream field, Trim type, Lower/Upper, Padding, Pad char, Pad Length, InitCap, Escape, Digits, and Remove Special character. The table contains two rows:

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits	Remove Special character
1	nombre		both	upper	none			N	None	none	none
2	intervalo		both	upper	none			N	None	none	none

*Ilustración 85 - Normalización TR\_DIM\_GRUPO\_EDAD.*

## Secuenciación

Otro aspecto a destacar es que las dimensiones ya tienen claves primarias, por lo tanto vamos a definir la misma como un autonumérico incrementándose de uno en uno:

**Add sequence**

Step name:

Name of value:

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection:

Schema name:

Sequence name:

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☒

Counter name (optional):

Start at value:

Increment by:

Maximum value:

*Ilustración 86 - Secuenciación TR\_DIM\_GRUPO\_EDAD.*

## Guardado

Finalmente, realizamos el guardado en la dimensión indicando la tabla destino como “DIM\_Grupo\_Edad” y asociamos los atributos:

**Table output**

Step name:

Connection:

Target schema:

Target table:

Commit size:

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

**Main options** **Database fields**

Partition data over tables: ☐

Partitioning field:

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

Field that contains name of table:

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field:

*Ilustración 87 - Guardado TR\_DIM\_GRUPO\_EDAD.*

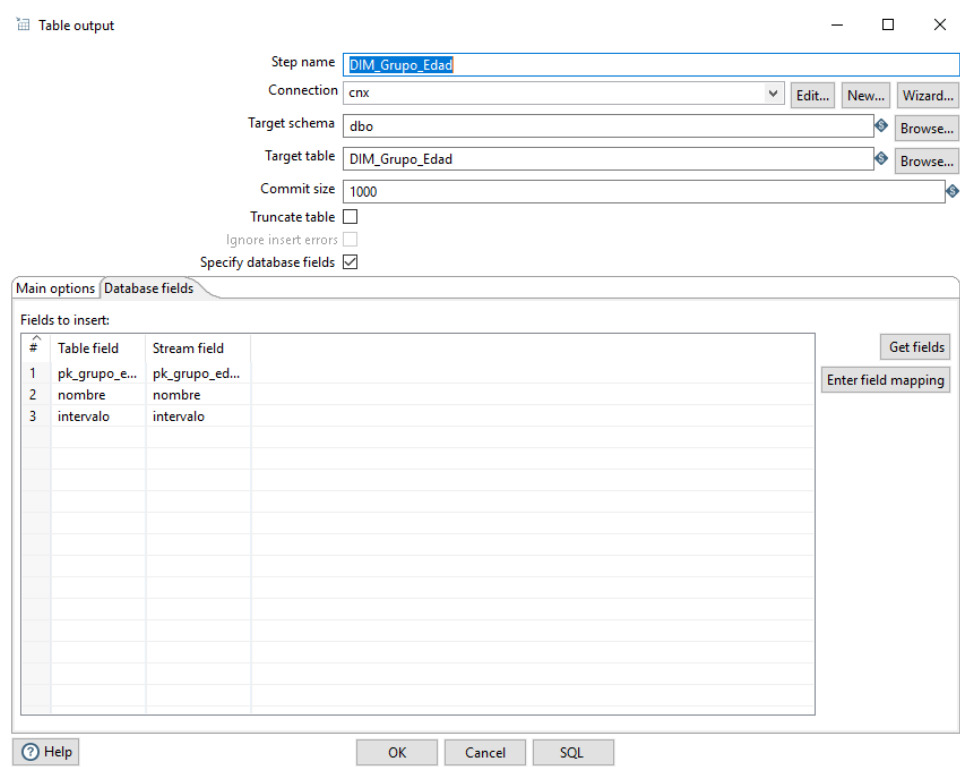


Ilustración 88 - Guardado TR\_DIM\_GRUPO\_EDAD.

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

Execution Results													
Logging Execution History Step Metrics Performance Graph Metrics Preview data													
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado DIM_Grupo_Edad	0	0	1	0	0	0	0	0	Finished	0.0s	67	-
2	Data grid	0	0	7	0	0	0	0	0	Finished	0.0s	368	-
3	String operations	0	7	7	0	0	0	0	0	Finished	0.0s	292	-
4	Add sequence	0	7	7	0	0	0	0	0	Finished	0.0s	259	-
5	DIM_Grupo_Edad	0	7	7	0	7	0	0	0	Finished	0.1s	119	-

Ilustración 89 - Métricas TR\_DIM\_GRUPO\_EDAD.

Como podemos observar generamos los 7 registros creados manualmente y guardamos todos en la base de datos.

### 3.3.2. Transformación TR\_DIM\_Medicion

La segunda transformación que vamos a realizar se llama “TR\_DIM\_GRUPO\_EDAD”, su objetivo es almacenar las diferentes medidas que vamos a usar en la tala de hechos mediciones, el resultado de esta transformación va a ser los datos almacenados en “DIM\_Medicion”.

La transformación nos ha quedado de la siguiente forma:



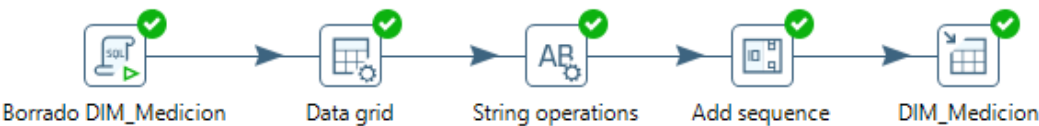


Ilustración 90 - TR\_DIM\_MEDICION.

**Borrado**

Lo primero que debemos de hacer es borrar todos los registros que hay en la tabla, por si había previos:

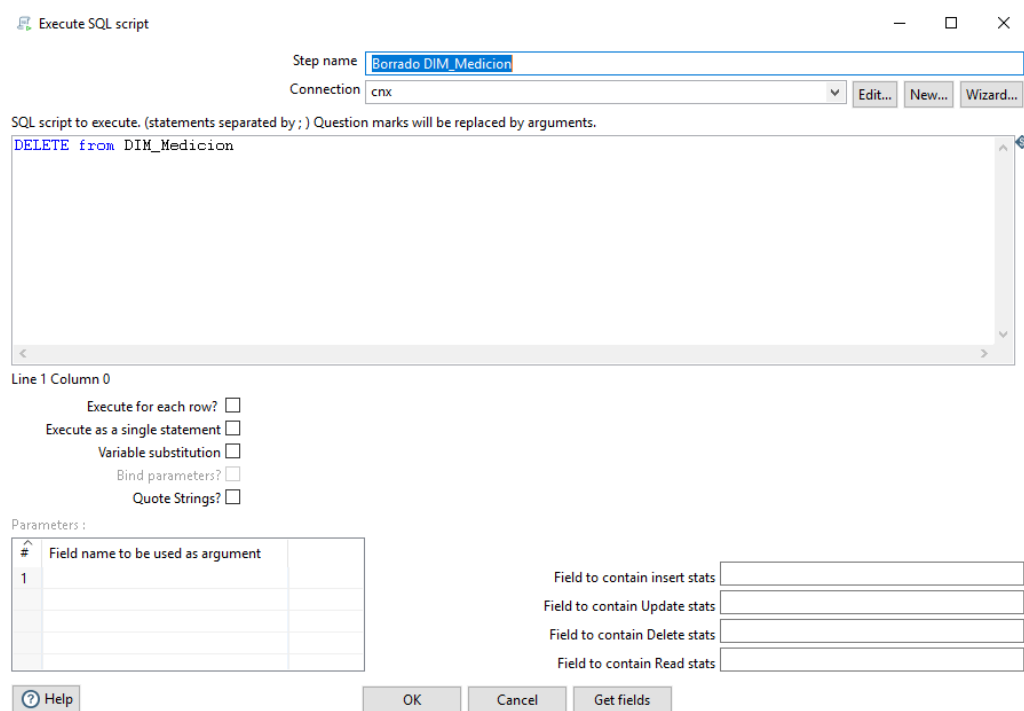


Ilustración 91 - Borrado TR\_DIM\_MEDICION.

**Grid**

Como los datos de esta dimensión no se encuentran en ningún fichero, la única solución que tenemos es introducirlos de forma manual, es por ello que hemos creado el siguiente grid:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Set empty string?
1	nombre	String		100					N
2	unidad_medida	String		20					N

Ilustración 92 - Grid TR\_DIM\_DIM\_MEDICION.

Una vez definidos los campos, introducimos los registros de forma manual:

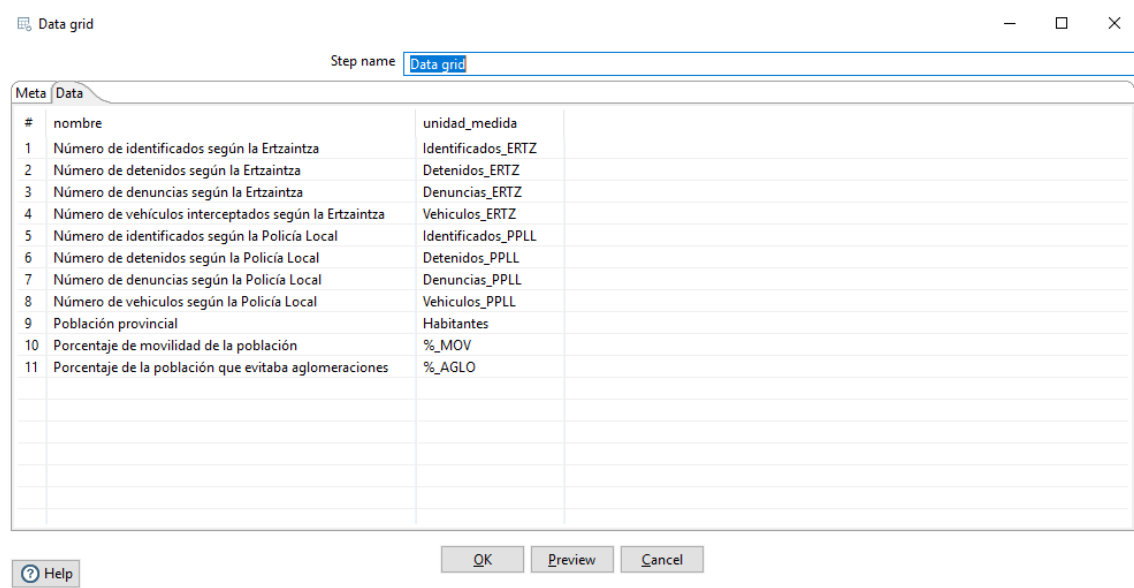


Ilustración 93 - Grid TR\_DIM\_MEDICION.

Normalización

Normalizamos tanto el nombre como la unidad de mediada, para que así todo esté en mayúsculas y sin espacios:

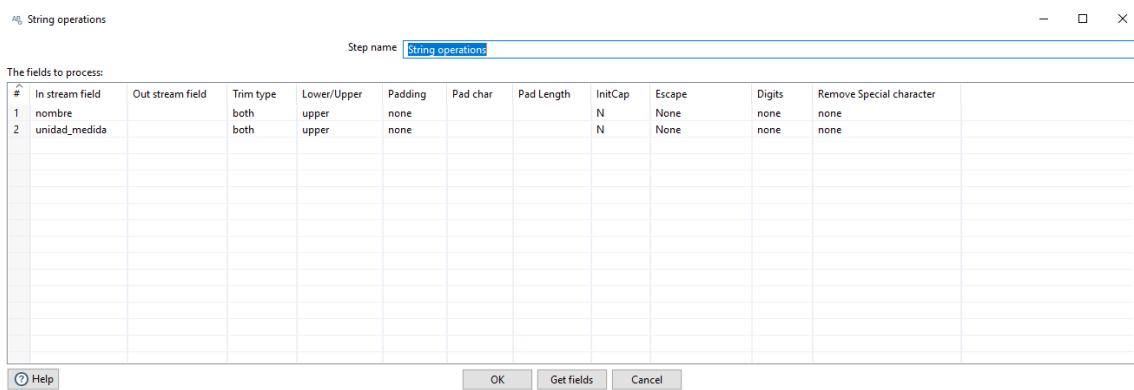


Ilustración 94 - Normalización TR\_DIM\_MEDICION.

Secuenciación

Al igual que sucedía antes, las tablas dimensiones ya tienen claves primarias, por lo que tenemos que definir la misma para esta dimensión:

Step name: Add sequence

Name of value: pk\_medicion

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection: cnx

Schema name:

Sequence name: SEQ\_

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

Help OK Cancel

*Ilustración 95 - Secuenciación TR\_DIM\_MEDICION.*

## Guardado

Una vez que ya tenemos todos los datos de forma correcta, procedemos a realizar el guardado en la tabla correspondiente, en nuestro caso “DIM\_Medicion”:

Step name: DIM\_Medicion

Connection: cnx

Target schema: dbo

Target table: DIM\_Medicion

Commit size: 1000

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

Partition data over tables: ☐

Partitioning field:

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

Field that contains name of table:

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field:

Help OK Cancel SQL

*Ilustración 96 - Guardado TR\_DIM\_MEDICION.*

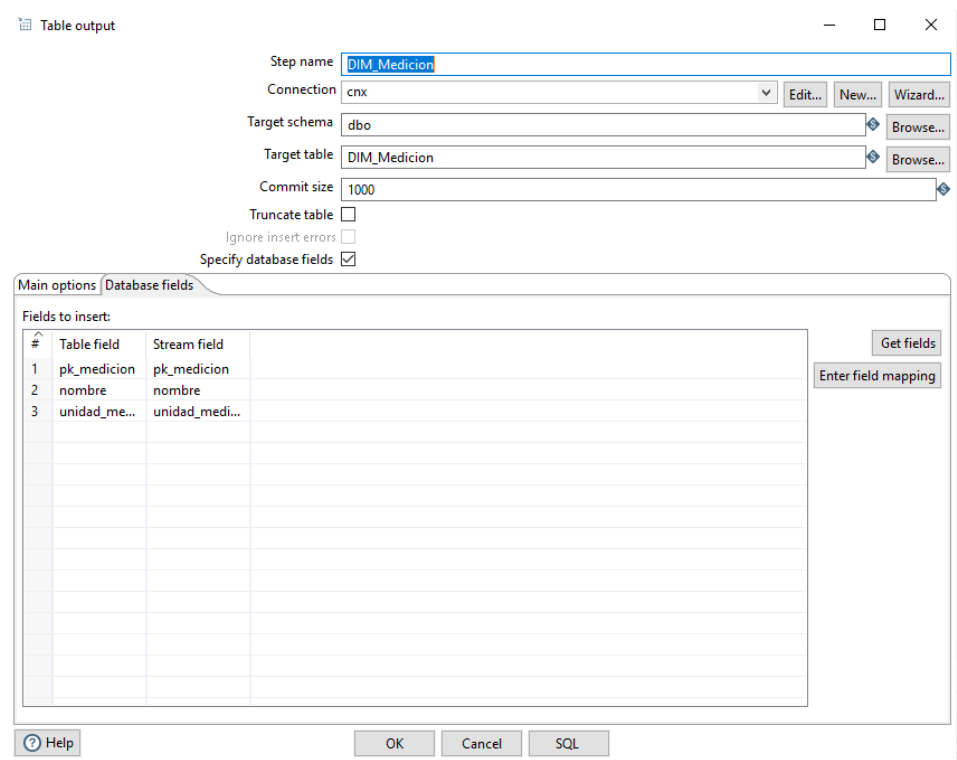


Ilustración 97 - Guardado TR\_DIM\_MEDICION.

Al ejecutar la anterior transformación nos proporciona las siguientes métricas:

**Execution Results**

Logging Execution History Step Metrics Performance Graph Metrics Preview data													
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado DIM_Medicion	0	0	1	0	0	0	0	0	Finished	0.0s	42	-
2	Data grid	0	0	11	0	0	0	0	0	Finished	0.0s	-	-
3	String operations	0	11	11	0	0	0	0	0	Finished	0.0s	458	-
4	Add sequence	0	11	11	0	0	0	0	0	Finished	0.0s	407	-
5	DIM_Medicion	0	11	11	0	11	0	0	0	Finished	0.1s	147	-

Ilustración 98 - Métricas TR\_DIM\_MEDICIONES.

Como podemos observar generamos los 11 registros de forma manual y los guardamos perfectamente en la base de datos.

3.3.3. Transformación TR\_DIM\_TIPOLOGIA

La tercera transformación de este bloque se corresponde con “TR\_DIM\_TIPOLOGIA”, su objetivo es almacenar las diferentes tipologías en las llamadas al 112 en Cataluña, el resultado de esta transformación va a ser los datos almacenados en “DIM\_Tipologia”.

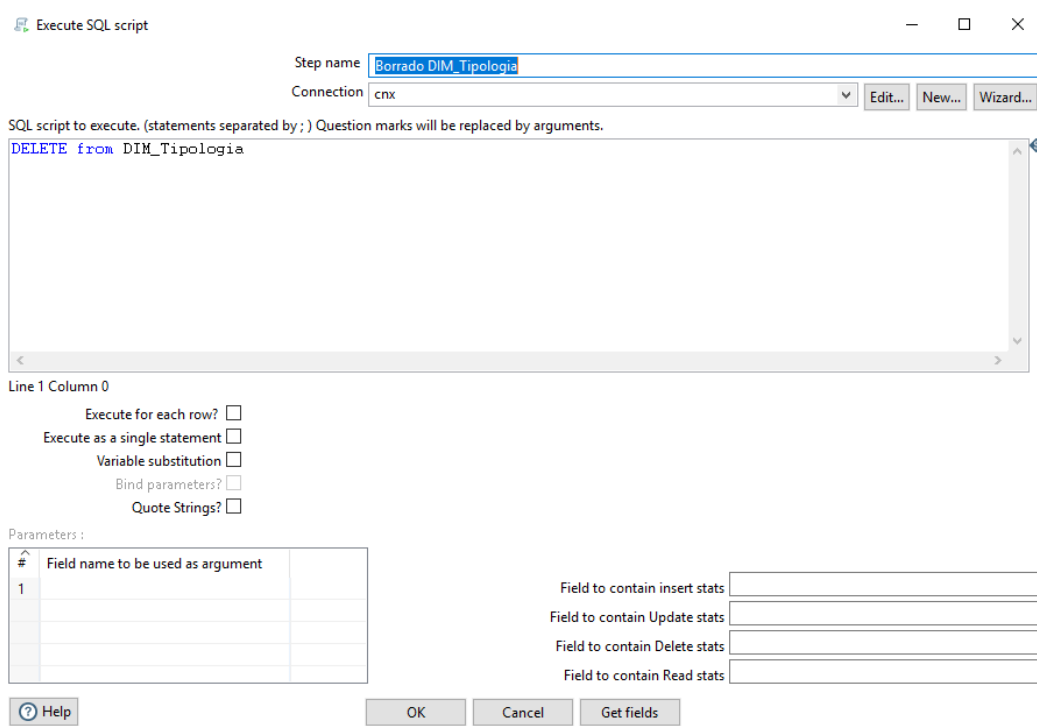
La transformación nos ha quedado de la siguiente forma:



*Ilustración 99 - TR\_DIM\_TIPOLOGIA.*

## Borrado

Al igual que en las transformaciones anteriores lo primero que debemos de hacer es el borrado de los registros que tenemos en la dimensión:

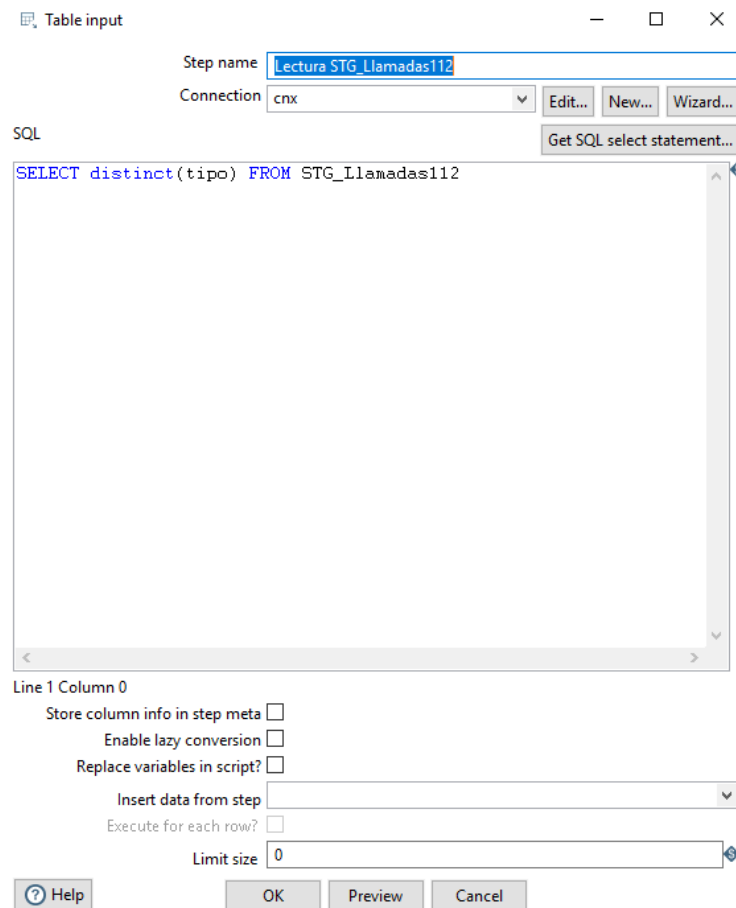


*Ilustración 100 - Borrado TR\_DIM\_TIPOLOGIA.*

## Lectura

Aunque en este caso hay también pocas tipologías, para ser más exactos hay 10, podríamos haber usado un grid pero hemos considerado que lo mejor es hacer la lectura de la tabla intermedia “SGT\_Llamadas112” porque la información no es fija, es decir, en un futuro pueden pasarnos tipologías nuevas y de no hacerlo así tendríamos que modificar la transformación.

Al hacer la lectura indicamos el campo “tipo”, la tabla “STG\_Llamadas112” y con la función distinct nos quedamos con todas las tipologías diferentes:



*Ilustración 101 - Lectura TR\_DIM\_TIPOLOGIA.*

### **Secuenciación**

Al igual que en las transformaciones anteriores, definimos la clave primaria de “DIM\_Tipologia” a partir de una secuencia numérica:

**Add sequence**

Step name: Add sequence

Name of value: pk\_tipologia

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection: cnx

Schema name:

Sequence name: SEQ\_

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

Help OK Cancel

*Ilustración 102 - Secuenciación TR\_DIM\_TIPOLOGIA.*

## Guardado

Finalmente, realizamos el guardado en la dimensión indicando la tabla destino como “DIM\_Tipologia” y asociamos los campos:

**Table output**

Step name: DIM\_Tipologia

Connection: cnx

Target schema: dbo

Target table: DIM\_Tipologia

Commit size: 1000

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

Main options Database fields

Partition data over tables: ☐

Partitioning field:

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field? ☐

Field that contains name of table:

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field:

Help OK Cancel SQL

*Ilustración 103 - Guardado TR\_DIM\_TIPOLOGIA.*

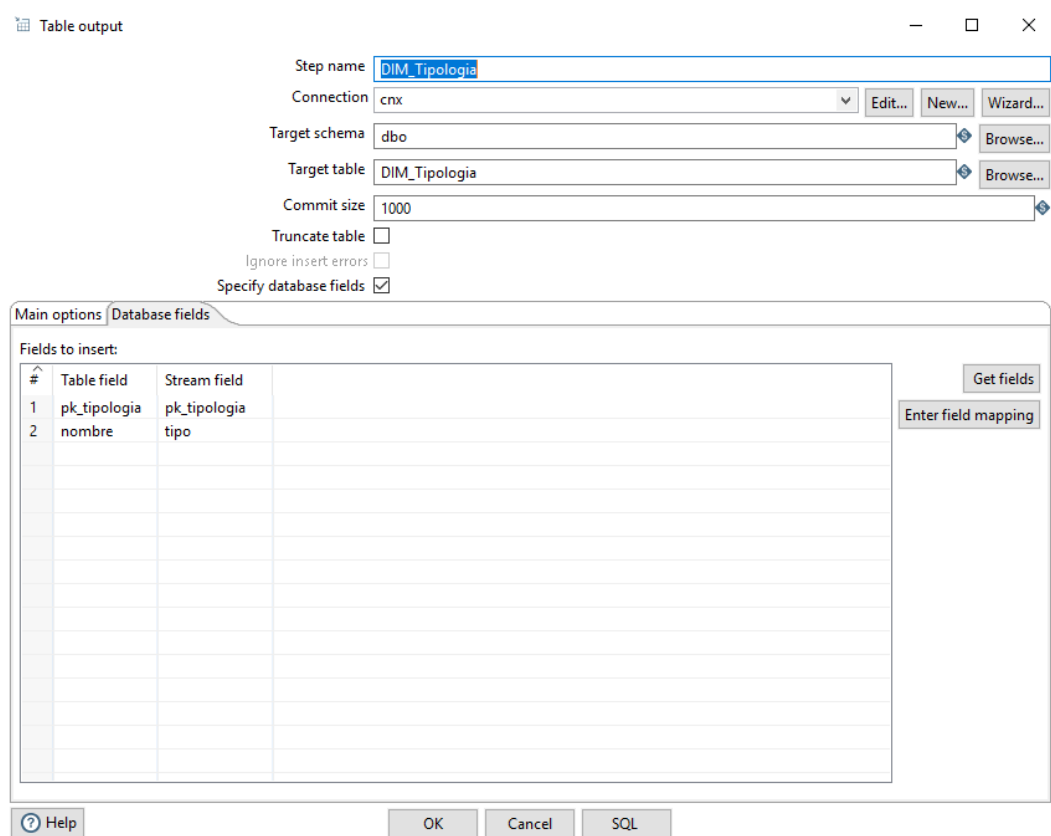


Ilustración 104 - Guardado TR\_DIM\_TIPOLOGIA.

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

**Execution Results**

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado DIM_Tipologia	0	0	1	0	0	0	0	0	Finished	0.0s	33	-
2	Lectura STG_Llamadas112	0	0	10	10	0	0	0	0	Finished	0.4s	26	-
3	Add sequence	0	10	10	0	0	0	0	0	Finished	0.4s	25	-
4	DIM_Tipologia	0	10	10	0	10	0	0	0	Finished	0.4s	23	-

Ilustración 105 - Métricas TR\_DIM\_TIPOLOGIA.

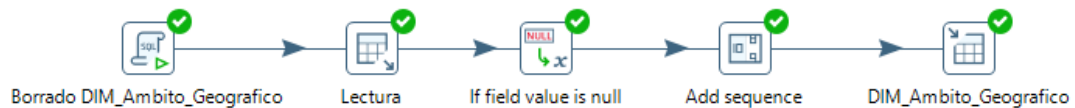
3.3.4. Transformación TR\_DIM\_AMBITO\_GEOGRAFICO

La cuarta transformación se corresponde con una dimensión compartida por ambos hechos, esta transformación se llama “TR\_DIM\_AMBITO\_GEOGRAFICO” y se encarga de almacenar todos los datos geográficos recogidos de la fuentes proporcionadas, es decir, datos que se encuentran en las tablas intermedias.

Una vez que hemos leído todos los datos los vamos a almacenar a la tabla “DIM\_Ambito\_Geografico”, ya que es ésta la que se corresponde con la dimensión.

La transformación nos ha quedado de la siguiente forma:

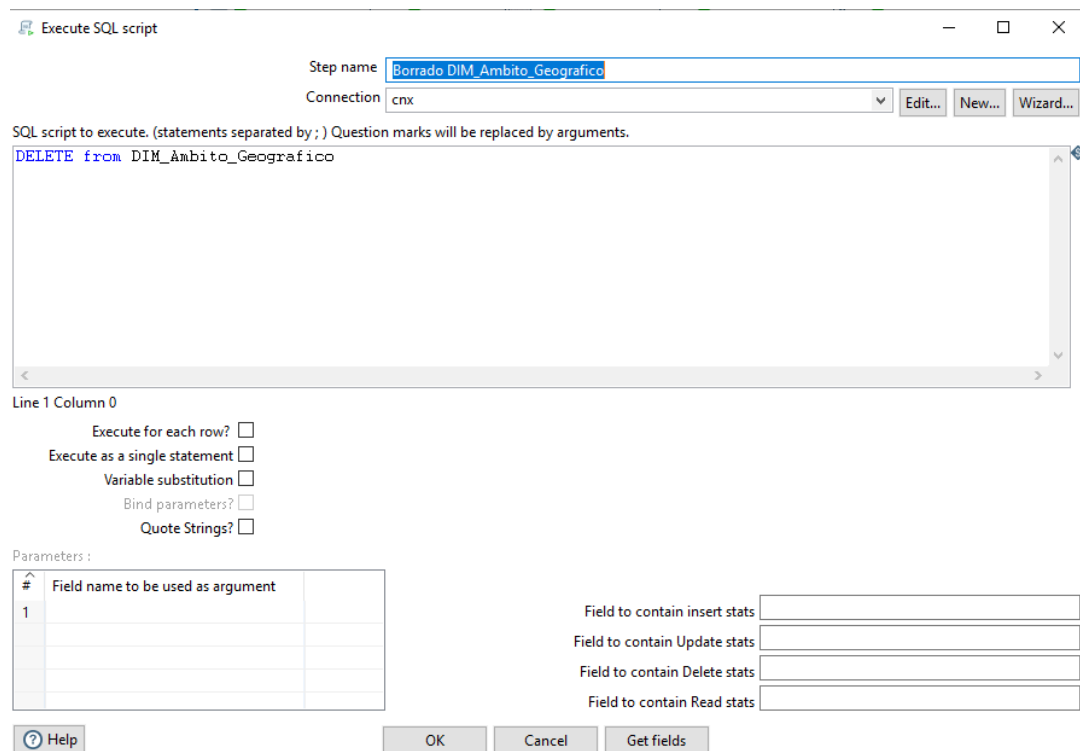




*Ilustración 106 - TR\_DIM\_AMBITO\_GEOGRAFICO.*

### Borrado

Lo primero que debemos de hacer es un borrado de los registros (si hay) de “DIM\_Ambito\_Geografico”, para ello escribimos directamente la sentencia SQL y la ejecutamos:



*Ilustración 107 - Borrado TR\_DIM\_AMBITO\_GEOGRAFICO.*

### Lectura

En este caso tenemos que introducir en la dimensión todos los datos relativos al ámbito geográfico que tenemos en las tablas intermedias. Tal y como está definida la dimensión los atributos “provincia\_codigo” y “provincia\_nombre” son obligatorios (no pueden ser nulos), es por ello que hacemos diferentes joins entre las tablas implicadas (STG\_Evitar\_Aglomeracion, STG\_Poblacion, STG\_Llamadas112) ya que no todas tienen el atributo “provincia\_codigo”. Además aprovechando los joins, establecemos el nombre de la comunidad a cada provincia.

El script necesario para realizar la operación comentada en el párrafo anterior es el siguiente:

```

select provincia_codigo, provincia_nombre, comunidad_autonoma, comarca, municipio
from STG_Evitar_Aglomeracion

RIGHT JOIN STG_Poblacion
ON STG_Evitar_Aglomeracion.provincia = STG_Poblacion.provincia_nombre

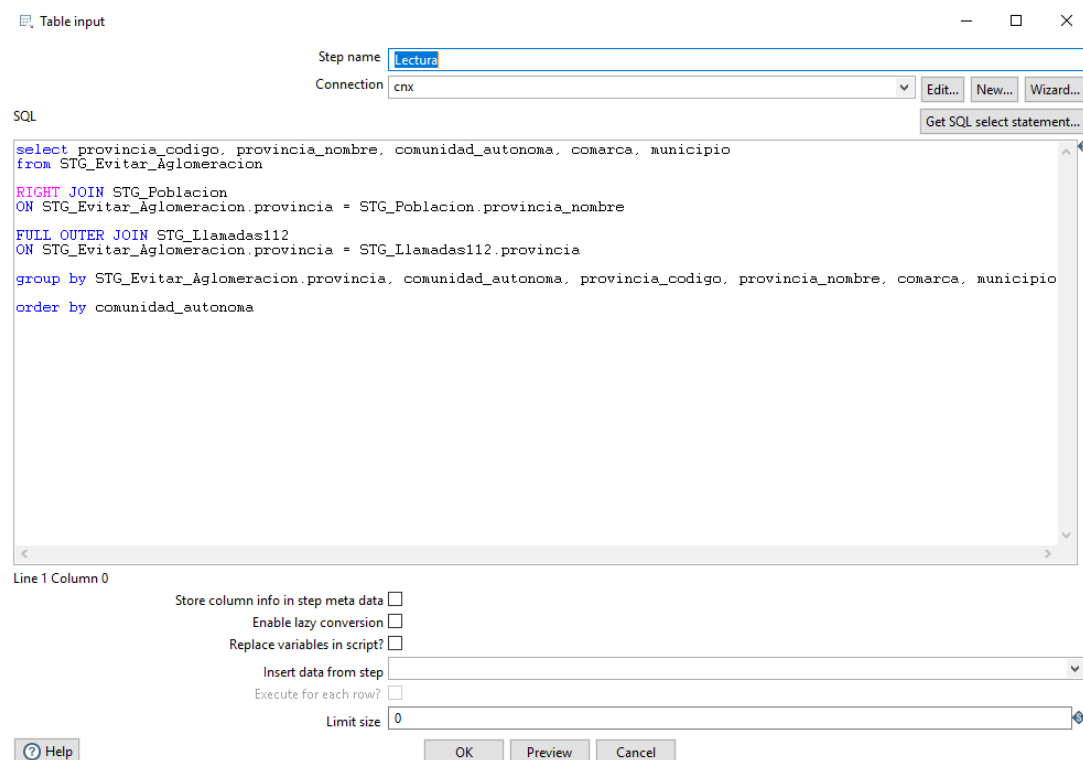
FULL OUTER JOIN STG_Llamadas112
ON STG_Evitar_Aglomeracion.provincia = STG_Llamadas112.provincia

group by STG_Evitar_Aglomeracion.provincia, comunidad_autonoma, provincia_codigo, provincia_nombre, comarca, municipio
_order by comunidad_autonoma

```

*Ilustración 108 - Lectura TR\_DIM\_AMBITO\_GEOGRAFICO.*

De tal forma, al agrupar por los campos que apreciamos en la imagen anterior, obtenemos todos los datos geográficos de forma única, es decir, no tenemos duplicados. Finalmente, ordenamos dichos valores por su comunidad para que sea más legible, una vez hecho todo esto usamos Spoon para realizar la carga:



*Ilustración 109 - Lectura TR\_DIM\_AMBITO\_GEOGRAFICO.*

## Nulos

Al hacer los joins anteriores en determinados atributos (comunidad autónoma, comarca y municipio) no siempre tienen valor, por ejemplo, la ciudad Ceuta o Melilla no tienen una comunidad como tal, porque son ciudades autónomas pero no comunidades.

Para solventar estos problemas sustituimos los valores nulos de todos los campos por "NA", esto significa que no es aplicable, de tal forma en Spoon nos quedaría la siguiente configuración:

If field value is null

Step name

If field value is null

Replace Null for all fields

Replace by value

NA

Set empty string?

Mask (Date)

Select fields

Select value type

Value types

#	Type	Replace by value	Conversion mask (Date)	Set empty string?

Fields

#	Field	Replace by value	Conversion mask (Date)	Set empty string?
1				

Help

OK

Get Fields

Cancel

Ilustración 110 - Nulos TR\_DIM\_AMBITO\_GEOGRAFICO.

Secuenciación

En este caso no normalizamos los datos porque ya lo hicimos al crear las tablas STG, de tal forma que todos los datos están en mayúsculas y sin espacios.

Otro aspecto a tener en cuenta es la creación de la clave primaria para esta dimensión, por lo que vamos a definir la misma como un autonumérico incrementándose de uno en uno:

**Add sequence**

Step name:

Name of value:

Use a database to generate the sequence

Use DB to get sequence? ☐

Connection:

Schema name:

Sequence name:

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☒

Counter name (optional):

Start at value:

Increment by:

Maximum value:

*Ilustración 111 - Secuenciación TR\_DIM\_AMBITO\_GEOGRAFICO.*

## Guardado

Finalmente, realizamos el guardado en la dimensión indicando la tabla destino como “DIM\_Ambito\_Geografico” y asociamos los atributos:

**Table output**

Step name:

Connection:

Target schema:

Target table:

Commit size:

Truncate table: ☐

Ignore insert errors: ☐

Specify database fields: ☒

**Main options** **Database fields**

Partition data over tables: ☐

Partitioning field:

Partition data per month: ☒

Partition data per day: ☐

Use batch update for inserts: ☒

Is the name of the table defined in a field?: ☐

Field that contains name of table:

Store the tablename field: ☒

Return auto-generated key: ☐

Name of auto-generated key field:

*Ilustración 112 - Guardado TR\_DIM\_AMBITO\_GEOGRAFICO.*

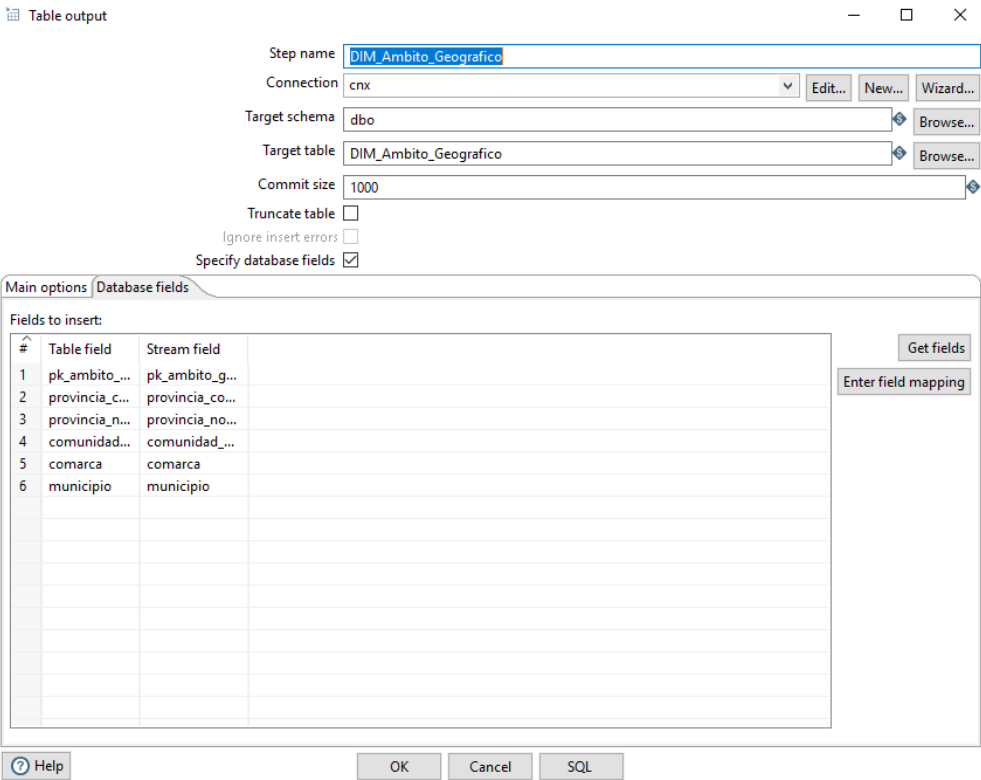


Ilustración 113 - Guardado TR\_DIM\_AMBITO\_GEOGRAFICO.

El ejecutar la anterior transformación obtenemos las siguientes métricas:

**Execution Results**

Logging Execution History Step Metrics Performance Graph Metrics Preview data

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado DIM_Ambito_Geografico	0	0	1	0	0	0	0	0	Finished	0.0s	48	-
2	Lectura	0	0	1019	1019	0	0	0	0	Finished	22.6s	45	-
3	If field value is null	0	1019	1019	0	0	0	0	0	Finished	22.7s	45	-
4	Add sequence	0	1019	1019	0	0	0	0	0	Finished	22.7s	45	-
5	DIM_Ambito_Geografico	0	1019	1019	0	1019	0	0	0	Finished	22.7s	45	-

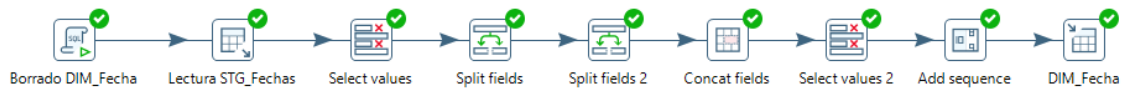
Ilustración 114 - Métricas TR\_DIM\_AMBITO\_GEOGRAFICO.

Como podemos observar leemos 1019 registros y almacenamos en la dimensión los mismos 1019 registros.

### 3.3.5. Transformación TR\_DIM\_FECHA

La última transformación respecto a las dimensiones es “TR\_DIM\_FECHA”, su objetivo es almacenar todas las fecha que se encuentran en las tablas intermedias y almacenarlas en “DIM\_Fecha”.

La transformación nos ha quedado de la siguiente forma:



*Ilustración 115 - TR\_DIM\_FECHA.*

## Borrado

Lo primero que debemos hacer es el borrado de los registros que contenía la dimensión, para ello escribimos directamente la sentencia SQL y la ejecutamos:

Execute SQL script

Step name: **Borrado DIM\_Fecha**

Connection: cnx

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

```
DELETE from DIM_Fecha
```

Line 1 Column 0

Execute for each row? ☐

Execute as a single statement ☐

Variable substitution ☐

Bind parameters? ☐

Quote Strings? ☐

Parameters :

#	Field name to be used as argument
1	

Field to contain insert stats

Field to contain Update stats

Field to contain Delete stats

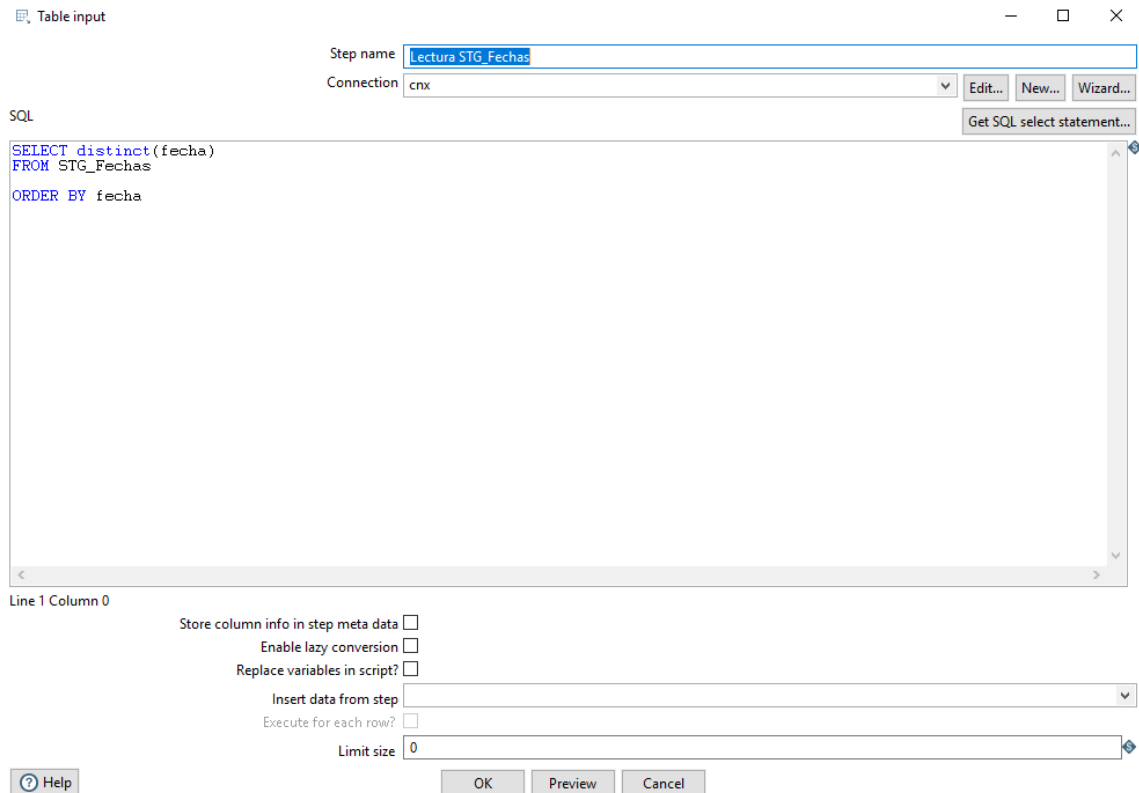
Field to contain Read stats

Help OK Cancel Get fields

*Ilustración 116 - Borrado TR\_DIM\_FECHA.*

## Lectura

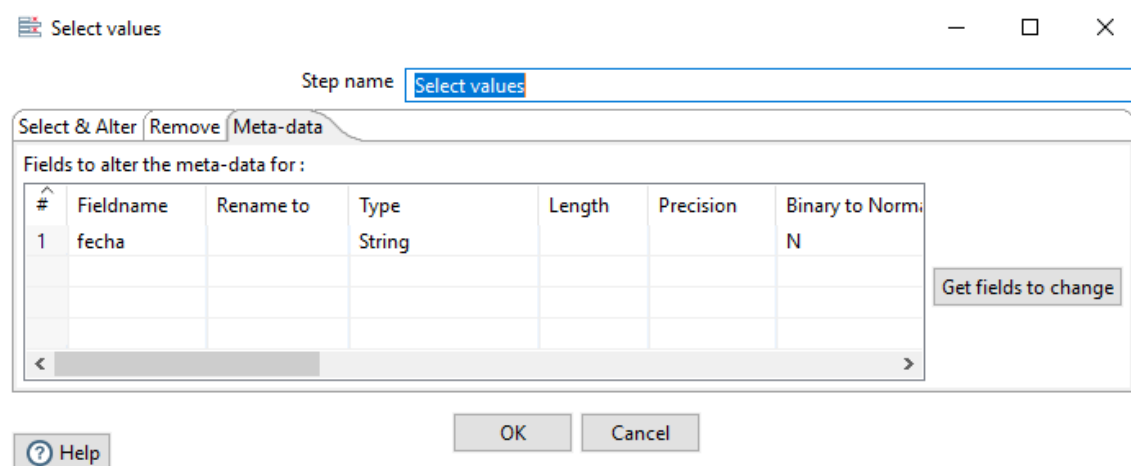
Para hacer la lectura de las fechas tenemos que cargar todos los registros distintos de “STG\_Fechas”, ya que es en esta tabla intermedia donde están almacenadas todas las fechas de todos los ficheros fuentes que nos han proporcionado.



*Ilustración 117 - Lectura TR\_DIM\_FECHA.*

## Conversión

Lo siguiente que debemos hacer es convertir el campo fecha que hemos leído en el paso anterior a string, ya que para esta dimensión no solo tenemos que guardar la fecha como tal, sino que también el día, mes y año por separado. Por lo tanto, convertimos a string la fecha:



*Ilustración 118 - Conversión String TR\_DIM\_FECHA.*

Split

La fecha que tenemos en la base de datos nos proporciona también la hora, pero estos datos no nos interesan, es por ello que usamos split, le indicamos que divida la fecha a partir de un espacio en blanco:

Split fields

Step name

Split fields

Field to split

fecha

Delimiter

Enclosure

Fields

#	New field	ID	Remove ID?	Type	Length	Precision	Format	Group	Decimal	Currency	Nullif	Default	Trim type
1	fecha		N	String									both
2	time		N	String									both

Help

OK

Cancel

Ilustración 119 - Split TR\_DIM\_FECHA.

Split

Una vez que tenemos solamente la fecha en formato string, volvemos a hacer un split de los campos para obtener el día, mes y año en formato numérico:

Split fields

Step name

Split fields 2

Field to split

fecha

Delimiter

/

Enclosure

Fields

#	New field	ID	Remove ID?	Type	Length	Precision	Format	Group	Decimal	Currency	Nullif	Default	Trim type
1	año		N	Integer									none
2	mes		N	Integer									none
3	día		N	Integer									none

Help

OK

Cancel

Ilustración 120 - Split TR\_DIM\_FECHA.

Concatenación

Al hacer split sobre el campo fecha hemos perdido ese campo como tal, es decir, hemos perdido el campo que tenía tanto el mes, día y año con el formato “yyyy/MM/dd”, es por ello que a partir de los campos creados en el paso anterior volvemos a crear la fecha:



Step name: Concat fields

Target Field Name: fecha

Length of Target Field: 0

Separator: /

Enclosure:

Fields

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	anyo	String							none	
2	mes	String							none	
3	dia	String							none	

Get Fields Minimal width

Help OK Cancel

*Ilustración 121 - Concatenación TR\_DIM\_FECHA.*

## Conversión

Ese nuevo campo fecha está en formato string, pero en la dimensión necesitamos que sea de tipo date, es por ello que la convertimos a dicho tipo:

Step name: Select values 2

Select & Alter Remove Meta-data

Fields to alter the meta-data for:

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normi
1	fecha		Date			N

Get fields to change

Help OK Cancel

*Ilustración 122 - Conversión TR\_DIM\_FECHA.*

## Secuenciación

Al igual que sucedía con las dimensiones anteriores, necesitamos definir una clave primaria, es por ello que creamos un nuevo campo autonumérico que se va incrementado de uno en uno:

**Add sequence**

Step name:

Name of value:

Use a database to generate the sequence

Use DB to get sequence? ☐

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:  ...

Increment by:  ...

Maximum value:  ...

? Help OK Cancel

*Ilustración 123 - Secuenciación TR\_DIM\_FECHA.*

## Guardado

Finalmente realizamos el guardado de todo este proceso en la “DIM\_Fecha”, para ello asociamos los campos con la tabla de la base de datos:

[illegible]

*Ilustración 124 - Guardado TR DIM FECHA.*

Al ejecutar la anterior transformación obtenemos las siguientes métricas:

**Execution Results**

Execution Results													
Logging Execution History Step Metrics Performance Graph Metrics Preview data													
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/output
1	Borrado DIM_Fecha	0	0	1	0	0	0	0	0	Finished	0.0s	71	-
2	Lectura STG_Fechas	0	0	170	170	0	0	0	0	Finished	0.0s	6,296	-
3	Select values	0	170	170	0	0	0	0	0	Finished	0.0s	3,953	-
4	Split fields	0	170	170	0	0	0	0	0	Finished	0.1s	3,400	-
5	Split fields 2	0	170	170	0	0	0	0	0	Finished	0.1s	1,405	-
6	Concat fields	0	170	170	0	0	0	0	0	Finished	0.1s	1,338	-
7	Select values 2	0	170	170	0	0	0	0	0	Finished	0.1s	1,288	-
8	Add sequence	0	170	170	0	0	0	0	0	Finished	0.1s	1,141	-
9	DIM_Fecha	0	170	170	0	170	0	0	0	Finished	0.2s	867	-

*Ilustración 125 - Métricas TR\_DIM\_FECHA.*

Como podemos apreciar de la anterior ilustración, tenemos 170 fechas únicas entre todos los ficheros proporcionados, y éstas mismas 170 se almacenan en la dimensión de forma correcta.

---

## 4. Bibliografía

---

a