SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

https://www.123rf.com/photo_130776341_st 1

BUSINESS INTELLIGENCE & DATA WAREHOUSE

GROUP ASSIGNMENT #2

DATA INTEGRATION DESIGN

GROUP D

Anxhelo Zylyftari
Amos Mtaita
Jacobo De Leon Garcia
Jose Ricardo Gomes Dos Santos
Iman Hdairis

ABSTRACT

This project consists of designing a one-time historic load of the German Farm Subsidies data using an ETL (Extraction, Transformation and Loading) tool.

At the end of this project the following ETL documents will be ready:

Data Mapping document: A document that describes the default strategy for extracting data from data source, data mapping process, and data quality tracking and metadata approach.

ETL Scripts: The collection of ETL processes created with a data integration tool (i.e. PDI to be specific)

Database backup: A backup file of the database and schema was created after the ETL process has been executed successfully

OBJECTIVE OF THE ETL PROCESS:

- **Extracting:** Gathering raw data from the source systems and writing it to disk in the ETL environment before any significant restructuring of the data takes place.

- **Cleaning and conforming:** Sending source data through a series of processing steps in the ETL system to improve the quality of the data received from the source, and merging data from two or more sources to create and enforce con-formed dimensions and conformed metrics.

- **Delivering:** Physically structuring and loading the data into the presentation server's target dimensional models.

- **Managing.** Managing the related systems and processes of the ETL environment in a coherent manner.

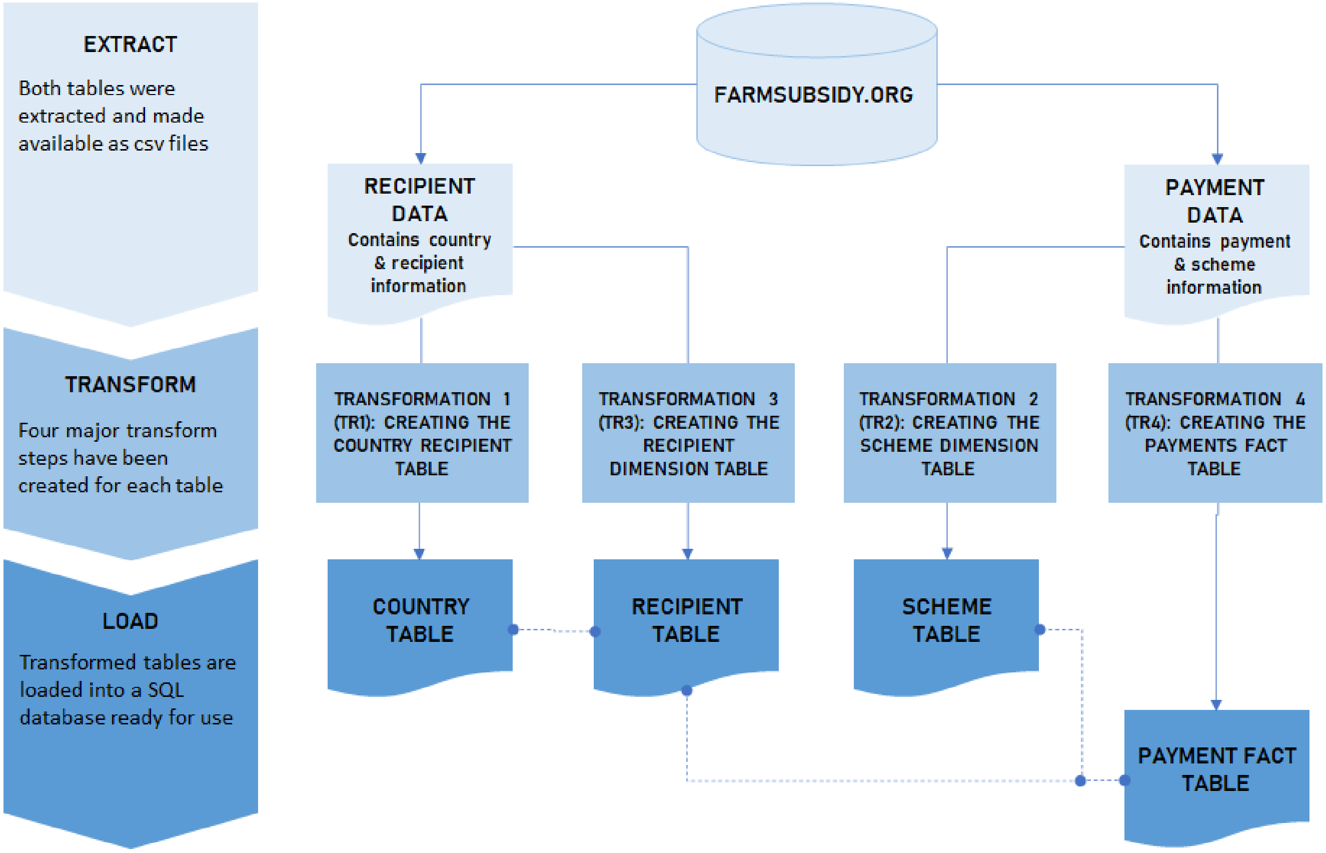
1.0 DEVELOP ETL PLAN

A high-level plan is beneficial in order to avoid redesign and rework during the process.

1.1 DRAW THE HIGH-LEVEL PLAN

The high-level schematic provides an overview of the ETL process from the source to the final tables that were loaded into the database. The German farm subsidies data is gathered from two main files; **Recipients (see appendix 1)** file as well as the **Payments (see appendix 2)** file.

High Level Data Staging Plan Schematic


EXTRACT
Both tables were extracted and made available as csv files
TRANSFORM
Four major transform steps have been created for each table
LOAD
Transformed tables are loaded into a SQL database ready for use
FARMSUBSIDY.ORG
RECIPIENT DATA
Contains country & recipient information
PAYMENT DATA
Contains payment & scheme information
TRANSFORMATION 1 (TR1): CREATING THE COUNTRY RECIPIENT TABLE
TRANSFORMATION 3 (TR3): CREATING THE RECIPIENT DIMENSION TABLE
TRANSFORMATION 2 (TR2): CREATING THE SCHEME DIMENSION TABLE
TRANSFORMATION 4 (TR4): CREATING THE PAYMENTS FACT TABLE
COUNTRY TABLE
RECIPIENT TABLE
SCHEME TABLE
PAYMENT FACT TABLE

1.2 CHOOSE AN ETL TOOL

There are multiple ETL tools available in the market. Pentaho is the ETL tool used for this course. Pentaho Data Integration features and benefits include: quick installations, 100% Java with cross platform support for Windows, Linux, and Macintosh. Easy to use graphical designer with over 100 out-of-the-box mapping objects including inputs, transforms, and outputs.

https://help.pentaho.com/Documentation/6.0/0J0/0C0/000

1.3 DEVELOP DEFAULT STRATEGIES

Now that the overview plan has been drafted and the ETL tool has been selected, a set of default strategies were developed for the activities in the ETL system.

Kimball highlighted 7 activities in The Data Warehouse Toolkit book, however in this project only the following activities were taken:

1. **Extract from each major source system.**

   The default method for extracting data from each source system was identified. The **Recipient** and **Payments** tables were inputted as csv files into Pentaho. The Schema in mySQL workbench was integrated into transformations.

2. **Police data quality for dimensions and particularly facts.**

   Kimball emphasizes that this is the step that ETL systems add value to the data. The data quality must be monitored during the ETL process rather than waiting for business users to find data problems. Therefore, the following measures were taken:

   For the **Recipient table** (german.subsidies.recipient.csv):

   - Removed the **address2, geo_4, geo1_natlang, geo2_natlang, geo3_natlang, geo4_natlang** columns as they don't contain any data and are not relevant.
   - Deleted **countrypayment** because it contained DE for all rows
   - In the **countryrecipnt** column, the "99" values were replaced by "DE".
   - In the **name** column, the "Unknown recipient" values were replaced with a blank (to do easier Queries using Null functions).
   - In the **geo_1** column, the "Unknown" values were replaced with a blank.
   - In the **town** column, there were 5 values that included the zip code, these zip code values were removed and only the town was kept.

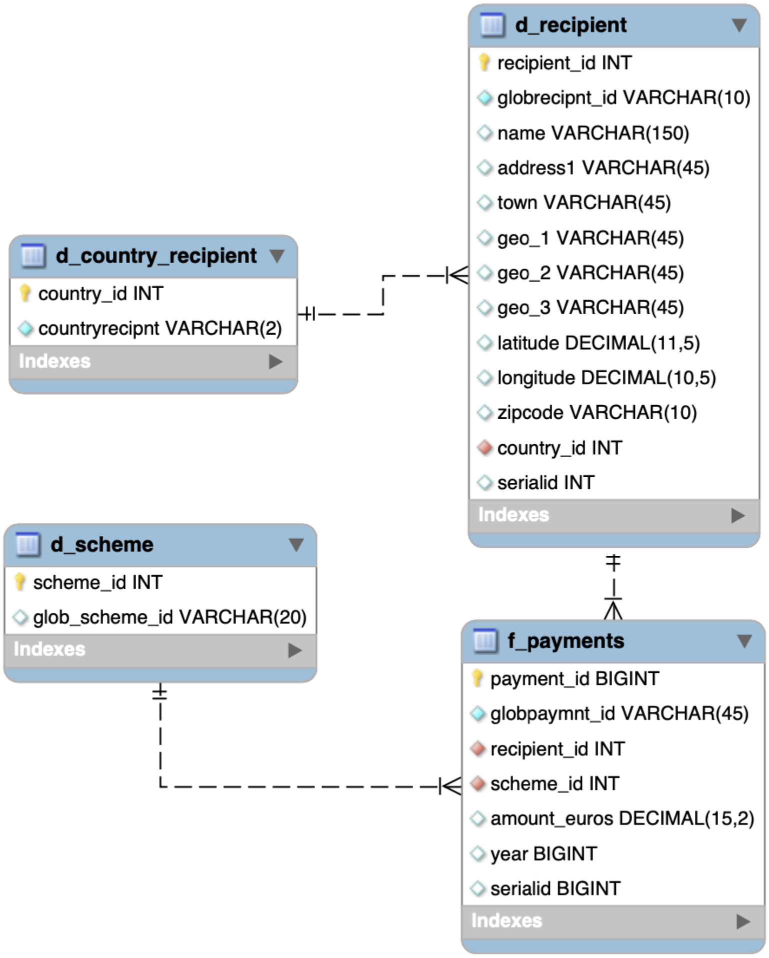
   One recommendation is to put constraints on the data entry user, whereby some of these steps could be removed. For example, the data type for countryrecipnt could be predetermined as string.

   For the **Payments table** (german.subsidies.payments.csv):

   - The **globrecipnt_idx** attribute is used to identify the recipients of payments, however given that the **globalrecipnt_id** field is already a unique identifier, it was decided to consider the field as redundant and therefore it was removed.
   - Removed the **amount_nat_cur** as it doesn't contain any data and is not relevant.
   - Deleted **countrypayment** because it contained DE for all rows.

1.4 UPDATED SNOWFLAKE SCHEMA

Taking the above points into account, below screenshot is the amended schema.



We decided to remove the Country_Payment table because there was only one row "DE" which made it irrelevant.

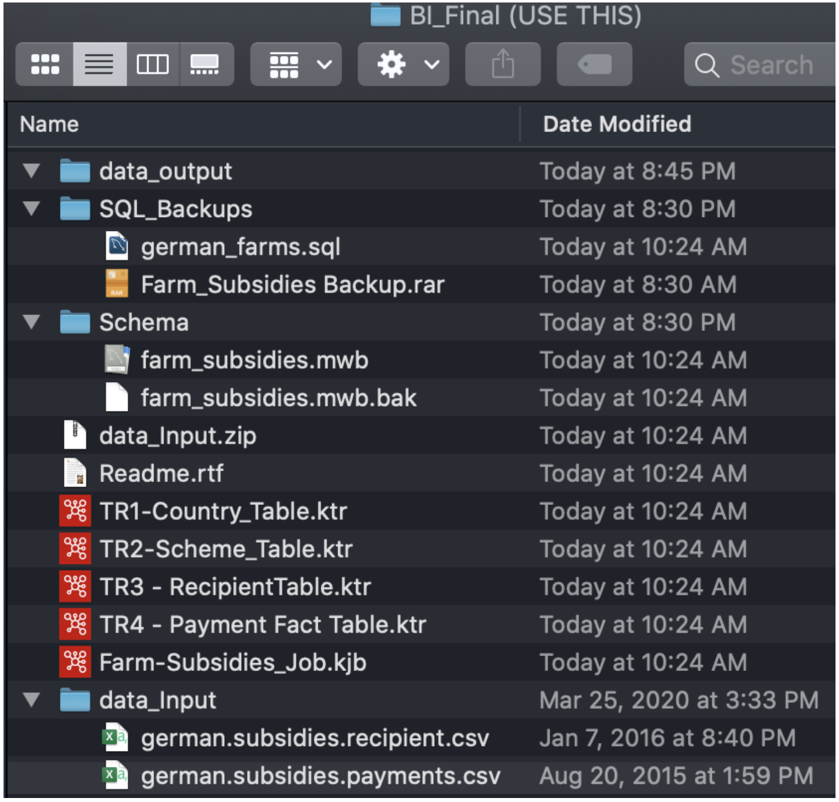
We also removed the Recipient Location table because of the missing values in the location attributes we had in the recipient table (for example zipcode attribute was missing 492 values).

1.5 EXECUTING THE PROJECT

In order to run the transformations and job smoothly, it is necessary to follow the following steps:

- Unzip the “data_input” folder
- Upload the .mwb file on MySQL Workbench and Forward Engineer to generate the schema
- Use the job/transformations directly from the enclosed folder (do not move the files from their parent folder)

Note: the contents of the parent file should be as follows:



BI_Final (USE THIS)

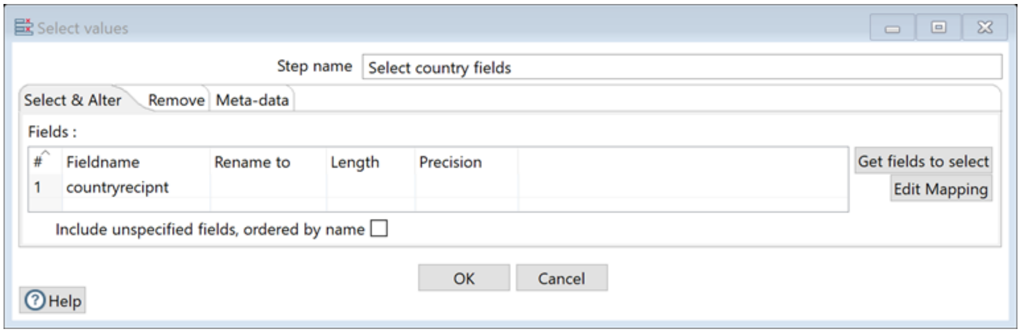
Name	Date Modified
▼ data_output	Today at 8:45 PM
▼ SQL_Backups	Today at 8:30 PM
german_farms.sql	Today at 10:24 AM
Farm_Subsidies Backup.rar	Today at 8:30 AM
▼ Schema	Today at 8:30 PM
farm_subsidies.mwb	Today at 10:24 AM
farm_subsidies.mwb.bak	Today at 10:24 AM
data_Input.zip	Today at 10:24 AM
Readme.rtf	Today at 10:24 AM
TR1-Country_Table.ktr	Today at 10:24 AM
TR2-Scheme_Table.ktr	Today at 10:24 AM
TR3 - RecipientTable.ktr	Today at 10:24 AM
TR4 - Payment Fact Table.ktr	Today at 10:24 AM
Farm-Subsidies_Job.kjb	Today at 10:24 AM
▼ data_Input	Mar 25, 2020 at 3:33 PM
german.subsidies.recipient.csv	Jan 7, 2016 at 8:40 PM
german.subsidies.payments.csv	Aug 20, 2015 at 1:59 PM

2.0 DATA TRANSFORMATION STEPS

2.1 TRANSFORMATION 1 (TR1): CREATING THE COUNTRY RECIPIENT TABLE D_COUNTRY_RECIPIENT

Step 1 - Input recipient Table: The recipient table is loaded into PDI by using the 'input CSV file' connector as the data provided from the source was in this format. Once loaded, run this step to ensure data is added into PDI ready for transformation. (see appendix 1)

Step 2 - Select Fields needed in the Country Recipient Table: The 'Select Values' transformation phase is used to eliminate the columns that will not be needed in the final fact table. After adding this step run the system to ensure only needed columns remain in the table.



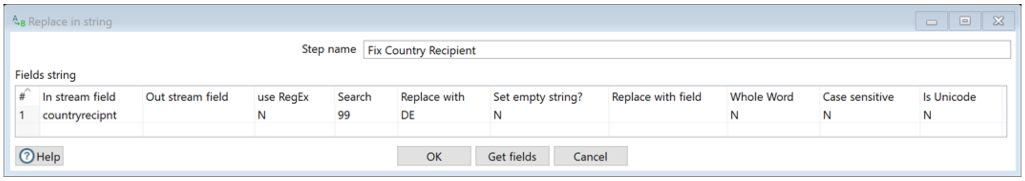
Step name: Select country fields

Fields :

#	Fieldname	Rename to	Length	Precision
1	countryrecipnt			

Include unspecified fields, ordered by name □

Step 3 - Fix Country Recipient: As part of ensuring data quality, the "99" values in the countryrecipnt column are replaced by "DE".

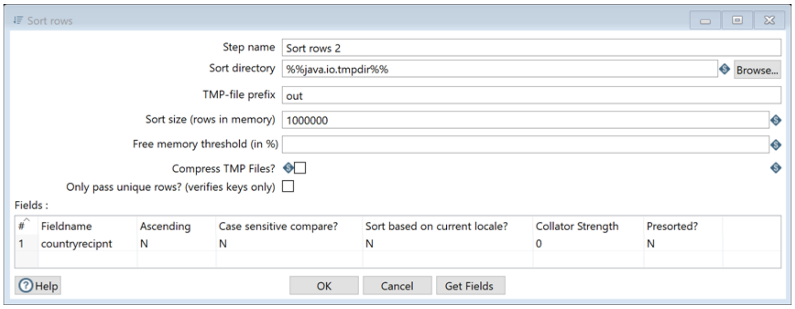


Step name: Fix Country Recipient

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	countryrecipnt		N	99	DE	N		N	N	N

Step 4 - Sort the table based on the Country Recipient Table: The table was sorted so we can extract only unique values in the next transformation step.

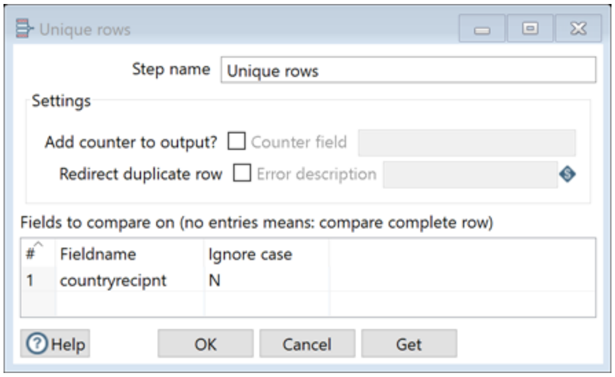


Step name: Sort rows 2
Sort directory: %%java.io.tmpdir%%
TMP-file prefix: out
Sort size (rows in memory): 1000000
Free memory threshold (in %):
Compress TMP Files? □
Only pass unique rows? (verifies keys only) □

Fields :

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

Step 5 - Extract only Unique countryrecipnt Values: Unique values were extracted ensuring data quality.



Step name: Unique rows

Settings

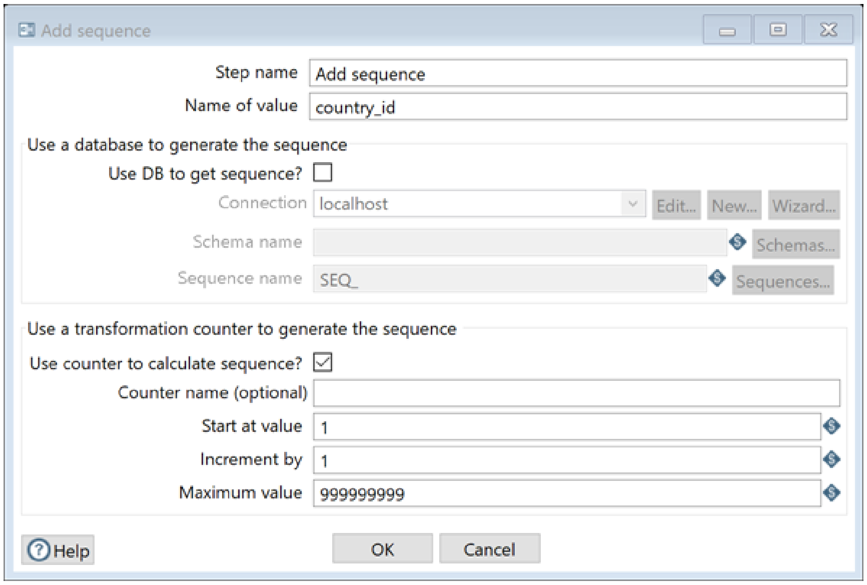
Add counter to output? □ Counter field

Redirect duplicate row □ Error description

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

#	Fieldname	Ignore case
1	countryrecipnt	N

Step 7 - Add sequence country_id: The sequence step was used to create the country_id which was added as the primary key.



Step name: Add sequence

Name of value: country_id

Use a database to generate the sequence

Use DB to get sequence? □

Connection: localhost

Schema name:

Sequence name: SEQ_

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☑

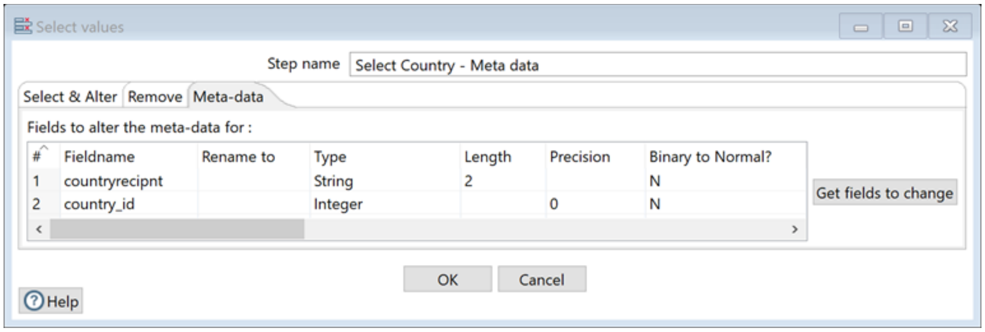
Counter name (optional):

Start at value: 1

Increment by: 1

Maximum value: 999999999

Step 8 - Select Metadata for all Columns: Review the metadata of the table now including the country_id to ensure the correct data type. Run this step to ensure all adjusted metadata information matches with the data in the table.

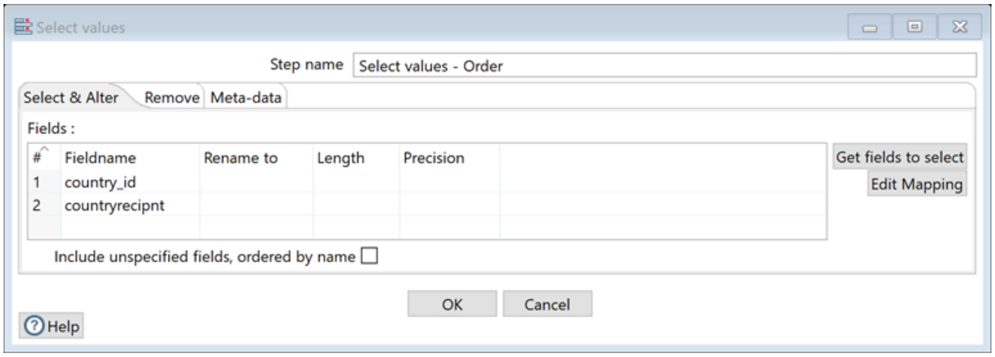


Step name: Select Country - Meta data

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?
1	countryrecipnt		String	2		N
2	country_id		Integer		0	N

Step 9 - Reorder the Table Columns : Reordering the table to prepare the table for integration in the database.



Step name: Select values - Order

Fields :

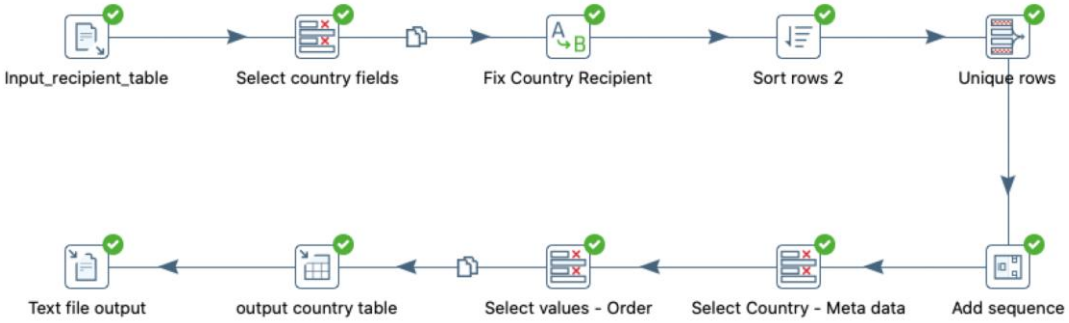
#	Fieldname	Rename to	Length	Precision
1	country_id			
2	countryrecipnt			

Include unspecified fields, ordered by name □

Step 10 - Create a Table Output Linked to a MySQL Database: Table is ready for loading into a database. This is done by using the 'Table Output' designer which enables us to link the table to a database while also automatically creating an SQL script that creates the table and also loads the data onto the table. (see appendix 3)

Step 11 - Create a CSV Table: Create a CSV table which will be used for data quality review and checks. (see appendix 3)

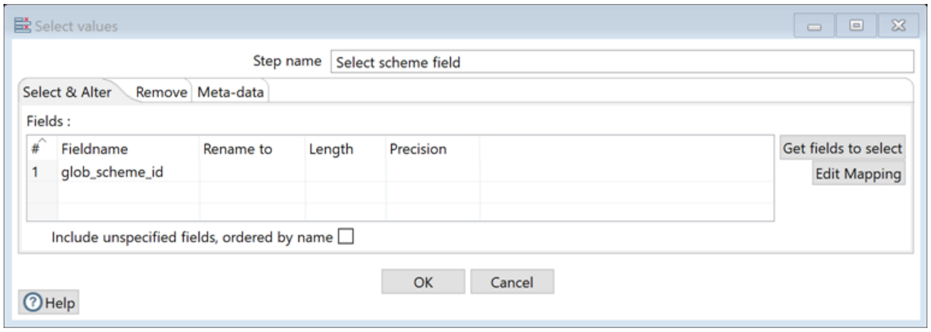
Step by Step Overview of Transformation 1 (TR1)



2.2 TRANSFORMATION 2 (TR2): CREATE THE SCHEME DIMENSION TABLE D_SCHEME

Step 1 - Input Payment Table: The payment table is loaded into PDI by using the ‘input CSV file’ connector as the data provided from the source was in this format. Once loaded, run this step to ensure data is added into PDI ready for transformation. (see appendix 2)

Step 2 - Select Fields needed in the Scheme Table: Select values needed for this table.



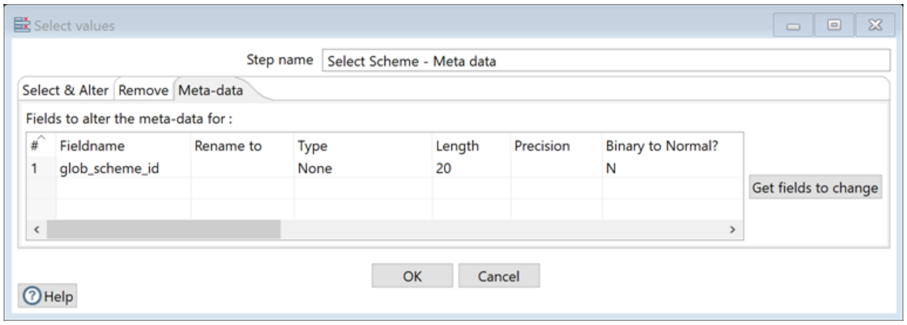
Step name: Select scheme field

Fields :

#	Fieldname	Rename to	Length	Precision
1	glob_scheme_id			

Include unspecified fields, ordered by name □

Step 3 - Select Metadata for all Columns: Review the metadata information that was loaded together with the table and make adjustments where needed.

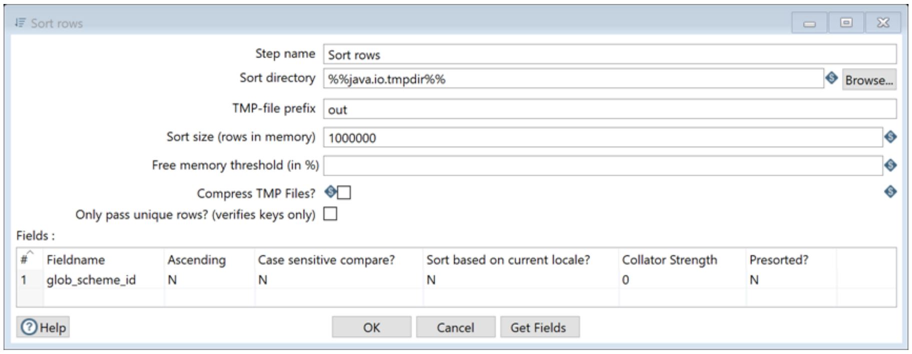


Step name: Select Scheme - Meta data

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?
1	glob_scheme_id		None	20		N

Step 4 - Sort the table based on the Glob_Scheme_ID: Sort the table so we can extract only unique values in the next transformation step.

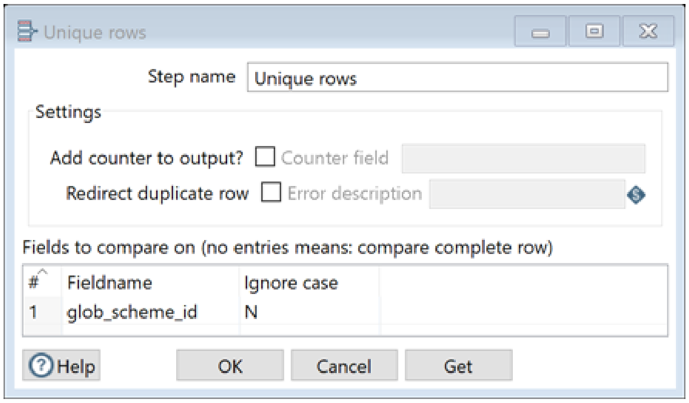


Step name: Sort rows
Sort directory: %%java.io.tmpdir%%
TMP-file prefix: out
Sort size (rows in memory): 1000000
Free memory threshold (in %):
Compress TMP Files? □
Only pass unique rows? (verifies keys only) □

Fields :

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

Step 5 - Extract only Unique Glob_Scheme_ID Values: Extract only unique values in order to deduplicate the table ensuring the the scheme table has unique ID's.


Unique rows

Step name: Unique rows

Settings

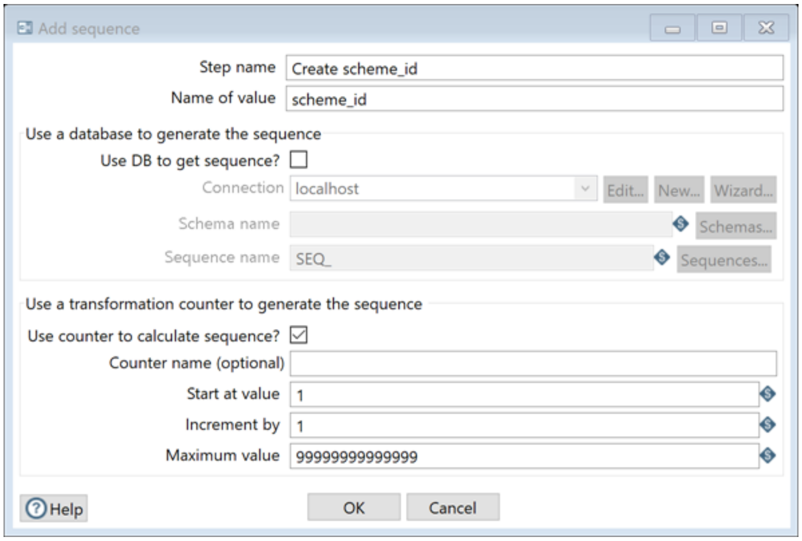
Add counter to output? □ Counter field

Redirect duplicate row □ Error description

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

#	Fieldname	Ignore case
1	glob_scheme_id	N

Step 6 - Create the Scheme_ID: Create a new unique_id column which will be used as the primary key of the table.


Add sequence

Step name: Create scheme_id

Name of value: scheme_id

Use a database to generate the sequence

Use DB to get sequence? □

Connection: localhost

Schema name:

Sequence name: SEQ_

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☑

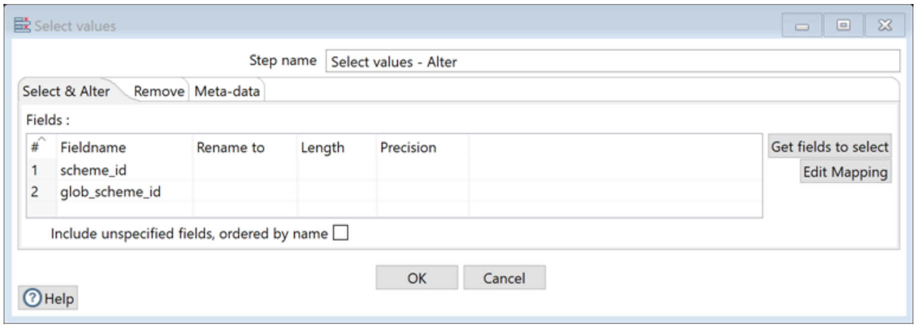
Counter name (optional):

Start at value: 1

Increment by: 1

Maximum value: 999999999999999

Step 7 - Add the Scheme_ID to the main Table: The created scheme_id is added to the table with the glob_scheme_id in order to create the final d_scheme table.


Select values

Step name: Select values - Alter

Select & Alter | Remove | Meta-data

Fields :

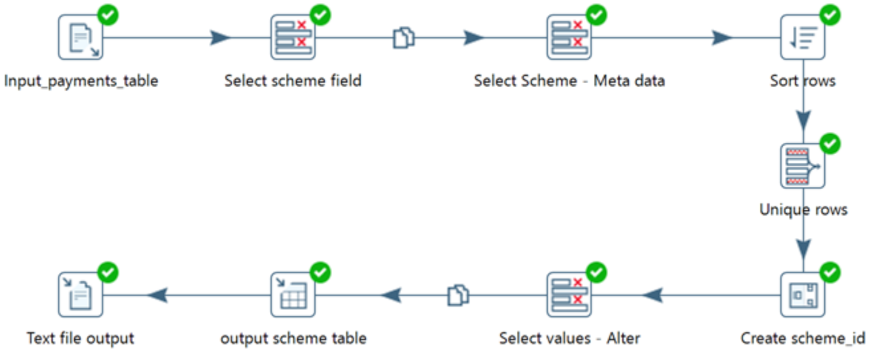
#	Fieldname	Rename to	Length	Precision
1	scheme_id			
2	glob_scheme_id			

Include unspecified fields, ordered by name □

Step 8 - Create a Table Output Linked to a MySQL Database: The table is ready for loading into a database. This is done by using the 'Table Output' designer which enables us to link the table to a database while also automatically creating an SQL script that creates the table and also loads the data onto the table. (see appendix 4)

Step 9 - Create a CSV Table: Create a CSV table which will be used for data quality review and checks. (see appendix 4)

Step by Step Overview of Transformation 2 (TR2)



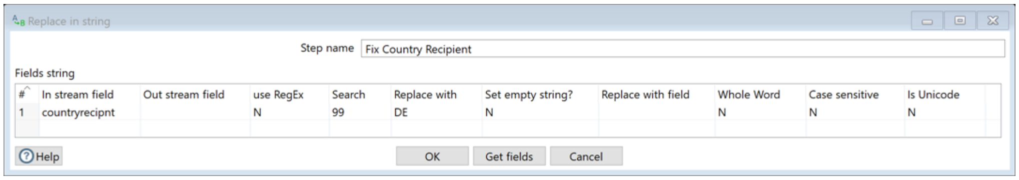
2.3 TRANSFORMATION 3 (TR3): CREATING THE RECIPIENT DIMENSION TABLE D_RECIPIENT

Step 1 - Input recipient Table: The recipient table is loaded into PDI by using the 'input CSV file' connector as the data provided from the source was in this format. (see appendix 1)

Step 2 - Select Columns for the Recipient Dimension Table: The 'Select Values' transformation phase was used to eliminate the columns that will not be needed in the recipient dimension table. (see appendix 5)

Step 3 - Select Metadata for all Columns: The metadata information that was loaded together with the table was reviewed and adjusted where needed. (see appendix 6)

Step 4 - Fix Country Recipient: During this step, as part of ensuring data quality, the "99" values in the countryrecipnt column are replaced by "DE".

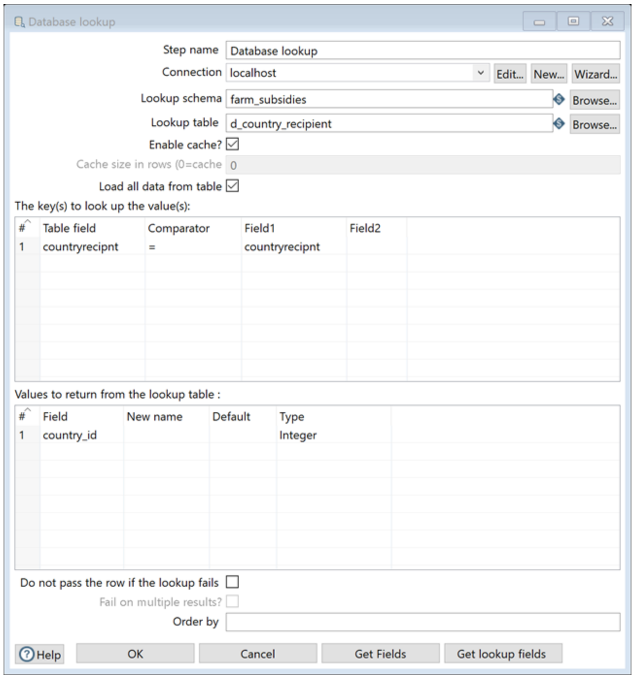


Step name: Fix Country Recipient

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	countryrecipnt		N	99	DE	N		N	N	N

Step 5 - Lookup the countryrecipnt: Look up the countryrecipnt into our table as the country_id so it can be later used as the foreign key linked to the scheme table.



Step name: Database lookup
Connection: localhost
Lookup schema: farm_subsidies
Lookup table: d_country_recipient
Enable cache? ☑
Cache size in rows (0=cache: 0
Load all data from table ☑

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

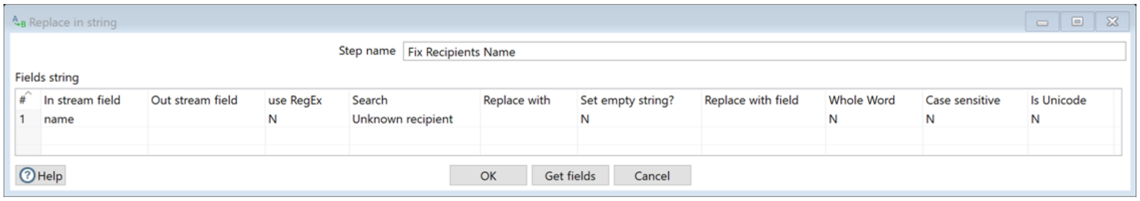
#	Table field	Comparator	Field1	Field2
1	countryrecipnt	=	countryrecipnt	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	country_id			Integer

Do not pass the row if the lookup fails □
Fail on multiple results? □
Order by:

Step 6 - Fix Recipients Name: As part of the data quality check, there are values in the name column which are classified as "Unknown recipient", these values are replaced with blanks.



Replace in string

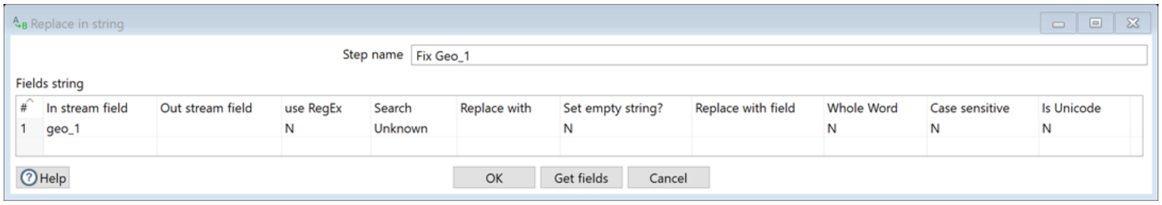
Step name: Fix Recipients Name

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	name		N	Unknown recipient		N		N	N	N

Step 7 - Fix Town: There are 5 values in the town column which include the zip code, these values should be removed and only the town should remain. (see appendix 7)

Step 8 - Fix Geo_1: The “Unknown” values in the Geo_1 column are removed.



Replace in string

Step name: Fix Geo_1

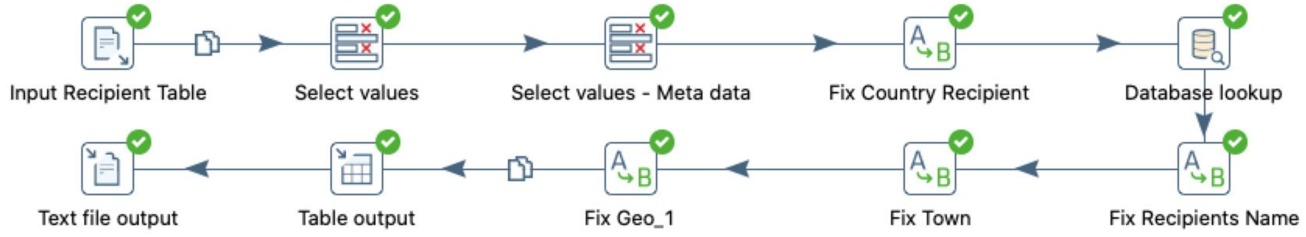
Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	geo_1		N	Unknown		N		N	N	N

Step 9 - Create a Table Output Linked to a MySQL Database: The table is ready for loading into a database. This is done by using the ‘Table Output’ designer which enables us to link the table to a database while also automatically creating an SQL script that creates the table and also loads the data onto the table. (see appendix 8)

Step 10 - Create a CSV Table: Create a CSV table which will be used for data quality review and checks. (see appendix 8)

Step by Step Overview of Transformation 3 (TR3)

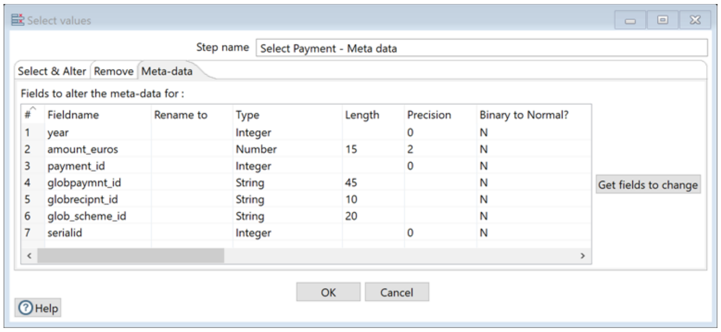


2.4 TRANSFORMATION 4 (TR4): CREATING THE PAYMENTS FACT TABLE F_PAYMENTS

Step 1 - Input Payment Table: The payment table is loaded into PDI by using the ‘input CSV file’ connector as the data provided from the source was in this format. (see appendix 2)

Step 2 - Select Columns for the Final Fact Table: Use the ‘Select Values’ transformation phase to eliminate the columns that will not be needed in the final fact table. (see appendix 9)

Step 3 - Select Metadata for all Columns: Review the metadata information that was loaded together with the table and make adjustments where needed.

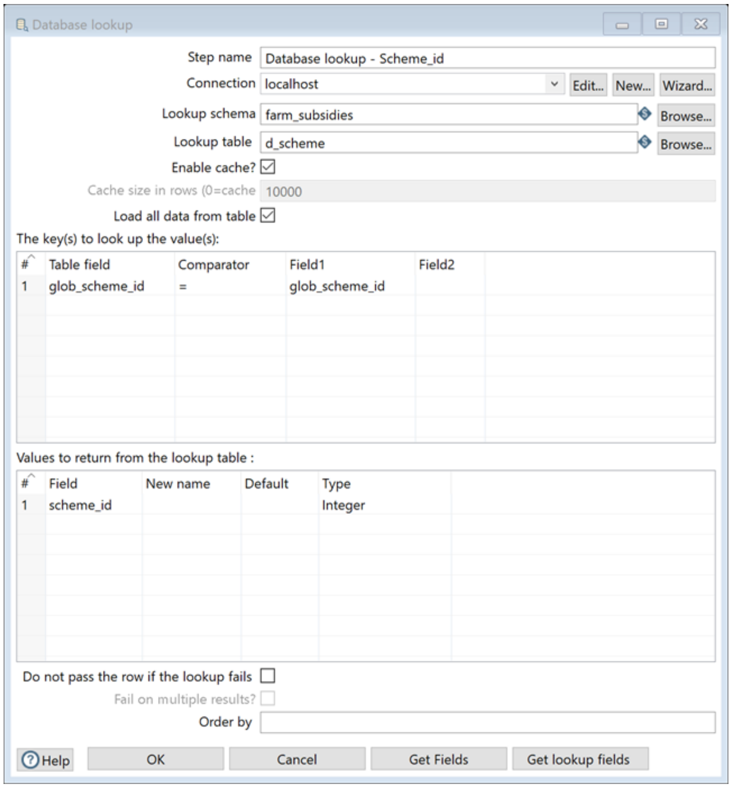


Step name: Select Payment - Meta data

Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?
1	year		Integer		0	N
2	amount_euros		Number	15	2	N
3	payment_id		Integer		0	N
4	globpaymnt_id		String	45		N
5	globrecipnt_id		String	10		N
6	glob_scheme_id		String	20		N
7	serialid		Integer		0	N

Step 4 - Lookup the Scheme_ID: Look up the glob_scheme_id into our table as the scheme_id so it can be later used as the foreign key linked to the scheme table.



Step name: Database lookup - Scheme_id

Connection: localhost

Lookup schema: farm_subsidies

Lookup table: d_scheme

Enable cache? ☑

Cache size in rows (0=cache: 10000

Load all data from table ☑

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

#	Table field	Comparator	Field1	Field2
1	glob_scheme_id	=	glob_scheme_id	

Values to return from the lookup table :

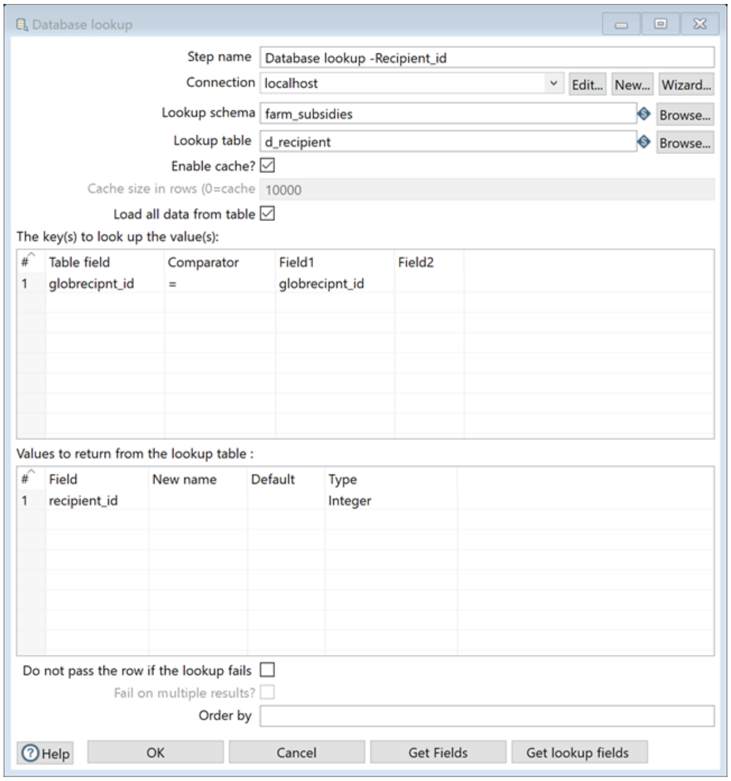
#	Field	New name	Default	Type
1	scheme_id			Integer

Do not pass the row if the lookup fails ☐

Fail on multiple results? ☐

Order by

Step 5 - Lookup the Recipient_ID: Look up the globrecipnt_id into our table as the recipient_id so it can be later used as the foreign key linked to the recipient table.



Step name: Database lookup -Recipient_id

Connection: localhost

Lookup schema: farm_subsidies

Lookup table: d_recipient

Enable cache? ☑

Cache size in rows (0=cache: 10000

Load all data from table ☑

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

#	Table field	Comparator	Field1	Field2
1	globrecipnt_id	=	globrecipnt_id	

Values to return from the lookup table :

#	Field	New name	Default	Type
1	recipient_id			Integer

Do not pass the row if the lookup fails ☐

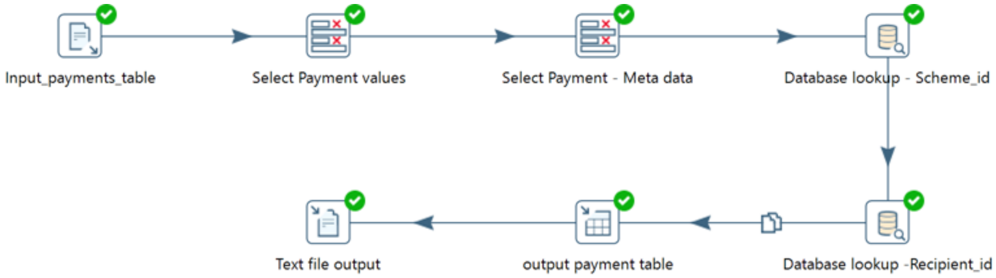
Fail on multiple results? ☐

Order by

Step 6 - Create a Table Output Linked to a MySQL Database: The table is ready for loading into a database. This is done by using the 'Table Output' designer which enables us to link the table to a database while also automatically creating an SQL script that creates the table and also loads the data onto the table. (see appendix 10)

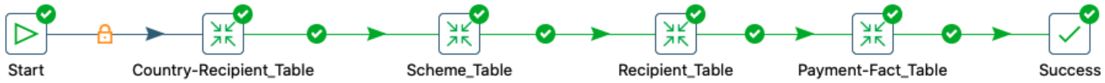
Step 7 - Create a CSV Table: Create a CSV table which will be used for data quality review and checks. (see appendix 10)

Step by Step Overview of Transformation 4 (TR4)


Input_payments_table
Select Payment values
Select Payment - Meta data
Database lookup - Scheme_id
Database lookup -Recipient_id
output payment table
Text file output

2.5 FINAL ETL JOB: COMBINING ALL TRANSFORMATIONS

The final process was the creation of an ETL job which combines all transformations into a single process which was used to then load all the data into the database.

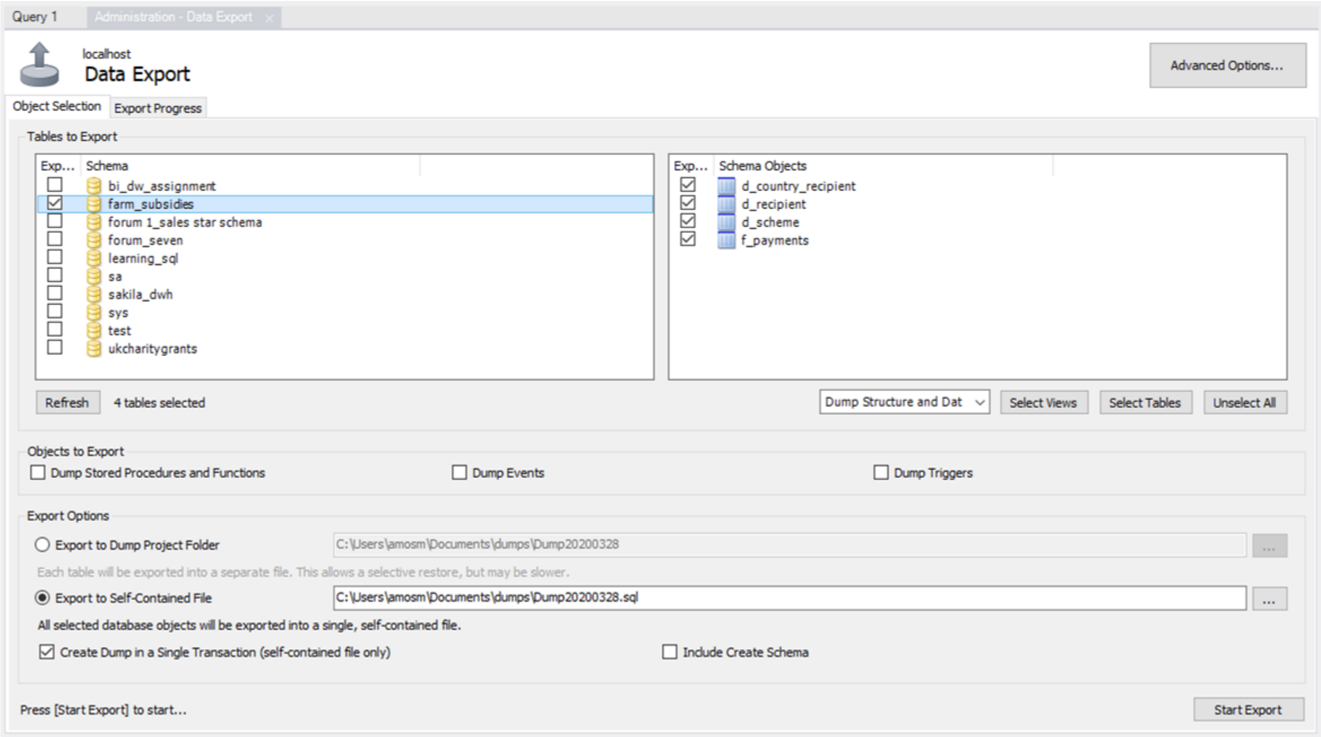


3.0 DATABASE BACKUP

A backup file is useful in order to share the database with the schema and the data with related team members, whereby the ELT process does not need to be executed on each machine. This process is faster.

The backup file is generated from the created farm_subsidies database

Under the administration section → select Data Export → select Farmsubsidies database (make sure to select: Dump Structure and Data, as well as Export to Self-Contained File- meaning only one sql script is created and rename the "dump" file to be "**german_farms**")→ click Export → "Export Completed"



Tables to Export

Exp...	Schema
□	bi_dw_assignment
☑	farm_subsidies
□	forum 1_sales star schema
□	forum_seven
□	learning_sql
□	sa
□	sakila_dwh
□	sys
□	test
□	ukcharitygrants

Exp...	Schema Objects
☑	d_country_recipient
☑	d_recipient
☑	d_scheme
☑	f_payments

4 tables selected

Dump Structure and Dat

Objects to Export

□ Dump Stored Procedures and Functions □ Dump Events □ Dump Triggers

Export Options

○ Export to Dump Project Folder C:\Users\amosm\Documents\dumps\Dump20200328

Each table will be exported into a separate file. This allows a selective restore, but may be slower.

● Export to Self-Contained File C:\Users\amosm\Documents\dumps\Dump20200328.sql

All selected database objects will be exported into a single, self-contained file.

☑ Create Dump in a Single Transaction (self-contained file only) □ Include Create Schema

Press [Start Export] to start...

To load a backfile: MySQL Workbench: Through my administration → under Management select Data Import/Restore → select Import from Self-Contained File → select the path which includes the backup file→ select Dump Structure and Data → click Start Import → "Import Completed"

APPENDIX

Appendix 1: Recipient Table (Source CSV File)

name	address1	address2	zipcode	town	countryrecipnt	countrypymnt	recipient_id	recipient_idx	globrecipnt_id	globrecipnt_idx	geo_1	geo_2	geo_3	geo_4	geo1_natlang	geo2_natlang	geo3_natlang	geo4_natlang	latitude	longitude	serialid
Centralmarkt Roisdorf/Straelen GmbH			DE-53332		DE	DE	1	1	DE1	DE1	Nordrhein-Westfalen	K√∂ln	Rhein-Sieg-Kreis						50.76376	6.9847	1
Campina GmbH & Co. KG			DE-50739		DE	DE	2	2	DE2	DE2	Nordrhein-Westfalen	K√∂ln	K√∂ln						50.97356	6.94629	2
NBV / UGA GmbH			DE-47638		DE	DE	3	3	DE3	DE3	Nordrhein-Westfalen	D√ºsseldorf	Kleve						51.44521	6.26844	3
Landgard Obst & Gem√ºse GmbH & Co. K			DE-53332		DE	DE	4	301244	DE4	DE1	Nordrhein-Westfalen	K√∂ln	Rhein-Sieg-Kreis						50.76376	6.9847	4
UGA Niederrhein GmbH			DE-47638		DE	DE	5	3	DE5	DE3	Nordrhein-Westfalen	D√ºsseldorf	Kleve						51.44521	6.26844	5

Appendix 2: Payments Table (Source CSV File)

year	countrypayment	amount_euros	amt_nat_currcy	payment_id	globpaymnt_id	globrecipnt_id	globrecipnt_idx	glob_scheme_id	serialid
2008	DE	3262		912573	DE912573	DE233142	DE233142	DE3	1
2008	DE	570		912574	DE912574	DE233145	DE233145	DE3	2
2008	DE	570		912575	DE912575	DE233146	DE233146	DE3	3
2008	DE	575.77		912576	DE912576	DE233147	DE233147	DE3	4
2008	DE	437.67		912577	DE912577	DE233149	DE233149	DE3	5

Appendix 3: d_country_recipient Table

country_id	Countryrecipnt
1	UK
2	NL
3	LU
4	IE
5	HU
6	GB
7	FR
8	ES
9	DK
10	DE
11	CZ
12	CH
13	BE
14	AT

Appendix 4: d_scheme Table

scheme_id	glob_scheme_id
1	DE9
2	DE8
3	DE7
4	DE6
5	DE5
6	DE4
7	DE3
8	DE2
9	DE16
10	DE15
11	DE14
12	DE13
13	DE12
14	DE11
15	DE10
16	DE1

Appendix 5: Selected Columns for d_recipient Table

Left Column	Type	Format	Length
recipient_id	Integer	#	15
globrecipnt_id	String		6
name	String		37
address1	String		25
town	String		45
geo_1	String		19
geo_2	String		12
geo_3	String		20
latitude	Number	#.#	8
longitude	Number	#.#	8
zipcode	String		8
countryrecipnt	String		2
serialid	Integer	#	15

Appendix 6: Reviewed Metadata for d_Recipient Table

Column	Rename to	Type	Length	Precision	Binary to Normal?
name		String	150		N
address1		String	45		N
zipcode		String	10		N
town		String	45		N
recipient_id		Integer	0	0	N
globrecipnt_id		String	10		N
geo_1		String	45		N
geo_2		String	45		N
geo_3		String	45		N
latitude		Number	11	5	N
longitude		Number	10	5	N
serialid		Integer		0	N
countryrecipnt		String	2		N

Appendix 7: Clean Town Column in Recipient Table

In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?
town		N	4389PC		N
town		N	8911AL		N
town		N	57537		N
town		N	9035EK		N
town		N	6523LD		N

Appendix 8: d_recipient Table

recipient_id	globrecipnt_id	name	address1	town	geo_1	geo_2	geo_3	latitude	longitude	zipcode	country recipnt	serialid	country
1	DE1	Centralmarkt Roisdorf/ Straelen GmbH	Raiffeisenstr. 10		Nordrhein-Westfalen	K√∂ln	Rhein-Sieg-Kreis	50.8	7	DE-53332	DE	1	10
2	DE2	Campina GmbH & Co. KG	Geldernstra√üe 35		Nordrhein-Westfalen	K√∂ln	K√∂ln	51	6.9	DE-50739	DE	2	10
3	DE3	NBV / UGA GmbH	Hans-Tenhaeff-Str. 44		Nordrhein-Westfalen	D√ºsseldorf	Kleve	51.4	6.3	DE-47638	DE	3	10
4	DE4	Landgard Obst & Gem√ºse GmbH & Co. K	Raiffeisenstr. 10		Nordrhein-Westfalen	K√∂ln	Rhein-Sieg-Kreis	50.8	7	DE-53332	DE	4	10
5	DE5	UGA Niederrhein GmbH			Nordrhein-Westfalen	D√ºsseldorf	Kleve	51.4	6.3	DE-47638	DE	5	10

Appendix 9: Columns for the f_payments Table

Fields	Type	Format	Length
amount_euros	Number	#.#	7
amt_nat_currcy	Boolean		
glob_scheme_id	String		20
globpaymnt_id	String		8
globrecipnt_id	String		8
payment_id	Integer	#	15
serialid	Integer	#	15
year	Integer	#	15

Appendix 10: f_payments Table

year	amount_euros	amt_nat_currcy	payment_id	globpaymnt_id	globrecipnt_id	glob_scheme_id	serialid	scheme_id	recipient_id
2008	3262		912573	DE912573	DE233142	DE3	1	7	233142
2008	570		912574	DE912574	DE233145	DE3	2	7	233145
2008	750		912575	DE912575	DE233146	DE3	3	7	233146
2008	575.8		912576	DE912576	DE233147	DE3	4	7	233147
2008	437.7		912577	DE912577	DE233149	DE3	5	7	233149