

PRA2- Almacén de datos para el análisis de estadísticas deportivas de las ligas de baloncesto WNBA y NBA

Autos: Issam Fakhari

Contenido

Contenido.....	1
I. Identificación de los procesos ETL	2
A. Bloque IN (de las fuentes a tablas intermedias)	3
B. Bloque TR (de las tablas intermedias a nuestro almacén)	3
II. Diseño y desarrollo de los procesos ETL	5
A. Creación de tablas intermedias (<i>staging area</i>)	5
a) Tablas intermedias IN_STATS.....	5
b) Tablas intermedias IN_PLAYERS.....	6
c) Tablas intermedias IN_THROWS.....	7
d) Tablas intermedias IN_TEAMS	7
B. Creación del modelo multidimensional	8
e) Dimensiones	8
f) Tablas de hechos	11
C. Creación del proceso de extracción, transformación y carga (ETL)	13
a) Variables de entorno.....	13
b) Conexión a la base de datos SQL Server	14
c) Bloque IN.....	14
d) Bloque TR	46
III. Implementación de trabajos con procesos ETL.....	86
IV. Diseño de modelo OLAP.....	90

I. Identificación de los procesos ETL

- **Bloque IN:** procesos de carga de los datos desde las fuentes a las tablas intermedias en el área intermedia (*staging area*). 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 de los procesos de transformación para la carga de las tablas de hechos. Estos procesos se distinguen con el prefijo TR_ en el nombre.

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

Nombre ETL	Descripción	Orígenes de datos	Tabla de destino (Stage)
IN_STATS	Carga de los datos correspondientes a las estadísticas de juego por temporadas de los jugadores de la NBA y jugadoras de la WNBA	WNBA_Seasons_Stats_2005_2017.xls NBA_Seasons_Stats_1950_2017.csv	STG_WNBA_Seasons_Stats_2005_2017 STG_NBA_Seasons_Stats_1950_2017
IN_PLAYERS	Carga de los datos correspondientes a la información personal de los jugadores	WNBA_Players_List.txt NBA_Players_List.xls NBA_players_data.json	STG_WNBA_Players STG_NBAPlaylist_Active STG_NBA_Players
IN_THROWS	Carga de los datos correspondientes a los tiros libre realizados durante las temporadas	nba_free_throws.csv	STG_NBA_free_throws
IN_TEAMS	Carga de los datos correspondientes a los equipos de la NBA y la WNBA	WNBA_Teams.xls NBA_Teams.xml TeamCodes.txt Estados Unidos.json	STG_WNBA_Teams STG_NBA_Teams STG_Teams_Codes STG_Estados Unidos

B. Bloque TR (de las tablas intermedias a nuestro almacén)

Nombre ETL	Descripción	Tabla origen	Tabla de destino (dimensión)
------------	-------------	--------------	------------------------------

TR_DIM_GAMES	Carga y transformación de la dimensión con datos de los partidos jugados	STG_NBA_free_throws	DIM_Partidos
TR_DIM_PLAYS	Carga y transformación de la dimensión con datos de los tipos de jugadas que se pueden realizar en el caso de los tiros libres	STG_NBA_free_throws	DIM_Jugadas
TR_DIM_PLAYERS	Carga y transformación de la dimensión con datos de la información personal de los jugadores		DIM_Jugadores
TR_DIM_STATES	Carga y transformación de la dimensión con datos de la información de los diferentes estados miembros de los EE.UU.	STG_Estados_Unidos	DIM_ESTADOS_EEUU
TR_DIM_TEAMS	Carga y transformación de la dimensión con datos de los equipos que participan en las ligas NBA y WNBA	STG_WNBA_Teams STG_NBA_Teams STG_Teams_Codes	DIM_Equipos
TR_DIM_DIVISIONES	Carga y transformación de la dimensión con datos de las divisiones de la NBA.	STG_NBA_Teams	DIM_Divisiones
TR_DIM_CONFERENCIAS	Carga y transformación de la dimensión con datos de las conferencias de la NBA y WNBA.	STG_NBA_Teams	DIM_Conferencias

Y el proceso del bloque de carga y transformación de la tabla de hechos.

Nombre ETL	Descripción	Tabla origen
TR_FACT_SEASON_STATS	Carga y transformación de la tabla de hechos FACT_SEASON_STATS	STG_WNBA_Seasons_Stats_2005_2017 STG_NBA_Seasons_Stats_1950_2017
TR_FACT_FREE_THROWS	Carga y transformación de la tabla de hechos FACT_FREE_THROWS	STG_NBA_free_throws

II. Diseño y desarrollo de los procesos ETL

A. Creación de tablas intermedias (*staging area*)

a) Tablas intermedias IN_STATS

```
STG_WNBA_Seasons_Stats_2005_2017
CREATE TABLE [dbo].[STG_WNBA_Season_Stats_2005_2017](
    [id_wnba_player][varchar](50) NULL,
    [player][nvarchar](50) NULL,
        [age][numeric](2,0) NULL,
        [active][numeric](2,0) NULL,
    [season][numeric](4,0) NULL,
        [temporada][varchar](50) NULL,
        [team][nvarchar](5) NULL,
        [league][nvarchar](5) NULL,
    [GP][decimal](8,2) NULL,
        [PTS][decimal](8,2) NULL,
        [MIN][decimal](8,2) NULL,
        [FGM][decimal](8,2) NULL,
        [FGA][decimal](8,2) NULL,
        [FG%][decimal](8,2) NULL,
        [_3PM][decimal](8,2) NULL,
        [_3PA][decimal](8,2) NULL,
        [_3P%][decimal](8,2) NULL,
        [FTM][decimal](8,2) NULL,
        [FTA][decimal](8,2) NULL,
        [FT%][decimal](8,2) NULL,
        [OREB][decimal](8,2) NULL,
        [DREB][decimal](8,2) NULL,
        [REB][decimal](8,2) NULL,
        [AST][decimal](8,2) NULL,
        [TOV][decimal](8,2) NULL,
        [STL][decimal](8,2) NULL,
        [BLK][decimal](8,2) NULL,
        [PF][decimal](8,2) NULL,
        [EFF][decimal](8,2) NULL
    ) ON [PRIMARY]
GO
```

```
STG_NBA_Seasons_Stats_1950_2017
```

```
CREATE TABLE [dbo].[STG_NBA_Season_Stats_1950_2017](
    [year][numeric](4,0) NULL,
    [season][varchar](50) NULL,
    [player][varchar](50) NULL,
    [position][varchar](8) NULL,
    [age][numeric](4,0) NULL,
    [team][varchar](5) NULL,
    [G][numeric](8,0) NULL,
    [PTS][numeric](8,0) NULL,
    [MP][varchar](8) NULL,
    [FG][numeric](8,0) NULL,
    [FGA][numeric](8,0) NULL,
    [FG%][decimal](5,3) NULL,
    [_3P][varchar](8) NULL,
    [_3PA][varchar](8) NULL,
    [_3P%][varchar](8) NULL,
    [_2P][numeric](8,0) NULL,
    [_2PA][numeric](8,0) NULL,
    [_2P%][decimal](5,3) NULL,
    [FT][numeric](8,0) NULL,
```

```

        [FTA][numeric](8,0) NULL,
        [FT%][decimal](5,3) NULL,
        [eFG%][decimal](5,3) NULL,
        [ORB%][varchar](8) NULL,
        [ORB][varchar](8) NULL,
        [DRB][varchar](8) NULL,
        [DRB%][varchar](8) NULL,
        [AST][numeric](8,0) NULL,
        [TOV][varchar](8) NULL,
        [STL][varchar](8) NULL,
        [BLK][varchar](8) NULL,
        [PF][numeric](8,0) NULL,
        [TRB][varchar](8) NULL,
        [EFF][numeric](8,0) NULL,
    ) ON [PRIMARY]
GO

```

b) Tablas intermedias IN_PLAYERS

```

STG_NBAPlaylist_Active (IN_NBA_PlayersList)
CREATE TABLE [dbo].[STG_NBAPlaylist_ACTIVE](
    [active][varchar](100) NOT NULL,
) ON [PRIMAR]

STG_NBA_PLAYERS (IN_PLAYERS)
CREATE TABLE [dbo].[STG_NBA_PLAYERS](
    [pk_jugador][numeric](4,0) NULL,
    [liga][varchar](5) NULL,
    [cod_jugador][varchar](100) NULL,
    [nombre][varchar](255) NULL,
    [posicion_juego][varchar](100) NULL,
    [sexo][numeric](4,0) NULL,
    [activo][varchar](10) NULL,
    [altura][varchar](50) NULL,
    [peso][varchar](50) NULL,
    [shoots][varchar](50) NULL,
    [universidad][varchar](255) NULL,
    [fecha_nacimiento][varchar](50) NULL,
    [ciudad_nacimiento][varchar](255) NULL,
    [career_AST][varchar](10) NULL,
    [career_FG%][varchar](10) NULL,
    [career_FG3%][varchar](10) NULL,
    [career_FT%][varchar](10) NULL,
    [career_G][varchar](10) NULL,
    [career_PER][varchar](10) NULL,
    [career PTS][varchar](10) NULL,
    [career_TRB][varchar](10) NULL,
    [career_WS][varchar](10) NULL,
    [career_eFG%][varchar](10) NULL
) ON [PRIMARY]

STG_WNBA_PLAYERS (IN_WNBA_PlayerList)
CREATE TABLE [dbo].[STG_WNBA_Players](
    [player][nvarchar](50) NULL,
    [active][nvarchar](2) NULL
) ON [PRIMARY]
GO

```

c) Tablas intermedias IN_THROWNS

```
CREATE TABLE [dbo].[STG_NBA_free_throws](
    [end_result] [varchar](50) NULL,
    [game] [varchar](50) NULL,
    [game_id] [numeric](11,1) NULL,
    [period] [numeric](11,0) NULL,
    [play] [varchar](200) NULL,
    [play_tipo] [varchar](200) NULL,
    [player] [varchar](100) NULL,
    [play_cod] [varchar](50) NULL,
    [playoffs] [varchar](50) NULL,
    [score] [varchar](50) NULL,
    [time] [varchar](10) NULL,
    [season] [varchar](50) NULL
) ON [PRIMARY]
GO
```

d) Tablas intermedias IN_TEAMS

STG_WNBA_Teams (IN_WNBA_Teams)

```
CREATE TABLE [dbo].[STG_WNBA_Teams](
    [conferencia] [varchar](20) NULL,
    [equipo] [varchar](20) NULL,
    [J] [numeric](2,0) NULL,
    [G] [numeric](2,0) NULL,
    [ciudad] [varchar](20) NULL,
    [estado] [varchar](20) NULL
) ON [PRIMARY]
GO
```

STG_NBA_Teams (IN_NBA_Teams)

```
CREATE TABLE [dbo].[STG_NBA_Teams](
    [Conferencia] [varchar](100) NULL,
    [Division] [varchar](100) NULL,
    [Equipo] [varchar](255) NULL,
    [Ciudad] [varchar](255) NULL,
    [Estado] [varchar](10) NULL,
    [Pabellon] [varchar](255) NULL,
    [Fundado] [numeric](4,0) NOT NULL,
    [Patrocinio] [varchar](255) NULL
) ON [PRIMARY]
GO
```

STG_Teams_Codes (IN_Teams_Codes)

```
CREATE TABLE [dbo].[STG_Team_Codes](
    [league] [varchar](5) NULL,
    [team] [varchar](50) NULL,
    [code] [varchar](10) NULL
) ON [PRIMARY]
GO
```

STG_Estados_Unidos (IN_EEUU)

```
CREATE TABLE [dbo].[STG_Estados_Unidos](
    [codigo] [varchar](50) NULL,
    [nombre] [varchar](255) NULL,
    [Nombre oficial] [varchar](100) NULL,
    [superficie] [numeric](8,0) NULL,
    [poblacion] [numeric](8,0) NULL,
    [capital] [varchar](100) NULL,
    [densidadPoblacion] [decimal](8,2) NULL
) ON [PRIMARY]
```

GO

B. Creación del modelo multidimensional

e) Dimensiones

DIM_Tiempo

```
CREATE TABLE [dbo].[DIM_Tiempo](
    [pk_date] [numeric](4,0) NOT NULL,
    [date_year] [smallint] NOT NULL,
    [date_month] [tinyint] NOT NULL,
    [date_day] [tinyint] NOT NULL,
    [date_date] [date],
    [temporada] [varchar](50) NOT NULL,
    CONSTRAINT [PK_DIM_Tiempo] PRIMARY KEY CLUSTERED
    (
        [pk_date] 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_Minutos

```
CREATE TABLE [dbo].[DIM_Minutos](
    [pk_minutoSegundo] [numeric](4,0) NOT NULL,
    [minuto] [tinyint] NOT NULL,
    [segundo] [tinyint] NOT NULL,
    [minutoSegundo] [varchar](10) NOT NULL,
    CONSTRAINT [PK_DIM_Minutos] PRIMARY KEY CLUSTERED
    (
        [pk_minutoSegundo] 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_PosicionesJuego

```
CREATE TABLE [dbo].[DIM_PosicionesJuego](
    [pk_posicion_juego] [numeric](2,0) NOT NULL,
    [codigo_posicion] [varchar](10) NOT NULL,
    [desc_posicion_EN] [varchar](50) NOT NULL,
    [desc_posicion_ES] [varchar](50) NOT NULL,
    [num_posicion] [numeric](4,0) NOT NULL,
    CONSTRAINT [PK_DIM_PosicionesJuego] PRIMARY KEY CLUSTERED
    (
        [pk_posicion_juego] 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_Partidos

```
CREATE TABLE [dbo].[DIM_Partidos](
    [pk_partido] [numeric](8,0) NOT NULL,
    [partido] [varchar](50) NOT NULL,
    [playoffs] [varchar](50) NOT NULL,
```

```

        [temporada][varchar](50) NOT NULL,
        [resultado][varchar](50) NOT NULL,
        CONSTRAINT [PK_DIM_Partidos] PRIMARY KEY CLUSTERED
        (
        [pk_partido] 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_Jugadas
CREATE TABLE [dbo].[DIM_Jugadas](
    [pk_jugada][numeric](4,0) NOT NULL,
    [codigo_jugada][varchar](50) NOT NULL,
    [dec_jugada][varchar](100) NOT NULL,
    CONSTRAINT [PK_DIM_Jugadas] PRIMARY KEY CLUSTERED
    (
    [pk_jugada] 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_Jugadores
CREATE TABLE [dbo].[DIM_Jugadores](
    [pk_jugador][numeric](4,0) NOT NULL,
    [liga][varchar](5) NOT NULL,
    [cod_jugador][varchar](100) NOT NULL,
    [nombre][varchar](255) NOT NULL,
    [posicion_juego][varchar](100) NOT NULL,
    [sexo][numeric](4,0) NOT NULL,
    [activo][numeric](4) NOT NULL,
    [altura][varchar](50) NOT NULL,
    [peso][varchar](50) NOT NULL,
    [shoots][varchar](50) NOT NULL,
    [universidad][varchar](255) NOT NULL,
    [fecha_nacimiento][date] NOT NULL,
    [ciudad_nacimiento][varchar](255) NOT NULL,
    [career_AST][decimal](8,2) NULL,
    [career_FG%][decimal](8,2) NULL,
    [career_FG3%][decimal](8,2) NULL,
    [career_FT%][decimal](8,2) NULL,
    [career_G][decimal](8,2) NULL,
    [career_PER][decimal](8,2) NULL,
    [career PTS][decimal](8,2) NULL,
    [career_TRB][decimal](8,2) NULL,
    [career_WS][decimal](8,2) NULL,
    [career_eFG%][decimal](8,2) NULL
    CONSTRAINT [PK_DIM_Jugadores] PRIMARY KEY CLUSTERED
    (
    [pk_jugador] 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_Estados_EEUU
CREATE TABLE [dbo].[DIM_Estados_EEUU](
    [pk_estado_EEUU][numeric](4,0),
    [codigo][varchar](50) NOT NULL,
    [nombre][varchar](255) NOT NULL,
    [Nombre oficial][varchar](100) NOT NULL,
    [superficie][numeric](8,0) NOT NULL,
    [poblacion][numeric](8,0) NOT NULL,
    [capital][nvarchar](100) NOT NULL,
    [densidadPoblacion][decimal](8,2) NOT NULL,
    CONSTRAINT [PK_DIM_Estados_EEUU] PRIMARY KEY CLUSTERED
    (
        [pk_estado_EEUU] 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_Equipos
CREATE TABLE [dbo].[DIM_Equipos](
    [pk_equipo][numeric](2,0) NOT NULL,
    [id_estado_EEU][numeric](4,0),
    [id_division][numeric](2,0) NOT NULL,
    [liga][varchar](5) NOT NULL,
    [codigo_equipo][varchar](10) NOT NULL,
    [nombre_equipo][varchar](50) NOT NULL,
    [ciudad][varchar](100) NOT NULL,
    [pabellon][varchar](100) NOT NULL,
    [fundado][numeric](4,0) NULL,
    [patrocinio][varchar](100) NOT NULL,
    CONSTRAINT [PK_DIM_Equipos] PRIMARY KEY CLUSTERED
    (
        [pk_equipo] 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_Equipos] WITH CHECK ADD CONSTRAINT
[FK_Estado_Equipo] FOREIGN KEY([id_estado_EEU])
REFERENCES[dbo].[DIM_Estados_EEUU]([pk_estado_EEUU])
GO

ALTER TABLE[dbo].[DIM_Equipos] CHECK CONSTRAINT[FK_Estado_Equipo]
GO

ALTER TABLE[dbo].[DIM_Equipos] WITH CHECK ADD CONSTRAINT
[FK_Division_Equipo] FOREIGN KEY([id_division])
REFERENCES[dbo].[DIM_Divisiones]([pk_division])
GO

ALTER TABLE[dbo].[DIM_Equipos] CHECK CONSTRAINT[FK_Division_Equipo]
GO

DIM_Divisiones
CREATE TABLE [dbo].[DIM_Divisiones](

```

```

[pk_division][numeric](2,0) NOT NULL,
[codigo_division][varchar](50) NOT NULL,
[Division][varchar](100) NOT NULL,
[id_conferencia][numeric](4,0) NOT NULL,
CONSTRAINT [PK_DIM_Divisiones] PRIMARY KEY CLUSTERED
(
    [pk_division] 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_Divisiones] WITH CHECK ADD CONSTRAINT
[FK_conferencia] FOREIGN KEY([id_conferencia])
REFERENCES[dbo].[DIM_Conferencias]([pk_conferencia])
GO

DIM_Conferencias
CREATE TABLE [dbo].[DIM_Conferencias](
    [pk_conferencia][numeric](4,0)NOT NULL,
    [codigo_conferencia][varchar](50) NOT NULL,
    [conferencia][varchar](100) NOT NULL,
    CONSTRAINT [PK_DIM_Conferencias] PRIMARY KEY CLUSTERED
    (
        [pk_conferencia] 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

```

f) Tablas de hechos

```

FACT_SEASONS_STATS
CREATE TABLE [dbo].[FACT_SEASONS_STATS](
    [id_season][numeric](4,0) NULL,
    [id_player][numeric](4,0) NULL,
    [id_team][numeric](2,0) NULL,
    [id_position][numeric](2,0) NULL,
    [league][varchar](5) NULL,
    [player][varchar](100) NOT NULL,
    [G][varchar](5) NULL,
    [PTS][varchar](5) NULL,
    [MP][varchar](20) NULL,
    [FG][varchar](5) NULL,
    [FGA][varchar](5) NULL,
    [FG%][varchar](5) NULL,
    [_3P][varchar](8) NULL,
    [_3PA][varchar](8) NULL,
    [_3P%][varchar](8) NULL,
    [FT][varchar](5) NULL,
    [FTA][varchar](5) NULL,
    [FT%][varchar](5) NULL,
    [ORB][varchar](8) NULL,
    [DRB][varchar](8) NULL,
    [TRB][varchar](8) NULL,
    [AST][varchar](5) NULL,
    [TOV][varchar](5) NULL,
    [STL][varchar](5) NULL,
    [BLK][varchar](5) NULL,

```

```

        [PF][varchar](5) NULL,
        [EFF][varchar](5) NULL,
    ) ON [PRIMARY]
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] WITH CHECK ADD CONSTRAINT
[FK_Season] FOREIGN KEY([id_season])
REFERENCES[dbo].[DIM_Tiempo]([pk_date])
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] CHECK CONSTRAINT[FK_Season]
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] WITH CHECK ADD CONSTRAINT
[FK_Player] FOREIGN KEY([id_player])
REFERENCES[dbo].[DIM_Jugadores]([pk_jugador])
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] CHECK CONSTRAINT[FK_Player]
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] WITH CHECK ADD CONSTRAINT
[FK_Team] FOREIGN KEY([id_team])
REFERENCES[dbo].[DIM_Equipos]([pk_equipo])
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] CHECK CONSTRAINT[FK_Team]
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] WITH CHECK ADD CONSTRAINT
[FK_Position] FOREIGN KEY([id_position])
REFERENCES[dbo].[DIM_PosicionesJuego]([pk_posicion_juego])
GO

ALTER TABLE[dbo].[FACT_SEASONS_STATS] CHECK CONSTRAINT[FK_Position]
GO

FACT_FREE_THROWS
CREATE TABLE [dbo].[FACT_FREE_THROWS](
    [id_temporada][numeric](4,0) NOT NULL,
    [id_partido][numeric](8,0) NOT NULL,
    [id_jugador][numeric](4,0) NOT NULL,
    [id_jugada][numeric](4,0) NOT NULL,
    [id_Minuto][numeric](4,0) NOT NULL,
    [periodo_juego][int] NOT NULL,
    [resultado_tiro_libre][numeric](8,2) NOT NULL,
) ON [PRIMARY]
GO

ALTER TABLE[dbo].[FACT_FREE_THROWS] WITH CHECK ADD CONSTRAINT
[FK_FREE_THROWS_Season] FOREIGN KEY([id_temporada])
REFERENCES[dbo].[DIM_Tiempo]([pk_date])
GO

ALTER TABLE[dbo].[FACT_FREE_THROWS] CHECK CONSTRAINT[FK_FREE_THROWS_Season]
GO

ALTER TABLE[dbo].[FACT_FREE_THROWS] WITH CHECK ADD CONSTRAINT
[FK_Partido] FOREIGN KEY([id_partido])
REFERENCES[dbo].[DIM_Partidos]([pk_partido])
GO

ALTER TABLE[dbo].[FACT_FREE_THROWS] CHECK CONSTRAINT[FK_Partido]

```

GO

```
ALTER TABLE[dbo].[FACT_FREE_THROWS] WITH CHECK ADD CONSTRAINT  
[FK_Jugador] FOREIGN KEY([id_jugador])  
REFERENCES[dbo].[DIM_Jugadores]([pk_jugador])  
GO
```

```
ALTER TABLE[dbo].[FACT_FREE_THROWS] CHECK CONSTRAINT[FK_Jugador]  
GO
```

```
ALTER TABLE[dbo].[FACT_FREE_THROWS] WITH CHECK ADD CONSTRAINT  
[FK_Jugada] FOREIGN KEY([id_jugada])  
REFERENCES[dbo].[DIM_Jugadas]([pk_jugada])  
GO
```

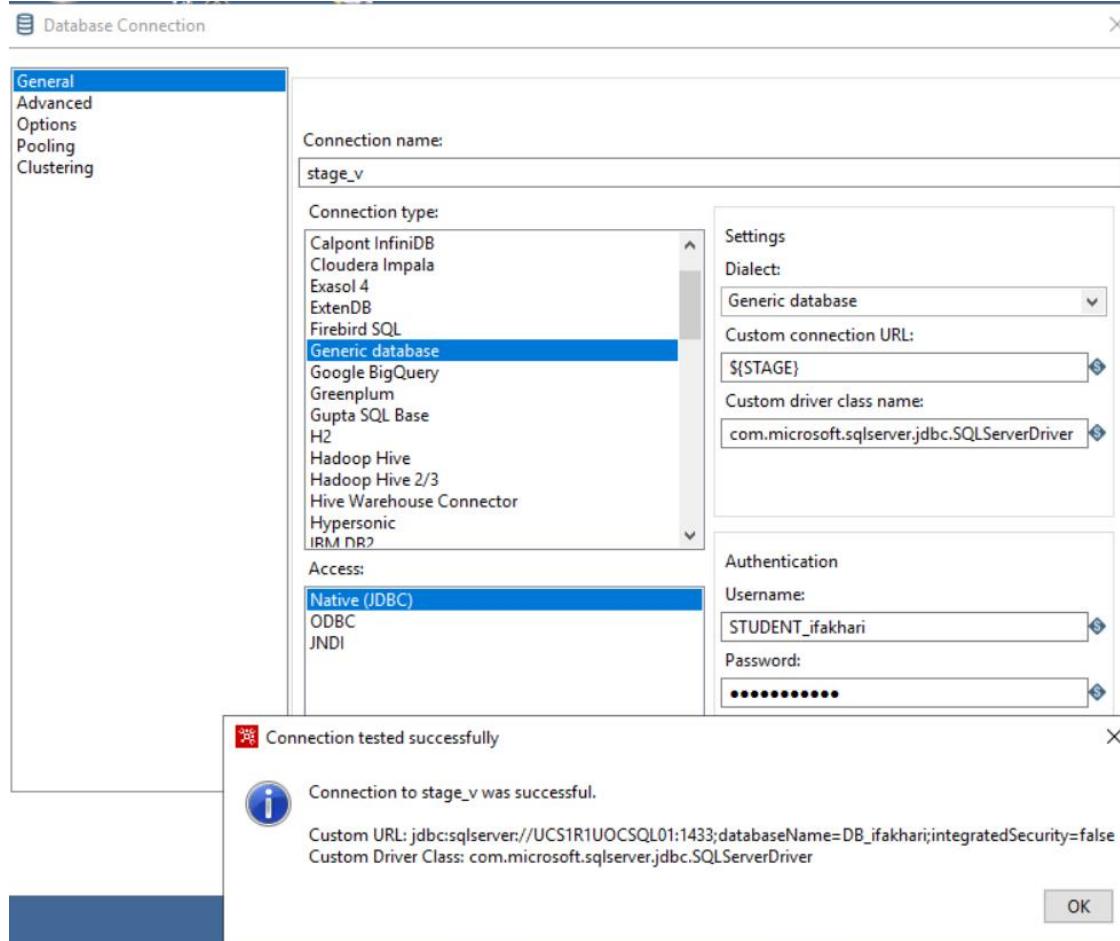
```
ALTER TABLE[dbo].[FACT_FREE_THROWS] CHECK CONSTRAINT[FK_Jugada]  
GO
```

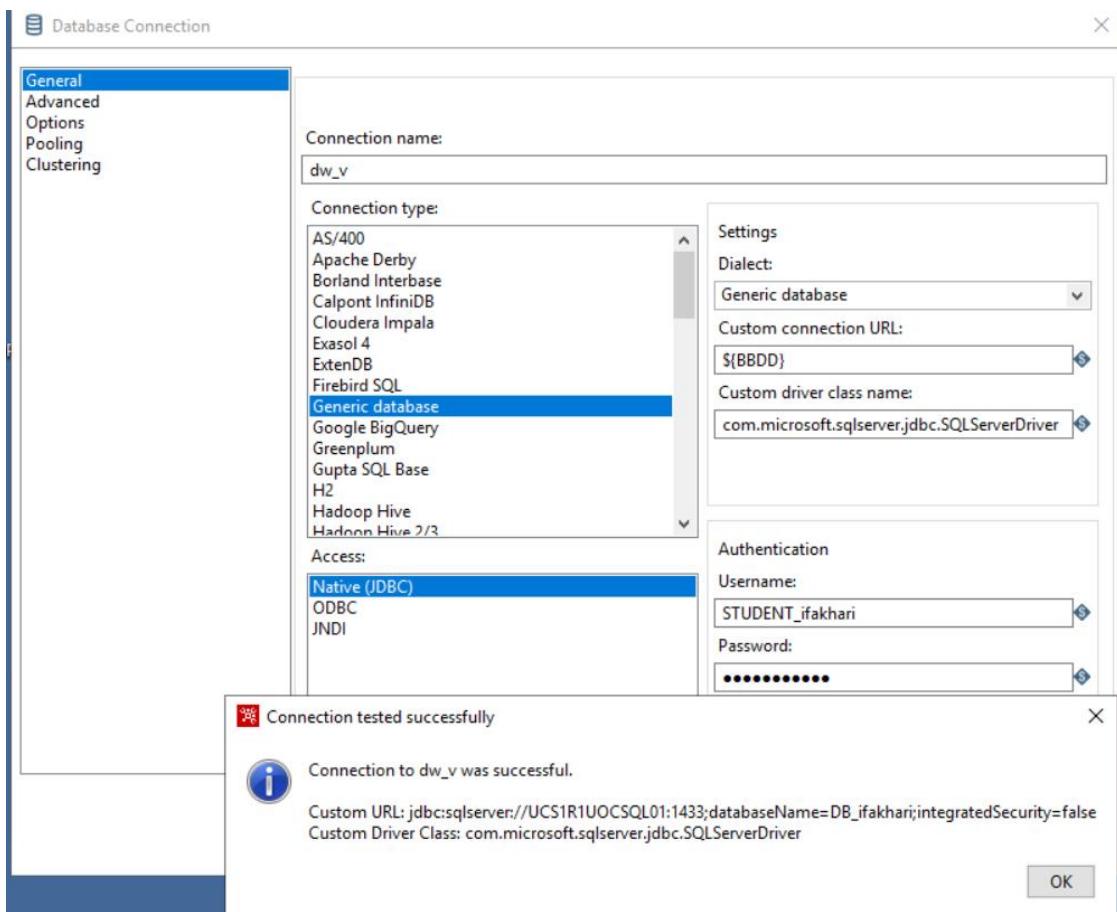
```
ALTER TABLE[dbo].[FACT_FREE_THROWS] WITH CHECK ADD CONSTRAINT  
[FK_Minuto] FOREIGN KEY([id_Minuto])  
REFERENCES[dbo].[DIM_Minutos]([pk_minutoSegundo])  
GO
```

```
ALTER TABLE[dbo].[FACT_FREE_THROWS] CHECK CONSTRAINT[FK_Minuto]  
GO
```

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

a) Variables de entorno





- b) Conexión a la base de datos SQL Server
- c) Bloque IN

Transformaciones IN_STATS

STG_WNBA_Seasons_Stats_2005_2017

La transformación IN_WNBA_Seasons_Stats_2005_2017 contiene las siguientes transformaciones: lectura del fichero xls, desnormalización de las filas, campo constante correspondiente a la liga WNBA, generación de ID para las jugadoras de la WNBA, cálculo del estadístico EFF para la liga WNBA, se añade campo year2 para crear la temporada correspondiente, concatenación de columna season y 'year2' creada previamente para generar temporada, filtro para la columna 'age' de la jugadora , corrección estableciendo instancia vacía, incorporación de los datos, metadatado de los campos para establecer la tipología.

El segundo bloque de los pasos en esta transformación consiste en obtener un campo activo para indicar si las jugadoras se encuentran inactivas o de lo contrario. Para ello se parte de los datos de entrada de STG_WNBA_Players que cuenta con campo 'player' y 'active', de manera que se indica mediante el valor 1 cuando si activas.

Para comprobar si las jugadoras están en activo, usamos los campos de la tabla cargada. Fuzzy match mide la distancia de edición entre los campos de nombres de jugadores. Se emplea el algoritmo Jaro Winkler para calcular la cercanía de las cadenas que contiene los datos. Hemos establecido un valor mínimo de 0.9 para que sea lo más aproximado posible, pudiendo dejar entrada a errores tan solo de algún carácter. Para aquellos nombres que coinciden devuelve el nombre y el valor del algoritmo. Si no cumple los requisitos el valor devuelto es null.

Tras los resultados, usamos value mapper para indicar con 0 las jugadoras que no están en activo.

Finalmente, cargamos los datos a la tabla intermedia STG_WNBA_Seasons_Stats_2005_2017

En las siguientes capturas se muestra el proceso:

- Carga de los datos

#	Name	Type	Length	Precision	Trim type	Repeat	Format	Currency	Decimal	Grouping
1	ID	String	-1	-1	none	N				
2	Data	String	-1	-1	none	N				
3	Type	String	-1	-1	none	N				
4	Stat	String	-1	-1	none	N				

- Desnormalizar las filas. De esta manera se indica la agrupación por ID, los campos que queremos obtener y en qué campo inicial se encuentra al cargar el fichero.

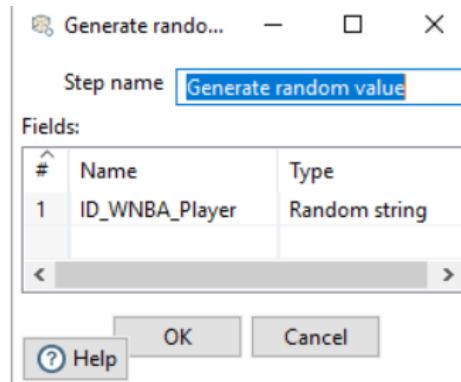
#	Group field
1	ID

#	Target fieldname	Value fieldname	Key value	Type	Format	Length	Precision	Currency	Deci
1	season	Data	SEASON	Integer					
2	player	Data	PLAYER	String					
3	GP	Data	GP	Number					
4	MIN	Data	MIN	Number					
5	FGM	Data	FGM	Number					
6	FGA	Data	FGA	Number					
7	FG%	Data	FG%	Number					
8	3PM	Data	3PM	Number					
9	3PA	Data	3PA	Number					
10	3P%	Data	3P%	Number					
11	FTM	Data	FTM	Number					
12	FTA	Data	FTA	Number					
13	FT%	Data	FT%	Number					
14	OREB	Data	OREB	Number					
15	DREB	Data	DREB	Number					
16	REB	Data	REB	Number					
17	AST	Data	AST	Number					
18	STL	Data	STL	Number					
19	BLK	Data	BLK	Number					
20	TOV	Data	TOV	Number					
21	PF	Data	PF	Number					

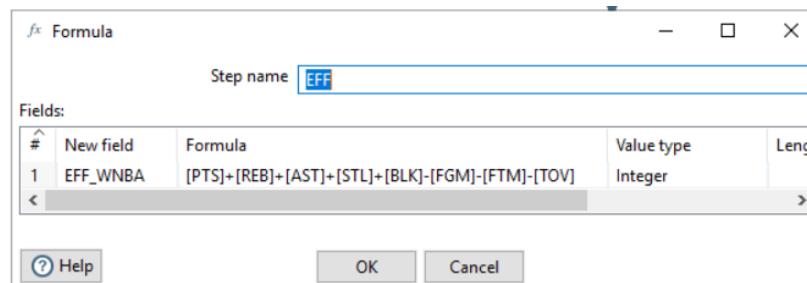
- Crear un campo league con valores constantes tipo string 'WNBA'

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	league	String							WNBA	N

- Se genera el id para las jugadoras de la WNBA



- Cálculo del estadístico EFF para la liga WNBA según los campos necesarios.



- Se obtiene el campo year2, que trata de calcular un año adicional, de manera que podamos recrear un campo temporada por concatenación.

Two overlapping configuration windows are shown:

Top Window (Formula Step):

- Step name: 'Formula'
- Fields table:

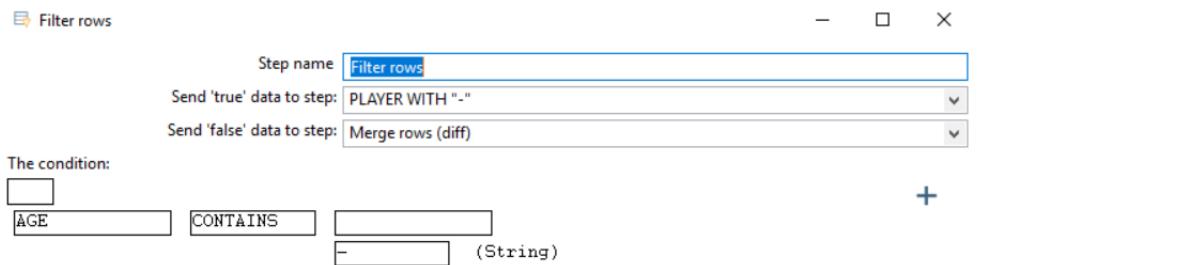
#	New field	Formula	Value type	Length	Precision	Replace value
1	year2	[season]+1	Integer			

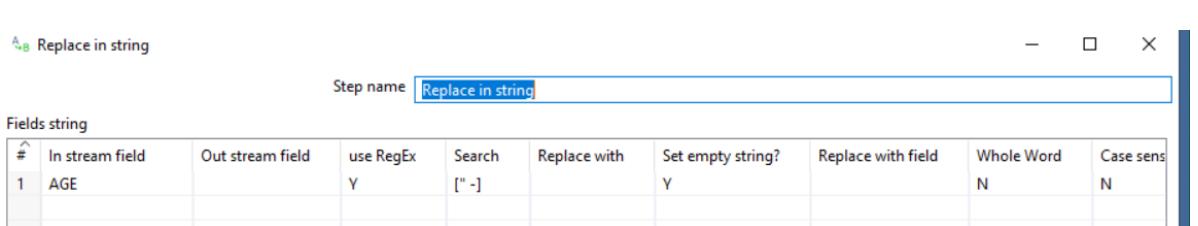
Bottom Window (Concat fields Step):

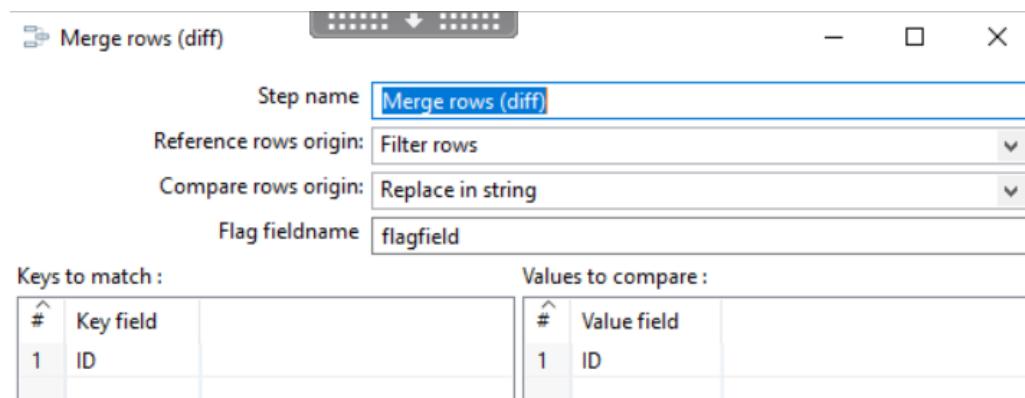
- Step name: 'Concat fields'
- Target Field Name: 'temporada'
- Length of Target Field: '0'
- Separator: '-' (with 'Insert TAB' button)
- Enclosure: '' (empty quotes)
- Fields table:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type
1	season	None							none
2	year2	None							none

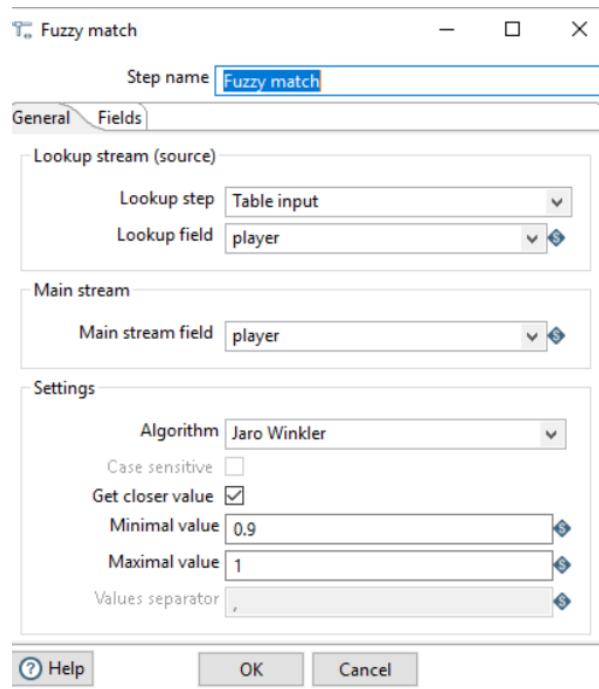
- Los pasos siguientes tratan de encontrar en el campo age el error, corregirlo estableciendo en blanco e incorporando los datos de nuevo.







- Los siguientes pasos realizan las comprobaciones para obtener si una jugada está en activo o no, como se ha explicado previamente.



Para aquellas jugadoras que no están en activo, y por lo tanto tienen valor en null se reemplaza con valor 0.

Value mapper

Step name: 0_inactive

Fieldname to use: active

Target field name (empty=overwrite):

Default upon non-matching:

Field values:

#	Source value	Target value
1		0

- Realizamos la carga a la tabla intermedia.

Table output

Step name: Table output

Connection: stage_v

Target schema:

Target table: STG_WNBA_Season_Stats_2005_2017

Commit size: 1000

Truncate table:

Ignore insert errors:

Specify database fields:

Main options | Database fields

Fields to insert:

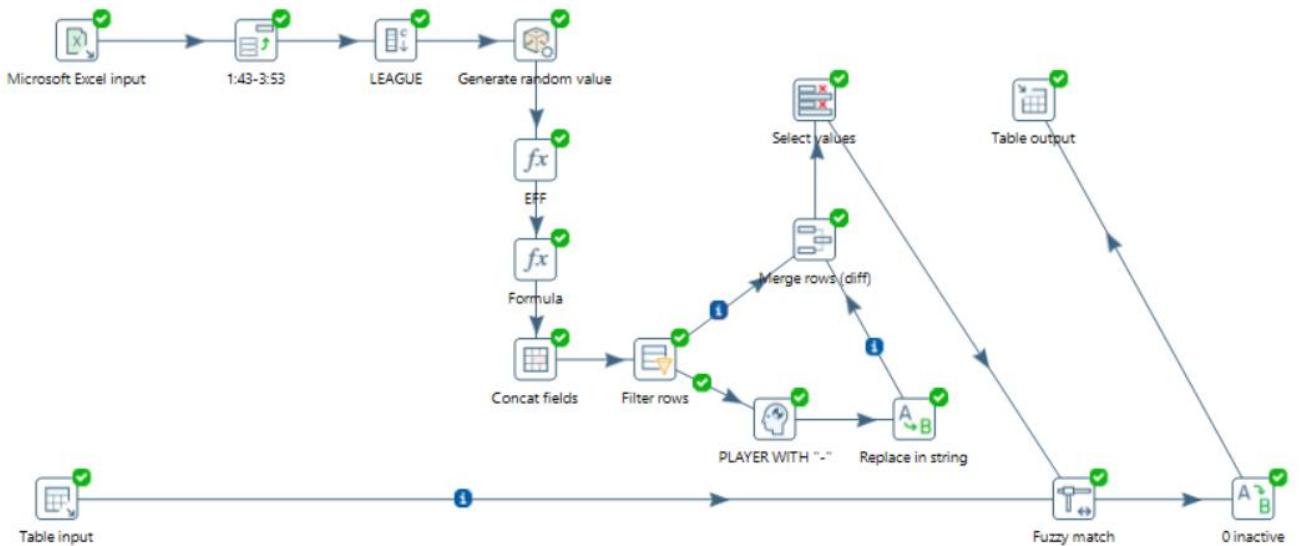
#	Table field	Stream field
1	season	season
2	MIN	MIN
3	FGM	FGM
4	FGA	FGA
5	FG%	FG%
6	_3PM	3PM
7	_3PA	3PA
8	_3P%	3P%
9	FTM	FTM
10	FTA	FTA
11	FT%	FT%
12	OREB	OREB
13	DREB	DREB
14	REB	REB
15	AST	AST
16	STL	STL
17	BLK	BLK
18	TOV	TOV
19	PF	PF
20	PTS	PTS

Get fields

Enter field mapping

Help | OK | Cancel | SQL

La transformación completa es la siguiente:



El resultado de la ejecución es el siguiente:

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	Table input	0	0	963	963	0	0	0	0	Finished
2	Microsoft Excel input	0	0	64635	64635	0	0	0	0	Finished
3	1:43-3:53	0	64635	2085	0	0	0	0	0	Finished
4	LEAGUE	0	2085	2085	0	0	0	0	0	Finished
5	Generate random value	0	2085	2085	0	0	0	0	0	Finished
6	EFF	0	2085	2085	0	0	0	0	0	Finished
7	Formula	0	2085	2085	0	0	0	0	0	Finished
8	Concat fields	0	2085	2085	0	0	0	0	0	Finished
9	Filter rows	0	2085	2085	0	0	0	0	0	Finished
10	PLAYER WITH "-"	0	1	1	0	0	0	0	0	Finished
11	Replace in string	0	1	1	0	0	0	0	0	Finished
12	Merge rows (diff)	0	2085	2085	0	0	0	0	0	Finished
13	Select values	0	2085	2085	0	0	0	0	0	Finished
14	Fuzzy match	0	3048	2085	0	0	0	0	0	Finished
15	0 inactive	0	2085	2085	0	0	0	0	0	Finished
16	Table output	0	2085	2085	0	2085	0	0	0	Finished

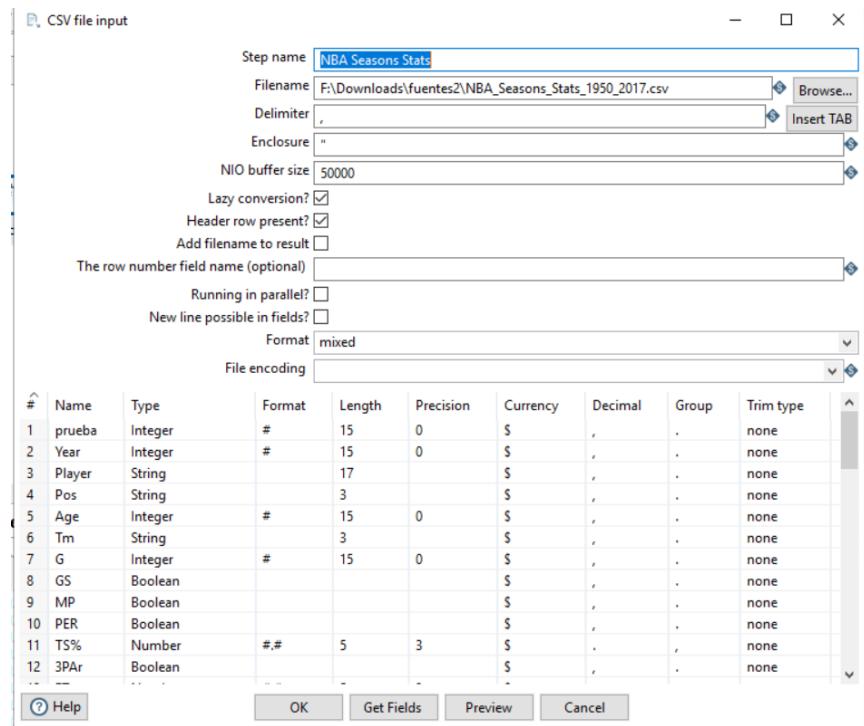
STG_NBA_Seasons_Stats_1950_2017

Previo a la transformación se realiza una modificación manual en el archivo csv, añadiendo un campo, de manera cuando se lea en la transformación no desplace los demás campos obviando la columna en blanco.

La transformación extrae los datos del archivo NBA_Seasons_Stats_1950_2017.csv y carga los datos a la tabla intermedia STG_NBA_Seasons_Stats_1950_2017.

Esta transformación contiene los siguientes pasos: lectura del fichero csv, obtención de los campos de interés, cambio de los campos string a mayúsculas, obtención del estadístico EEF de la liga NBA, obtención de campo 'year2' para crear la temporada y carga a la tabla intermedia.

- Lectura del fichero



- Selección de los campos de nuestro interés

#	Fieldname	Rename to	Length	Precision	
1	Year				
2	Player				
3	Pos				
4	Age				
5	Tm				
6	G				
7	MP				
8	ORB%				
9	DRB%				
10	FG				
11	FGA				
12	FG%				
13	3PA				
14	3P%				
15	2PA				
16	2P%				
17	eFG%				
18	FTA				
19	FT%				
20	ORB				
21	DRB				
22	AST				
23	STL				
24	BLK				
25	TOV				
26	PF				
27	PTS				
28	3P				

Include unspecified fields, ordered by name

- Cambio a mayúsculas

ABD 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	Player			upper	none			N	None	none
2	Pos		none	upper	none			N	None	none
3	Tm		none	upper	none			N	None	none

- Obtención del estadístico EFF

[fx Formula](#)

Step name: EFF

Fields:

#	New field	Formula	Value type	Length	Prec
1	EFF	[PTS]+[TRB]+[AST]+[STL]+[BLK]-[FG]-[FT]	Number		

Buttons: [Help](#) [OK](#) [Cancel](#)

- Obtención del campo 'year2' y concatenación para crear la temporada correspondiente.

[fx Formula](#)

Step name: Year2

Fields:

#	New field	Formula	Value type	Length	Precision	Replace value
1	Year2	[Year]+1	Integer			

Buttons: [Help](#) [OK](#) [Cancel](#)

Concat fields

Step name: season

Target Field Name: season

Length of Target Field: 0

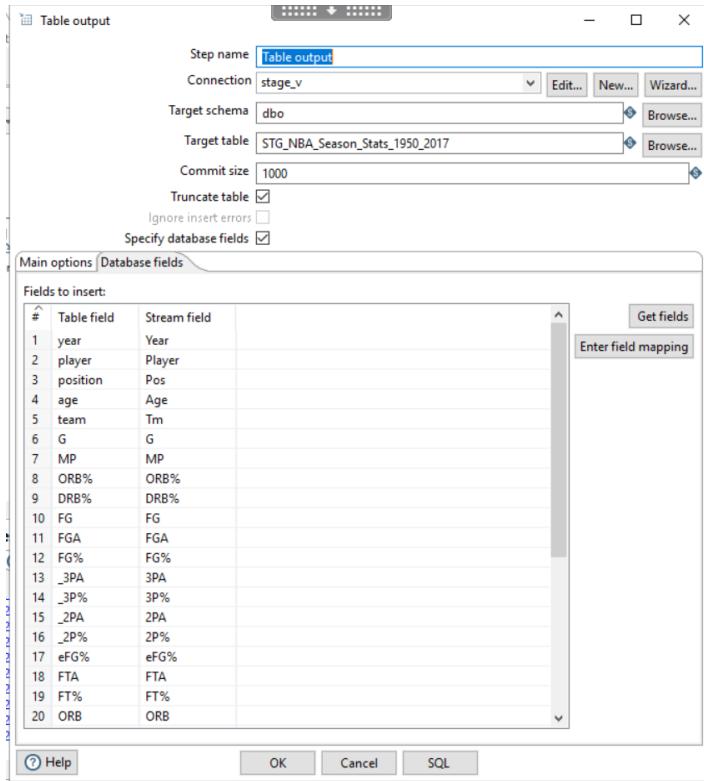
Separator: -

Enclosure: "

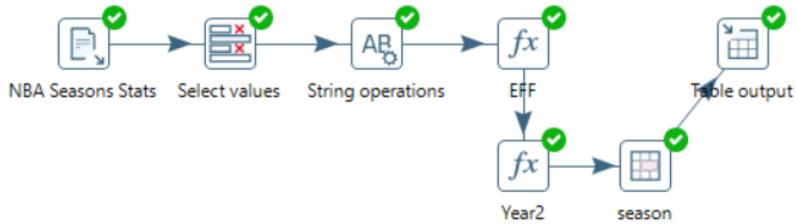
Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	Year	None							none	
2	Year2	None							none	

- Carga final a la tabla intermedia.



La transformación es la siguiente:



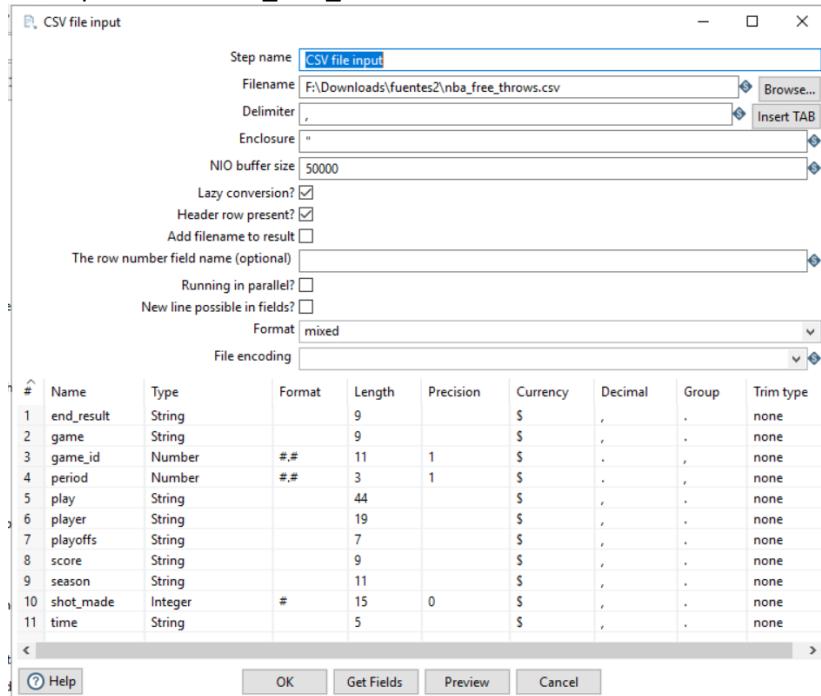
El resultado de la ejecución de la transformación completa es el siguiente:

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed
1	NBA Seasons Stats	0	0	24691	24692	0	0	0	0	Finished	0.7s	3
2	Select values	0	24691	24691	0	0	0	0	0	Finished	0.9s	2
3	String operations	0	24691	24691	0	0	0	0	0	Finished	1.0s	2
4	EFF	0	24691	24691	0	0	0	0	0	Finished	1.1s	2
5	Year2	0	24691	24691	0	0	0	0	0	Finished	1.1s	2
6	season	0	24691	24691	0	0	0	0	0	Finished	1.6s	1
7	Table output	0	24691	24691	0	24691	0	0	0	Finished	2.2s	1

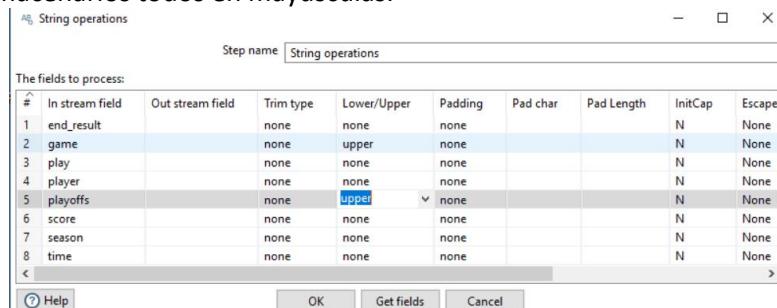
Transformación IN_THROWS

Esta transformación contiene los siguientes pasos: lectura del fichero csv, cambio a mayúsculas, obtención del tipo de juego para cada jugador, generar un código de juego y carga a la tabla intermedia STG_NBA_free_Throws.

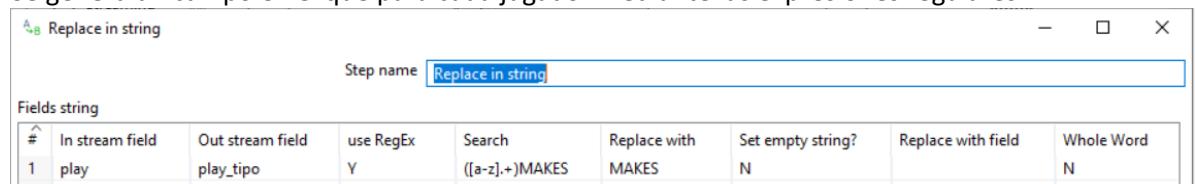
- Paso CSV file input lectura nba_free_throws.csv.



- Paso operaciones con cadenas para normalizar los campos de tipo string en la base de datos y almacenarlos todos en mayúsculas.



- Se genera un campo en el que para cada jugador mediante las expresiones regulares.



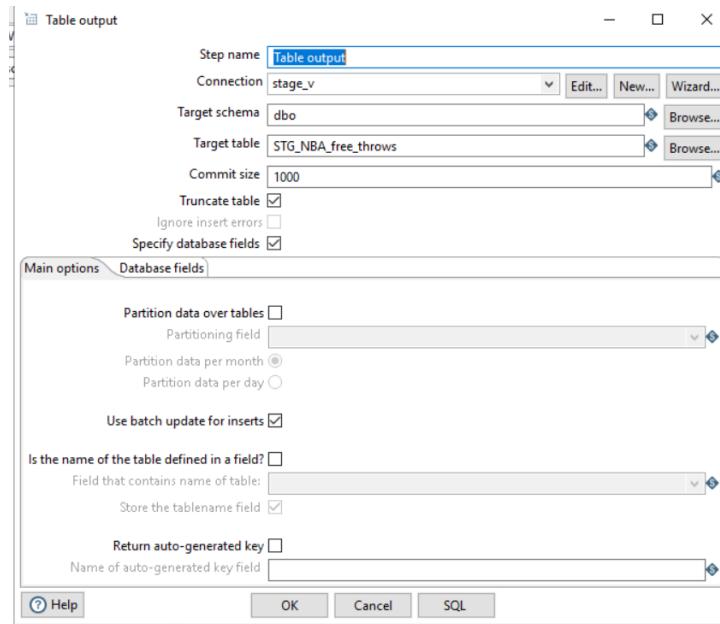
Replace in string

Step name: Replace in string 3

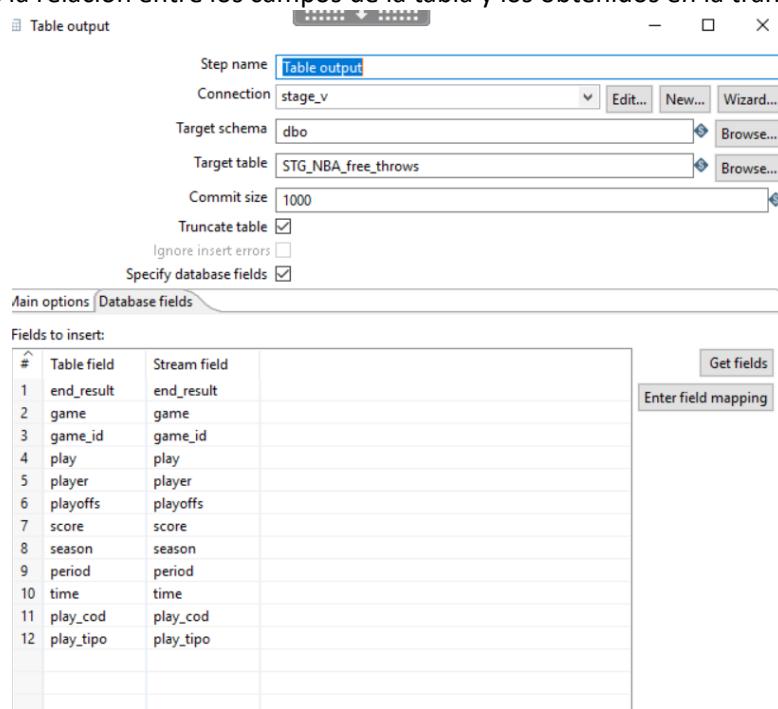
Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word
1	play_tipo		Y	([a-z].+)MISSES	MISSES	N		N

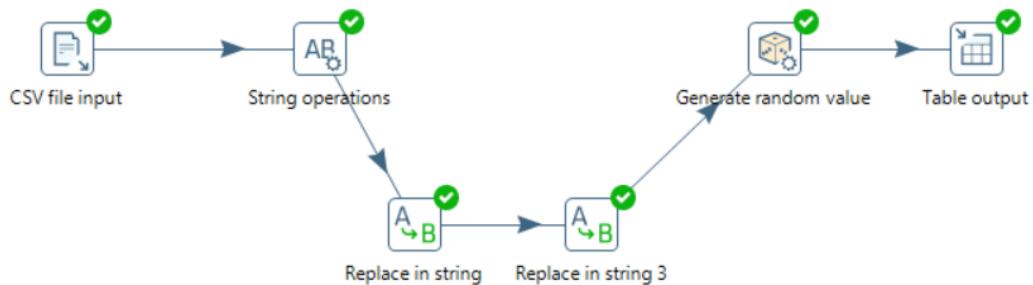
- Paso salida de tabla carga STG_NBA_free_throws. Cargamos los datos de la fuente origen a la tabla intermedia.



- Indicamos la relación entre los campos de la tabla y los obtenidos en la transformación



La transformación completa es la siguiente:



El resultado de la ejecución de la transformación completa es el siguiente:

Execution Results									
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Error:
1	CSV file input	0	0	618019	618020	0	0	0	0
2	String operations	0	618019	618019	0	0	0	0	0
3	Replace in string	0	618019	618019	0	0	0	0	0
4	Replace in string 3	0	618019	618019	0	0	0	0	0
5	Generate random value	0	618019	618019	0	0	0	0	0
6	Table output	0	618019	618019	0	618019	0	0	0

Como se observa en las métricas se cargan los 618019 registros del fichero csv a la tabla intermedia.

Transformaciones IN_TEAMS

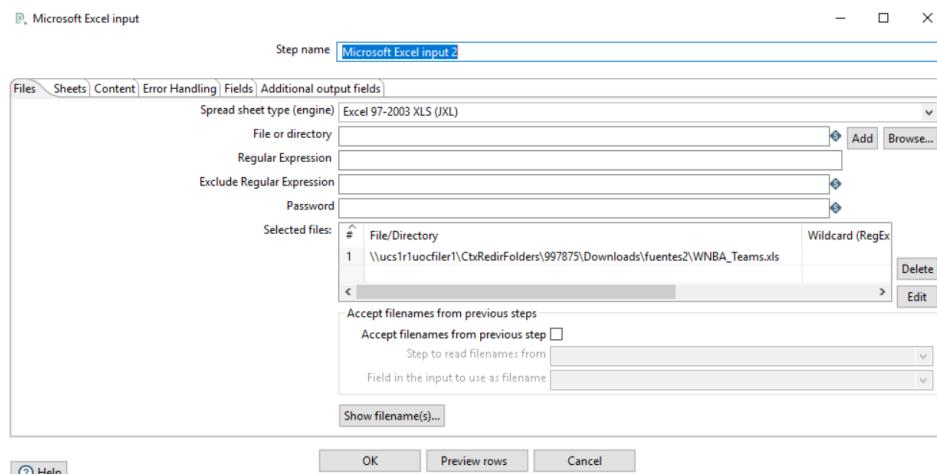
STG_WNBA_Teams (IN_WNBA_Teams)

Previo a la transformación manipulamos el archivo xls para configurar el campo correspondiente a la Conferencia de los equipos. El resto de acciones se realizan mediante la transformación que detallaremos.

3	Equipo	Pts	J	G	P	p.	c.	dif	Ciudad, Estado
4	Conferencia Este	Indiana Fever	6	3	3	0	216	189	27 Indianapolis, Indiana
5	Conferencia Este	Connecticut Sun	5	3	2	1	265	216	49 Uncasville, connecticut
6	Conferencia Este	Atlanta Dream	5	3	2	1	238	221	17 Atlanta, Georgia
7	Conferencia Este	Washington Mystics	3	2	1	1	154	150	4 Washington D.C., Washington
8	Conferencia Este	New York Liberty	3	3	0	3	232	290	-58 White Plains, New York
9	Conferencia Este	Chicago Sky	2	2	0	2	123	145	-22 Chicago, Illinois
10									
11									
12									
13									
14	Equipo	Pts	J	G	P	p.	c.	dif	Ciudad, Estado
15	Conferencia Oeste	Minnesota Lynx	4	2	2	0	165	154	11 Minneapolis, Minnesota
16	Conferencia Oeste	Phoenix Mercury	4	2	2	0	169	159	10 Phoenik, Arizona
17	Conferencia Oeste	Dallas Wings	4	3	1	2	196	220	-24 Arlington, Texas
18	Conferencia Oeste	Los Angeles Sparks	3	2	1	1	167	167	0 Los Angeles, California
19	Conferencia Oeste	Seattle Storm	2	2	0	2	169	179	-10 Seattle, Washington
20	Conferencia Oeste	Las Vegas Aces	1	1	0	1	75	79	-4 Paradises, Nevada

La transformación contiene los siguientes pasos: lectura del fichero xls, filtrado de filas, cambio a mayúsculas de campos string, obtención del campo ciudad y Estado por separado y carga a la tabla intermedia STG_WNBA_Teams.

- Para la lectura usamos el tipo Microsoft Excel input. Añadimos el fichero.



En la pestaña Sheets, indicamos el nombre de la hoja de cálculo que queremos procesar.

List of sheets to read			
#	Sheet name	Start row	Start column
1	Teams	0	0

Para recuperar los campos, lo realizamos manualmente en el orden que están en el fichero de las celdas verticales, ya por defecto no lo realiza con precisión en este caso concreto.

Additional output fields										
#	Name	Type	Length	Precision	Trim type	Repeat	Format	Currency	Decimal	Grouping
1	Conferencia	String	-1	-1	none	N				
2	Equipo	String	-1	-1	none	N				
3	Pts	None	-1	-1	none	N				
4	J	None	-1	-1	none	N				
5	G	None	-1	-1	none	N				
6	P	None	-1	-1	none	N				
7	p_min	None	-1	-1	none	N				
8	c	None	-1	-1	none	N				
9	dif	None	-1	-1	none	N				
10	Ciudad,Estado	String	-1	-1	none	N				

- Filtramos las líneas que repiten el encabezado de los campos

The condition:

NOT

Conferencia IS NULL

#	Conferencia	Equipo	Pts	J	G	P	p_min	c	dif	Ciudad,Estado
1	<null>	Equipo	Pts	J	G	P	p.	c.	dif	Ciudad, Estado
2	<null>	Equipo	Pts	J	G	P	p.	c.	dif	Ciudad, Estado

- Cambios los campos string a mayúsculas

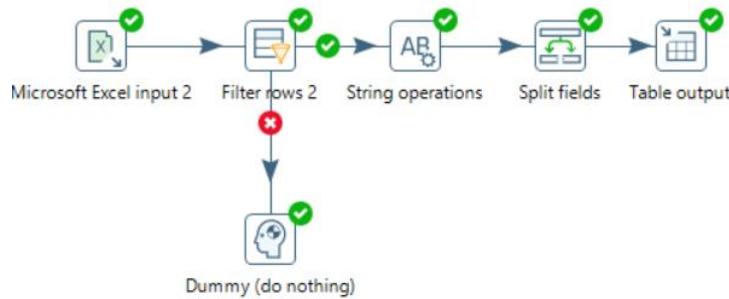
#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits	Remove
1	Conferencia		none	upper	none			N	None	none	none
2	Equipo		none	upper	none			N	None	none	none
3	Pts		none	none	none			N	None	none	none
4	J		none	none	none			N	None	none	none
5	G		none	none	none			N	None	none	none
6	P		none	none	none			N	None	none	none
7	p_min		none	none	none			N	None	none	none
8	c		none	none	none			N	None	none	none
9	dif		none	none	none			N	None	none	none
10	Ciudad,Estado		none	upper	none			N	None	none	none

- Obtenemos los campos Ciudad y Estado por separado, pues para realizar operaciones para enlazar dimensiones más adelante será sencillo. Separamos los campos mediante el delimitador coma (",")

#	New field	ID	Remove ID?	Type	Length	Precision	Format	Group	Decimal	Currency	Nullif	Default	Trim type
1	Ciudad	N	String										none
2	Estado	N	String										none

- Finalmente cargamos los datos en la tabla intemedia STG_WNBA_Teams

La transformación completa es la siguiente:



El resultado de la ejecución de la transformación completa es el siguiente:

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	Microsoft Excel input 2	0	0	14	14	0	0	0	0	Finished
2	Filter rows 2	0	14	14	0	0	0	0	0	Finished
3	String operations	0	12	12	0	0	0	0	0	Finished
4	Split fields	0	12	12	0	0	0	0	0	Finished
5	Table output	0	12	12	0	12	0	0	0	Finished
6	Dummy (do nothing)	0	2	2	0	0	0	0	0	Finished

STG_NBA_Teams (IN_NBA_Teams)

Esta transformación permite la carga de los valores de los equipos de la NBA. El fichero NBA_Teams.xml se cargará en el área intermedia para completar la dimensión de los equipos, divisiones y conferencias.

La transformación consta de los siguientes pasos: lectura del fichero NBA_Teams.xml, filtrar filas, conversión campo FUNDADO a numérico, cambiar a mayúsculas y carga a la tabla intermedia STG_NBA_Teams

- Paso GET Data from XML lectura del fichero

Get data from XML

Step name: **Get data from XML**

File Content Fields Additional output fields

XML source from field

XML source is defined in a field? XML source is a filename? Read source as Uri?
getXML source from a field

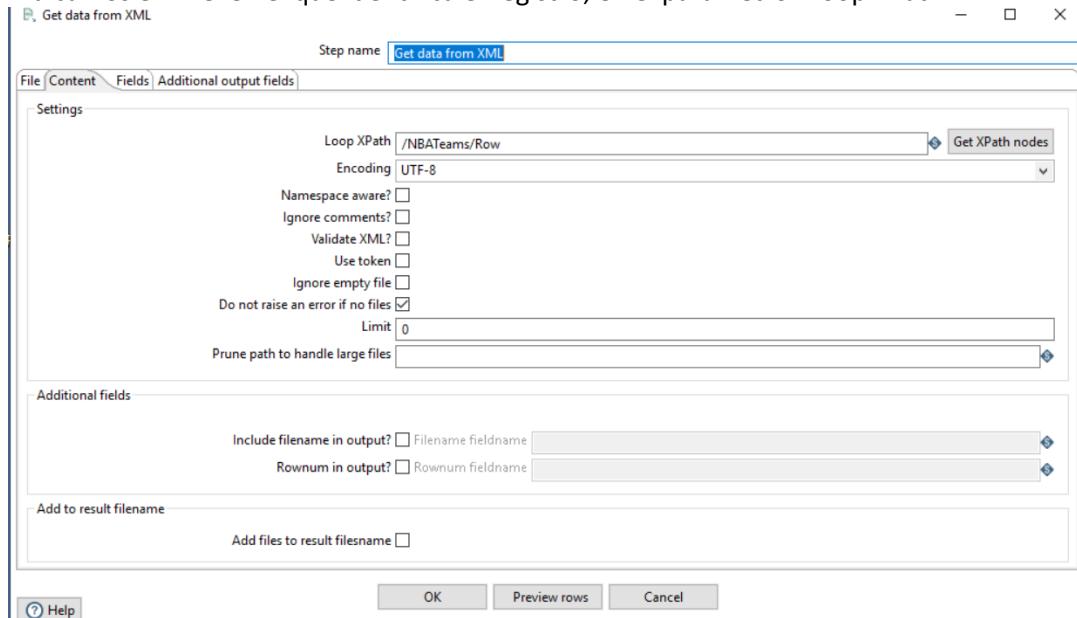
File or directory Add Browse
Regular Expression
Exclude Regular Expression

Selected files:	File/Directory	Wildcard (RegExp)	Exclude wildcard	Required
1	F:\Downloads\fuentes2\NBA_Teams.xml		N	<button>Delete</button> <button>Edit</button>

Show filename(s)...

OK Preview rows Cancel

Indicamos el nivel en el que identifica el registro, en el parámetro “Loop XPath”.



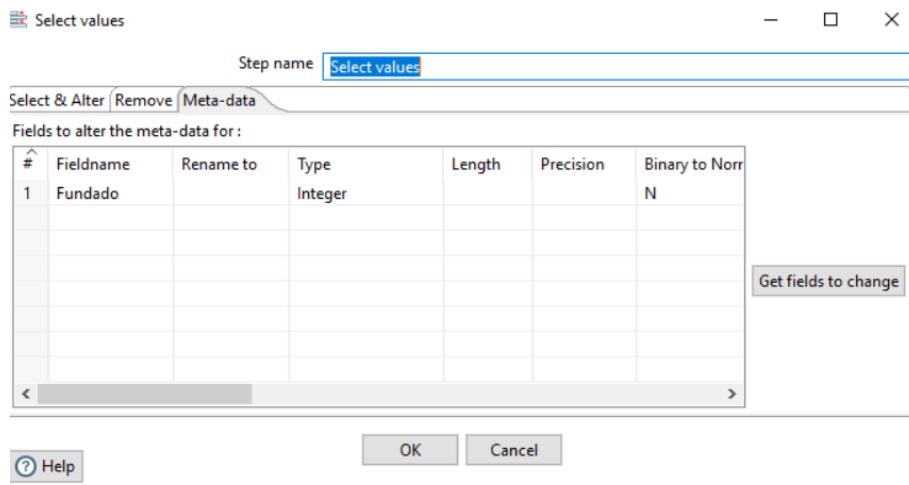
En la pestaña “Fields”, indicamos el XPATH correspondiente a cada campo dentro del registro definido.

#	Name	XPath	Element	Result type	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Conferencia	Data[1]	Node	Value of	None							none
2	Division	Data[2]	Node	Value of	String							none
3	Equipo	Data[3]	Node	Value of	String							none
4	Ciudad	Data[4]	Node	Value of	String							none
5	Estado	Data[5]	Node	Value of	String							none
6	Pabellon	Data[6]	Node	Value of	String							none
7	Fundado	Data[7]	Node	Value of	String							none
8	Patrocinio	Data[8]	Node	Value of	String							none

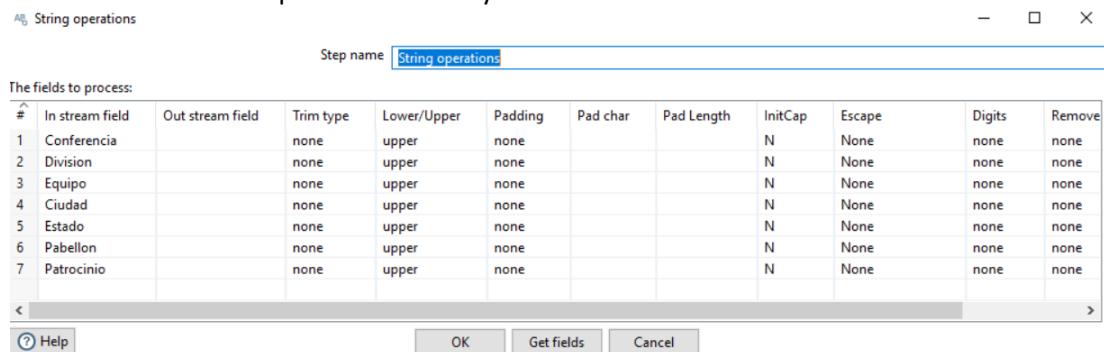
- Filtrar primera línea que repite los cabeceros de las columnas

#	Conferencia	Division	Equipo	Ciudad	Estado	Pabellon
1	Conferencia	División	Equipo	Ciudad	Estado	Pabellón
2	Conferencia Oeste	Noroeste	Denver Nuggets	Denver	CO	Pepsi Center
3	Conferencia Oeste	Noroeste	Minnesota Timberwolves	Minneapolis	MN	Target Center

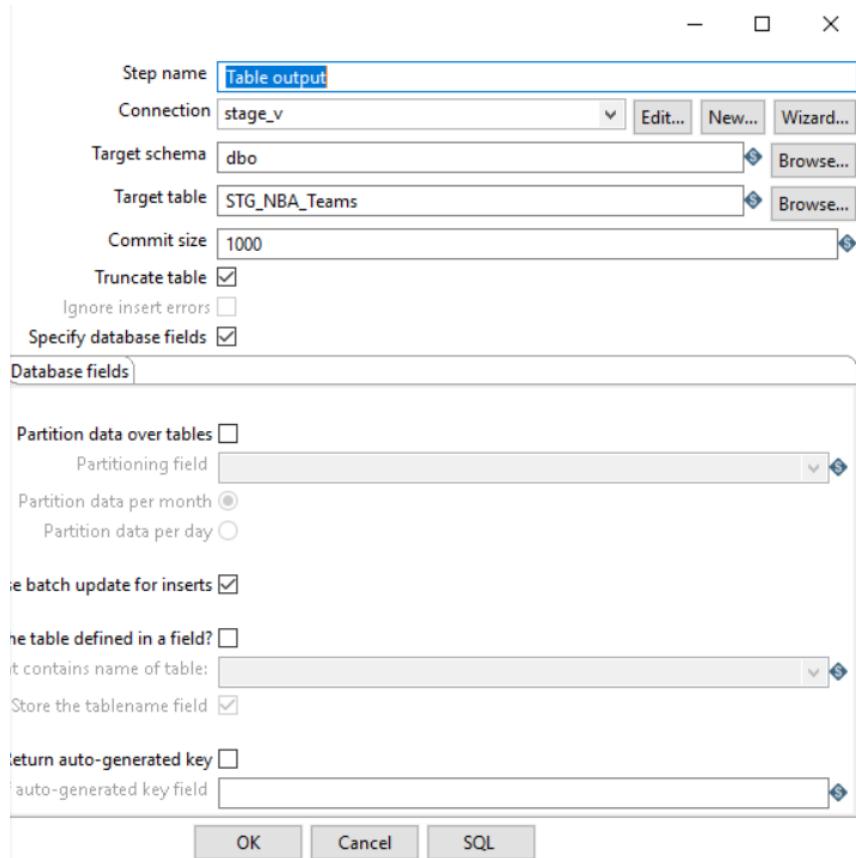
- Conversión del campo a tipo numérico



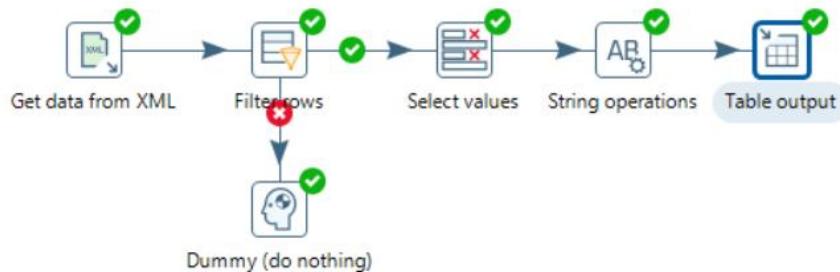
- Conversión de los campos STRING a mayúsculas



- Finalmente, cargamos los datos a la tabla intermedia STG_NBA_Teams



La transformación completa es la siguiente:



El resultado de la ejecución de la transformación completa es el siguiente:

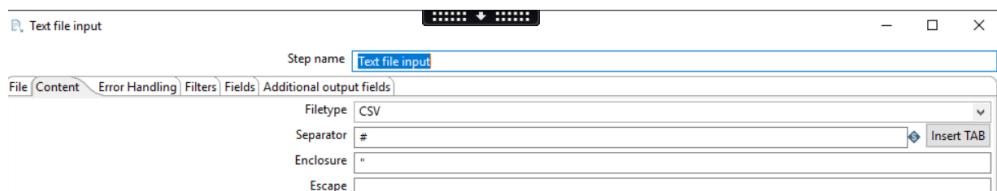
Execution Results										
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	Get data from XML	0	0	31	31	0	0	0	0	Finished
2	Filter rows	0	31	31	0	0	0	0	0	Finished
3	Select values	0	30	30	0	0	0	0	0	Finished
4	String operations	0	30	30	0	0	0	0	0	Finished
5	Table output	0	30	30	0	30	0	0	0	Finished
6	Dummy (do nothing)	0	1	1	0	0	0	0	0	Finished

STG_Teams_Codes (IN_Teams_Codes)

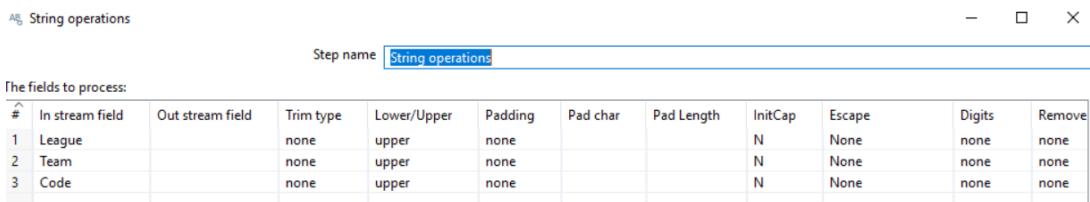
Extrae la información del fichero de texto TeamCodes.txt, que contiene información adicional de los equipos de la NBA y la WNBA y carga los datos en la tabla intermedia STG_Team_Codes.

Esta transformación contiene los siguientes pasos: lectura del fichero txt, cambio de campos a mayúsculas, identificar únicos equipos y carga a la tabla intermedia.

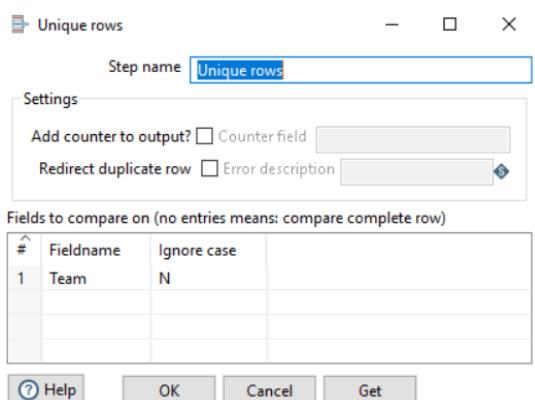
- Paso Text file input, indicando el tipo de separador (#)



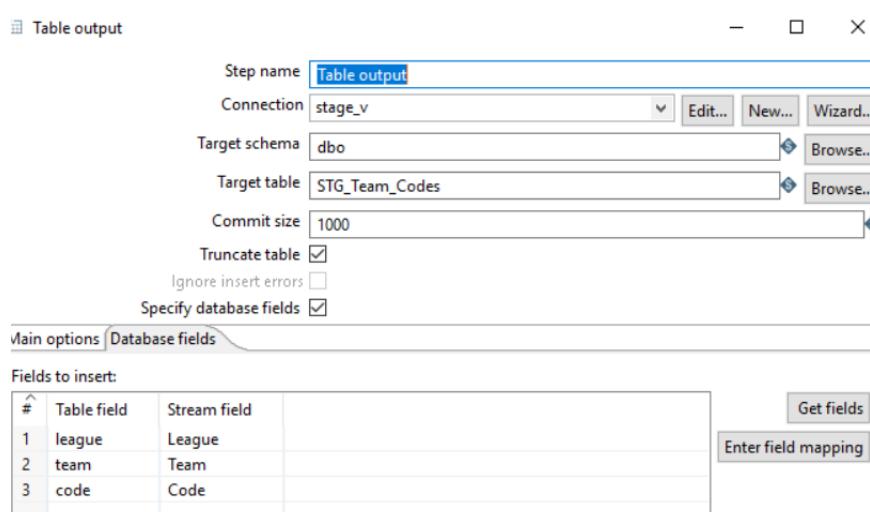
- Cambio a mayúscula de los campos



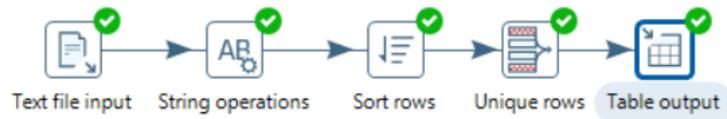
- Verificar que los equipos no están repetidos antes de cargar a la tabla intermedia



- Por último, cargamos los datos a la tabla intermedia STG_Team_Codes



La transformación completa es la siguiente:



El resultado de la ejecución de la transformación completa es el siguiente:

Execution Results

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time
1	Text file input	0	0	42	43	0	1	0	0	Finished	0
2	String operations	0	42	42	0	0	0	0	0	Finished	0
3	Sort rows	0	42	42	0	0	0	0	0	Finished	0
4	Unique rows	0	42	42	0	0	0	0	0	Finished	0
5	Table output	0	42	42	0	42	0	0	0	Finished	0

STG_Estados_Unidos (IN_EEUU)

Esta transformación permite la carga de los valores de los Estados Unidos. El fichero Estados_Unidos.json se cargará en el área intermedia para completar la dimensión de los diferentes estados miembros de los EE.UU.

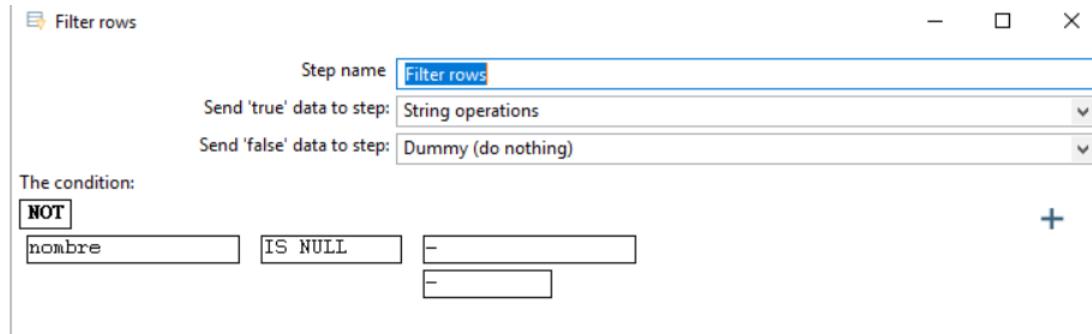
La transformación consta de los siguientes pasos: lectura del fichero Estados_Unidos.json, filtrar filas, cambiar a mayúsculas y carga a la tabla intermedia STG_Estados_Unidos.

- Paso JSON Input lectura Estados_Unidos.json. La lectura de los campos se realiza mediante el path.

The screenshot shows the configuration of a JSON input step. The step name is "JSON input". The "Fields" tab is selected, displaying a list of fields with their corresponding paths and types. The fields are:

#	Name	Path	Type	Format	Length	Precision	Currency	Decim
1	nombre	\$.[*].[Estado]	String					
2	Nombre oficial	\$.[*].[Nombre oficial]	String					
3	superficie	\$.[*].[Superficie (km2)]	Number					
4	codigo	\$.[*].[Abrev.]	String					
5	poblacion	\$.[*].[Población (2010)]	Integer					
6	capital	\$.[*].[Capital]	String					
7	densidadPoblacion	\$.[*].[Densidad de Población (hab/km2)]	Number					

- Filtrar aquellos Estados, cuyos campos están nulos y que además el su nombre oficial no figura en inglés.



- Paso operaciones con cadenas para normalizar el campo de nombre, nombre oficial, código y capital de tipo string en la base de datos y almacenarlos todos en mayúsculas en la tabla intermedia.

String operations

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits	Remove Special
1	nombre		none	upper	none			N	None	none	none
2	Nombre oficial		none	upper	none			N	None	none	none
3	codigo		none	upper	none			N	None	none	none
4	capital		none	upper	none			N	None	none	none

- Paso salida de tabla carga IN_EEUU. Cargamos los datos de la fuente origen JSON a la tabla intermedia.

Table output

Step name: Table output

Connection: stage_v

Target schema: dbo

Target table: STG_Estados_Unidos

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:

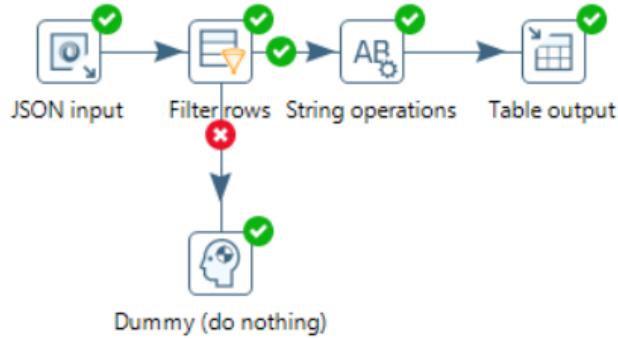
Store the tablename field:

Return auto-generated key:

Name of auto-generated key field:

OK Cancel SQL

La transformación completa es la siguiente:



El resultado de la ejecución de la transformación completa es el siguiente:

Execution Results										
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	JSON input	0	0	53	53	0	0	0	0	Finished
2	Filter rows	0	53	53	0	0	0	0	0	Finished
3	Dummy (do nothing)	0	3	3	0	0	0	0	0	Finished
4	String operations	0	50	50	0	0	0	0	0	Finished
5	Table output	0	50	50	0	50	0	0	0	Finished

Transformación IN_PLAYERS

Por un lado, realizaremos dos transformaciones que nos permitirán la carga de los datos sobre la información personal de los jugadores. En la primera, el fichero NBA_Player_List.xls cargará los datos de los jugadores en activo para realizar la comprobación con el segundo fichero, NBA_player_data.json. Este último se cargará en otra área intermedia distinta para luego completar la dimensión de los jugadores.

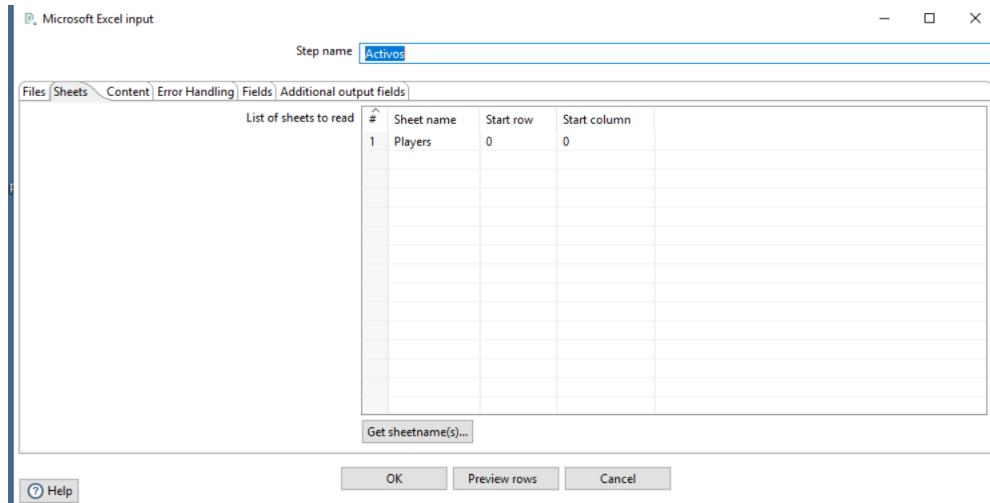
Por otro lado, realizamos una transformación que nos permitirá la carga de los datos sobre las jugadoras de la WNBA.

STG_NBAPlaylist_Active (IN_NBA_PlayersList)

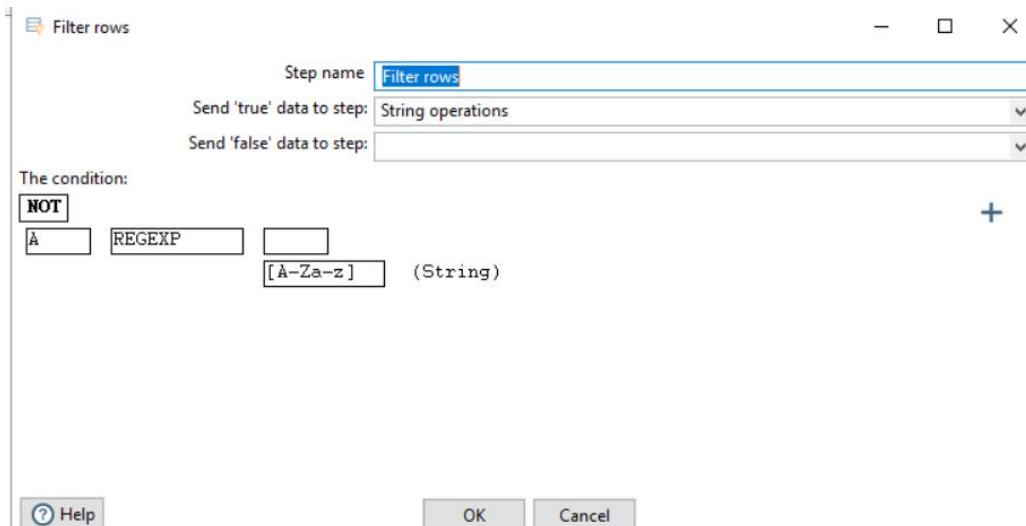
La transformación trata de cargar los nombres de los jugadores en la pestaña Player del fichero NBA_Player_List.xls, que contiene el listado de jugadores de la NBA en activo. Solo escogemos estos para posteriormente realizar la comparación con los datos de nombres del fichero NBA_player_data.json, y crear un campo que indique si se encuentran activos o no.

La transformación IN_NBA_PlayerList contiene los siguientes pasos: lectura del fichero, eliminar las instancias que corresponden a las letras del abecedario, cambio de campo a mayúsculas, división del campo por apellido y nombre, concatenación de nombre completo (nombre espacio apellido) y carga a la tabla intermedia.

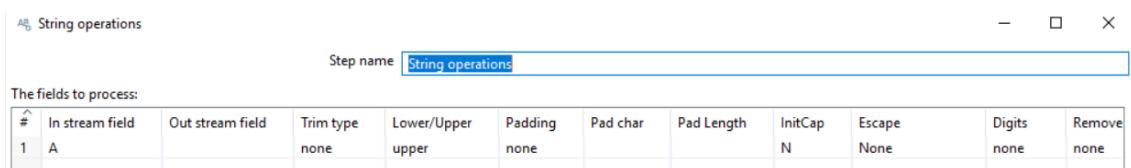
- Como se trata de un fichero xls, utilizamos la entrada de tipo Microsoft Excel input. En la pestaña Sheets indicamos el nombre de la hoja de cálculo que nos interesa de jugadores en activo.



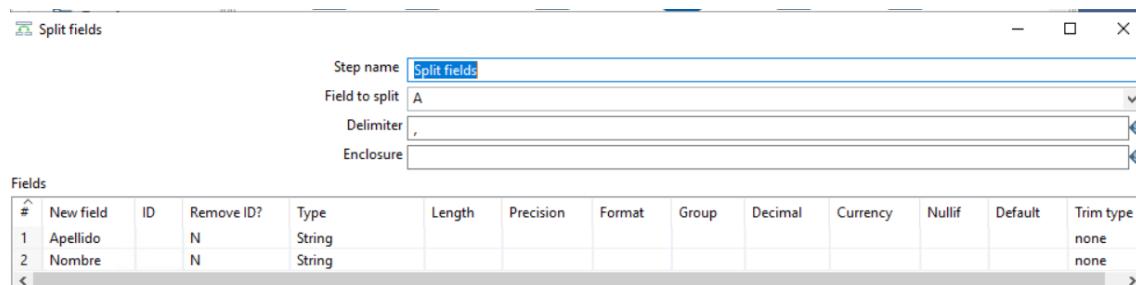
- Usamos regex para filtrar las instancias que corresponden a las letras del abecedario.



- Cambio a mayúsculas del campo



- División del campo por apellido nombre, indicando el delimitador (,)



- Concatenación para crear el nombre completo. En el separador usamos un espacio tabulador (individual) para poder realizar la comprobación con el nombre completo en el fichero posterior

Step name: Concat fields

Target Field Name: Jugadores_Activo

Length of Target Field: 0

Separator:

Enclosure:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	Nombre	String							none	
2	Apellido	String							none	

- Finalmente, realizamos la carga a la tabla intermedia

Step name: Table output

Connection: stage_v

Target schema: dbo

Target table: STG_NBAPlaylist_ACTIVE

Commit size: 1000

Truncate table:

Ignore insert errors:

Specify database fields:

Main options	Database fields						
Fields to insert:	<table border="1"> <thead> <tr> <th>#</th> <th>Table field</th> <th>Stream field</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>active</td> <td>Jugadores_A...</td> </tr> </tbody> </table>	#	Table field	Stream field	1	active	Jugadores_A...
#	Table field	Stream field					
1	active	Jugadores_A...					

La transformación completa es la siguiente:



El resultado de la ejecución es el siguiente:

Execution Results

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Activ
1	Activos	0	0	547	547	0	0	0	0	Finis
2	Filter rows	0	547	525	0	0	0	0	0	Finis
3	String operations	0	525	525	0	0	0	0	0	Finis
4	Split fields	0	525	525	0	0	0	0	0	Finis
5	Concat fields	0	525	525	0	0	0	0	0	Finis
6	Table output	0	525	525	0	525	0	0	0	Finis

STG_NBA_PLAYERS (IN_PLAYERS)

Esta transformación permite la carga de los datos de información personal de los jugadores.

La transformación contiene los siguientes pasos: lectura del fichero NBA_player_data.json, conversión del campo de fecha de nacimiento a tipo dato fecha, conversión a mayúsculas de los campos string, comprobar si el jugador está activo, generar código para cada jugador,

general la clave primaria de los jugadores, establecer el campo de liga como NBA, establecer el valor de la columna sexo como 1, generar el código de jugadores y carga a la tabla intermedia STG_NBA_PLAYERS.

- Paso JSON Input lectura de NBA_player_data.json. Indicamos el path para llegar a cada uno de los campos que hay que leer.

#	Name	Path	Type	Format	Length	Precision	Currency	Decimal	Group	Trim
1	birthDate_str	\$..birthDate_str	String							noni
2	birthPlace	\$..birthPlace	String							noni
3	State_Country	\$..State_Country	String							noni
4	career_AST	\$..career_AST	String							noni
5	career_FG%	\$..career_FG%	String							noni
6	career_FG3%	\$..career_FG3%	String							noni
7	career_G	\$..career_G	String							noni
8	career_FT%	\$..career_FT%	String							noni
9	career_PER	\$..career_PER	String							noni
10	career PTS	\$..career PTS	String							noni
11	career_TRB	\$..career_TRB	String							noni
12	career_WS	\$..career_WS	String							noni
13	career_eFG%	\$..career_eFG%	String							noni
14	college	\$..college	String							noni
15	height	\$..height	String							noni
16	highSchool	\$..highSchool	String							noni
17	name	\$..name	String							noni
18	position	\$..position	String							noni
19	shoots	\$..shoots	String							noni
20	weight	\$..weight	String							noni

- Formateamos el tipo de dato a fecha del campo de fecha de nacimiento.

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Format Lenient?
1	birthDate_str		Date			N	dd/MM/yyyy	Y

- Conversión a mayúscula de los campos string.

#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits	Remove
1	birthPlace		none	upper	none			N	None	none	none
2	State_Country		none	upper	none			N	None	none	none
3	career_AST		none	none	none			N	None	none	none
4	career_FG%		none	none	none			N	None	none	none
5	career_FG3%		none	none	none			N	None	none	none
6	career_G		none	none	none			N	None	none	none
7	career_FT%		none	none	none			N	None	none	none
8	career_PER		none	none	none			N	None	none	none
9	career PTS		none	none	none			N	None	none	none
10	career_TRB		none	none	none			N	None	none	none
11	career_WS		none	none	none			N	None	none	none
12	career_eFG%		none	none	none			N	None	none	none
13	college		none	upper	none			N	None	none	none
14	height		none	none	none			N	None	none	none
15	highSchool		none	upper	none			N	None	none	none
16	name		none	upper	none			N	None	none	none
17	position		none	upper	none			N	None	none	none
18	shoots		none	upper	none			N	None	none	none
19	weight		none	none	none			N	None	none	none

- Cargamos la tabla intermedia NBA_player_data.json

Table input

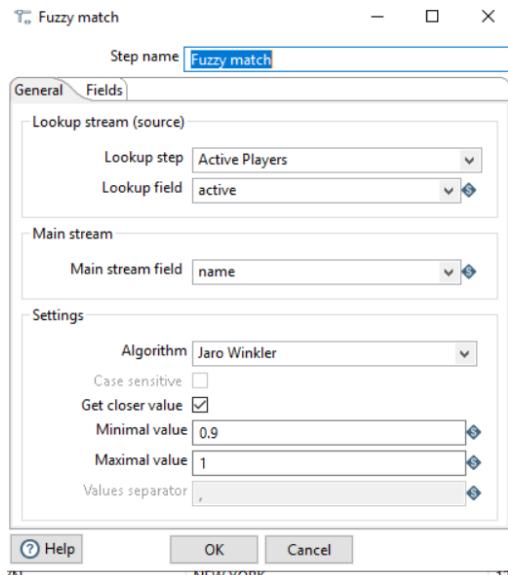
Step name: Active Players

Connection: stage_v

SQL:

```
SELECT
    active
FROM dbo.STG_NBAPlaylist_ACTIVE
```

- Para comprobar si los jugadores están en activo, usamos el campo de la tabla cargada. Fuzzy match mide la distancia de edición entre los campos de nombres de jugadores. Se emplea el algoritmo Jaro Winkler para calcular la cercanía de las cadenas que contiene los datos. Hemos establecido un valor mínimo de 0.9 para que sea lo más aproximado posible, pudiendo dejar entrada a errores tan solo de algún carácter. Para aquellos nombres que coinciden devuelve el nombre y el valor del algoritmo. Si no cumple los requisitos el valor devuelto es null.



Indicamos el nombre del campo de valor y el valor del campo.

Fuzzy match

Step name Fuzzy match

General Fields

Output fields

Match field	Champ valeur
Value field	Champ mesure

Specify the additional fields (not mandatory) to retrieve from lookup stream

#	Field	New name		Get fields
1	active			

- Conversión de los valores null por 0. Significará que los jugadores están inactivos.

Value mapper

Step name : Inactive

Fieldname to use : active

Target field name (empty=overwrite) :

Default upon non-matching :

Field values:

#	Source value	Target value
1		0

OK Cancel Help

- Para asignar los valores de active por 1, como jugadores que se encuentran en activo, realizamos diversos pasos. Estos son, reemplazar aquellos jugadores por espacio en blanco, crear una variable constante con valor 1, concatenar el campo active (que tiene valores 0 y instancias vacías) y el campo constante. De este modo los valores del campo serán (01 para jugadores en inactivo y 1 para jugadores en activo). Finalmente, para completar el campo, mapeamos los valores 01 por valor 0.

Replace in string

Step name Replace with regex

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is
1	active		Y	[a-zA-Z-]		Y		N	N	N

Add constants

Step name: CONSTANTE_1

Fields :

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string
1	CONSTANTE_1	Integer							1	N

OK Cancel

Concat fields

Step name: Concat fields

Target Field Name: ACTIVO

Length of Target Field: 0

Separator:

Enclosure: "

Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	active	None							none	none
2	CONSTANTE_1	None								

Value mapper

Step name: Value mapper

Fieldname to use: ACTIVO

Target field name (empty=overwrite) :

Default upon non-matching :

Field values:

#	Source value	Target value
1	01	0

OK Cancel

Help

- Generamos la clave primaria para cada jugador

Add sequence

Step name	PK_JUGADOR
Name of value	pk_jugador
- Use a database to generate the sequence	
Use DB to get sequence?	<input type="checkbox"/>
Connection	<input type="button" value="Edit..."/>
Schema name	<input type="button" value="Schemas..."/>
Sequence name	SEQ_ <input type="button" value="Sequences..."/>
- Use a transformation counter to generate the sequence	
Use counter to calculate sequence?	<input checked="" type="checkbox"/>
Counter name (optional)	<input type="text"/>
Start at value	1 <input type="button" value=""/>
Increment by	1 <input type="button" value=""/>
Maximum value	999999999 <input type="button" value=""/>

Help **OK** **Cancel**

- Creamos columna constante para indicar la liga con valor NBA.

Add constants

Step name	LIGA									
Fields :										
#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	liga	String							NBA	N

Help **OK** **Cancel**

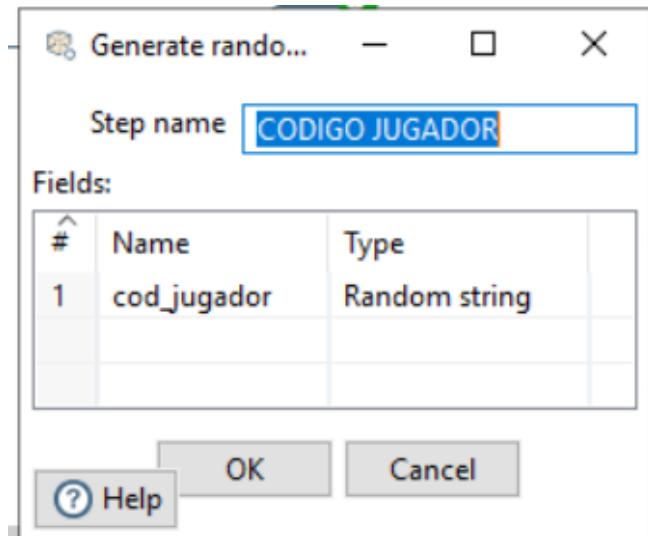
- Creamos columna constante para indicar el sexo (hombre = 1), ya que se presume de este dato por ser liga masculina.

Add constants

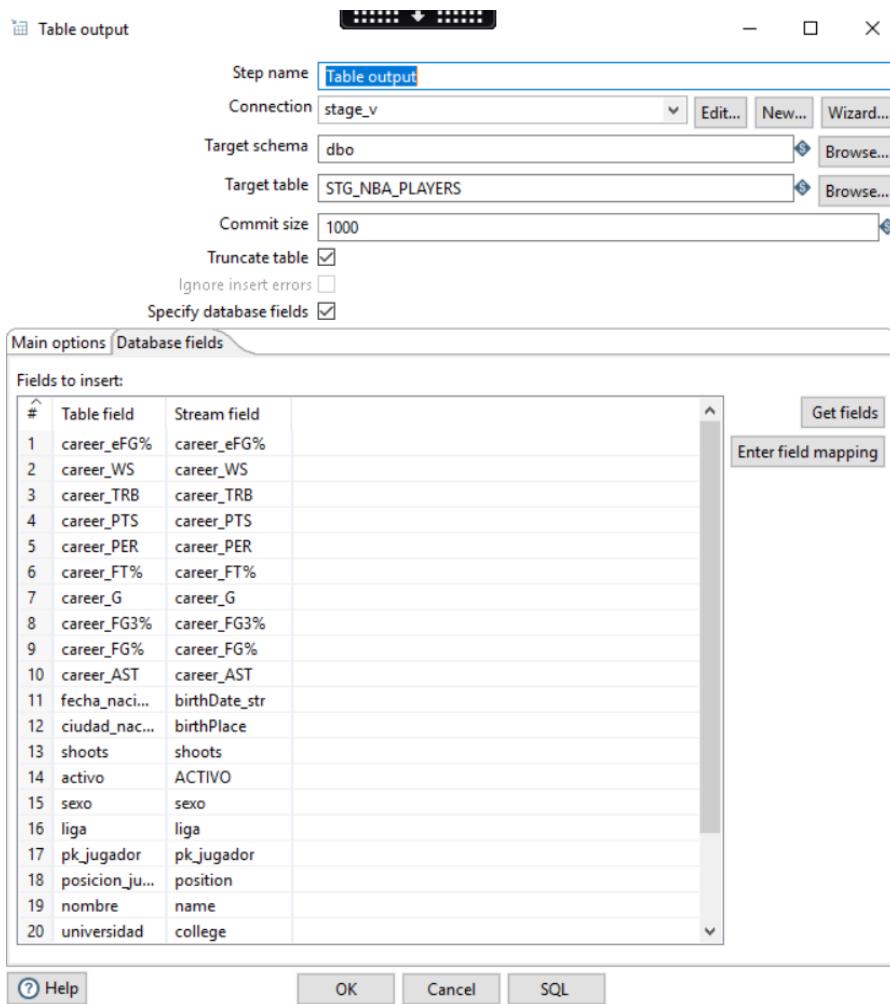
Step name	SEXO									
Fields :										
#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	sexo	Integer							1	N

Help **OK** **Cancel**

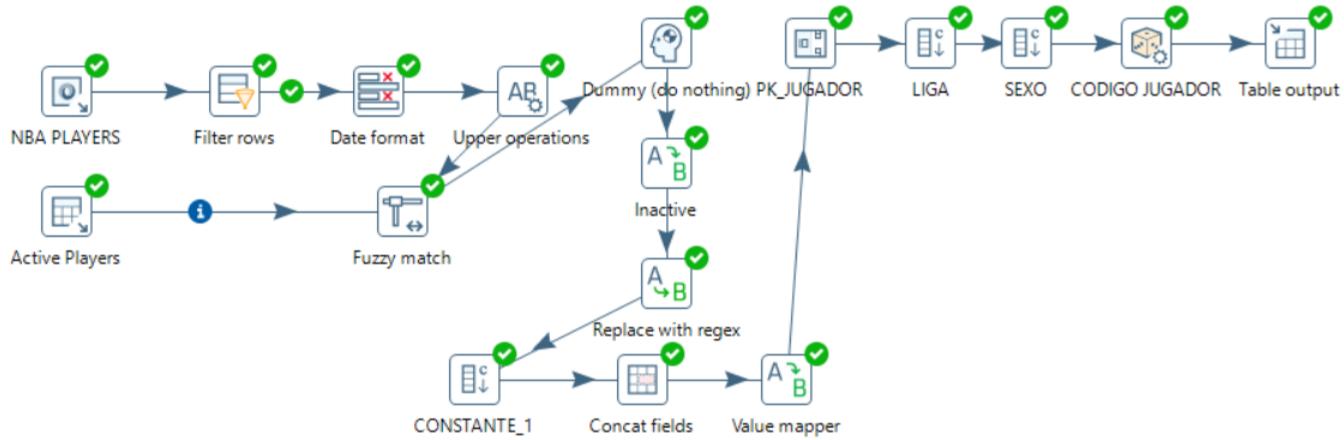
- Generamos código para cada jugador



- Carga a la tabla intermedia.



La transformación completa es la siguiente:



El resultado de la ejecución es el siguiente:

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)
1	NBA PLAYERS	0	0	4639	4639	0	0	0	0	Finished	1.0s	4,620
2	Active Players	0	0	1050	1050	0	0	0	0	Finished	0.0s	95,454
3	Filter rows	0	4639	4612	0	0	0	0	0	Finished	1.0s	4,593
4	Date format	0	4612	4612	0	0	0	0	0	Finished	1.0s	4,548
5	Upper operations	0	4612	4612	0	0	0	0	0	Finished	1.0s	4,526
6	Fuzzy match	0	5662	4612	0	0	0	0	0	Finished	7.7s	734
7	Dummy (do nothing)	0	4612	4612	0	0	0	0	0	Finished	7.7s	598
8	Inactive	0	4612	4612	0	0	0	0	0	Finished	7.7s	597
9	Replace with regex	0	4612	4612	0	0	0	0	0	Finished	7.7s	597
10	CONSTANTE_1	0	4612	4612	0	0	0	0	0	Finished	7.7s	596
11	Concat fields	0	4612	4612	0	0	0	0	0	Finished	7.7s	596
12	Value mapper	0	4612	4612	0	0	0	0	0	Finished	7.7s	596
13	PK_JUGADOR	0	4612	4612	0	0	0	0	0	Finished	7.7s	596
14	LIGA	0	4612	4612	0	0	0	0	0	Finished	7.7s	595
15	SEXO	0	4612	4612	0	0	0	0	0	Finished	7.8s	595
16	CODIGO JUGADOR	0	4612	4612	0	0	0	0	0	Finished	7.8s	595
17	Table output	0	4612	4612	0	4612	0	0	0	Finished	7.8s	591

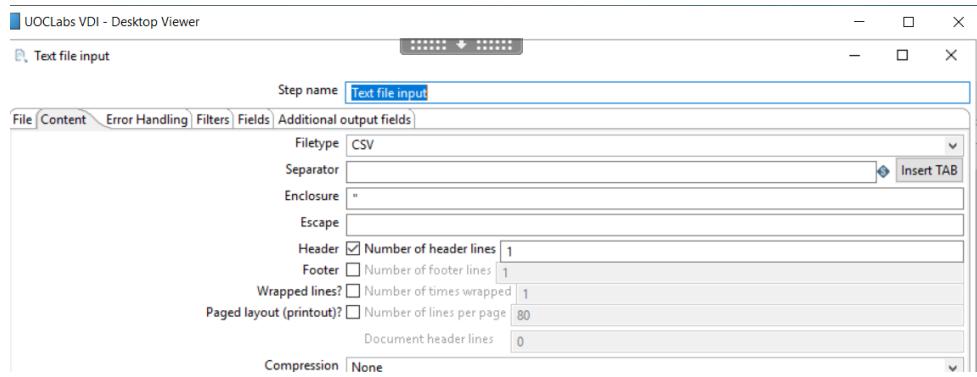
STG_WNBA_PLAYERS (IN_WNBA_PlayerList)

La transformación trata de obtener datos desde el fichero de texto WNBA_Players_List.txt que contiene la información sobre las jugadoras de la WNBA si se encuentran activas (valor 1) o no (valor 0).

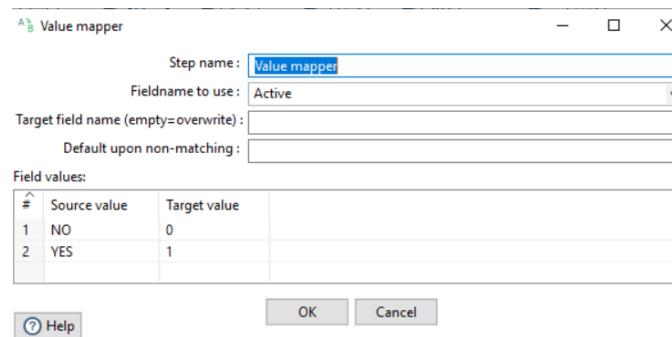
Los valores cargados a la tabla intermedia STG_WNBA_Players nos será de uso para realizar comparaciones con el fichero de estadísticos de las jugadoras de la WNBA e incluir en ello su actividad.

Las transformaciones realizadas son las siguientes: carga del fichero de extensión txt, mapeo de valores, separación de campo de jugadora para conformar posteriormente el nombre completo (nombre apellido), cambio a mayúsculas y carga a la tabla intermedia.

- Carga del fichero donde los campos se encuentran separados por tabulador.



- Mapeo de valores para indicar mediante 0 que no se encuentran activas y valor 1 para aquellas que sí lo están.



- División del campo jugadores, y concatenación para formar el nombre completo.

#	New field	ID	Remove ID?	Type	Length	Precision	Format	Group	Decimal	Currency	Nullif	Default
1	apellido		N	String								
2	Nombre		N	String								

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type
1	Nombre	None							none
2	apellido	None							none

- Cambio a mayúsculas del campo que contiene el nombre completo.

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
1	NombreCompleto			upper					

- Carga a la tabla intermedia

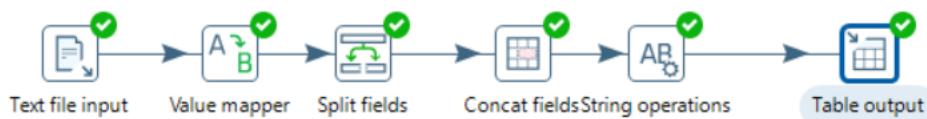
String operations

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
1	NombreCompleto			upper					

La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results

Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Spee
1 Text file input	0	0	963	964	0	1	0	0	Finished	0.1s	1
2 Value mapper	0	963	963	0	0	0	0	0	Finished	0.1s	
3 Split fields	0	963	963	0	0	0	0	0	Finished	0.2s	
4 Concat fields	0	963	963	0	0	0	0	0	Finished	0.2s	
5 String operations	0	963	963	0	0	0	0	0	Finished	0.2s	
6 Table output	0	963	963	0	963	0	0	0	Finished	0.2s	

d) Bloque TR

1. Transformaciones del bloque TR_DIM

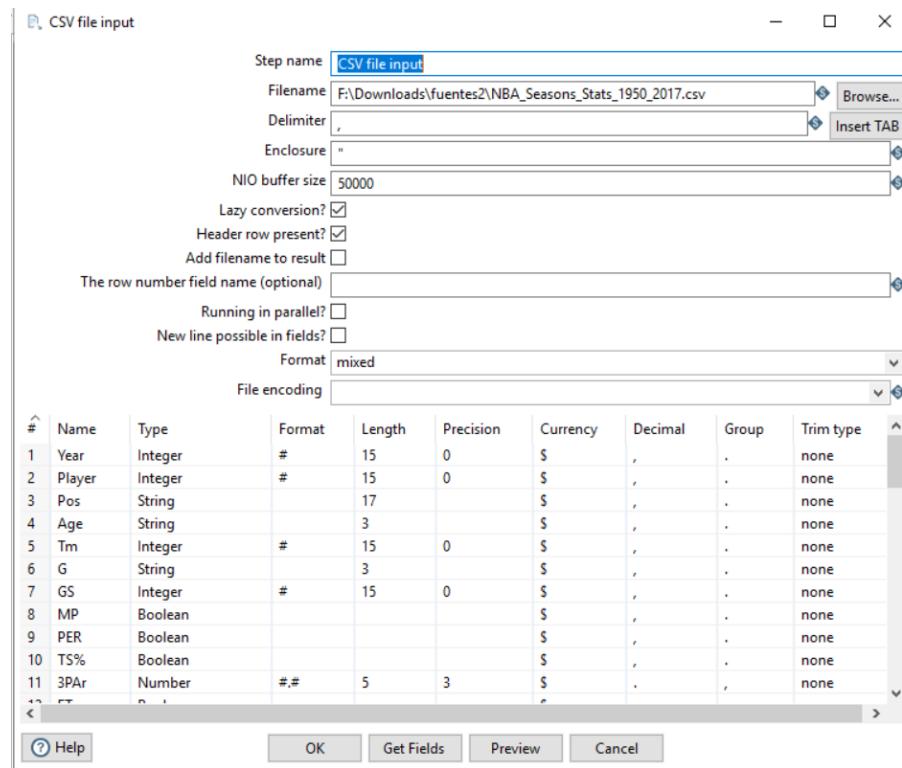
Transformación TR_DIM_Tiempo

La transformación TR_DIM_Tiempo corresponde a la carga de datos para formar la dimensión temporal del almacén.

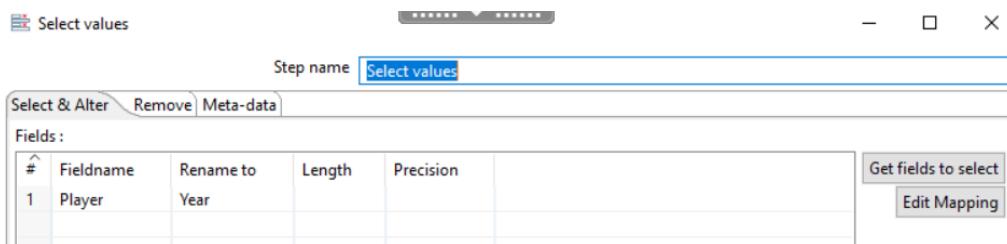
Los pasos de la transformación son los siguientes: carga del fichero csv, selección de los campos de manera correcta, obtención de los años en orden y únicos, se realiza filtro a valores nulos, obtención del segundo año correspondiente a temporada, obtención del campo temporada. Por defecto se establece en la dimensión el mes octubre, por ello se añade columna constante del mes (10). Por defecto, se establece el día 1 y se añade columna constante del día (1). Concatenación para formar la fecha completa. Metadatado para formar el tipo de datos del campo fecha. Se añade la secuencia que será la clave primaria de la dimensión temporal DIM_Tiempo y finalmente, se carga los datos a la dimensión.

Se adjunta la captura de los pasos

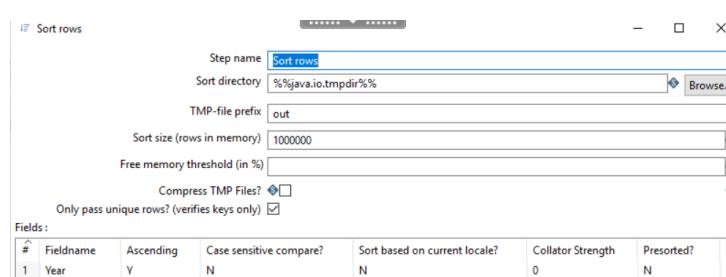
- Carga del fichero Seasons_Stats de la NBA para obtener los años. De esta manera la dimensión engloba los años para los estadísticos de la liga NBA y la liga WNBA.

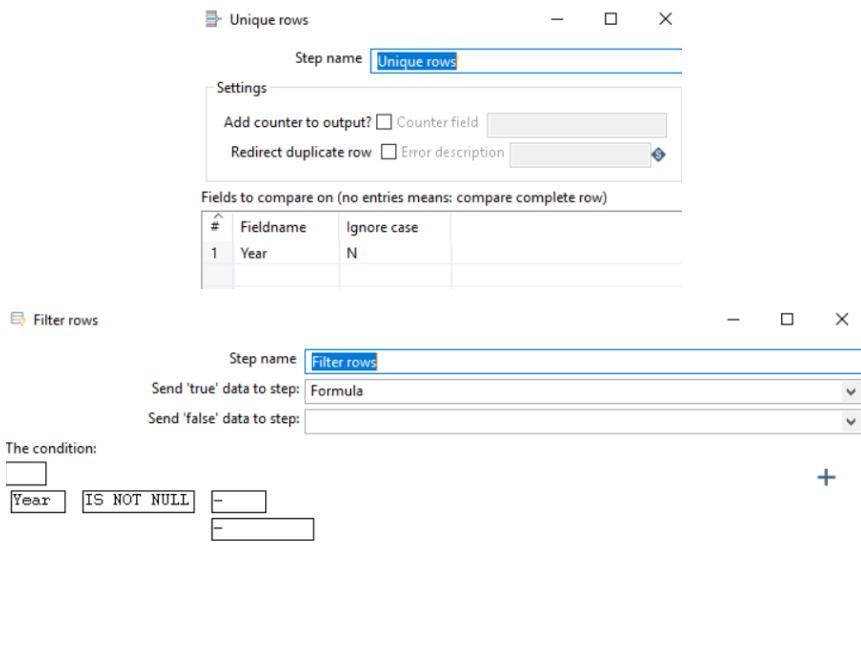


- Selección del campo de interés para componer la dimensión

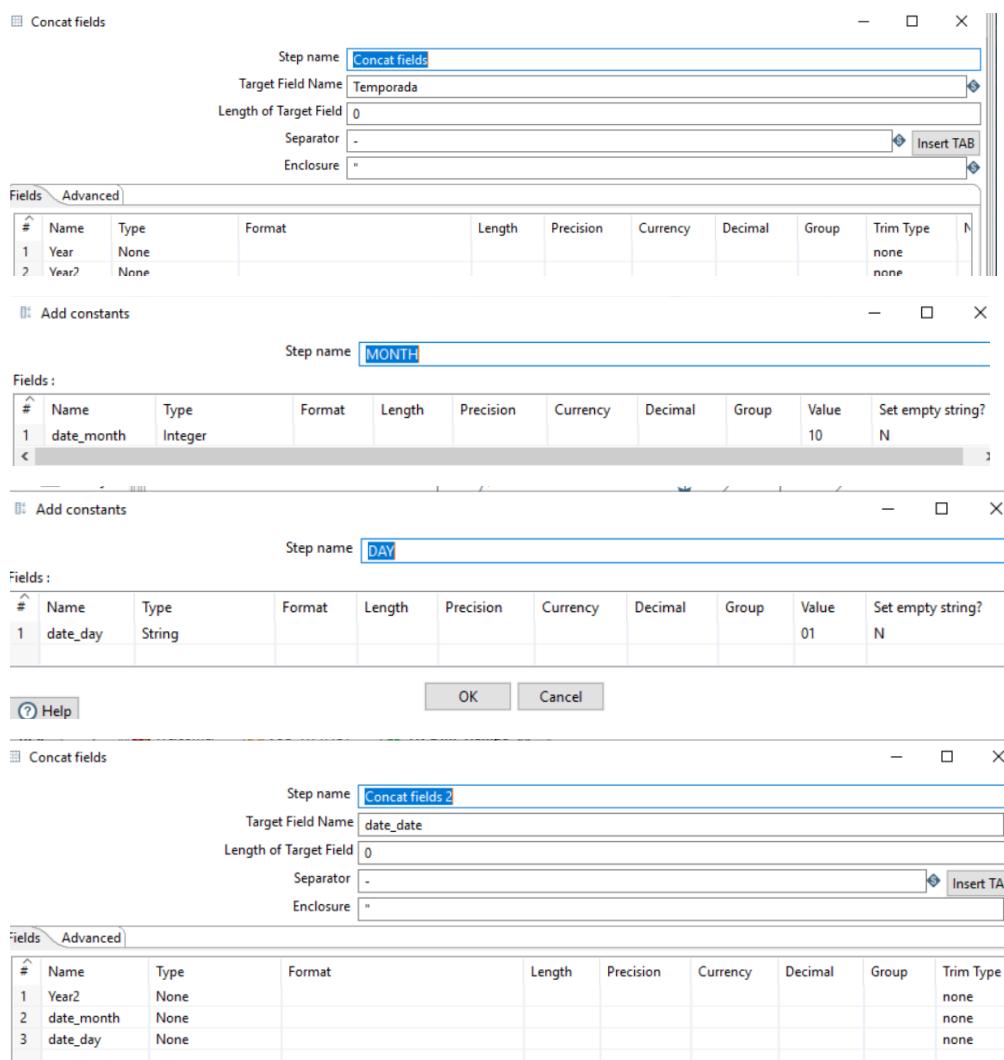


- Los siguientes pasos, tratan de ordenar los años, obtener los valores únicos y filtrar los valores nulos (correspondientes a instancias sin especificar)





- Los siguientes pasos, obtiene el segundo año para formar el campo temporada. Se concatena los campos, creación del campo mes, y día y sus valores por defecto.



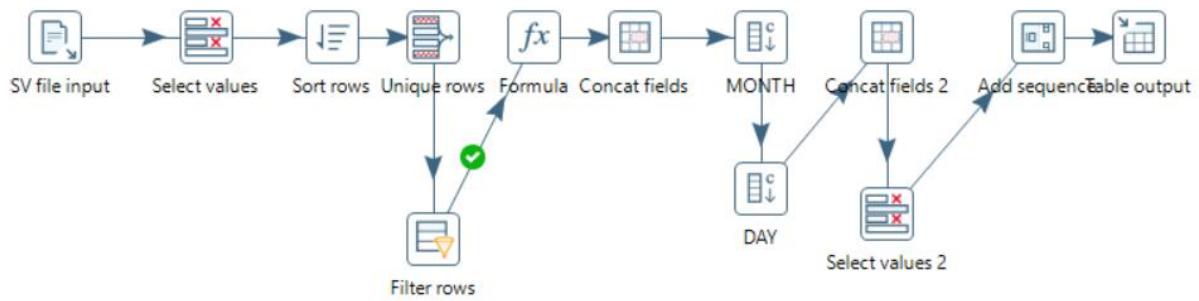
- Se transforma el tipo de dato a fecha. Se añade la secuencia que servirá de clave primaria y cargamos a la dimensión DIM_Tiempo

The screenshot shows the Informatica PowerCenter interface with three open transformation steps:

- Select values**: A table titled "Fields to alter the meta-data for:" with one row: # 1 Fieldname date_date, Rename to (empty), Type Date, Length (empty), Precision (empty), Binary to Normal? N.
- Add sequence**: A configuration dialog for generating a sequence. Step name: Add sequence, Name of value: pk_date. Under "Use a database to generate the sequence": Use DB to get sequence? (unchecked), Connection (dropdown), Schema name (dropdown), Sequence name: SEQ_. Under "Use a transformation counter to generate the sequence": Use counter to calculate sequence? (checked), Counter name (optional) (empty), Start at value: 1, Increment by: 1, Maximum value: 999999999.
- Table output**: A configuration dialog for outputting data to a table. Step name: Table output, Connection: dw_v, Target schema: dbo, Target table: DIM_Tiempo, Commit size: 1000. Options: Truncate table (checked), Ignore insert errors (unchecked). Under "Specify database fields": Main options tab, Fields to insert table:

#	Table field	Stream field
1	temporada	Temporada
2	date_year	date_year
3	pk_date	pk_date
4	date_date	date_date
5	date_month	date_month
6	date_day	date_day

La transformación es la siguiente:



Transformación TR_DIM_Minutos

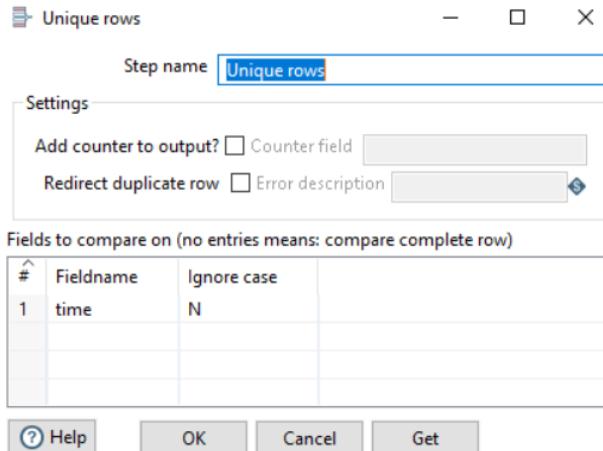
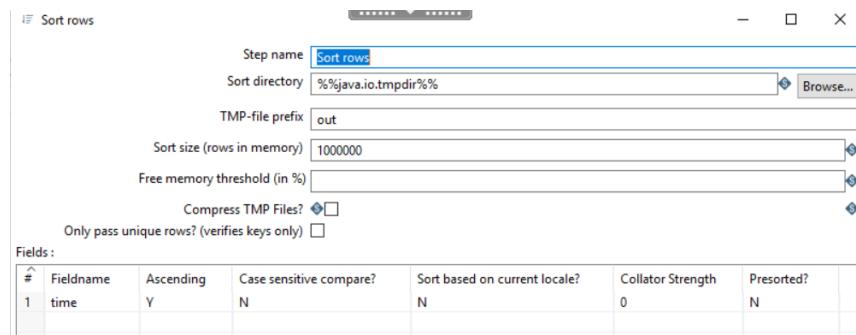
Mediante esta transformación pobaremos la dimensión DIM_Minutos, acotada a la duración de un partido e independiente de la dimensión temporal global creada en el apartado anterior

La transformación TR_DIM_Minutos contiene los siguientes pasos: lectura del fichero de tiros libres y obtención del campo time, ordenar la duración de los partidos y obtener valores únicos, dividir el campo time para obtener el campo minuto y segundo, crear nueva variable minutoSegundo, crear secuencia para la clave primaria de la dimensión DIM_Minutos y carga a la dimensión

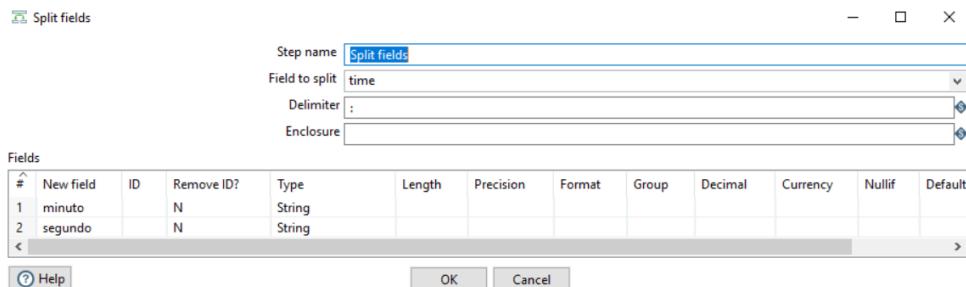
- Lectura del fichero y obtención del campo de interés

The screenshot shows the Talend Studio interface with two main windows. The top window is the 'CSV file input' configuration dialog, which includes fields for Step name (CSV file input), Filename (F:\Downloads\fuentes2\nba_free_throws.csv), Delimiter (,), Enclosure ("), NIO buffer size (50000), Lazy conversion? (checked), Header row present? (checked), Add filename to result (unchecked), The row number field name (optional) (empty), Running in parallel? (unchecked), New line possible in fields? (unchecked), Format (mixed), and File encoding (UTF-8). The bottom window is a 'Fields' dialog titled 'Select & Alter' with tabs for 'Select & Alter', 'Remove', and 'Meta-data'. It lists a single field 'time' with its details: Fieldname 'time', Rename to (empty), Length 5, Precision 0, and a 'Get fields to select' button.

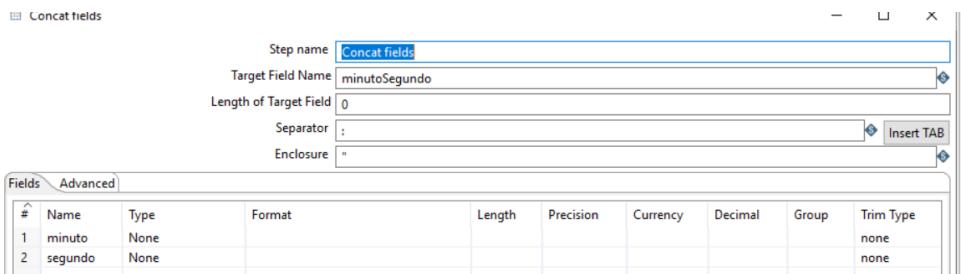
- Obtención de los valores en orden, únicos valores



- División del campo time por el delimitador entre minutos y segundos (:)



- Concatenación para obtener el campo minutoSegundo. No se usa el campo time , pues llegados a este paso hemos obtenido los valores únicos sin repetir.



- Se genera secuencia para formar la clave primaria de la dimensión y se cargan los datos a la DIM_Minutos

Add sequence

Step name: Add sequence
Name of value: pk_minutoSegundo

Use a database to generate the sequence
Use DB to get sequence?
Connection:
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

Table output

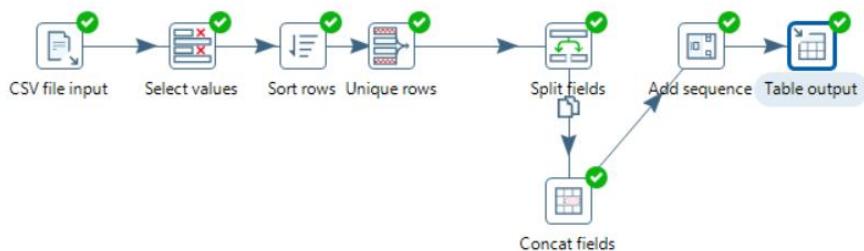
Step name: Table output
Connection: dw_v
Target schema: dbo
Target table: DIM_Minutos
Commit size: 1000
Truncate table:
Ignore insert errors:
Specify database fields:

Main options Database fields

Fields to insert:

#	Table field	Stream field
1	pk_minutoS...	pk_minutoSe...
2	minutoSegun...	minutoSegun...
3	segundo	segundo
4	minuto	minuto

La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results											
	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	
1	CSV file input	0	0	618019	618020	0	0	0	0	0	Finished
2	Select values	0	618019	618019	0	0	0	0	0	0	Finished
3	Sort rows	0	618019	618019	0	0	0	0	0	0	Finished
4	Unique rows	0	618019	534	0	0	0	0	0	0	Finished
5	Split fields	0	534	534	0	0	0	0	0	0	Finished
6	Concat fields	0	534	534	0	0	0	0	0	0	Finished
7	Add sequence	0	534	534	0	0	0	0	0	0	Finished
8	Table output	0	534	534	0	534	0	0	0	0	Finished

Transformación TR_DIM_PosicionesJuego

Para la DIM_PosicionesJuego no es necesario contar con ninguna tabla intermedia, sino que se crea manualmente.

Para ello introducimos los valores de los campos, generamos una secuencia de números que servirán de claves primarias y cargaremos los datos a la dimensión DIM_PosicionesJuego.

- El primero paso ha sido crear un data grid e introducir los valores

Data grid						
Step name Data grid						
Meta	Data	#	codigo_posicion	desc_posicion_EN	desc_posicion_ES	num_posicion
1	PG	1	POINT GUARD	BASE	1	
2	SG	2	SHOOTING GUARD	ESCOLTA	2	
3	SF	3	SMALL FORWARD	ALERO	3	
4	PF	4	POWER FORWARD	ALA-PIVOT	4	
5	C	5	CENTER	PIVOT	5	

- Generamos la secuencia para la clave primaria de la dimensión

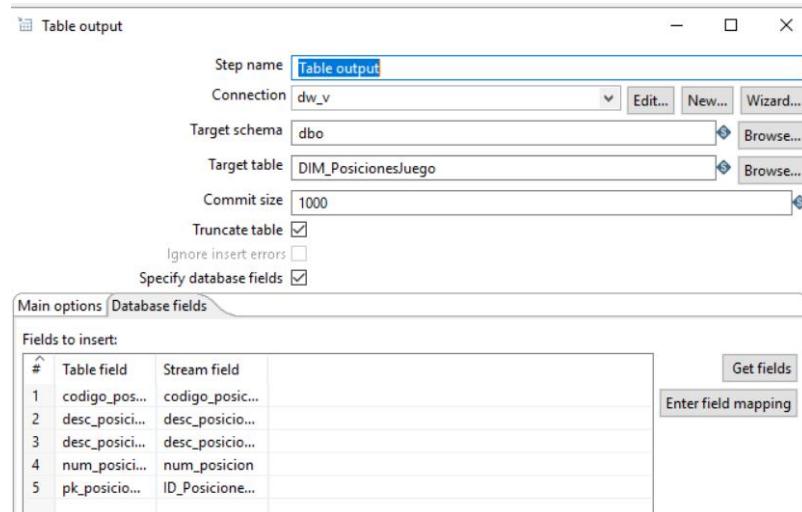
Add sequence

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

- Por último, cargamos los datos en la DIM_PosicionesJuego



El resultado de la transformación y los resultados de ejecución:



Execution Results											
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time
1	Data grid	0	0	5	0	0	0	0	0	Finished	0.0s
2	Add sequence	0	5	5	0	0	0	0	0	Finished	0.0s
3	Table output	0	5	5	0	5	0	0	0	Finished	0.2s

Transformación TR_DIM_GAMES

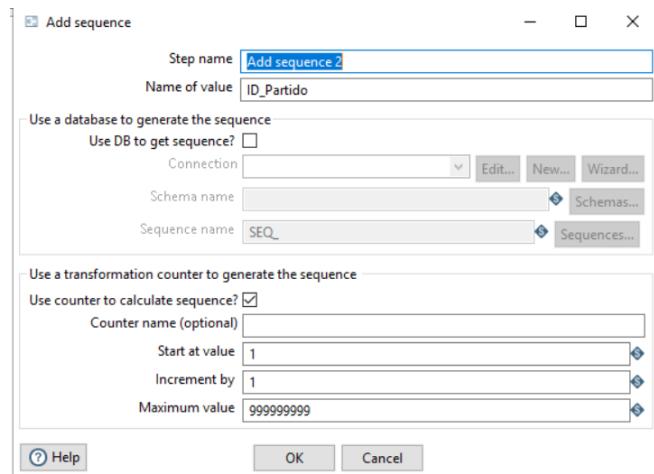
Mediante esta transformación obtendremos los valores de la dimensión DIM_Partidos, utilizando los datos que contiene la table intermedia STG_NBA_free_throws.

La transformación TR_DIM_GAMES contiene pasos: lectura de la tabla intermedia, obtención de secuencia y carga de la dimensión a la tabla DIM_Partidos.

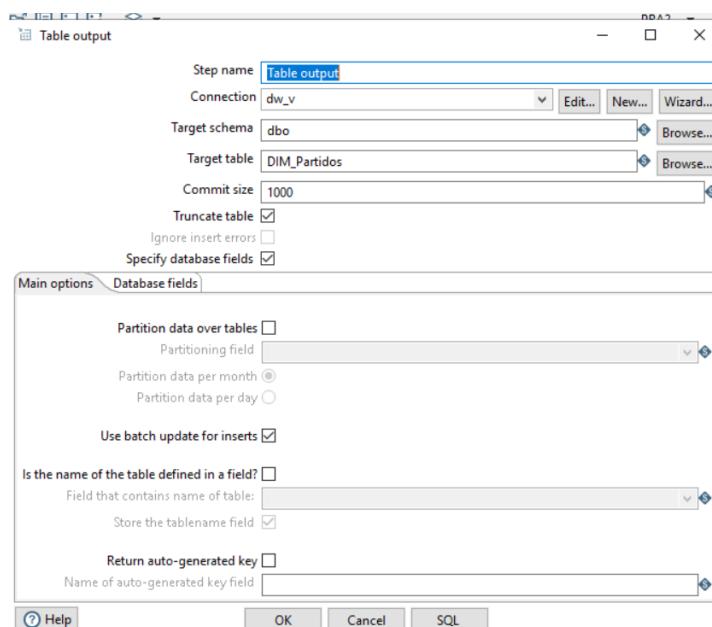
- Paso entrada tabla para obtener de los valores de los distintos campos



- Paso obtener valor de la secuencia de la base de datos para conseguir un campo ID_Partido mediante una secuencia de valores que comienza por 1 y que servirá de campo clave de la dimensión DIM_Partidos.



- Paso salida de tabla a DIM_Partidos, para cargar los datos en la tabla de la dimensión del almacén.



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



#	Stepname	Copynr	Read	Written	Input	Output	Update
1	Table input 2	0	0	618019	618019	0	
2	Add sequence 2	0	618019	618019	0	0	
3	Table output 2	0	618019	618019	0	618019	

Transformación TR_DIM_PLAYS

Este bloque contiene la transformación para obtener los valores de la dimensión DIM_Jugadas utilizando los datos cargados en la tabla intermedia STG_NBA_free_throws.

La transformación consta de los siguientes pasos: lectura de tabla intermedia, eliminar el nombre de jugador de jugadas, obtener los tipos de juegos, generar los códigos de las jugadas, generar identificadores para la dimensión y carga a la tabla DIM_Jugadas.

- Paso entrada de tabla para obtener los valores de las jugadas

Table input

Step name: Table input 2

Connection: dw_v

SQL:

```
SELECT
play
FROM dbo.STG_NBA_free_throws
```

- Para obtener los tipos de jugadas reemplazamos mediante expresiones regulares la parte del jugador hasta los strings “MAKES” y “MISSES” y los reemplazamos por el mismo string.

Replace in string

Step name: Replace in string 2

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field
1	play		Y	([a-z].+)MAKES	MAKES	N	
2	play		Y	([a-z].+)MISSES	MISSES	N	

- Para obtener los distintos valores únicos de cada jugada, realizamos en el diseño el paso sort ya que facilita la obtención.

Unique rows

Step name: Unique rows

Settings

Add counter to output? Counter field: contador

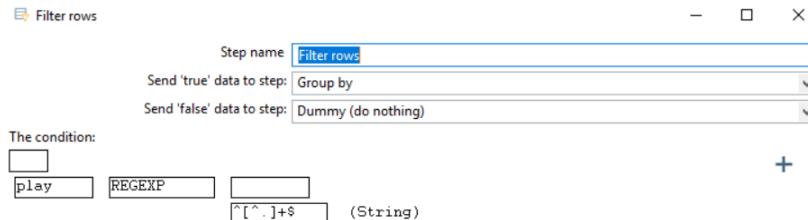
Redirect duplicate row Error description:

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

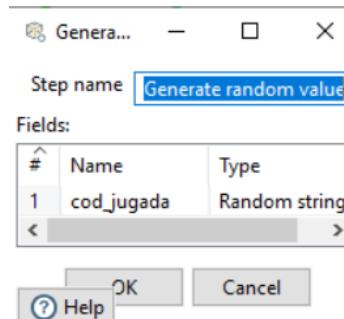
#	Fieldname	Ignore case
1	play	Y

Help OK Cancel Get

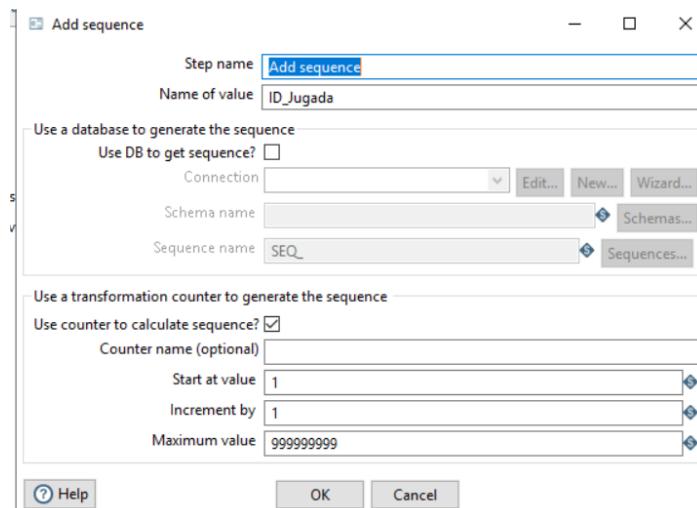
- Filtramos los datos de jugadas, ya que nos encontramos con un registro cuya descripción se extiende. El registro contiene un punto y será por el cual usaremos la expresión regular para filtrarlo. Tras este paso, los valores que corresponden al tipo de jugadas se agrupan para proseguir con la transformación.



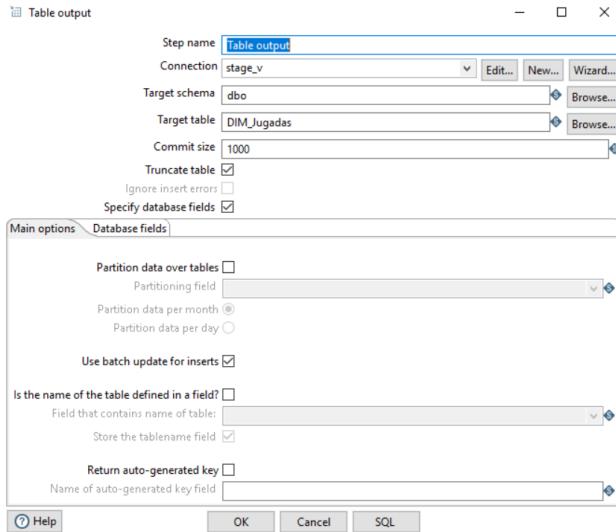
- Generamos identificaciones únicas para cada tipo de jugada



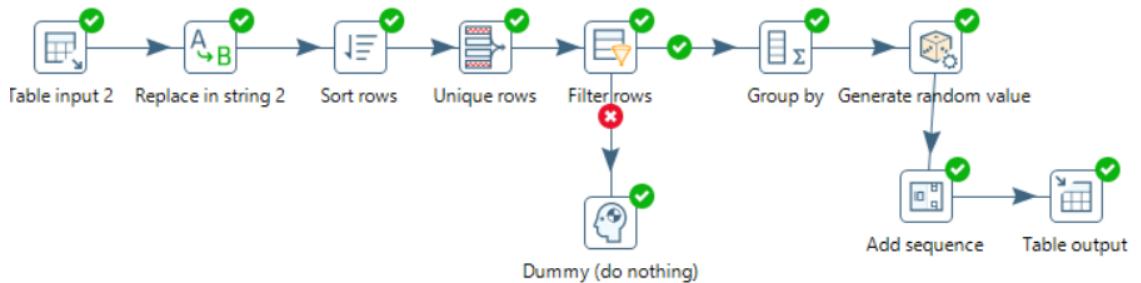
- Generamos secuencia de números para el campo primario



- Cargamos los datos a la DIM_Jugadas



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



El resultado de la ejecución es el siguiente:

Execution Results									
	Logging	Execution History	Step Metrics	Performance Graph	Metrics	Preview data			
Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Activ
Table input 2	0	0	618019	618019	0	0	0	0	Finish!
Replace in string 2	0	618019	618019	0	0	0	0	0	Finish!
Sort rows	0	618019	618019	0	0	0	0	0	Finish!
Unique rows	0	618019	38	0	0	0	0	0	Finish!
Filter rows	0	38	38	0	0	0	0	0	Finish!
Group by	0	37	37	0	0	0	0	0	Finish!
Generate random value	0	37	37	0	0	0	0	0	Finish!
Add sequence	0	37	37	0	0	0	0	0	Finish!
Dummy (do nothing)	0	1	1	0	0	0	0	0	Finish!
Table output	0	37	37	0	37	0	0	0	Finish!

Transformación TR_DIM_STATES

Este bloque contiene la transformación para obtener los valores de la dimensión DIM_Estados_EEUU utilizando los datos cargados en la tabla intermedia STG_Estados_Unidos.

La transformación consta de los siguientes pasos: lectura de tabla intermedia, obtener los valores únicos de Estado, generar los códigos de las jugadas, generar identificadores para la dimensión y carga a la tabla DIM_Estados_EEUU.

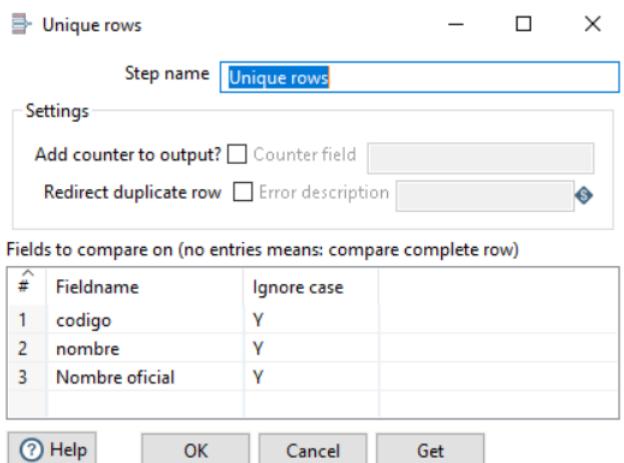
- Paso entrada de tabla para obtener los valores de los campos

```

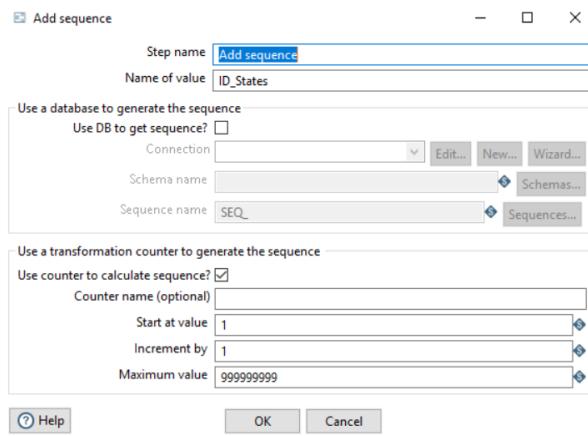
Step name Table input
Connection dw_v
SQL
SELECT
    codigo
, nombre
, "Nombre oficial"
, superficie
, poblacion
, capital
, densidadPoblacion
FROM dbo.STG_Estados_ Unidos

```

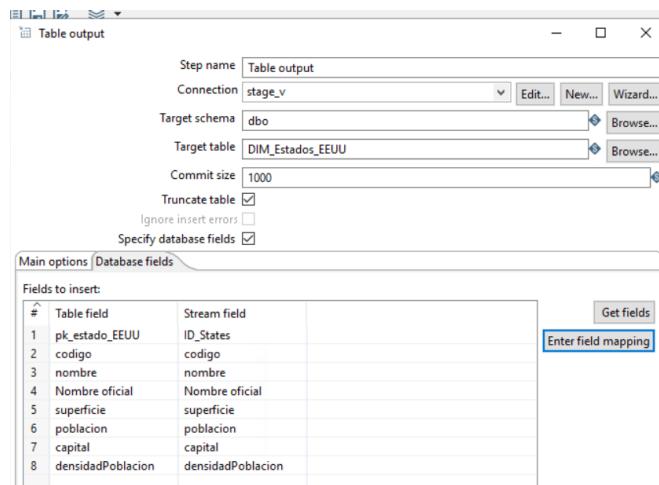
- Obtener los distintos valores únicos de nombre de Estado, realizamos en el diseño el paso sort ya que facilita la obtención.



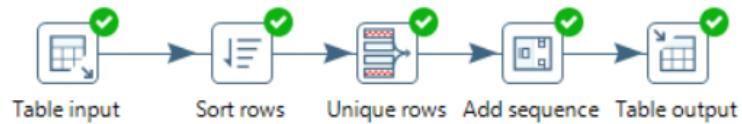
- Generamos secuencia de números para el campo primario



- Cargamos los datos a la DIM_Estados_EUU



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



El resultado de la ejecución es el siguiente:

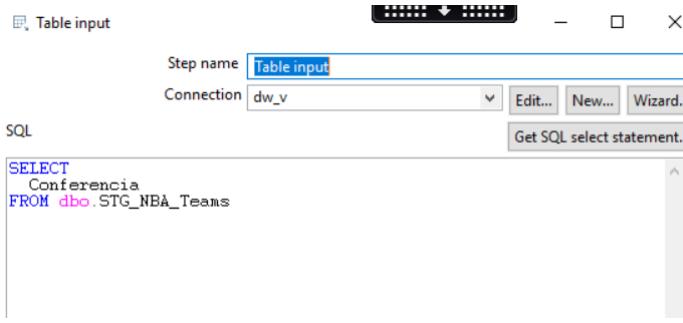
Execution Results											
#	Stepname	Copymr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time
1	Table input	0	0	50	50	0	0	0	0	Finished	0.0s
2	Sort rows	0	50	50	0	0	0	0	0	Finished	0.1s
3	Unique rows	0	50	50	0	0	0	0	0	Finished	0.1s
4	Add sequence	0	50	50	0	0	0	0	0	Finished	0.1s
5	Table output	0	50	50	0	50	0	0	0	Finished	0.2s

Transformación TR_DIM_CONFERENCIAS

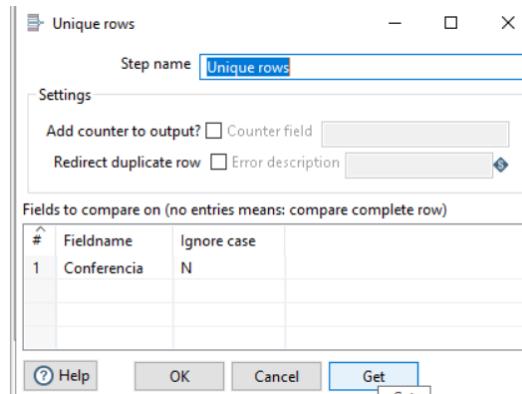
Este bloque contiene la transformación para obtener los valores de la dimensión DIM_Conferencias utilizando los datos cargados en la tabla intermedia STG_NBA_Teams.

La transformación TR_DIM_CONFERENCIAS contiene los siguientes pasos: lectura de la tabla intermedia, obtención tipos únicos de conferencia, generar código propio para cada conferencia, añadir una secuencia de claves primaria y carga a la DIM_Conferencias

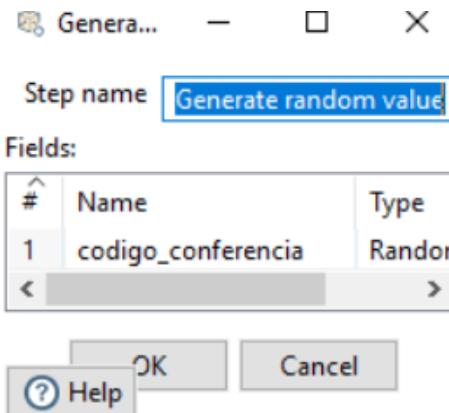
- Seleccionamos el campo de interés desde la tabla intermedia



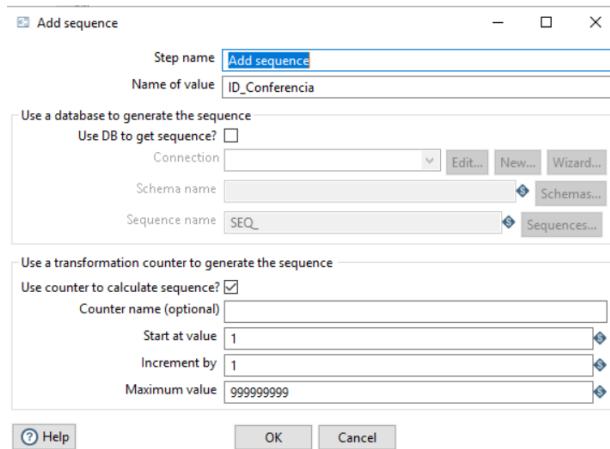
- Obtenemos tipos de conferencia



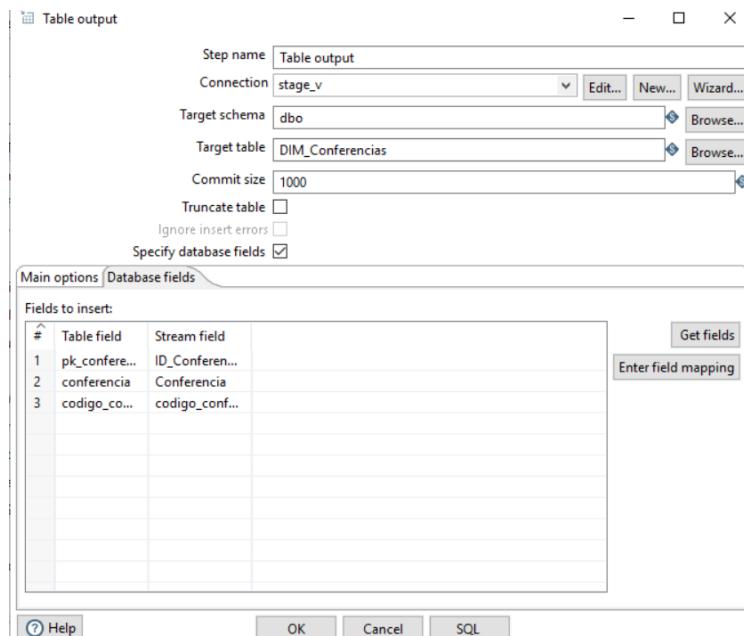
- Generamos el código para cada conferencia



- Secuencia para los valores primarios de la dimensión



- Por último, cargamos los correspondientes campos a la dimensión



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results										
	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	Table input	0	0	30	30	0	0	0	0	Finished
2	Unique rows	0	30	2	0	0	0	0	0	Finished
3	Generate random value	0	2	2	0	0	0	0	0	Finished
4	Add sequence	0	2	2	0	0	0	0	0	Finished
5	Table output	0	2	2	0	2	0	0	0	Finished

Transformación TR_DIM_DIVISIONES

Mediante esta transformación obtendremos los valores de la dimensión DIM_Divisiones, utilizando los datos sobre divisiones que contiene la tabla intermedia STG_NBA_Teams.

La transformación consta de los siguientes pasos: lectura de tabla intermedia, sacamos valores únicos de divisiones, obtener id_división, añadir código para cada división, añadir secuencia y carga a la tabla DIM_Divisiones.

- Paso entrada tabla obtener división. Se realiza la lectura desde la tabla intermedia STG_NBA_Teams.

Table input

Step name: Table input

Connection: dw_v

SQL:

```
SELECT
    Conferencia
    ,Division
FROM dbo.STG_NBA_Teams
```

- Obtener valores únicos de las divisiones

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	Division	N
2	Conferencia	N

Help OK Cancel Get

First rows Last rows Off

#	Conferencia	Division
1	CONFERENCIA OESTE	NOROESTE
2	CONFERENCIA OESTE	SUROESTE
3	CONFERENCIA OESTE	PACÍFICO
4	CONFERENCIA ESTE	ATLÁNTICO
5	CONFERENCIA ESTE	CENTRAL
6	CONFERENCIA ESTE	SURESTE

- Paso búsqueda en base datos para obtener el id_conferencia.

Database lookup

Step name: Database lookup

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_Conferencias

Enable cache?

Cache size in rows (0=cache) 0

Load all data from table

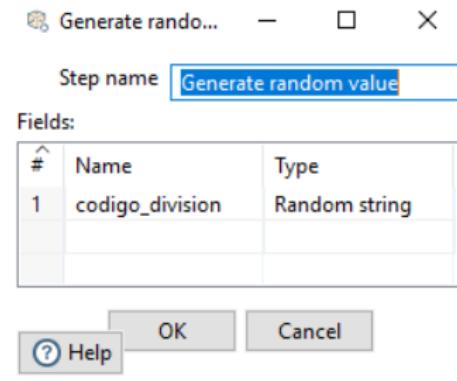
Key(s) to look up the value(s):

Table field	Comparator	Field1	Field2
Conferencia	=	conferencia	

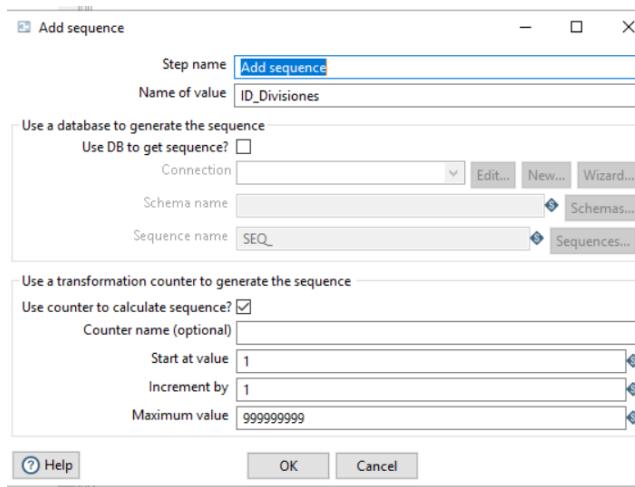
Values to return from the lookup table:

Field	New name	Default	Type
pk_conferencia	id_conferencia		None

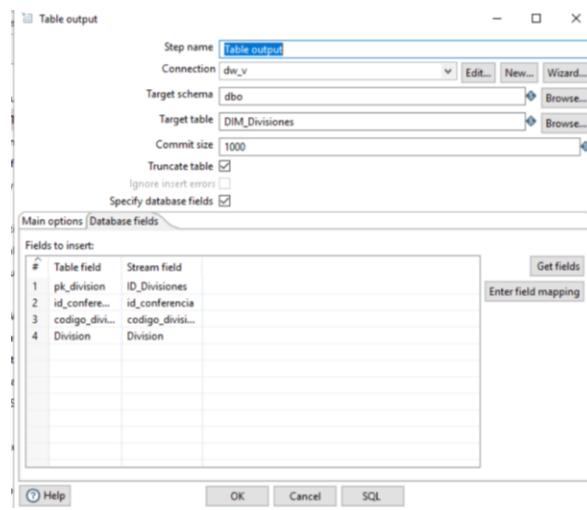
- Obtenemos código para cada división.



- Paso obtener valor de la secuencia de la base de datos para conseguir un capo ID_Divisiones mediante una secuencia de valores que comienza por 1 y que servirá de campo clave de la dimensión DIM_Divisiones.



- Paso salida de tabla carga DIM_Divisiones.



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results										
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	Table input	0	0	30	30	0	0	0	0	Finished
2	Unique rows	0	30	6	0	0	0	0	0	Finished
3	Database lookup	0	6	6	6	0	0	0	0	Finished
4	Generate random value	0	6	6	0	0	0	0	0	Finished
5	Add sequence	0	6	6	0	0	0	0	0	Finished
6	Table output	0	6	6	0	6	0	0	0	Finished

Transformación TR_DIM_TEAMS

Mediante esta transformación obtendremos los valores de la dimensión DIM_Equipos, utilizando diferentes tablas intermedias. En concreto usaremos STG_Team_Codes, STG_WNBA_Teams y STG_NBA_Teams.

La transformación TR_DIM_TEAMS contiene los siguientes pasos agrupados: carga de los datos de la tabla intermedia, concatenar de campos, obtener las claves foráneas, añadir secuencia para la clave primaria de la dimensión, ajuste de columnas y finalmente la carga de datos a la dimensión.

- Paso entrada tabla obtener liga, codigo_equipo y nombre equipo de STG_Team_Codes

Table input
Step name: Table input
Connection: stage_v
SQL:
SELECT league, team, code FROM dbo.STG_Team_Codes

Cargamos los datos de los equipos de WNBA y NBA mediante el valor común con la tabla inicial por el nombre de equipos.

La siguiente imagen es de la obtención de los campos de los equipos de la WNBA

Database lookup
Step name: lookup
Connection: stage_v
Lookup schema: dbo
Lookup table: STG_WNBA_Teams
Enable cache:
Cache size in rows (Discard): 0
Load all data from table:
The key(s) to look up the value(s):
Table field Comparator Field1 Field2
1 equipo = team
Values to return from the lookup table:
Field New name Default Type
1 conferencia None
2 equipo None
3 ciudad None
4 estado None

La siguiente imagen es de la obtención de los campos de los equipos de la NBA

Database lookup

Step name: nba

Connection: stage_v

Lookup schema: dbo

Lookup table: STG_NBA_Teams

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	Equipo	=	team	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	Division			None
2	Equipo			None
3	Ciudad			None
4	Estado			None
5	Pabellon			None
6	Fundado			None
7	Patrocinio			None
8	Conferencia			None

Do not pass the row if the lookup fails

- Los siguientes pasos tratan de unir bajo el mismo campo, los valores de conferencia, ciudad y estado obtenidos en cada tabla intermedia de los equipos de la NBA Y WNBA.

Concat fields

Step name: CONF

Target Field Name: CONF

Length of Target Field: 0

Separator: Insert TAB

Enclosure: "

Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	conferencia	None							none	none
2	Conferencia_1	None							none	none

Concat fields

Step name: CIUDAD

Target Field Name: CIU

Length of Target Field: 0

Separator: Insert TAB

Enclosure: "

Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	ciudad	None							none	none
2	Ciudad_1	None							none	none

Concat fields

Step name: STATE

Target Field Name: STATE

Length of Target Field: 0

Separator: Insert TAB

Enclosure: "

Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	Estado_1	None							none	none
2	estado	None							none	none

- Paso búsqueda en base de datos para obtener el id_división de la dimensión DIM_Divisiones. Se usan dos correspondencias. La primera se realiza por medio de la DIM_Conferencias mediante la conferencia y se obtiene el campo pk_conferencia. El mismo campo, se usa para la correspondencia con la DIM_Divisiones y así obtener la clave pk_division.

Top Screenshot (Step name: dw_v):

Step name:	dw_v
Connection:	dw_v
Lookup schema:	dbo
Lookup table:	DIM_Conferencias
Enable cache?	<input type="checkbox"/>
Cache size in rows (0=cache)	0
Load all data from table	<input type="checkbox"/>

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

#	Table field	Comparator	Field1	Field2
1	conferencia	=	CONF	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_conference			None

Bottom Screenshot (Step name: id_division):

Step name:	id_division
Connection:	stage_v
Lookup schema:	dbo
Lookup table:	DIM_Divisiones
Enable cache?	<input type="checkbox"/>
Cache size in rows (0=cache)	0
Load all data from table	<input type="checkbox"/>

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

#	Table field	Comparator	Field1	Field2
1	id_conference	=	pk_conference	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_division			None

- Paso búsqueda en base de datos para obtener el id_estado_EEUU de la dimensión DIM_Estados_EEUU. Se usan la correspondencia mediante el código del Estado obtenido en la base de NBA y por nombre en la base de equipos de la WNBA. Finalmente se guardan los valores de ambos resultados en el mismo campo.

Database lookup

Step name: **estado_cod**

Connection: **stage_v**

Lookup schema: **dbo**

Lookup table: **DIM_Estados_EEUU**

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	codigo	=	Estado_1	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_estado_EEUU	estado_cod		None

Database lookup

Step name: **Database lookup 2**

Connection: **stage_v**

Lookup schema: **dbo**

Lookup table: **DIM_Estados_EEUU**

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	=	estado	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_estado_EEUU			None

Concat fields

Step name: **CONCAT ID ESTADO**

Target Field Name: **ID_ESTADO**

Length of Target Field: **0**

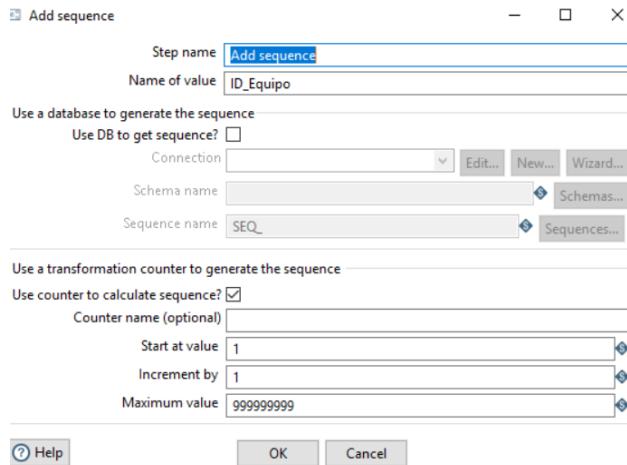
Separator: **Insert TAB**

Enclosure: **"**

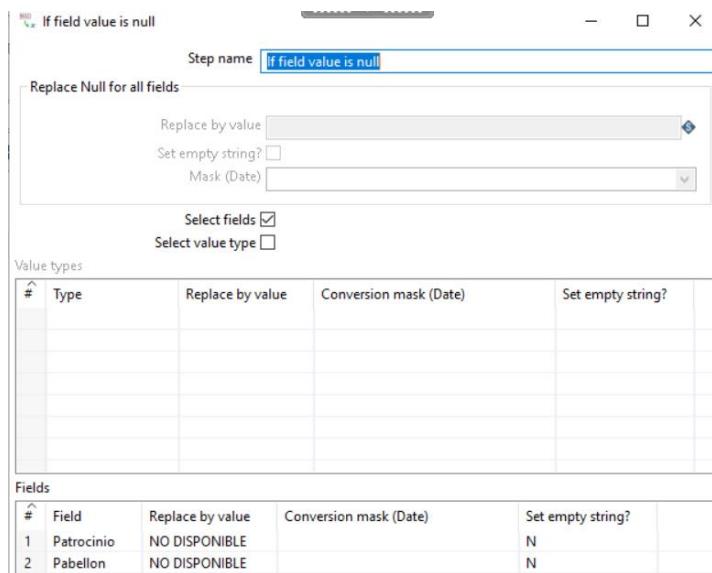
Fields Advanced

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	N
1	estado_cod	None							none	
2	pk_estado_EEUU	None							none	

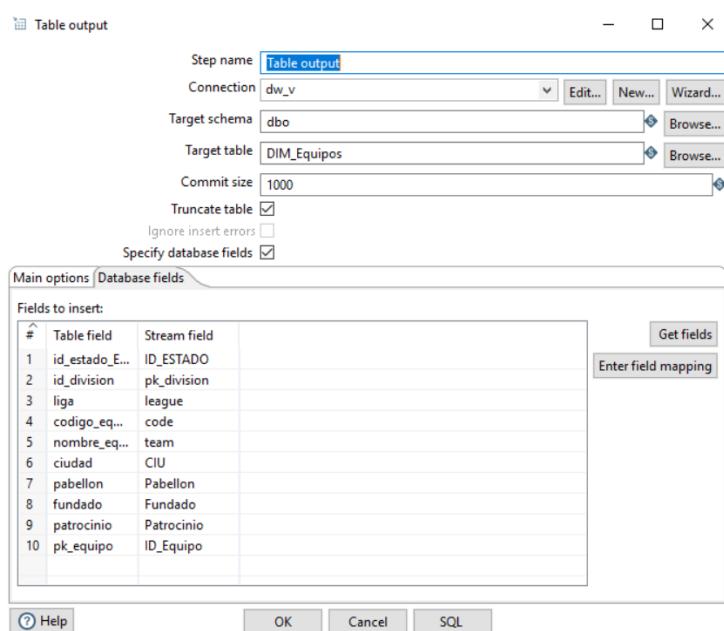
- Generamos secuencia que servirá de clave primera para la dimensión



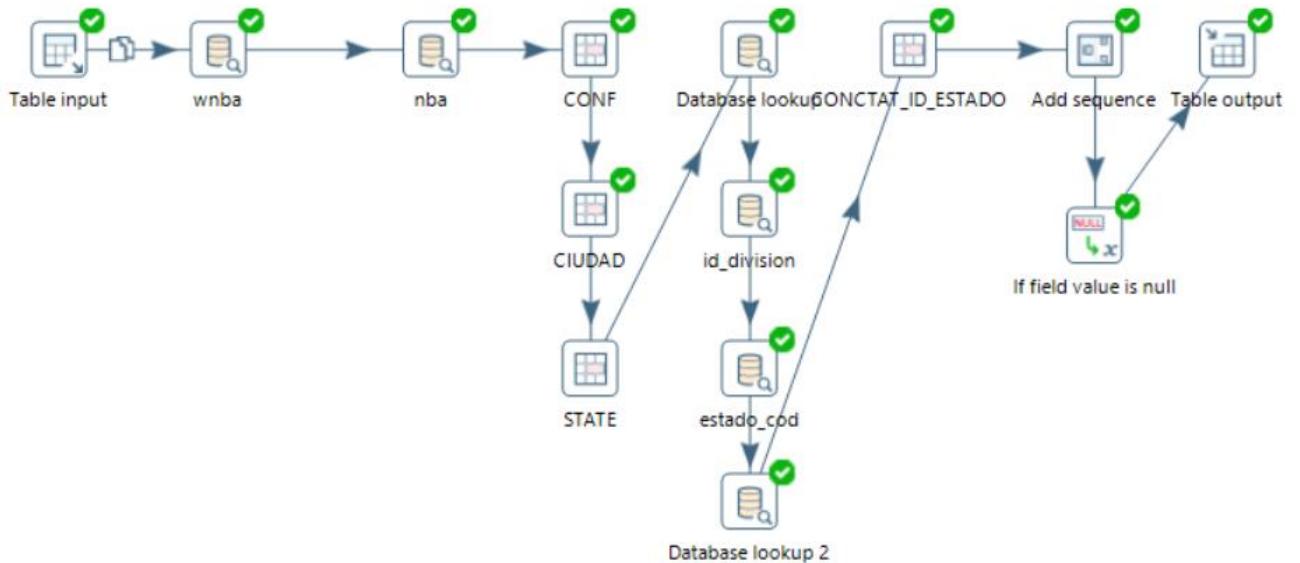
- Dado que los valores los campos Patrocinio y Pabellon no existen, los reemplazaos por string “NO DISPONIBLE” en la dimensión.



- Finalmente, realizamos la carga a la dimensión DIM_Equipos



La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Sp
1	Table input	0	0	42	42	0	0	0	0	Finished	0.2s	
2	wnba	0	42	42	12	0	0	0	0	Finished	2.4s	
3	nba	0	42	42	30	0	0	0	0	Finished	2.6s	
4	CONF	0	42	42	0	0	0	0	0	Finished	2.6s	
5	CIUDAD	0	42	42	0	0	0	0	0	Finished	2.6s	
6	Concat fields	0	42	42	0	0	0	0	0	Finished	2.7s	
7	Database lookup	0	42	42	42	0	0	0	0	Finished	3.1s	
8	id_division	0	42	42	42	0	0	0	0	Finished	3.3s	
9	estado_cod	0	42	42	29	0	0	0	0	Finished	3.3s	
10	Database lookup 2	0	42	42	12	0	0	0	0	Finished	3.5s	
11	CONCATAT_ID_ESTADO	0	42	42	0	0	0	0	0	Finished	3.5s	
12	Add sequence	0	42	42	0	0	0	0	0	Finished	3.5s	
13	If field value is null	0	42	42	0	0	0	0	0	Finished	3.5s	
14	Table output	0	42	42	0	42	0	0	0	Finished	4.0s	

Transformación TR_DIM_PLAYERS

Mediante esta transformación obtendremos los valores de la dimensión DIM_Jugadores, utilizando los datos que contiene la tabla intermedia STG_NBA_PLAYERS.

La transformación consta de los siguientes pasos: lectura de tabla intermedia, conversión campo a tipo numérico, unión de la tabla tras la corrección, reemplazo de valores, formateo de fecha y carga a la dimensión DIM_Jugadores.

- Paso entrada obtención de los campos

Table input

Step name: Table input

Connection: stage_v

SQL

```

SELECT
    pk_jugador
,   liga
,   cod_jugador
,   nombre
,   posicion_juego
,   sexo
,   activo
,   altura
,   peso
,   shoots
,   universidad
,   fecha_nacimiento
,   ciudad_nacimiento
,   career_AST
,   "career_FG%"
,   "career_FG3%"
,   "career_FT%"
,   career_G
,   career_PER
,   career PTS
,   career_TRB
,   career_WS
,   "career_eFG%"
FROM dbo.STG_NBA_PLAYERS

```

- Dado que la tabla intermedia la columna “contiene el campo con valor 1 como string realizamos un filtro para corregir y tras este paso unimos la tabla.

Filter rows

Step name: Filter rows

Send 'true' data to step: Dummy (do nothing)

Send 'false' data to step: Dummy (do nothing) 2

The condition:

activo CONTAINS 1 (String)

Realizamos un reemplazo a toda la sub-tabla tras el filtrado por condición establecida.

Set field value to a constant

Step name: Valor numerico activo 1

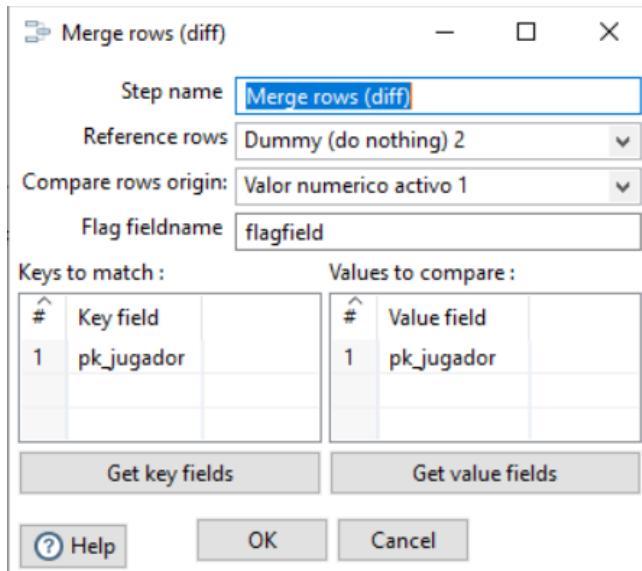
Use variable in constant

#	Field	Replace by value	Conversion mask (Date)	Set empty
1	activo	1		N

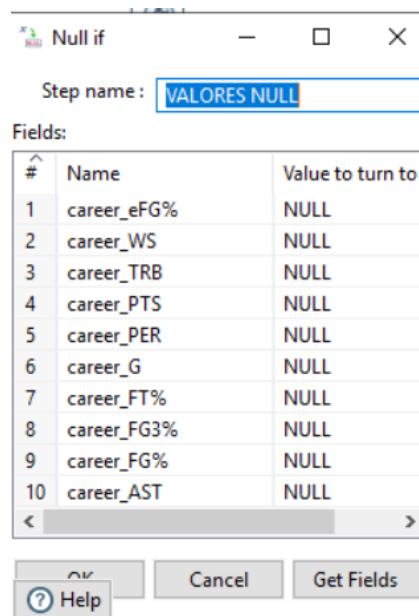
OK Get Fields Cancel

Help

- Unión de las tablas tras la corrección. Dado que las claves primarias son distintas para cada jugador, se concatena la tabla por los campos.



- Las estadísticas globales de los jugadores que no se encuentran disponibles contienen string “NULL”, cambiamos el valor por nulo de manera que se tiene en cuenta como falta de valor y no como una cadena.



- Conversión del formato de fecha que se ha recibido con minutos y segundos Acotarlo a simplemente el formato fecha yyyy-MM-dd

Select values

Step name: FORMATO FECHA

Select & Alter | Remove | Meta-data

Fields to alter the meta-data for:

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Format Lenien
1	fecha_nacimiento		Date			N	yyyy-MM-dd	N

Select values

Step name: FORMATO FECHA

Select & Alter | Remove | Meta-data

Fields:

#	Fieldname	Rename to	Length	Precision	
1	pk_jugador				
2	liga				
3	cod_jugador				
4	nombre				
5	posicion_juego				
6	sexo				
7	activo				
8	altura				
9	peso				
10	shoots				
11	universidad				
12	fecha_nacimiento				
13	ciudad_nacimiento				
14	career_AST				
15	career_FG%				
16	career_FG3%				
17	career_FT%				
18	career_G				
19	career_PER				
20	career PTS				
21	career_TRB				
22	career_WS				
23	career_eFG%				
24	flagfield				

Include unspecified fields, ordered by name

- Finalmente, cargamos los datos a la dimensión DIM_Jugadores

Table output

Step name: Table output

Connection: dw_y

Target schema: dbo

Target table: DIM_Jugadores

Commit size: 1000

Truncate table:

Ignore insert errors:

Specify database fields:

Main options | Database fields

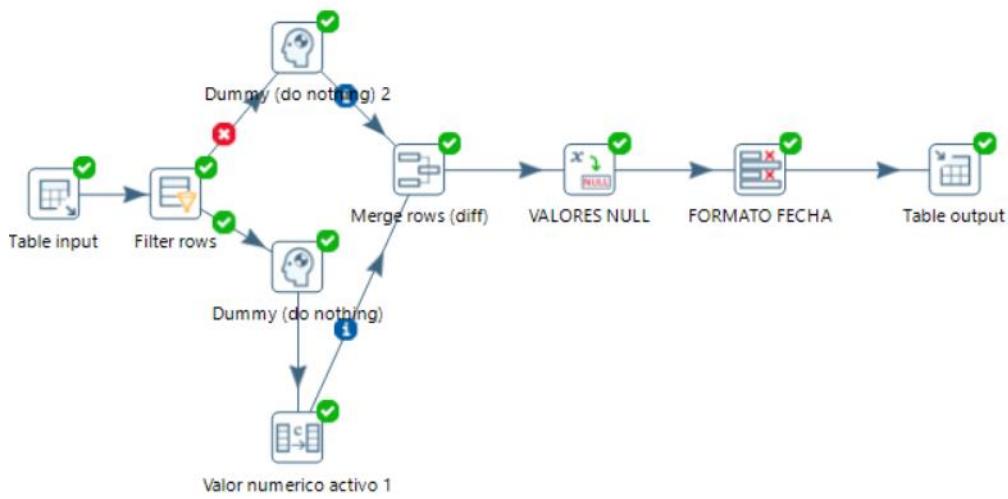
Fields to insert:

#	Table field	Stream field
1	fecha_naci...	fecha_nacimi...
2	career_eFG%	career_eFG%
3	career_WS	career_WS
4	career_TRB	career_TRB
5	career PTS	career PTS
6	career_PER	career_PER
7	career_G	career_G
8	career_FT%	career_FT%
9	career_FG3%	career_FG3%
10	career_FG%	career_FG%
11	career_AST	career_AST
12	ciudad_nac...	ciudad_naci...
13	shoots	shoots
14	universidad	universidad
15	peso	peso
16	altura	altura
17	activo	activo
18	sexo	sexo
19	posicion_ju...	posicion_jue...
20	nombre	nombre

Get fields | Enter field mapping

Help | OK | Cancel | SQL

La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



Execution Results											
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time
1	Table input	0	0	4612	4612	0	0	0	0	Finished	0.1s
2	Filter rows	0	4612	4612	0	0	0	0	0	Finished	0.1s
3	Dummy (do nothing)	0	202	202	0	0	0	0	0	Finished	0.1s
4	Dummy (do nothing) 2	0	4410	4410	0	0	0	0	0	Finished	0.1s
5	Valor numerico activo 1	0	202	202	0	0	0	0	0	Finished	0.2s
6	Merge rows (diff)	0	4612	4612	0	0	0	0	0	Finished	0.5s
7	VALORES NULL	0	4612	4612	0	0	0	0	0	Finished	0.5s
8	FORMATO FECHA	0	4612	4612	0	0	0	0	0	Finished	0.5s
9	Table output	0	4612	4612	0	4612	0	0	0	Finished	0.6s

2. Transformaciones del bloque TR_FACT

Transformación TR_FACT_SEASONS_STATS

Mediante esta transformación poblaremos la tabla de hechos FACT_SEASONS_STATS según el modelo multidimensional definido y que nos permitirá analizar los datos de diferentes estadísticas desde diferentes perspectivas.

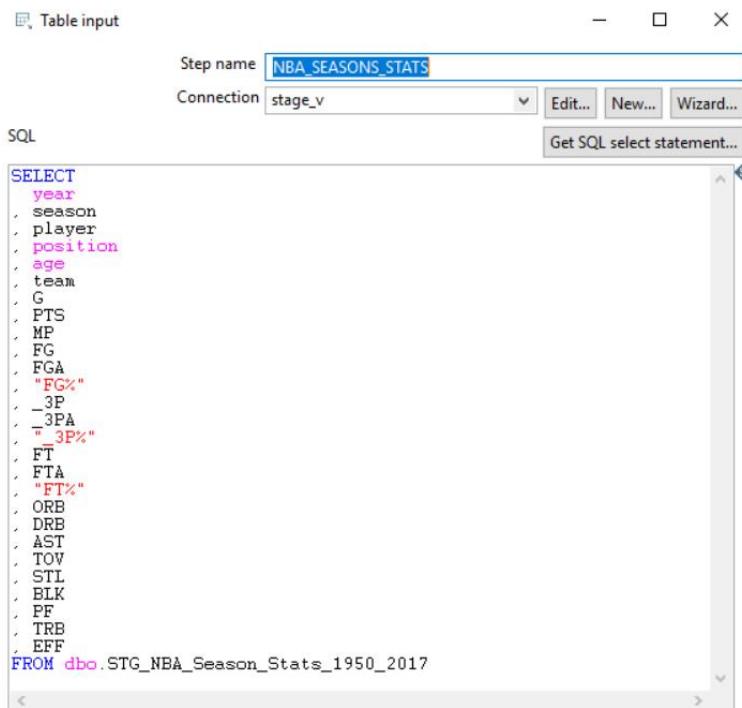
Para explicar la transformación dividiremos los pasos en dos bloques. El bloque para obtener los datos de estadísticas de la NBA y el bloque que lo hace para las estadísticas de la WNBA.

Para seguir como paso intermedio, se renombra los estadísticos equivalentes con el nombre que corresponde en la liga WNBA.

Una vez unidos los datos por los mismos campos, se obtiene el id de la temporada, el id de los jugadores (NBA), el id de los equipos, el id de la posición (caso de NBA), se seleccionan los datos de interés y se carga a la tabla de hechos.

Para el caso de la NBA estos son los pasos

- Se obtiene los estadísticos comunes (equivalentes en ligas) desde la STG_NBA_Seasons_Stats_1950_2017

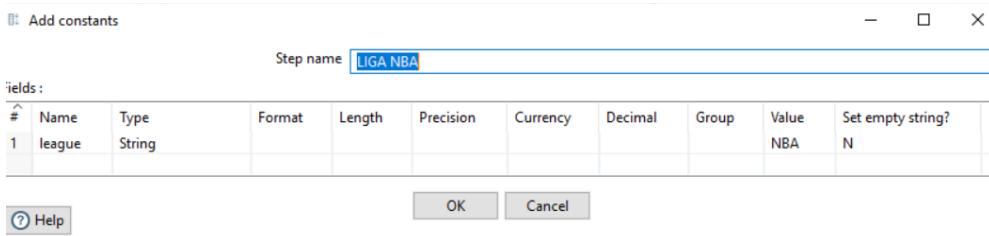


The screenshot shows the 'Table input' configuration window. The 'Step name' is set to 'NBA_SEASONS_STATS'. The 'Connection' is set to 'stage_v'. The 'SQL' field contains the following query:

```

SELECT
    year
    , season
    , player
    , position
    , age
    , team
    , G
    , PTS
    , MP
    , FG
    , FGA
    , "FG%"
    , -3P
    , -3PA
    , "-3P%"
    , FT
    , FTA
    , "FT%"
    , ORB
    , DRB
    , AST
    , TOV
    , STL
    , BLK
    , PF
    , TRB
    , EFF
FROM dbo.STG_NBA_Season_Stats_1950_2017
  
```

- Se genera campo con valor constante de la liga correspondiente (NBA)



- Se seleccionan los datos de interés y se realiza un sort indicando el orden de los campos, pues en el paso intermedio de unión con los datos de la WNBA será más sencillo.

Select values

Step name **Select values 2**

Select & Alter Remove Meta-data

Fields :

#	Fieldname	Rename to	Length	Precision
1	player			
2	age			
3	year			
4	season			
5	team			
6	league			
7	G			
8	PTS			
9	MP			
10	FG			
11	FGA			
12	FG%			
13	_3P			
14	_3PA			
15	_3P%			
16	FT			
17	FTA			
18	FT%			
19	ORB			
20	DRB			
21	TRB			
22	AST			
23	TOV			
24	STL			
25	BLK			
26	PF			
27	EFF			
28	position			

Include unspecified fields, ordered by name

Sort rows

Step name **Sort rows**

Sort directory **%%java.io.tmpdir%%**

TMP-file prefix **out**

Sort size (rows in memory) **1000000**

Free memory threshold (in %) **10**

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	player	Y	N	N	0	N
2	age	Y	N	N	0	N
3	year	Y	N	N	0	N
4	season	Y	N	N	0	N
5	team	Y	N	N	0	N
6	league	Y	N	N	0	N
7	G	Y	N	N	0	N
8	PTS	Y	N	N	0	N
9	MP	Y	N	N	0	N
10	FG	Y	N	N	0	N
11	FGA	Y	N	N	0	N
12	FG%	Y	N	N	0	N
13	_3P	Y	N	N	0	N
14	_3PA	Y	N	N	0	N
15	_3P%	Y	N	N	0	N
16	FT	Y	N	N	0	N
17	FTA	Y	N	N	0	N
18	FT%	Y	N	N	0	N
19	ORB	Y	N	N	0	N
20	DRB	Y	N	N	0	N
21	TRB	Y	N	N	0	N
22	AST	Y	N	N	0	N
23	TOV	Y	N	N	0	N

Para el caso de la WNBA estos son los pasos:

- Se cargan los datos de la STG_WNBA_Seasons_Stats_2005_2017

Step name: WNBA_SEASONS_STATS

Connection: stage_v

SQL:

```

SELECT
    id_wnba_player
    , player
    , age
    , active
    , season
    , temporada
    , team
    , league
    , GP
    , PTS
    , MIN
    , FGM
    , FGA
    , "FG%"
    , 3PM
    , 3PA
    , "3P%"
    , FTM
    , FTA
    , "FT%"
    , OREB
    , DREB
    , REB
    , AST
    , TOV
    , STL
    , BLK
    , PF
    , EFF
FROM dbo.STG_WNBA_Season_Stats_2005_2017

```

Line 1 Column 0

Store column info in step meta

Enable lazy conversion

Replace variables in script?

Insert data from step

Execute for each row?

Limit size: 0

OK Preview Cancel

- Se genera el campo de la liga correspondiente y se genera un campo referente a la posición de jugadoras (vacía), puesto que no contamos con este dato para las jugadoras de la liga WNBA.

Add constants

Step name: LIGA_WNBA

Fields:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	league	String							WNBA	N

OK Cancel

Add constants

Step name: POSITION

Fields:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Value	Set empty string?
1	position	String								Y

OK Cancel

- En la selección de los campos obtenidos para la liga WNBA, realizamos cambios de nombres, al mismo tiempo que especificamos el tipo de datos, de modo que se puedan integrar los datos.

Step name **Select values**

Select & Alter Remove Meta-data

Fields :

#	Fieldname	Rename to	Length	Precision	
1	player				
2	age				
3	season	year			
4	temporada	season			
5	team				
6	league				
7	GP	G			
8	PTS				
9	MIN	MP			
10	FGM	FG			
11	FGA				
12	FG%				
13	_3PM	_3P			
14	_3PA				
15	_3P%				
16	FTM	FT			
17	FTA				
18	FT%				
19	OREB	ORB			
20	DREB	DRB			
21	REB	TRB			
22	AST				
23	TOV				
24	STL				
25	BLK				
26	PF				
27	EFF				
28	position				

Include unspecified fields, ordered by name

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?
1	G		Integer			N
2	PTS		Integer			N
3	MP		String			N
4	FG		Integer			N
5	FGA		Integer			N
6	_3P		String			N
7	_3PA		String			N
8	_3P%		String			N
9	FT		Integer			N
10	FTA		Integer			N
11	ORB		String			N
12	DRB		String			N
13	TRB		String			N
14	AST		Integer			N
15	TOV		String			N
16	STL		String			N
17	BLK		String			N
18	PF		Integer			N
19	EFF		Integer			N

Una vez explicados los bloques, el siguiente paso trata de unir los datos. Ya que tenemos los datos con el mismo nombre de campo, es suficiente con indicar que la unión se hace en base al campo name.

Merge rows (diff)

Step name: Merge rows (diff)

Reference rows origin: Sort rows

Compare rows origin: Select values

Flag fieldname: flagfield

Keys to match :

#	Key field
1	player

Values to compare :

#	Value field
1	player

- Tras el paso, hemos obtenido datos vacíos correspondientes a la total de campos y que procedemos a filtrar.

Filter rows

Step name: FILTER PLAYER NULL

Send 'true' data to step: Complete Data

Send 'false' data to step:

The condition:

player IS NOT NULL

- Las siguientes capturas son las obtenciones del campo id_season (clave primaria de la dimensión DIM_Tiempo), id_player (clave primaria de los jugadores, en nuestro caso de la NBA), id_team (clave primaria de los equipos), id_positín (clave primaria de la dimensión de posiciones de juego, en este caso para los jugadores de la WNBA no está especificado por falta de datos). En estos casos se utiliza Database Lookup.

Database lookup

Step name: ID_SEASON

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_Tiempo

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	temporada	=	season	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_date			None

Database lookup

Step name: **ID_PLAYER**

Connection: **stage_v**

Lookup schema: **dbo**

Lookup table: **DIM_Jugadores**

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	=	player	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	pk_jugador			None

Database lookup

Step name: **ID_TEAM**

Connection: **stage_v**

Lookup schema: **dbo**

Lookup table: **DIM_Equipos**

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	codigo_equipo	=	team	
2	liga	=	league	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	pk_equipo			None

Database lookup

Step name: ID_POSICION

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_PosicionesJuego

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	codigo_posicion	=	position	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_posicion_juego			None

Finalmente se cargan los datos a la tabla de hechos FACT_SEASONS_STATS

Table output

Step name: Table output

Connection: dw_v

Target schema: dbo

Target table: FACT_SEASONS_STATS

Commit size: 1000

Truncate table:

Ignore insert errors:

Specify database fields:

Main options Database fields

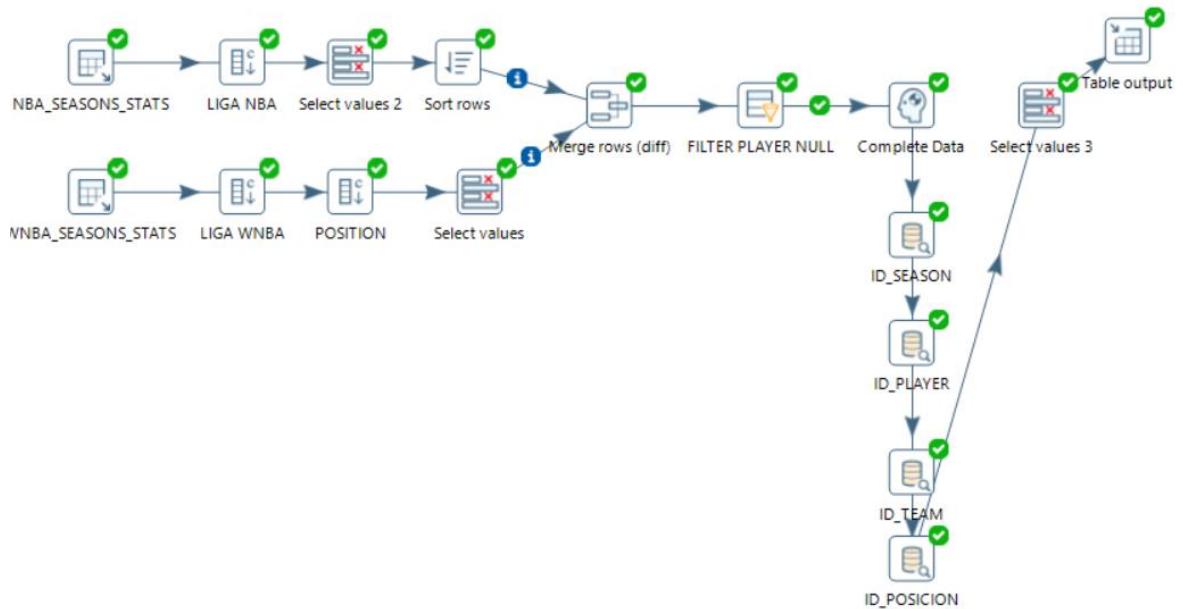
Fields to insert:

#	Table field	Stream field
1	id_season	pk_date
2	id_position	pk_posicion_j...
3	id_team	pk_equipo
4	id_player	pk_jugador
5	league	league
6	G	G
7	player	player
8	PTS	PTS
9	MP	MP
10	FG	FG
11	FGA	FGA
12	FG%	FG%
13	_3P	_3P
14	_3PA	_3PA
15	_3P%	_3P%
16	FT	FT
17	FTA	FTA
18	FT%	FT%
19	ORB	ORB
20	DRB	DRB

Get fields Enter field mapping

OK Cancel SQL

La transformación completa y los resultados de la ejecución de la transformación son los siguientes:



#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active
1	WNBA_SEASONS_STATS	0	0	2085	2085	0	0	0	0	Finished
2	LIGA WNBA	0	2085	2085	0	0	0	0	0	Finished
3	NBA_SEASONS_STATS	0	0	24691	24691	0	0	0	0	Finished
4	LIGA NBA	0	24691	24691	0	0	0	0	0	Finished
5	POSITION	0	2085	2085	0	0	0	0	0	Finished
6	Select values 2	0	24691	24691	0	0	0	0	0	Finished
7	Select values	0	2085	2085	0	0	0	0	0	Finished
8	Sort rows	0	24691	24691	0	0	0	0	0	Finished
9	Merge rows (diff)	0	26776	26776	0	0	0	0	0	Finished
10	FILTER PLAYER NULL	0	26776	26709	0	0	0	0	0	Finished
11	Complete Data	0	26709	26709	0	0	0	0	0	Finished
12	ID_SEASON	0	26709	26709	26709	0	0	0	0	Finished
13	ID_PLAYER	0	26709	26709	22786	0	0	0	0	Finished
14	ID_TEAM	0	26709	26709	14543	0	0	0	0	Finished
15	ID_POSICION	0	26709	26709	23883	0	0	0	0	Finished
16	Select values 3	0	26709	26709	0	0	0	0	0	Finished
17	Table output	0	26709	26709	0	26709	0	0	0	Finished

Transformación TR_FACT_FREE_THROWS

Mediante esta transformación poblaremos la tabla de hechos FACT_FREE_THROWS según el modelo multidimensional que contiene información que nos permite realizar el análisis de los datos de los tiros libres.

Los pasos que se siguen para realizar la transformación son los siguientes: carga de datos desde la tabla intermedia STG_NBA_free_throws, obtención del campo año para realizar la

comparación de valores al obtener la clave primaria de la DIM_Tiempo, obtención de la clave primaria de DIM_Tiempo, obtención clave primaria DIM_Partidos, obtención de la clave primera de la DIM_Jugadores, obtención de la clave primaria de la DIM_Minuto.

- Carga de los campos desde la tabla intermedia

Step name: Table input
Connection: stage_v
SQL:

```
SELECT
    game
,   period
,   play
,   player
,   play_cod
,   season
,   play_tipo
,   time
,   score
FROM dbo.STG_NBA_free_throws
```

- Se obtiene el año , acotando el campo season (string) desde la primera hasta la cuarta posición y se transforma a valor numérico.

Step name: Obtain year

#	In stream field	Out stream field	Cut from	Cut to
1	season	year	0	4

Step name: Year format

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?
1	year		Number	4	N	F

- En los siguientes pasos se trata de obtener las claves primarias mencionadas al inicio.

Database lookup

Step name: ID_SEASON

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_Tiempo

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	date_year	=	year	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_date		None	
2	temporada		None	

Database lookup

Step name: ID_PARTIDO

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_Partidos

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	partido	=	game	
2	temporada	=	temporada	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_partido		None	

Database lookup

Step name: ID_JUGADOR

Connection: dw_v

Lookup schema: dbo

Lookup table: DIM_Jugadores

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	=	player	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_jugador		None	

Database lookup

Step name: **ID_MINUTO**

Connection: **dw_v**

Lookup schema: **dbo**

Lookup table: **DIM_Minutos**

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	minutoSegundo	=	time	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_minutoSegundo			None

Database lookup

Step name: **ID_JUGADA**

Connection: **dw_v**

Lookup schema: **dbo**

Lookup table: **DIM_Jugadas**

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	dec_jugada	=	play_tipo	

Values to return from the lookup table:

#	Field	New name	Default	Type
1	pk_jugada			None

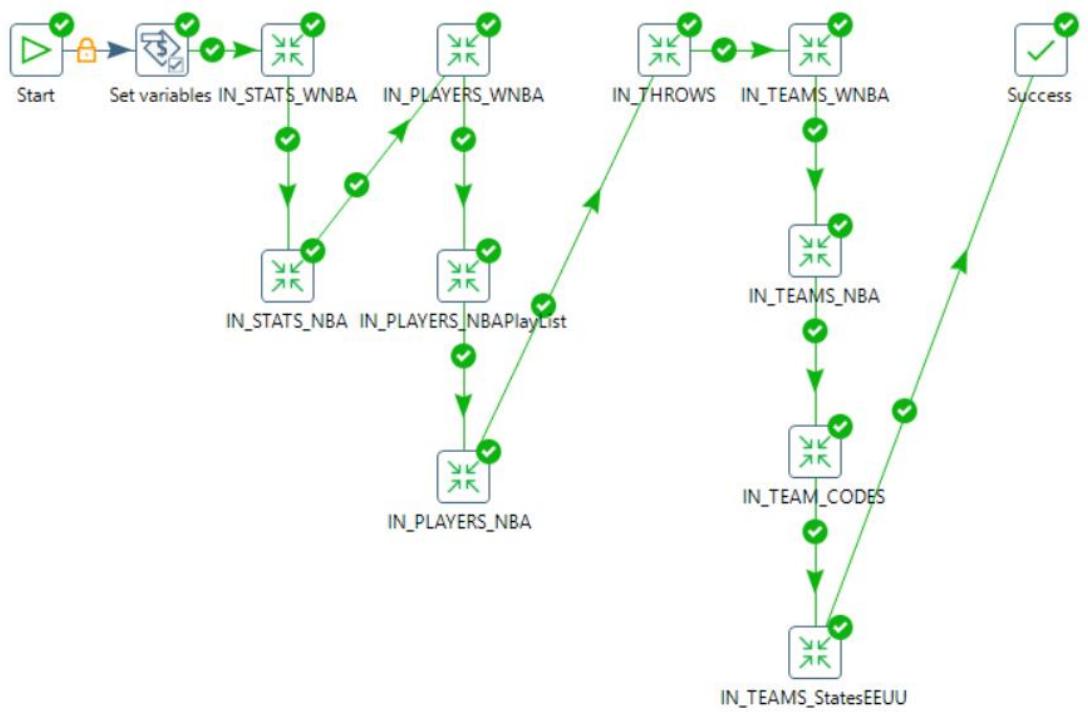
III. Implementación de trabajos con procesos ETL

Diseñamos los trabajos (jobs) mediante PDI que van a permitir la ejecución secuencial de todos los procesos ETL incluidos en cada bloque definido

Trabajo JOB_IN

El trabajo (job) JOB_IN procesa todas las transformaciones del bloque IN_ para la carga de datos desde las fuentes de datos proporcionadas al área intermedia.

El diseño es el siguiente:



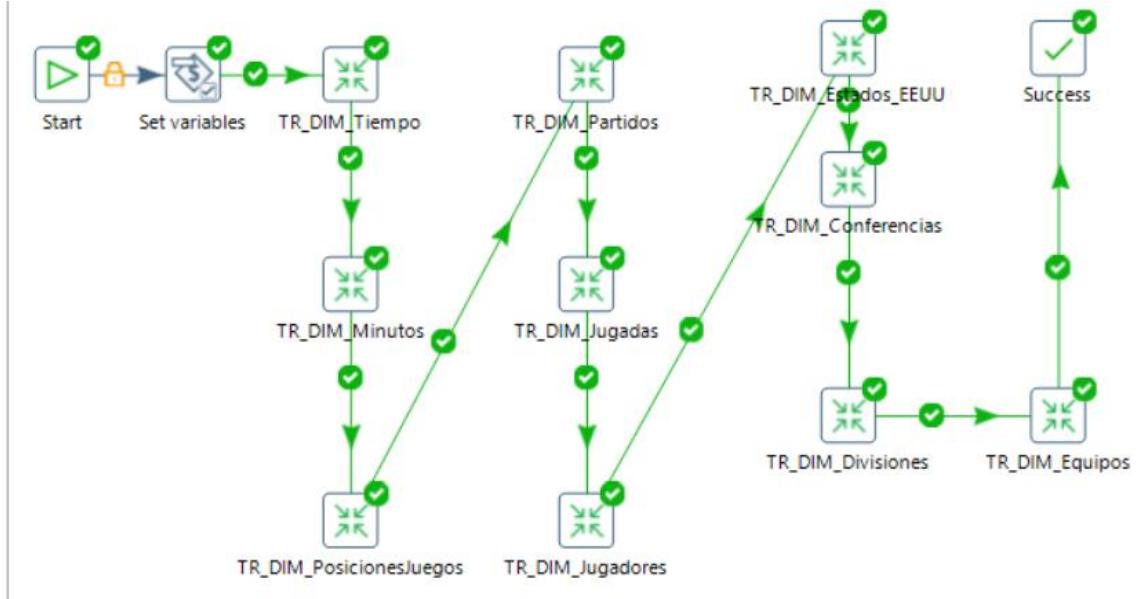
El resultado de la ejecución de la transformación es el siguiente:

Execution Results				
Job / Job Entry	Comment	Result	Reason	Filename
JOB_IN				
Job: JOB_IN	Start of job execution		start	
Start	Start of job execution		start	
Start	Job execution finished	Success		
Set variables	Start of job execution		Followed unconditional link	
Set variables	Job execution finished	Success		
IN_STATS_WNBA	Start of job execution		Followed link after success	
IN_STATS_WNBA	Job execution finished	Success		
IN_STATS_NBA	Start of job execution		Followed link after success	
IN_STATS_NBA	Job execution finished	Success		
IN_PLAYERS_NBA	Start of job execution		Followed link after success	
IN_PLAYERS_NBA	Job execution finished	Success		
IN_PLAYERS_NBAPlayList	Start of job execution		Followed link after success	
IN_PLAYERS_NBAPlayList	Job execution finished	Success		
IN_PLAYERS_NBA	Start of job execution		Followed link after success	
IN_PLAYERS_NBA	Job execution finished	Success		
IN_THROWS	Start of job execution		Followed link after success	
IN_THROWS	Job execution finished	Success		
IN_TEAMS_WNBA	Start of job execution		Followed link after success	
IN_TEAMS_WNBA	Job execution finished	Success		
IN_TEAMS_NBA	Start of job execution		Followed link after success	
IN_TEAMS_NBA	Job execution finished	Success		
IN_TEAM_CODES	Start of job execution		Followed link after success	
IN_TEAM_CODES	Job execution finished	Success		
IN_TEAMS_StatesEEUU	Start of job execution		Followed link after success	
IN_TEAMS_StatesEEUU	Job execution finished	Success		
Success	Start of job execution		Followed link after success	
Success	Job execution finished	Success		
Job: JOB_IN	Job execution finished	Success	finished	

Trabajo JOB_TR_DIM

El trabajo (job) JOB_TR_DIM procesa todas las transformaciones del bloque TR_DIM para la carga de datos desde las tablas intermedias a las tablas de dimensiones del almacén.

El diseño del trabajo (job) JOB_TR_DIM es el siguiente:



Se observa del resultado de la ejecución del trabajo, que todos los pasos se procesan con éxito.

Execution Results				
Job / Job Entry	Comment	Result	Reason	Filename
Job: JOB_TR_DIM	Start of job execution		start	
Start	Start of job execution		start	
Start	Job execution finished	Success		
Set variables	Start of job execution		Followed unconditional link	
Set variables	Job execution finished	Success		
TR_DIM_Tiempo	Start of job execution		Followed link after success	
TR_DIM_Tiempo	Job execution finished	Success		
TR_DIM_Minutos	Start of job execution		Followed link after success	
TR_DIM_Minutos	Job execution finished	Success		
TR_DIM_PosicionesJuegos	Start of job execution		Followed link after success	
TR_DIM_PosicionesJuegos	Job execution finished	Success		
TR_DIM_Partidos	Start of job execution		Followed link after success	
TR_DIM_Partidos	Job execution finished	Success		
TR_DIM_Jugadas	Start of job execution		Followed link after success	
TR_DIM_Jugadas	Job execution finished	Success		
TR_DIM_Jugadores	Start of job execution		Followed link after success	
TR_DIM_Jugadores	Job execution finished	Success		
TR_DIM_Estados_EEUU	Start of job execution		Followed link after success	
TR_DIM_Estados_EEUU	Job execution finished	Success		
TR_DIM_Conferencias	Start of job execution		Followed link after success	
TR_DIM_Conferencias	Job execution finished	Success		
TR_DIM_Divisiones	Start of job execution		Followed link after success	
TR_DIM_Divisiones	Job execution finished	Success		
TR_DIM_Equipos	Start of job execution		Followed link after success	
TR_DIM_Equipos	Job execution finished	Success		
Success	Start of job execution		Followed link after success	
Success	Job execution finished	Success		
Job: JOB_TR_DIM	Job execution finished	Success	finished	

IV. Diseño de modelo OLAP