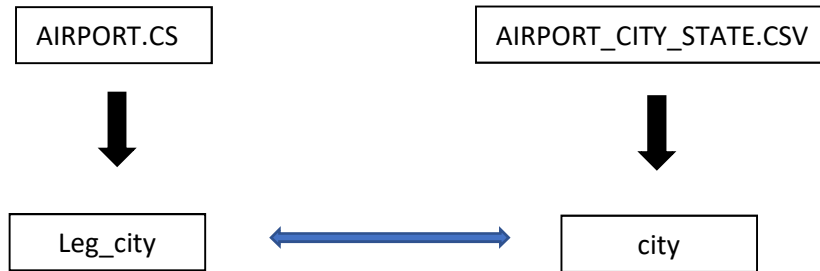


Task 3: Data transformation

El principal problema de esta tarea es la unión de las tablas (CSV) AIRPORT.CSV Y AIRPORT_CITY_STATE.CSV. Para conseguir esta unión hay que identificar la columna que comparten para que la unión sea efectiva. En este caso, el atributo/columna por la que hemos unido es:



Para conseguir esto se hace uso de la herramienta MERGE que incluye Pentaho Data Integration.

A continuación, se mostrarán paso a paso como se ha conseguido la carga de datos de la dimensión AIRPORT:

1. Se hace uso de CSV file input con airport.csv

The screenshot displays a Pentaho Data Integration (PDI) job configuration and its execution results. The job flow includes the following steps:

- CSV file input airport
- Sort rows
- Merge Join
- Add sequence
- Table output airport

The 'CSV Input' dialog box is open, showing the configuration for the 'CSV file input airport' step:

- Step name: CSV file input airport
- Filename: C:\Users\master\Documents\AIRPORT\Data\airport.csv
- Delimiter: ;
- Enclosure: "
- NIO buffer size: 50000
- Lazy conversion? ☒
- Header row present? ☒
- Add filename to result ☐
- The row number field name (optional):
- Running in parallel? ☐
- New line possible in fields? ☐
- File encoding: UTF-8

The 'Execution Results' tab shows the following data:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	leg_key	Integer	#	15	0	\$.		none
2	leg_name	String		15		\$.		none
3	leg_city	String		14		\$.		none
4	leg_type	String		13		\$.		none
5	leg_radar_type	String		3		\$.		none

2. Se debe de ordenar airport.csv ya que, de no ser así, la unión de los dos CSV no se haría correctamente debido a que la herramienta MERGE

Diagrama de flujo de datos:

- CSV file input airport → Sort rows → Merge Join
- CSV file input 2 airport_city_state → Sort rows 2 → Merge Join
- Merge Join → Add sequence → Table output airport

Configuración de la herramienta 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 %):
- 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	leg_city	Y	N	N	0	N

3. Se hace uso de CSV Input con el archivo airport_city_state.csv

Diagrama de flujo de datos:

- CSV file input airport → Sort rows → Merge Join
- CSV file input 2 airport_city_state → Sort rows 2 → Merge Join
- Merge Join → Add sequence → Table output airport

Configuración de la herramienta CSV Input:

- Step name: CSV file input 2 airport_city_state
- Filename: C:\Users\master\Documents\AIRPORT\Data\airport_city_state.csv
- Delimiter: ;
- Enclosure: "
- NIO buffer size: 50000
- Lazy conversion? ☒
- Header row present? ☒
- Add filename to result ☐
- The row number field name (optional):
- Running in parallel? ☐
- New line possible in fields? ☐
- File encoding:

Fields:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	city	String		14		\$.	,	none
2	state	String		2		\$.	,	none

4. Se ordena también este csv

The diagram shows a workflow with two input CSV files: 'CSV file input airport' and 'CSV file input 2 airport_city_state'. Both feed into 'Sort rows' and 'Sort rows 2' respectively. These then merge into a 'Merge Join' step, followed by 'Add sequence' and finally 'Table output airport'.

The 'Sort rows' configuration window is shown with the following settings:

- Step name: Sort rows 2
- Sort directory: %%java.io.tmpdir%%
- TMP-file prefix: out
- Sort size (rows in memory): 1000000
- Free memory threshold (in %):
- Compress TMP Files?: ☐
- Only pass unique rows? (verifies keys only): ☐

Fields table:

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	city	Y	N	N	0	N

5. Una vez ordenado por el atributo que comparten, se procede a unir ambos CSV

The diagram shows the same workflow as before, but with the 'Merge Join' step highlighted. The 'Sort rows' and 'Sort rows 2' steps are now connected to 'Merge Join'.

The 'Merge Join' configuration window is shown with the following settings:

- Step name: Merge Join
- First Step: Sort rows
- Second Step: Sort rows 2
- Join Type: INNER

Keys for 1st step:

#	Key field
1	leg_city

Keys for 2nd step:

#	Key field
1	city

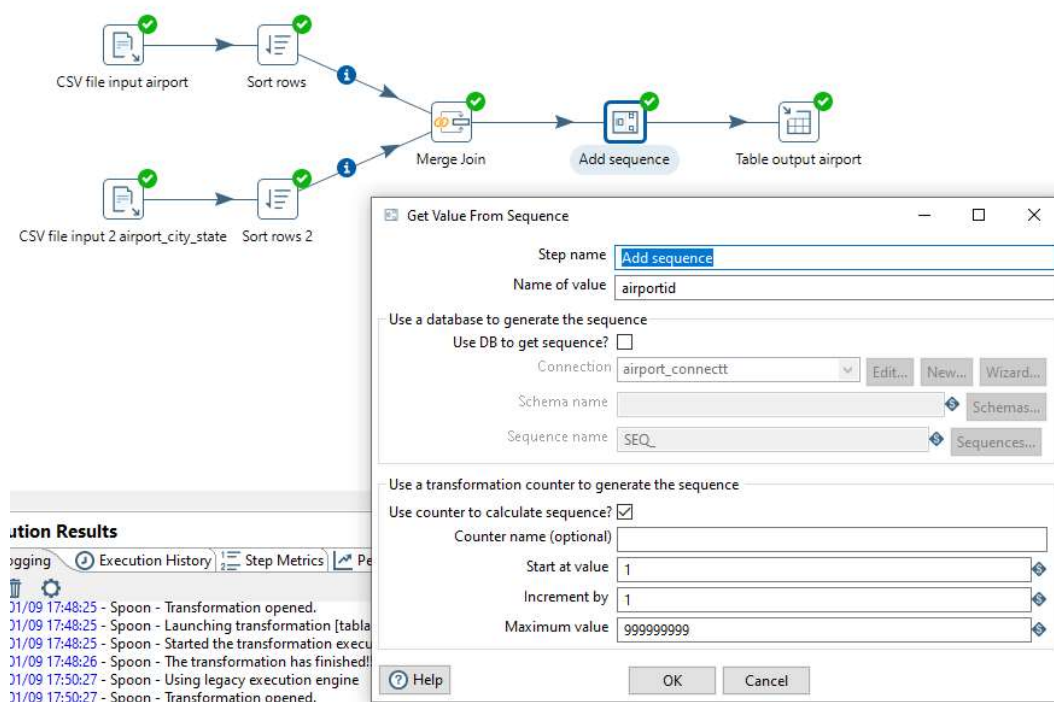
Buttons: Get key fields, Get key fields, Help, OK, Cancel.

Execution Results:

Logging | Execution History | Step Metrics | Performance

3/01/09 17:48:25 - Spoon - Transformation opened.
3/01/09 17:48:25 - Spoon - Launching transformation [tabla de datos]
3/01/09 17:48:25 - Spoon - Started the transformation execution.
3/01/09 17:48:26 - Spoon - The transformation has finished!!
3/01/09 17:50:27 - Spoon - Using legacy execution engine

6. Se crea el atributo airportid para identificar unánimemente a cada fila.



The diagram shows a workflow with two input paths: 'CSV file input airport' and 'CSV file input 2 airport_city_state'. Both paths go through 'Sort rows' and then merge at a 'Merge Join' step. The output of the 'Merge Join' goes to an 'Add sequence' step, which then feeds into a 'Table output airport'.

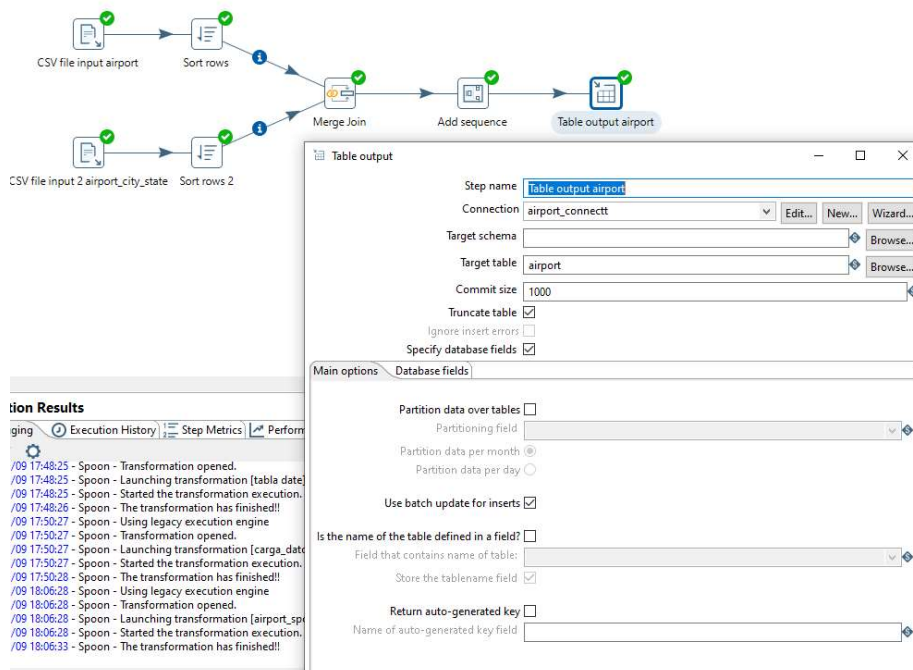
The 'Get Value From Sequence' dialog box is open, showing the configuration for the 'Add sequence' step. The 'Name of value' is set to 'airportid'. The 'Use a database to generate the sequence' section is unchecked. The 'Use a transformation counter to generate the sequence' section is checked, with 'Start at value' set to 1, 'Increment by' set to 1, and 'Maximum value' set to 999999999.

Execution Results

- 11/09 17:48:25 - Spoon - Transformation opened.
- 11/09 17:48:25 - Spoon - Launching transformation [table output airport].
- 11/09 17:48:25 - Spoon - Started the transformation execution.
- 11/09 17:48:26 - Spoon - The transformation has finished!!
- 11/09 17:50:27 - Spoon - Using legacy execution engine.
- 11/09 17:50:27 - Spoon - Transformation opened.

7. Por último, se carga la unión de estos CSV en la tabla creada anteriormente en el programa MYSQL

Para cargar estos datos, hay que seleccionar tanto la tabla a la que se va a cargar como la relación de las columnas del CSV con las de la tabla de MYSQL



The diagram shows the same workflow as in step 6, but the 'Add sequence' step is now highlighted in blue, indicating it is the active step.

The 'Table output' dialog box is open, showing the configuration for the 'Table output airport' step. The 'Connection' is set to 'airport_connectt'. The 'Target schema' is set to 'airport' and the 'Target table' is set to 'airport'. The 'Commit size' is set to 1000. The 'Truncate table' checkbox is checked. The 'Specify database fields' checkbox is checked. The 'Main options' tab is selected, showing 'Partition data over tables' unchecked, 'Partitioning field' set to 'airportid', 'Partition data per month' selected, and 'Use batch update for inserts' checked. The 'Is the name of the table defined in a field?' checkbox is unchecked, and the 'Return auto-generated key' checkbox is unchecked.

Execution Results

- 11/09 17:48:25 - Spoon - Transformation opened.
- 11/09 17:48:25 - Spoon - Launching transformation [table output airport].
- 11/09 17:48:25 - Spoon - Started the transformation execution.
- 11/09 17:48:26 - Spoon - The transformation has finished!!
- 11/09 17:50:27 - Spoon - Using legacy execution engine.
- 11/09 17:50:27 - Spoon - Transformation opened.
- 11/09 17:50:27 - Spoon - Launching transformation [carga_datos].
- 11/09 17:50:27 - Spoon - Started the transformation execution.
- 11/09 17:50:28 - Spoon - The transformation has finished!!
- 11/09 18:06:28 - Spoon - Using legacy execution engine.
- 11/09 18:06:28 - Spoon - Transformation opened.
- 11/09 18:06:28 - Spoon - Launching transformation [airport_spo].
- 11/09 18:06:28 - Spoon - Started the transformation execution.
- 11/09 18:06:33 - Spoon - The transformation has finished!!

Diagram showing a data flow process:

- CSV file input airport
- Sort rows
- CSV file input 2 airport_city_state
- Sort rows 2
- Merge Join
- Add sequence
- Table output airport

Table output configuration window:

Step name: **Table output airport**

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

Target schema: [Browse...]

Target table: **airport** [Browse...]

Commit size: **1000**

Truncate table: ☒

Ignore insert errors: ☐

Specify database fields: ☒

Main options / Database fields

Fields to insert:

#	Table field	Stream field
1	leg_key	leg_key
2	leg_name	leg_name
3	leg_city	leg_city
4	leg_type	leg_type
5	leg_radar_t...	leg_radar_type
6	airportID	airportid
7	state	state

Buttons: Get fields, Enter field mapping

Execution History:

- 1/09 17:48:25 - Spoon - Transformation opened.
- 1/09 17:48:25 - Spoon - Launching transformation [tabla date
- 1/09 17:48:25 - Spoon - Started the transformation execution.
- 1/09 17:48:26 - Spoon - The transformation has finished!!
- 1/09 17:50:27 - Spoon - Using legacy execution engine
- 1/09 17:50:27 - Spoon - Transformation opened.
- 1/09 17:50:27 - Spoon - Launching transformation [carga_data
- 1/09 17:50:27 - Spoon - Started the transformation execution.
- 1/09 17:50:28 - Spoon - The transformation has finished!!
- 1/09 18:05:38 - Spoon - Using legacy execution engine