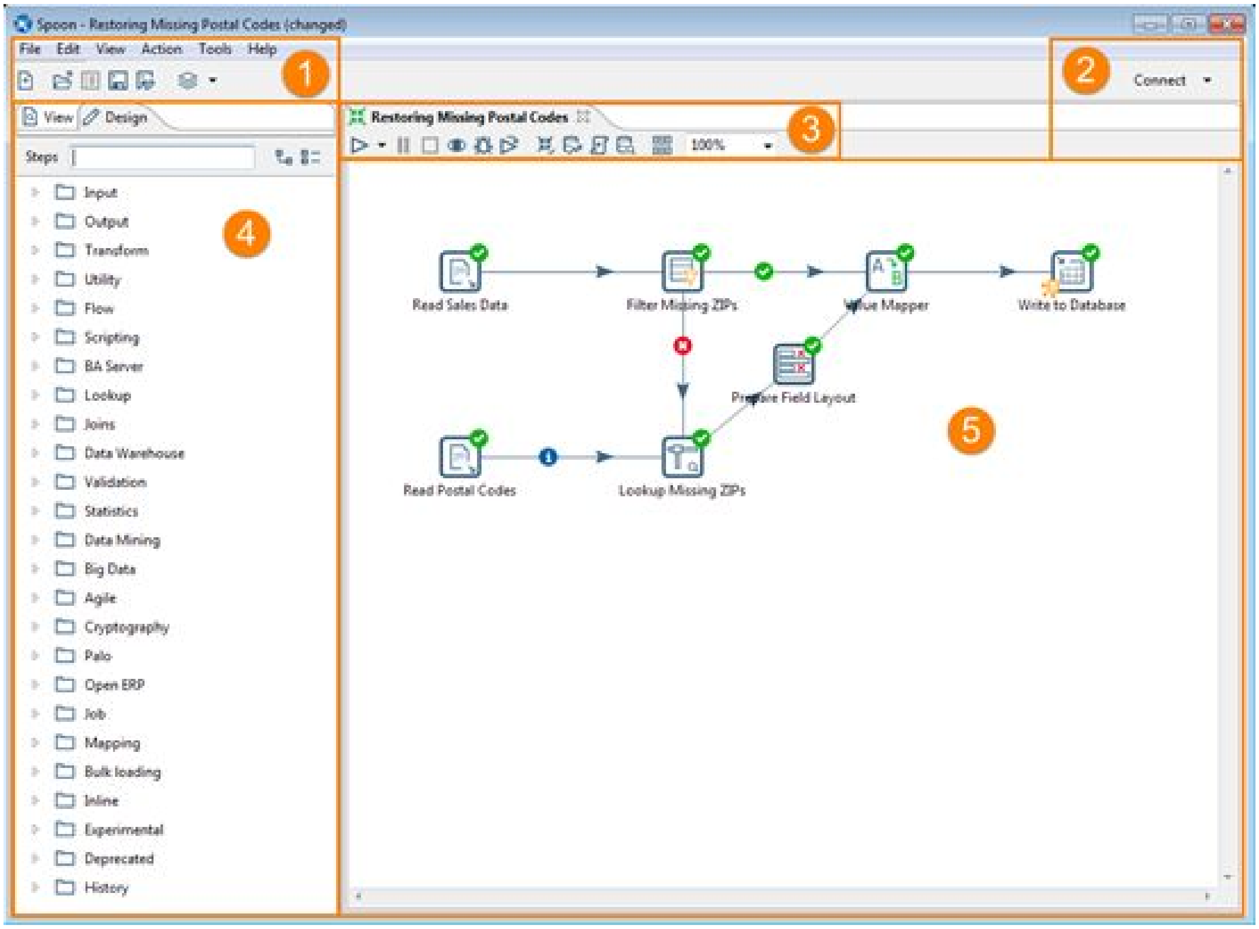
Introduction to Spoon

Spoon is the UI of the Pentaho ETL Tool or PDI. When you first open and run it, the Default "Perspective" is "Data integration". This is the perspective we will be working with.

The Data Integration perspective allows you to create transformations, jobs, and inspect your data allowing for iterative updates as you work.   Let us get ourselves familiar with Spoon.



1. Toolbar
2. Connect menu
3. Sub-toolbar
4. Design and View Tabs (Or sometimes called "Left pane")
5. Canvas

Component	Name	Description
①	Toolbar	Single-click access to common actions such as create a new file, opening existing documents, save and save as.
②	Connect Menu	Create and connect to repositories for centrally storing your ETL jobs and transformations. We are not going to use this functionality.
③	Sub-toolbar	Provides buttons for quick access to common actions specific to the transformation or job such as Run, Preview, and Debug.
④	Design and View Tabs	The **Design tab** of the Explore pane provides an organized list of transformation steps or job entries used to build transformations and jobs. Transformations are created by simply dragging transformation steps from the Design tab onto the canvas and connecting them with hops to describe the flow of data. The **View tab** of the Explore pane shows information for each job or transformation. This includes information such as available database connections and which steps and hops are used. In the image, the Design tab is selected.
⑤	Canvas	Main design area for building transformations and jobs describing the ETL activities you want to perform.

Icon	Description
🗋	Create a new job or transformation
📂	Open transformation/job from file if you are not connected to a repository or from the repository if you are connected to one
☰	Explore the repository
💾	Save the transformation/job to a file or to the repository
💾	Save the transformation/job under a different name or file name (Save as)
≋ ▾	Switch between the different perspectives. ● Data Integration — Create ETL transformations and jobs ● Schedule — Manage scheduled ETL activities on the Pentaho Server
▷ ▾	Run transformation/job and set run options; runs the current transformation from XML file or repository
‖	Pause transformation
□	Stop transformation

👁	Preview transformation: runs the current transformation from memory. You can preview the rows that are produced by selected steps.
🐞	Run the transformation in debug mode; allows you to troubleshoot execution errors
▷	Replay the processing of a transformation
✓	Verify transformation
⛁	Run an impact analysis of a transformation on the database
▤	Generate the SQL that is needed to run the transformation.
🔍	Launch the database explorer allowing you to preview data, run SQL queries etc. on the database.
☰	Show execution results pane
🔒	Lock transformation

Exercise 1 - Copy Text file to Table

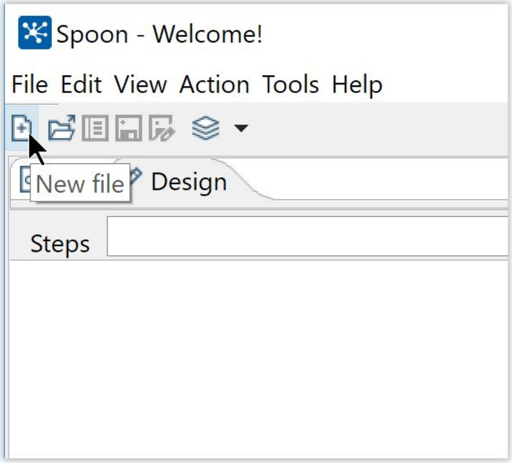
In this exercise, the objective is to familiarise you with basic operations of the Spoon tool in PDI. You should complete this exercise before attending the lecture.

Prerequisite:

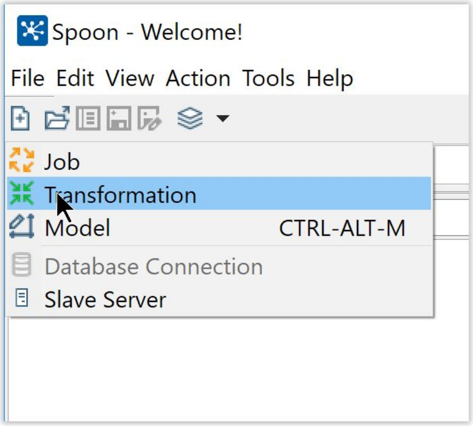
1. Start up your Postgres server from Bitnami Mapp/Wapp stack.
2. Note down the ipaddress, port number, user and password for connecting to Postgres server. Make sure you have a database named **postgres** in the server already. If not, create one.

STEPS

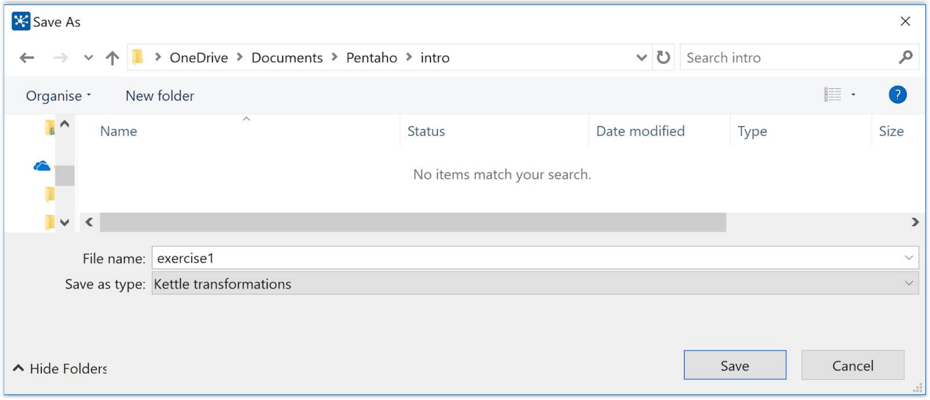
1. Click on New file icon



2. Select Transformation from drop down list



3. Save transformation file with name “exercise1”

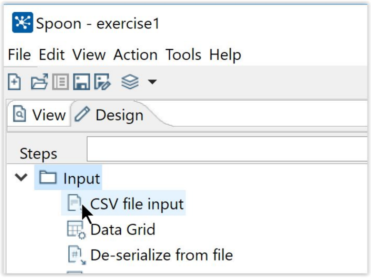


File name: exercise1

Save as type: Kettle transformations

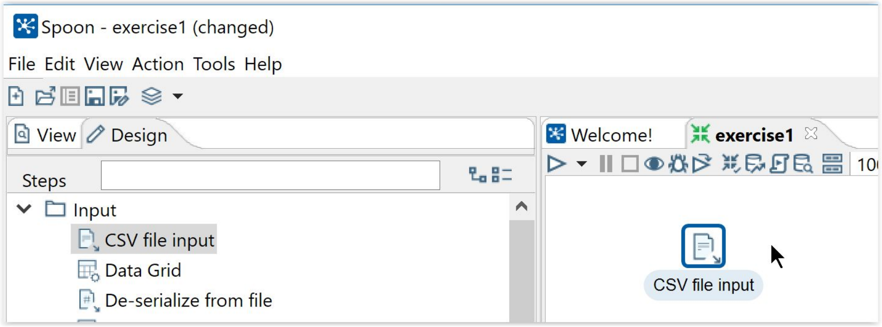
Transformation steps are saved with “.ktr”: file extension.

4. Drag “CSV file input” from left pane to the canvas



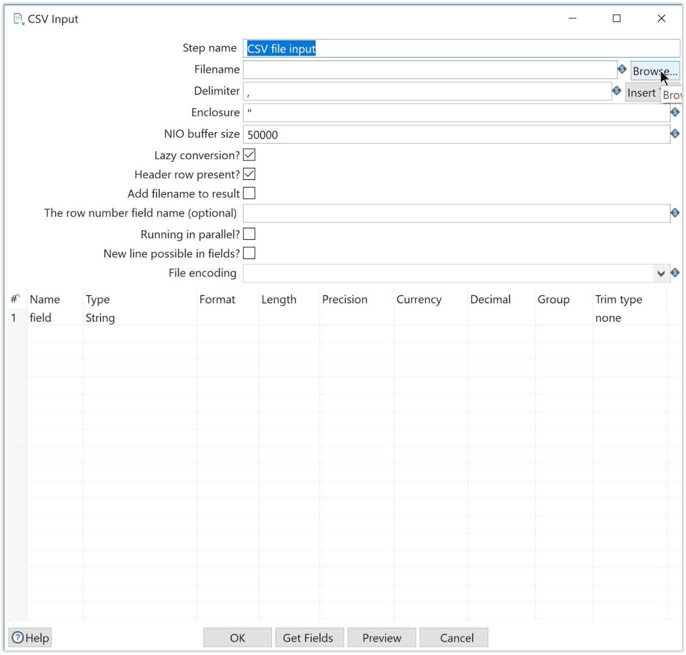
At the left pane, click on “> Input” to reveal drop down of steps for input. Click on “CSV file input” icon and drag.

You may also use the search box and enter “CSV” and the filtered list will show



Drag the icon to the canvas and release

5. Edit the “CSV file input” step by double clicking on it



Step name: CSV file input

Filename:

Delimiter: ,

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☑

Header row present? ☑

Add filename to result □

The row number field name (optional):

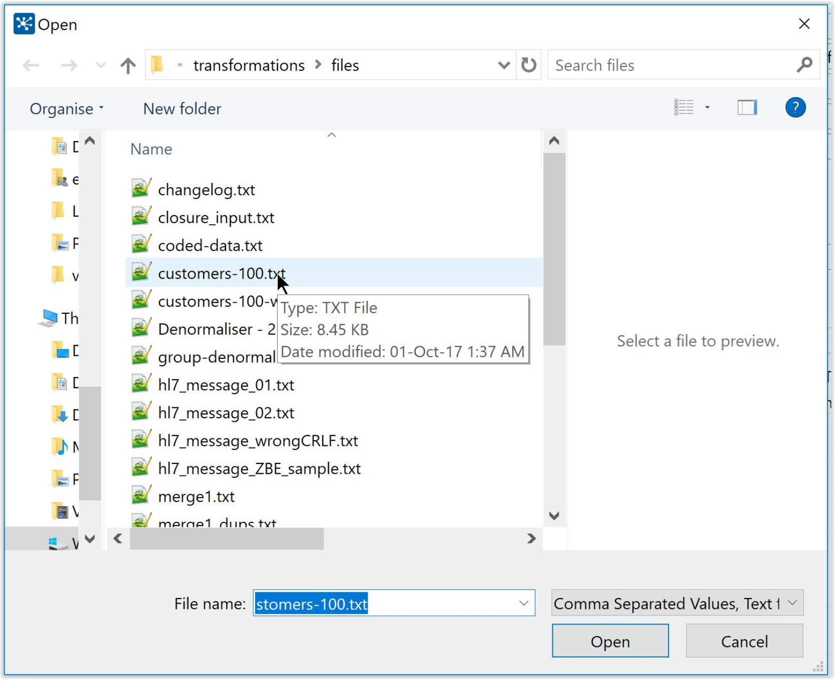
Running in parallel? □

New line possible in fields? □

File encoding:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	field	String							none

Click on browse and navigate to folder data-integration/samples/transformation/files



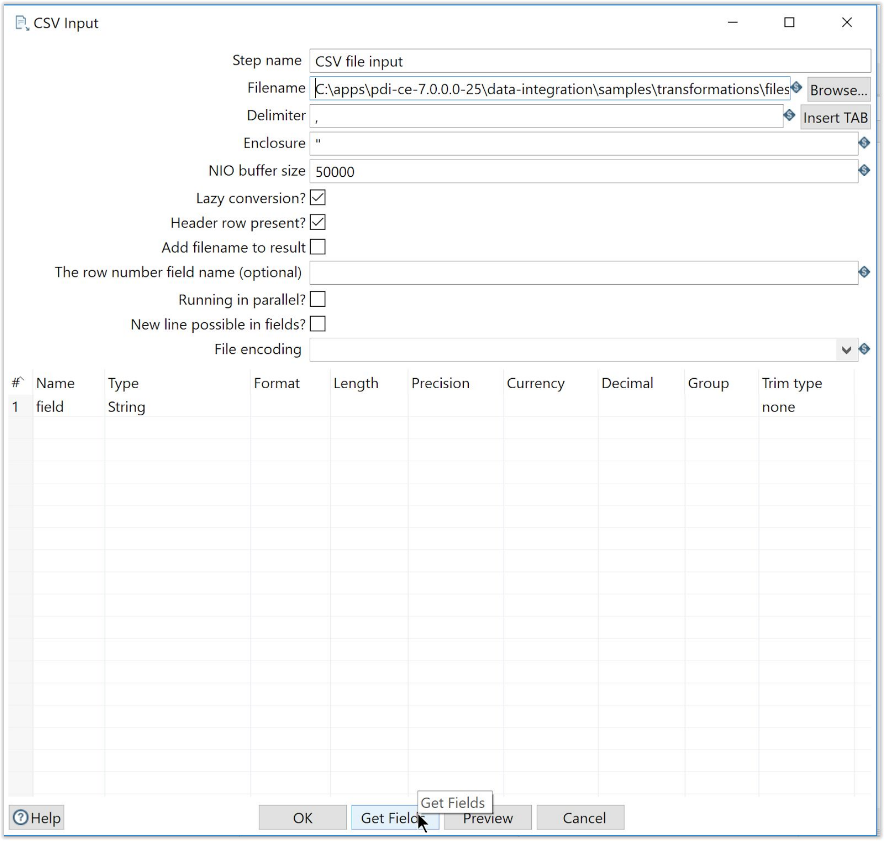
changelog.txt
closure_input.txt
coded-data.txt
customers-100.txt
customers-100-w
Denormaliser - 2
group-denormal
hl7_message_01.txt
hl7_message_02.txt
hl7_message_wrongCRLF.txt
hl7_message_ZBE_sample.txt
merge1.txt
merge1_dups.txt

Type: TXT File
Size: 8.45 KB
Date modified: 01-Oct-17 1:37 AM

File name: stomers-100.txt

Comma Separated Values, Text f

Choose customers-100.txt and click on “Open”



CSV Input

Step name: CSV file input

Filename: C:\apps\pdi-ce-7.0.0.0-25\data-integration\samples\transformations\files

Delimiter: ,

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☑

Header row present? ☑

Add filename to result □

The row number field name (optional):

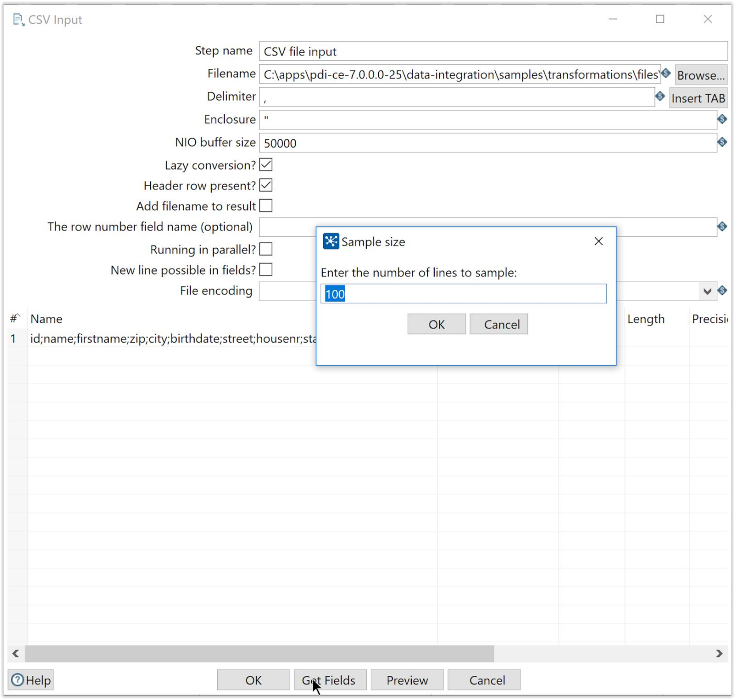
Running in parallel? □

New line possible in fields? □

File encoding:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	field	String							none

Click on Get fields. You will get a new dialog box



CSV Input

Step name: CSV file input

Filename: C:\apps\pdi-ce-7.0.0.0-25\data-integration\samples\transformations\files

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:

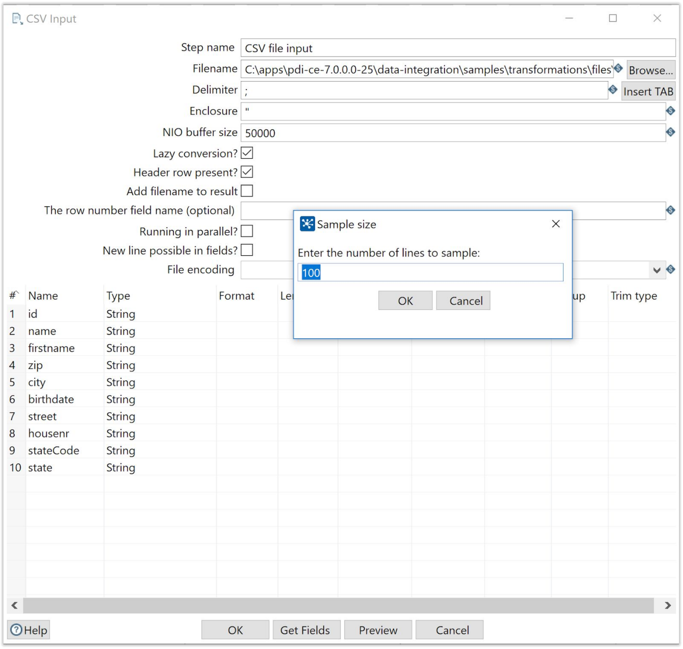
Sample size

Enter the number of lines to sample: 100

#	Name	Length	Precisi
1	id;name;firstname;zip;city;birthdate;street;housenr;sta		

Notice the field names are not separated into multiple lines due to the wrong delimiter “,” vs “;”

Click on “Cancel” on dialog “Sample size”



CSV Input

Step name: CSV file input

Filename: C:\apps\pdi-ce-7.0.0.0-25\data-integration\samples\transformations\files

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

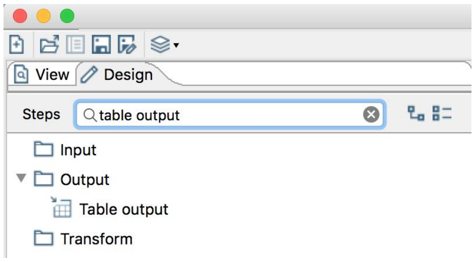
#	Name	Type	Format	Le	up	Trim type
1	id	String				
2	name	String				
3	firstname	String				
4	zip	String				
5	city	String				
6	birthdate	String				
7	street	String				
8	housenr	String				
9	stateCode	String				
10	state	String				

Sample size

Enter the number of lines to sample: 100

Go ahead and change the delimiter to “;” and click on Get fields again.
The fields are now correctly detected and data type is set to String. The dialog box “Sample size” is asking if you would like to set the data type automatically by sampling the first 100 rows. Click on “Cancel” for now as you want it to remain as STRING.

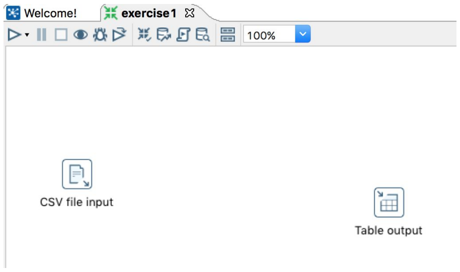
6. Drag and drop the Table output step
   a.



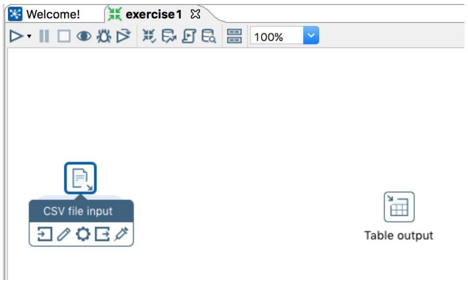
Steps: table output

- Input
- Output
  - Table output
- Transform

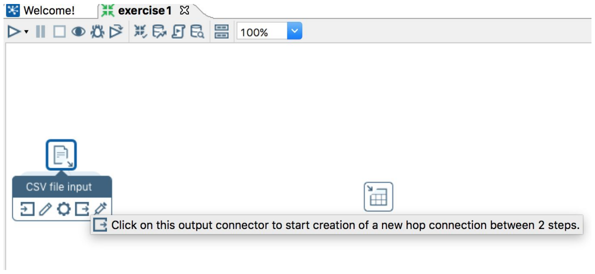
   b.



7. Connect the CSV file input to Table output

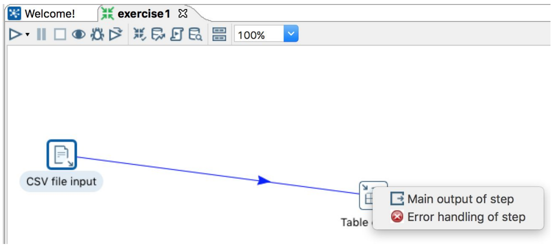


Click on the first step

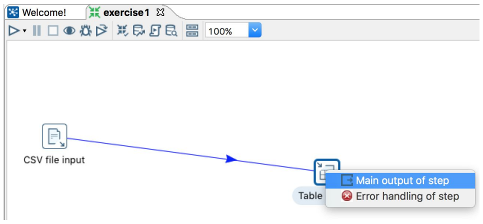


Click on this output connector to start creation of a new hop connection between 2 steps.

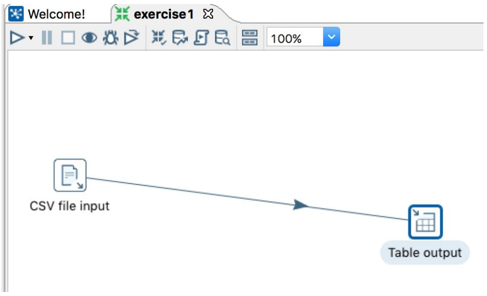
Click and drag the output connector icon



Drag the arrow point over the second step



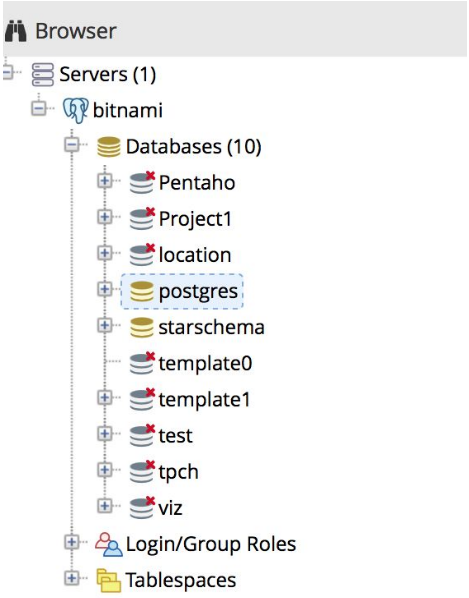
Release the mouse button and a tooltip will appear.
Click on “Main output of step”



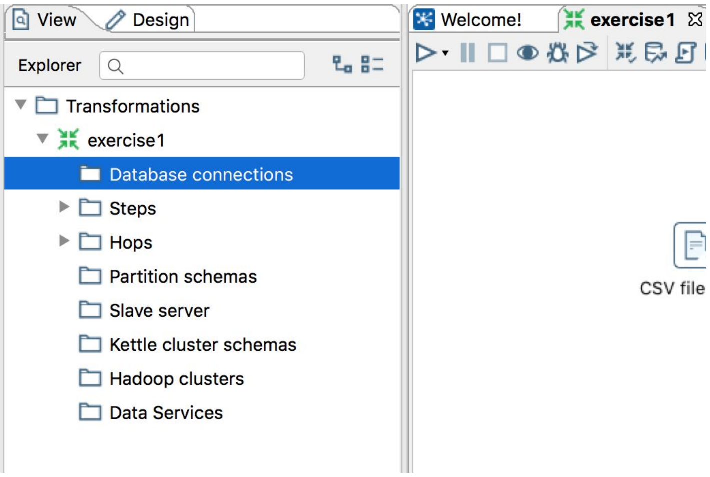
NOTE:
Other ways to connect the 2 steps using the mouse with clickable scroll button:
Go to https://help.pentaho.com/Documentation/6.0/0L0/0Y0/030/010
Paragraph: More About Hops

1. Use <SHIFT + left-click> on first step and drag to second step
2. left-click on the source step, hold down the middle mouse button, and drag the hop to the second step
3. Select both steps, then right-click and choose New Hop
4. Use <CTRL + left-click> to select two steps then right-click on the step and choose New Hop

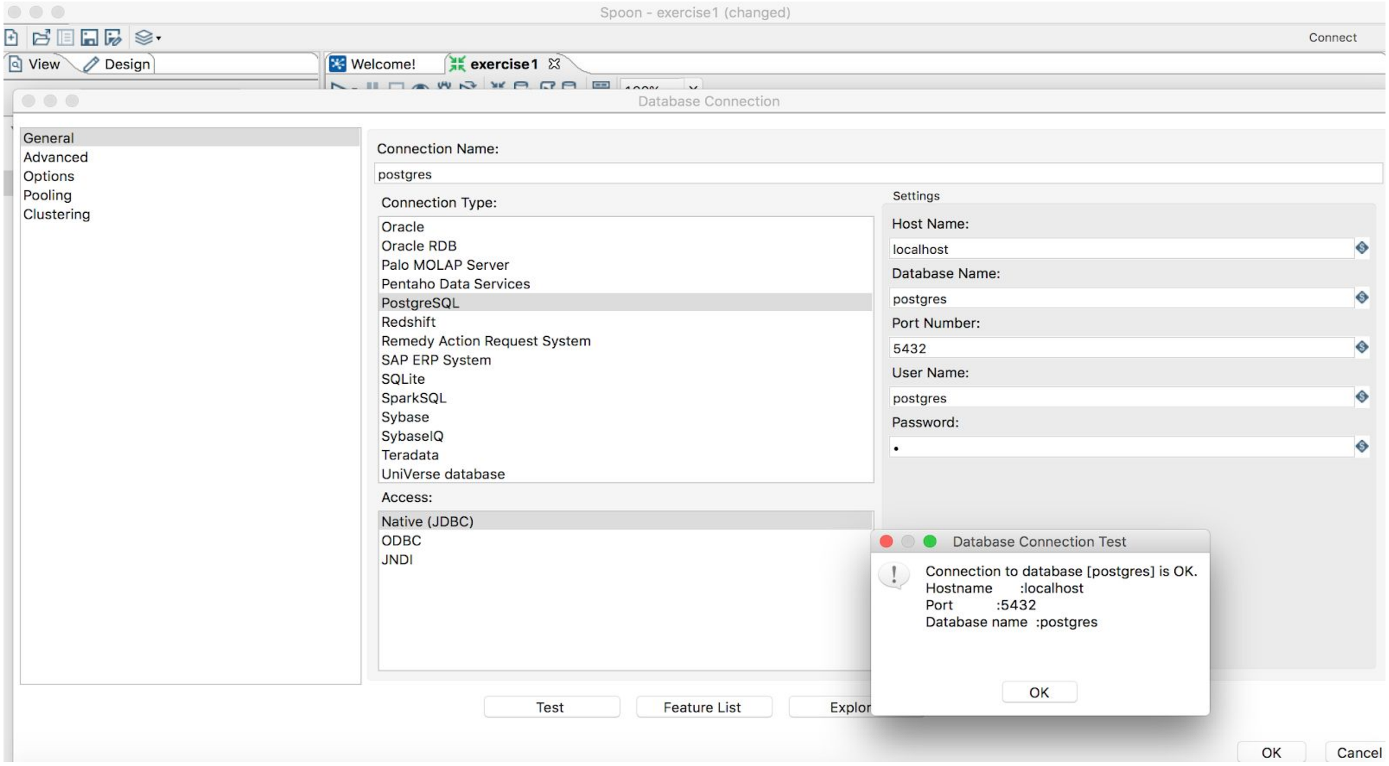
8. **Create a database and connect to it**
Open pgadmin and create a database named **postgres** , if it doesn't exist already.



Go back to Spoon.



Click on the View tab and expand “Database connections”. It should be empty, since you have not associated any database to Spoon yet.



Connection Name: postgres

Host Name: localhost

Database Name: postgres

Port Number: 5432

User Name: postgres

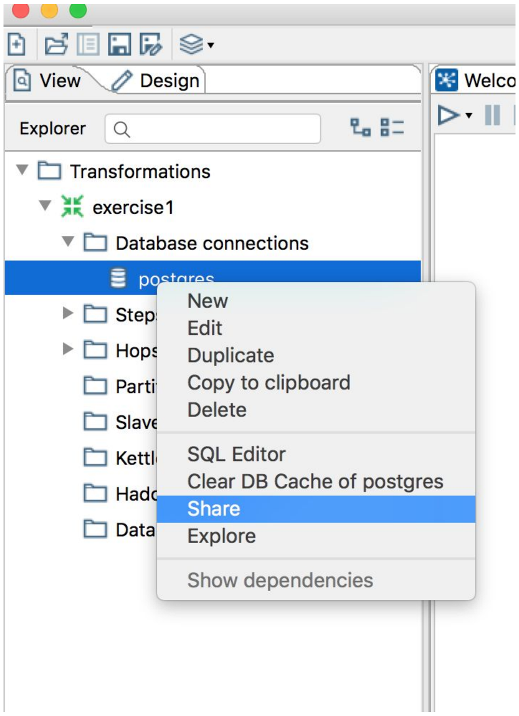
Password: •

Connection to database [postgres] is OK.
Hostname :localhost
Port :5432
Database name :postgres

Set the name of the connection

Fill in the configuration for the postgres database and click on “Test” and OK.

*****Common Errors:** You must test the connection before you use it. Since, we often put wrong password, username, wrong port etc.



Right Click on the postgres connection, and click Share. It is a good practice to Share the connection. As we create new transformations, the shared connection will automatically be taken as default connection.

9. Edit the Table output step

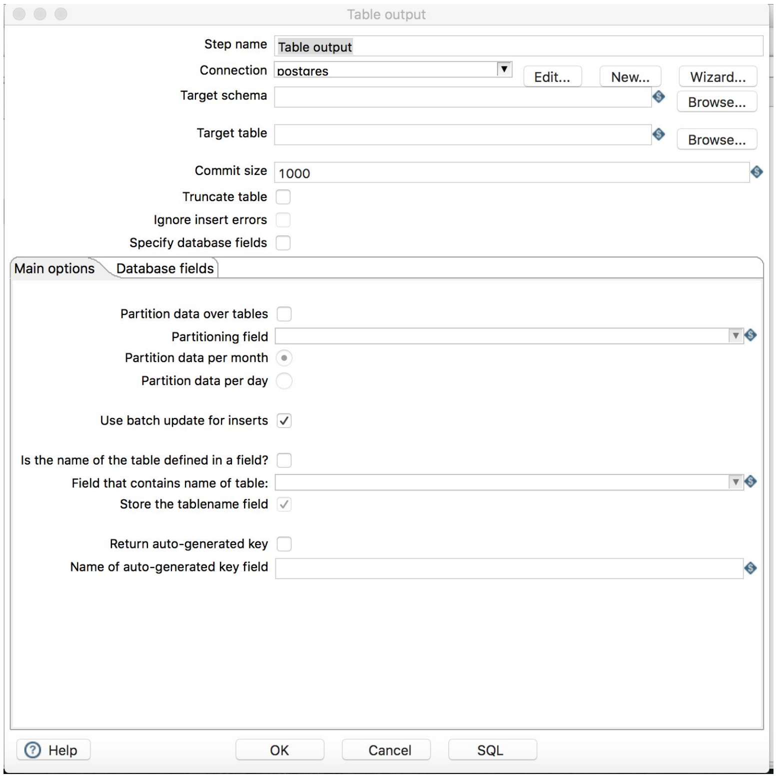


Table output

Step name	Table output
Connection	postgres
Target schema	
Target table	
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	

Change step name to “customers”

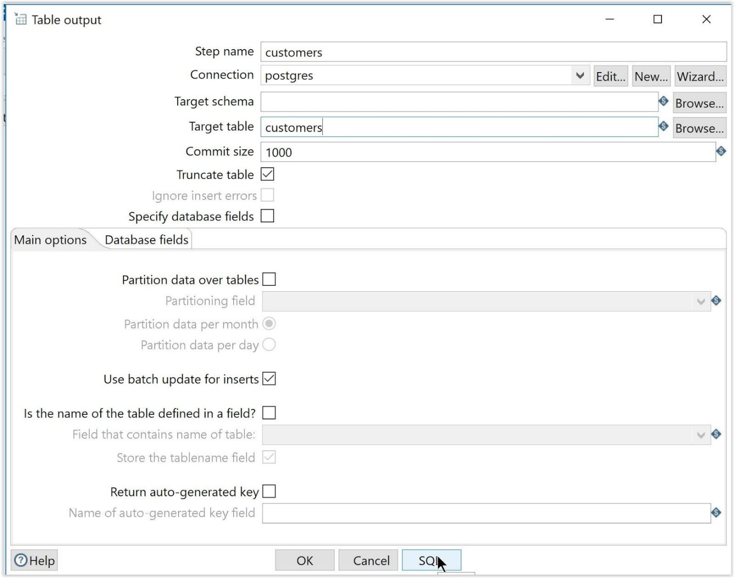


Table output

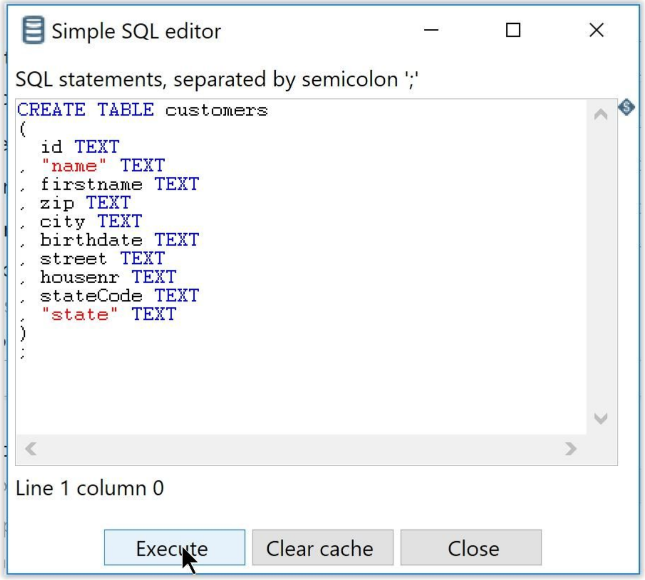
Step name: customers
Connection: postgres
Target schema:
Target table: customers
Commit size: 1000
Truncate table ☑
Ignore insert errors □
Specify database fields □

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:

Set the table name. Leave the other options as it is. We need to create a new table by executing a SQL Query.

Click on SQL button and a suggested SQL query dialog will appear. If the query dialog does not appear, then it might be that customers table already exists in the database. You need to delete it manually from pgadmin, if that is the case.



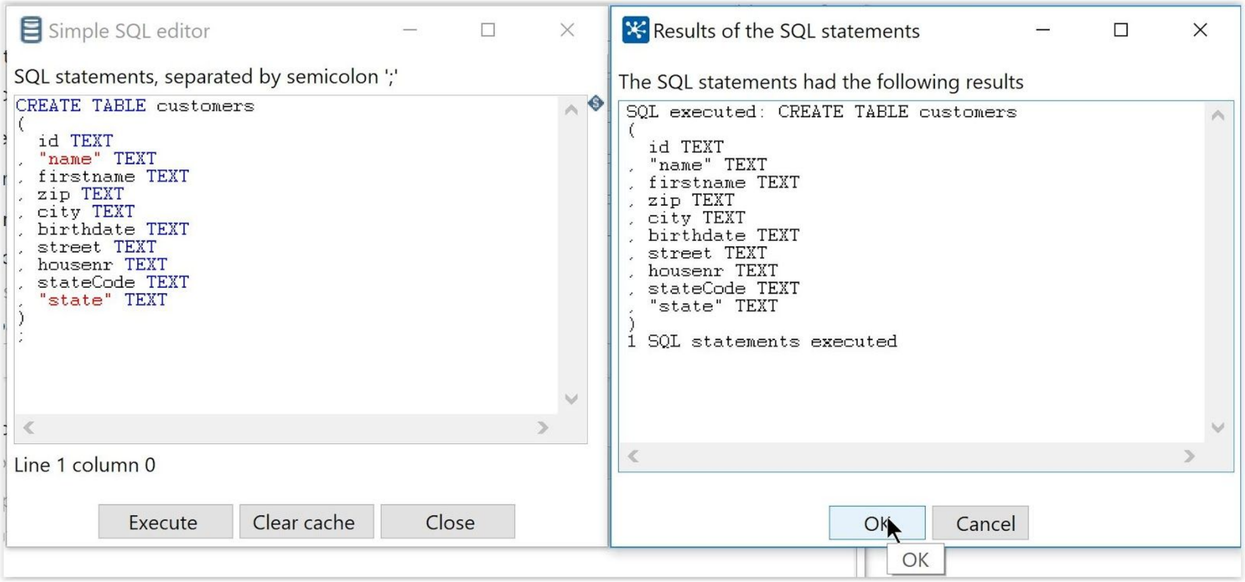
Simple SQL editor

SQL statements, separated by semicolon ';'

```
CREATE TABLE customers
(
  id TEXT
, "name" TEXT
, firstname TEXT
, zip TEXT
, city TEXT
, birthdate TEXT
, street TEXT
, housenr TEXT
, stateCode TEXT
, "state" TEXT
)
;
```

Line 1 column 0

Check the SQL query and edit it if necessary. Click on “Execute”



SQL statements, separated by semicolon ';'

```
CREATE TABLE customers
(
  id TEXT
, "name" TEXT
, firstname TEXT
, zip TEXT
, city TEXT
, birthdate TEXT
, street TEXT
, housenr TEXT
, stateCode TEXT
, "state" TEXT
)
;
```

Line 1 column 0

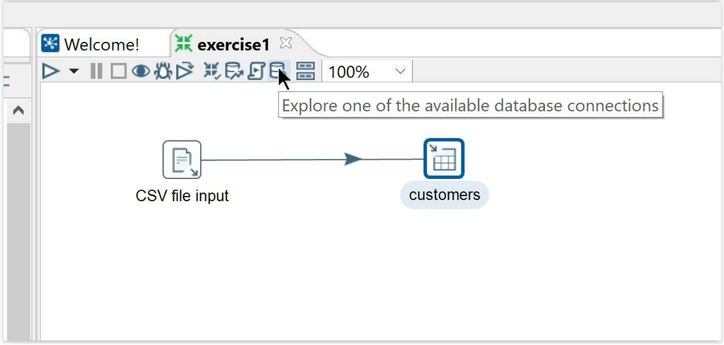
The SQL statements had the following results

```
SQL executed: CREATE TABLE customers
(
  id TEXT
, "name" TEXT
, firstname TEXT
, zip TEXT
, city TEXT
, birthdate TEXT
, street TEXT
, housenr TEXT
, stateCode TEXT
, "state" TEXT
)
1 SQL statements executed
```

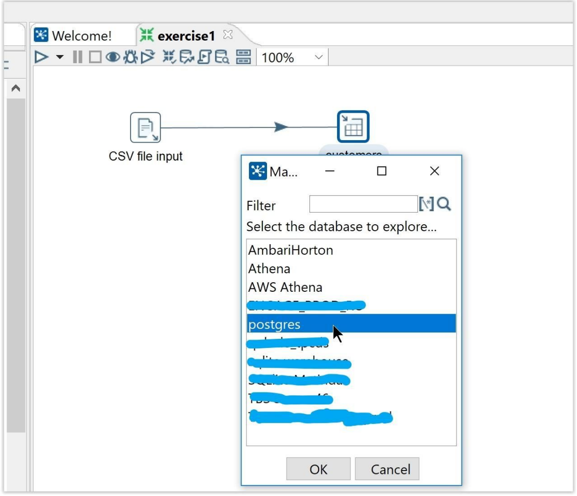
Check to see there is no error on execution. Click OK.

On the SQL dialog select all and copy the SQL Query and paste it in a notepad++ for your reference in future. This is good practice.

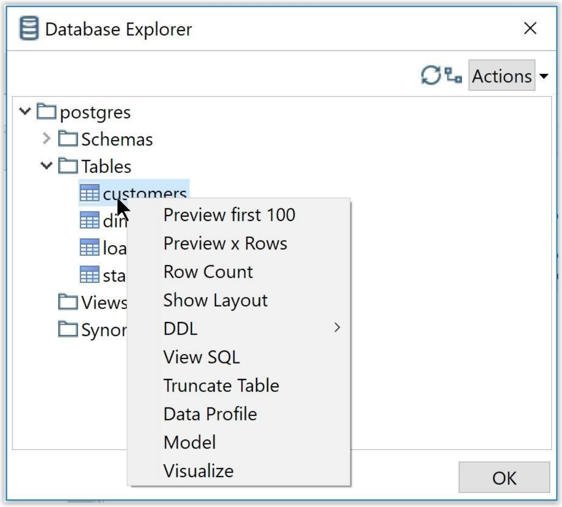
10. Check the database to see that your table is created.



Click on Database connection icon to browse the connections.



Choose the postgres connection that you have made before.



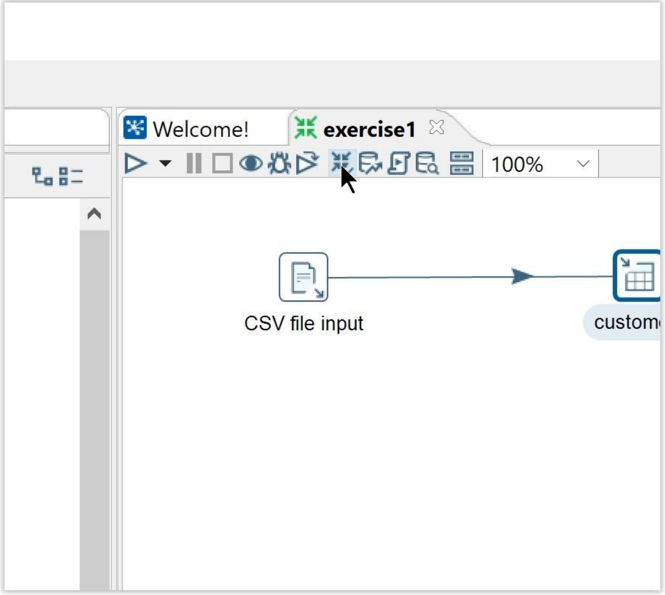
Navigate to the table and right-click to view the table.
There should be **no rows** in your table as you have just created the table.

10. Verify the transformation

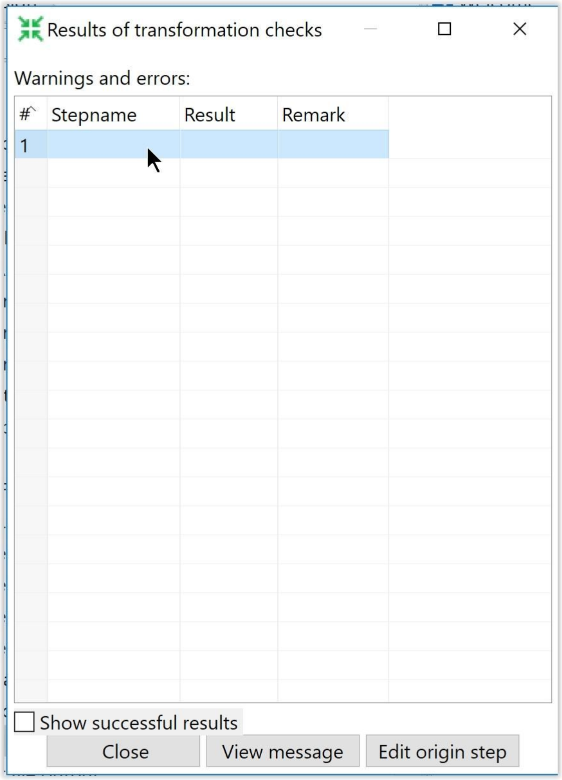
“Verify transformation” helps you to check that all steps are properly configured and all input fields and output fields connect from step to step.

*****Common Errors:** You may encounter:

1. **Errors:** Fields from the output of a step does not exist in the input of the next step. Output field name of a step does not match with the input field name of the next step.
2. **Warnings:** You may ignore them most of the time.



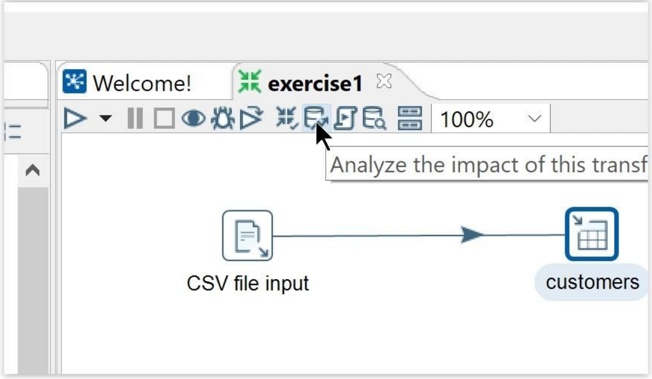
Click on the “Verify” icon



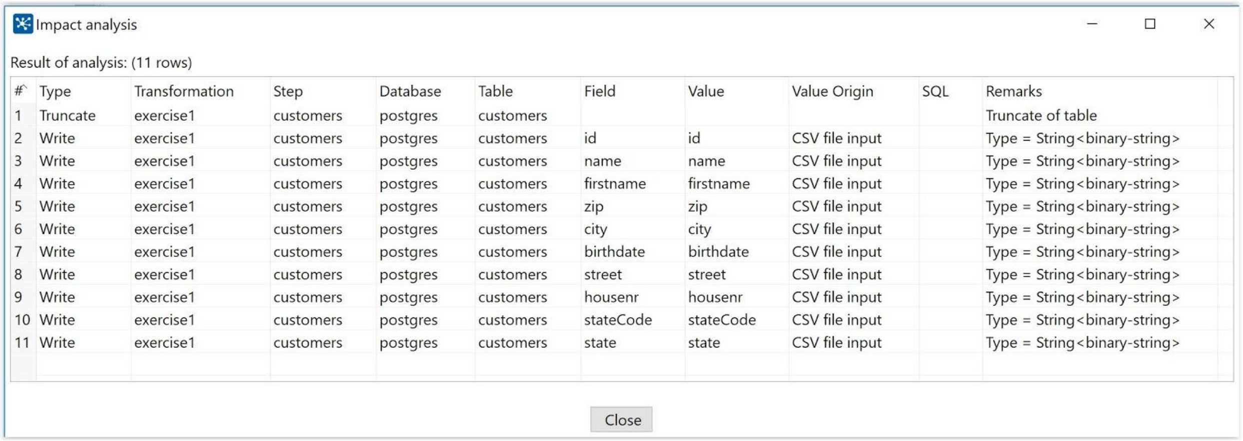
#	Stepname	Result	Remark
1			

You should get a blank table, showing that no errors were found.

11. Run an impact analysis of the transformation
You want to know before running the transform what impact it has on the resulting tables



Click on “Impact analysis” Icon



Result of analysis: (11 rows)

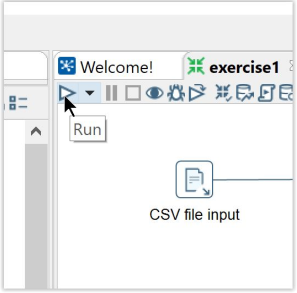
#	Type	Transformation	Step	Database	Table	Field	Value	Value Origin	SQL	Remarks
1	Truncate	exercise1	customers	postgres	customers					Truncate of table
2	Write	exercise1	customers	postgres	customers	id	id	CSV file input		Type = String<binary-string>
3	Write	exercise1	customers	postgres	customers	name	name	CSV file input		Type = String<binary-string>
4	Write	exercise1	customers	postgres	customers	firstname	firstname	CSV file input		Type = String<binary-string>
5	Write	exercise1	customers	postgres	customers	zip	zip	CSV file input		Type = String<binary-string>
6	Write	exercise1	customers	postgres	customers	city	city	CSV file input		Type = String<binary-string>
7	Write	exercise1	customers	postgres	customers	birthdate	birthdate	CSV file input		Type = String<binary-string>
8	Write	exercise1	customers	postgres	customers	street	street	CSV file input		Type = String<binary-string>
9	Write	exercise1	customers	postgres	customers	housenr	housenr	CSV file input		Type = String<binary-string>
10	Write	exercise1	customers	postgres	customers	stateCode	stateCode	CSV file input		Type = String<binary-string>
11	Write	exercise1	customers	postgres	customers	state	state	CSV file input		Type = String<binary-string>

*****Important/Common Errors**:

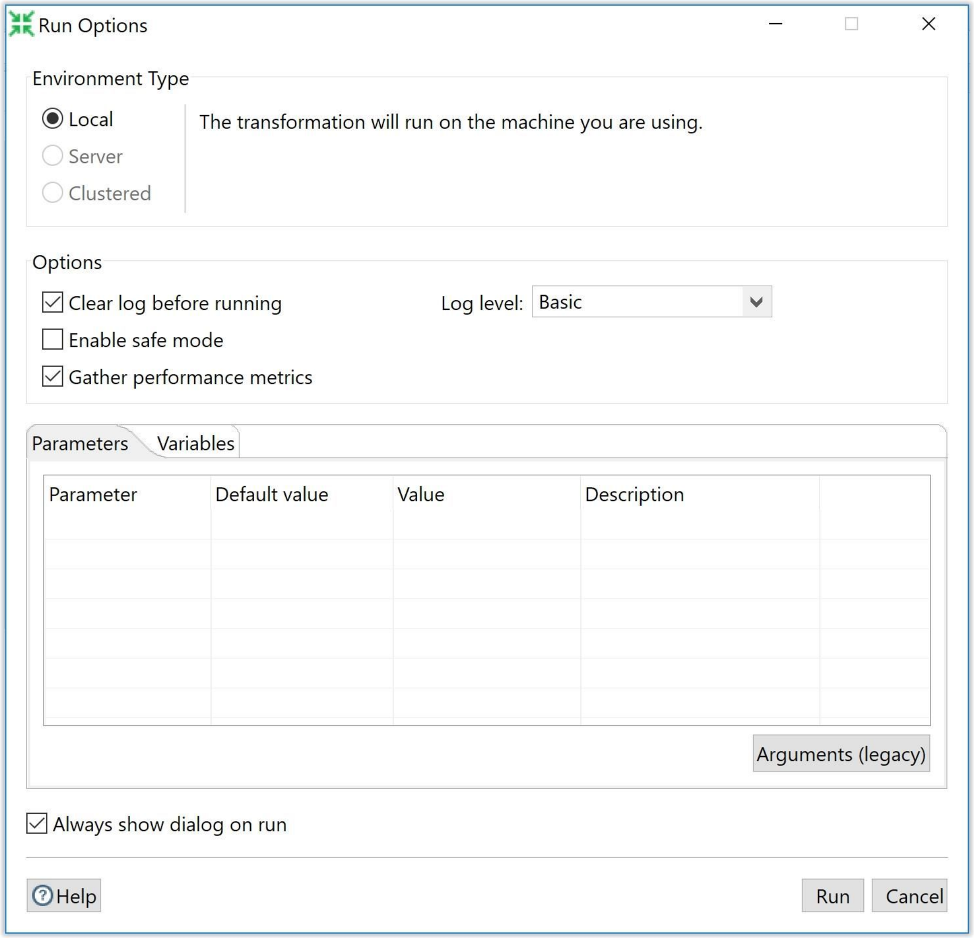
1. The last step “customers” which is an Table output step is set to TRUNCATE the table on every run. Make sure that this is the correct setting for the transformation that you want to do. This is okay if you are updating intermediate tables (if any) such as staging table (staging tables will be explained in the lecture), but **not okay** for Dimension tables and Fact table. Loading data into Dimension and fact table MUST NOT TRUNCATE. Since we are not creating dimension or fact table here, we are choosing truncate option.
2. Note the fields that will be written into the table and the source step of the value of each field. For this simple transform is only 2 steps. In a complex transform this will help you to debug.

12. Running the transformation

There are many options on running the transformation. Most important is that Pentaho will take the ktr file (“exercise1.ktr”) to run. If there are changes you have made you need to save it. The name of the transformation, “exercise1” on the tab will be bold if changes are not saved. In addition the run transformation step will ask you again to save the changes.



Click on “Run” Icon



Run Options

Environment Type

● Local

○ Server

○ Clustered

The transformation will run on the machine you are using.

Options

☑ Clear log before running

Log level: Basic

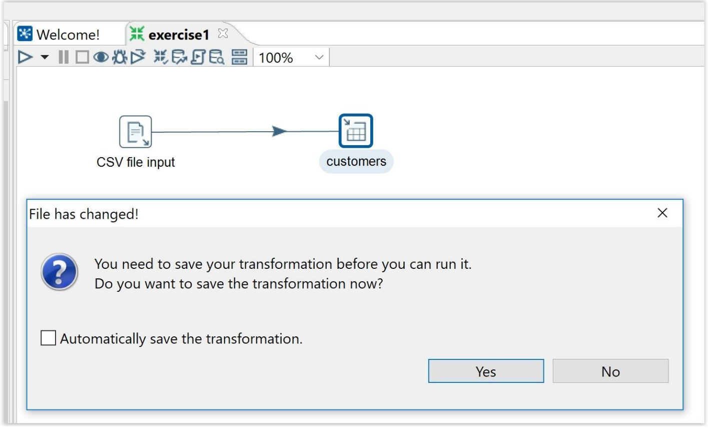
☐ Enable safe mode

☑ Gather performance metrics

Parameter	Default value	Value	Description

☑ Always show dialog on run

Click on “Run”

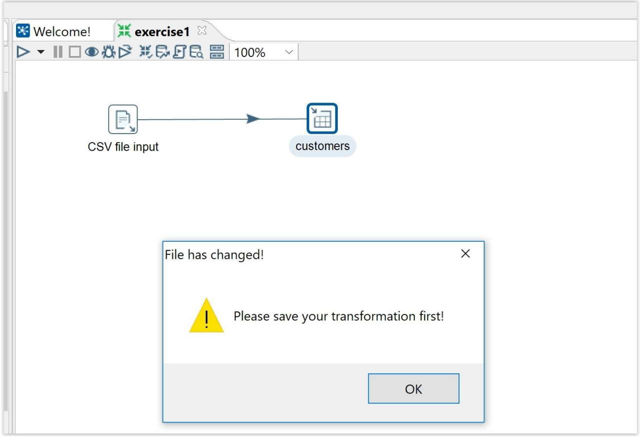


File has changed!

You need to save your transformation before you can run it.
Do you want to save the transformation now?

☐ Automatically save the transformation.

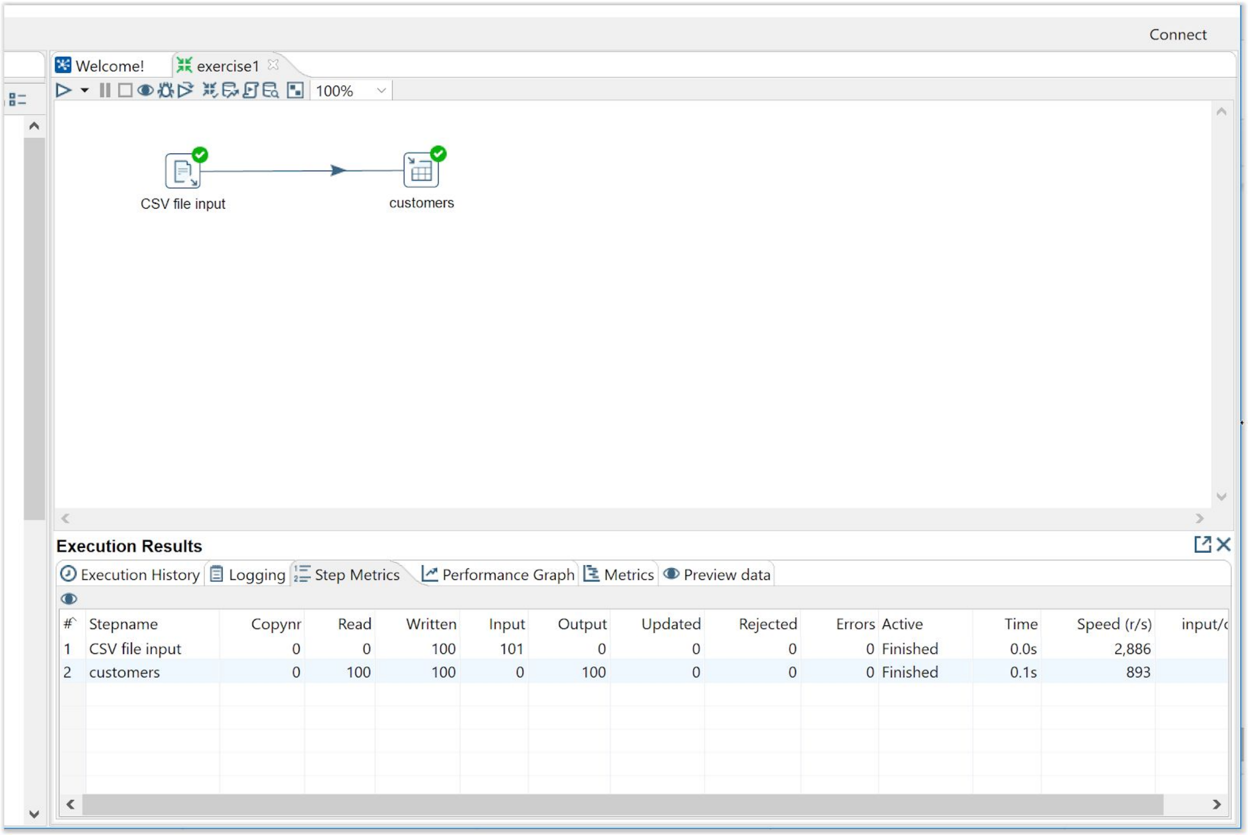
Note “exercise1” is BOLD , which means you have made changes but have not saved the file.
Click Yes



File has changed!

Please save your transformation first!

If you click on cancel, it will remind you to save the file first.



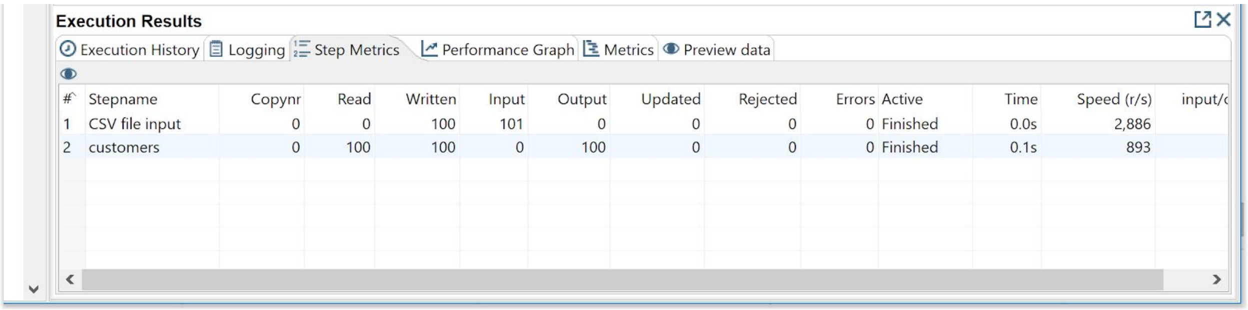
#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/c
1	CSV file input	0	0	100	101	0	0	0	0	Finished	0.0s	2,886	
2	customers	0	100	100	0	100	0	0	0	Finished	0.1s	893	

A new pane will appear at the bottom half of the canvas.

Most commonly used tabs are:

1. Step metrics
2. Logging
3. Metrics

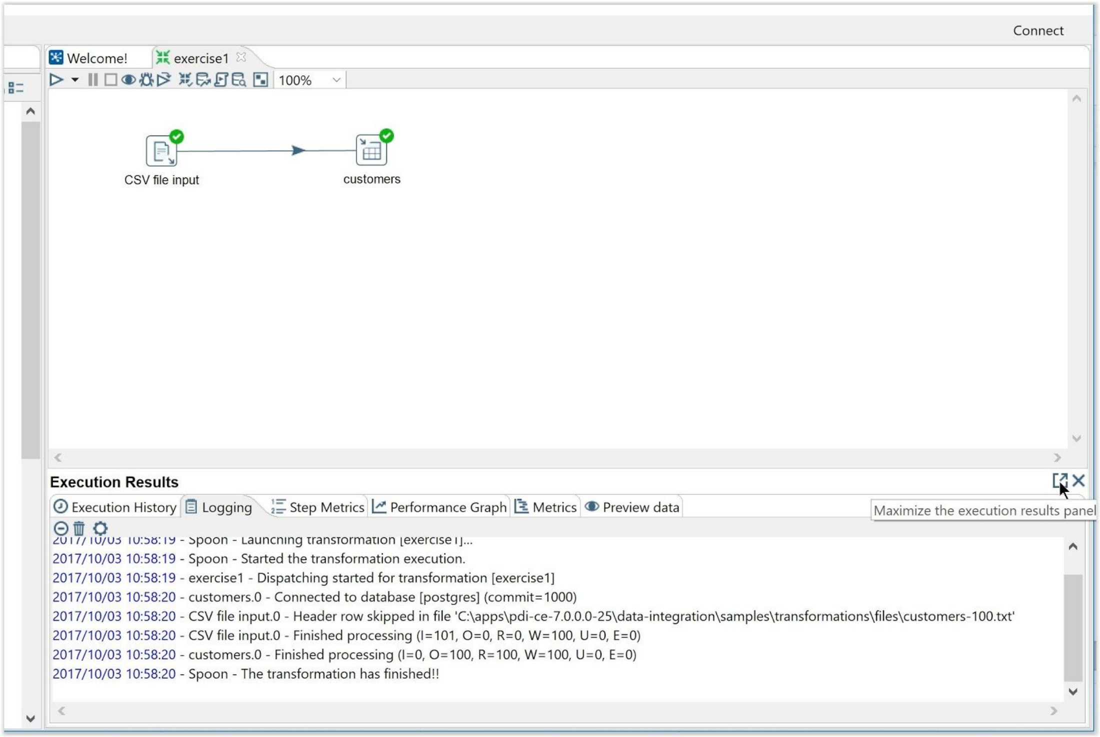
- Step metrics Tab



#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Time	Speed (r/s)	input/c
1	CSV file input	0	0	100	101	0	0	0	0	Finished	0.0s	2,886	
2	customers	0	100	100	0	100	0	0	0	Finished	0.1s	893	

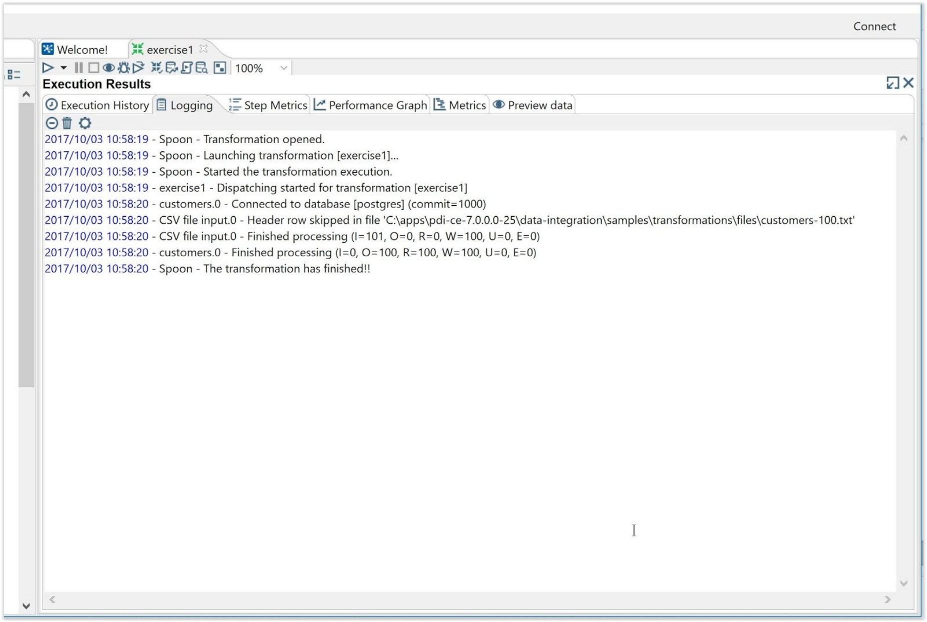
Every step is an object with an input and output. When the input is from internal steps it is counted as a **Read**, if the input is from outside of the transform (from a text file in this case), it is counted as an **Input** row count. When the output of a step is to another step that the rows counted are in the **Write** counter and if the output is to an external output, in this case it is a table in postgres, it is counted as an **Output**.

- Logging Tab



```
2017/10/03 10:58:19 - Spoon - Launching transformation [exercise1]...
2017/10/03 10:58:19 - Spoon - Started the transformation execution.
2017/10/03 10:58:19 - exercise1 - Dispatching started for transformation [exercise1]
2017/10/03 10:58:20 - customers.0 - Connected to database [postgres] (commit=1000)
2017/10/03 10:58:20 - CSV file input.0 - Header row skipped in file 'C:\apps\pdi-ce-7.0.0.0-25\data-integration\samples\transformations\files\customers-100.txt'
2017/10/03 10:58:20 - CSV file input.0 - Finished processing (I=101, O=0, R=0, W=100, U=0, E=0)
2017/10/03 10:58:20 - customers.0 - Finished processing (I=0, O=100, R=100, W=100, U=0, E=0)
2017/10/03 10:58:20 - Spoon - The transformation has finished!!
```

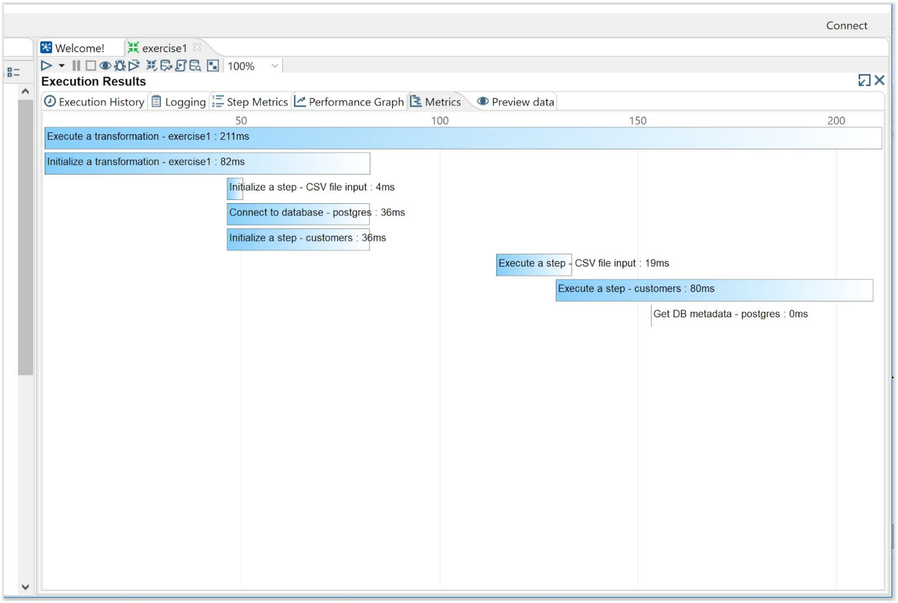
Logging tab contains the log of the run. You change the logging level at the Run dialog. The most detailed setting can be very verbose. Note you can click on the full pane Icon to see the full log.



```
2017/10/03 10:58:19 - Spoon - Transformation opened.
2017/10/03 10:58:19 - Spoon - Launching transformation [exercise1]...
2017/10/03 10:58:19 - Spoon - Started the transformation execution.
2017/10/03 10:58:19 - exercise1 - Dispatching started for transformation [exercise1]
2017/10/03 10:58:20 - customers.0 - Connected to database [postgres] (commit=1000)
2017/10/03 10:58:20 - CSV file input.0 - Header row skipped in file 'C:\apps\pdi-ce-7.0.0.0-25\data-integration\samples\transformations\files\customers-100.txt'
2017/10/03 10:58:20 - CSV file input.0 - Finished processing (I=101, O=0, R=0, W=100, U=0, E=0)
2017/10/03 10:58:20 - customers.0 - Finished processing (I=0, O=100, R=100, W=100, U=0, E=0)
2017/10/03 10:58:20 - Spoon - The transformation has finished!!
```

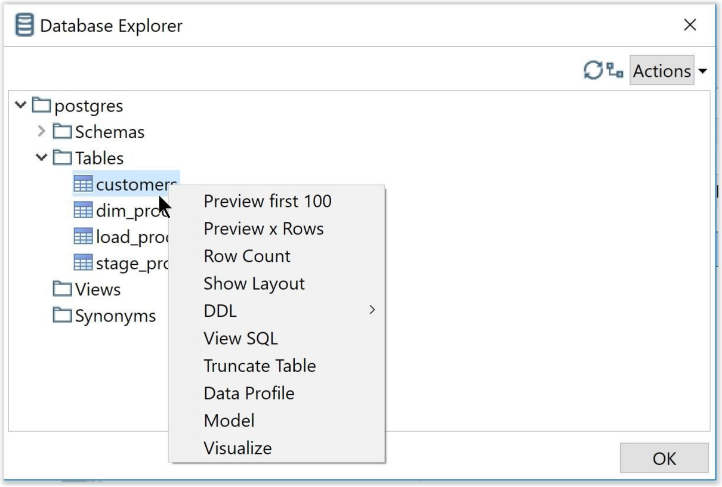
Full pane to view the logs.

- Metrics Tab

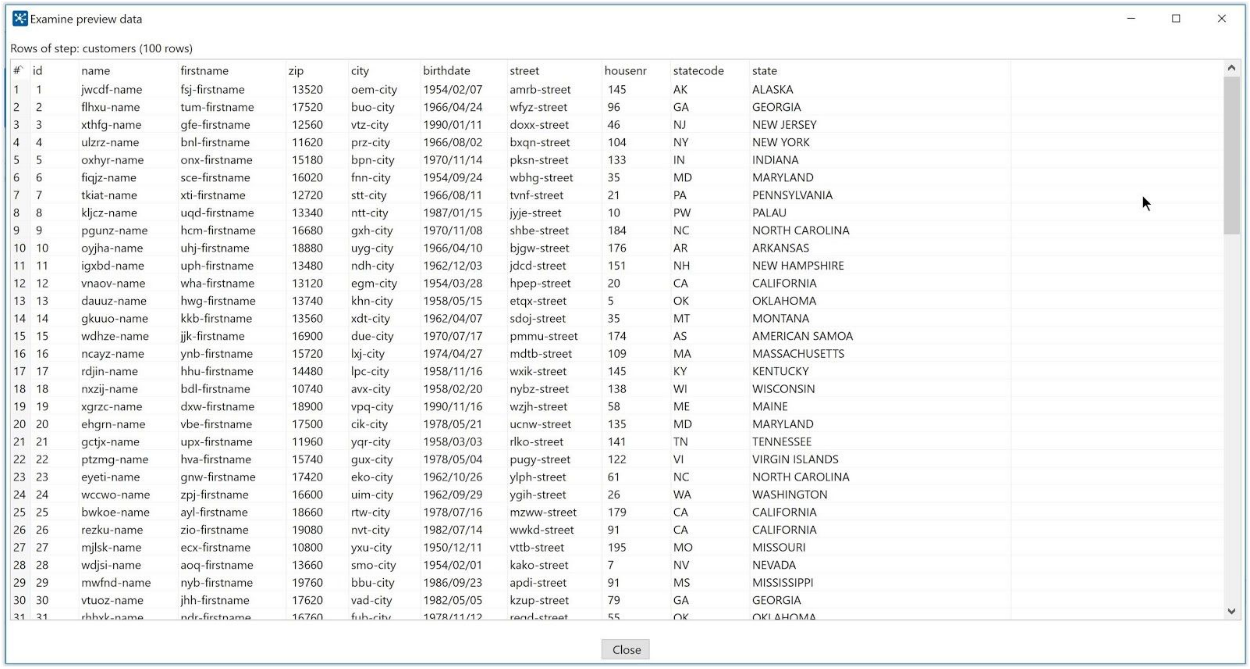

Connect
Welcome!
exercise1
100%
Execution Results
Execution History
Logging
Step Metrics
Performance Graph
Metrics
Preview data
50
100
150
200
Execute a transformation - exercise1 : 211ms
Initialize a transformation - exercise1 : 82ms
Initialize a step - CSV file input : 4ms
Connect to database - postgres : 36ms
Initialize a step - customers : 36ms
Execute a step - CSV file input : 19ms
Execute a step - customers : 80ms
Get DB metadata - postgres : 0ms

The Metrics tab shows a Gantt chart after your transformation executes. It contains information such as how long it takes to connect to a database, how much time is spent executing a SQL query, or how long it takes to load a transformation.

13. Check the table by browsing the table



Click on Browse Database connection Icon and navigate to customers table and right-click. Chose preview first 100 rows.



Rows of step: customers (100 rows)

#	id	name	firstname	zip	city	birthdate	street	housenr	statecode	state
1	1	jwcdf-name	fsj-firstname	13520	oem-city	1954/02/07	amrb-street	145	AK	ALASKA
2	2	flhxu-name	tum-firstname	17520	buo-city	1966/04/24	wfyz-street	96	GA	GEORGIA
3	3	xthfg-name	gfe-firstname	12560	vtz-city	1990/01/11	doxx-street	46	NJ	NEW JERSEY
4	4	ulzrz-name	bnl-firstname	11620	prz-city	1966/08/02	bxqn-street	104	NY	NEW YORK
5	5	oxhyr-name	onx-firstname	15180	bpn-city	1970/11/14	pksn-street	133	IN	INDIANA
6	6	fiqjz-name	sce-firstname	16020	fnn-city	1954/09/24	wbhg-street	35	MD	MARYLAND
7	7	tkiat-name	xti-firstname	12720	stt-city	1966/08/11	tvnf-street	21	PA	PENNSYLVANIA
8	8	kljcz-name	uqd-firstname	13340	ntt-city	1987/01/15	jyje-street	10	PW	PALAU
9	9	pgunz-name	hcm-firstname	16680	gxh-city	1970/11/08	shbe-street	184	NC	NORTH CAROLINA
10	10	oyjha-name	uhj-firstname	18880	uyg-city	1966/04/10	bjgw-street	176	AR	ARKANSAS
11	11	igxbd-name	uph-firstname	13480	ndh-city	1962/12/03	jdcd-street	151	NH	NEW HAMPSHIRE
12	12	vnaov-name	wha-firstname	13120	egm-city	1954/03/28	hpep-street	20	CA	CALIFORNIA
13	13	dauuz-name	hwg-firstname	13740	khn-city	1958/05/15	etqx-street	5	OK	OKLAHOMA
14	14	gkuuo-name	kkb-firstname	13560	xdt-city	1962/04/07	sdoj-street	35	MT	MONTANA
15	15	wdhze-name	jjk-firstname	16900	due-city	1970/07/17	pmmu-street	174	AS	AMERICAN SAMOA
16	16	ncayz-name	ynb-firstname	15720	lxj-city	1974/04/27	mdtb-street	109	MA	MASSACHUSETTS
17	17	rdjin-name	hhu-firstname	14480	lpc-city	1958/11/16	wxik-street	145	KY	KENTUCKY
18	18	nxzij-name	bdl-firstname	10740	avx-city	1958/02/20	nybz-street	138	WI	WISCONSIN
19	19	xgrzc-name	dxw-firstname	18900	vpq-city	1990/11/16	wzjh-street	58	ME	MAINE
20	20	ehgrn-name	vbe-firstname	17500	cik-city	1978/05/21	ucnw-street	135	MD	MARYLAND
21	21	gctjx-name	upx-firstname	11960	yqr-city	1958/03/03	rlko-street	141	TN	TENNESSEE
22	22	ptzmg-name	hva-firstname	15740	gux-city	1978/05/04	pugy-street	122	VI	VIRGIN ISLANDS
23	23	eyeti-name	gnw-firstname	17420	eko-city	1962/10/26	ylph-street	61	NC	NORTH CAROLINA
24	24	wccwo-name	zpj-firstname	16600	uim-city	1962/09/29	ygih-street	26	WA	WASHINGTON
25	25	bwkoe-name	ayl-firstname	18660	rtw-city	1978/07/16	mzww-street	179	CA	CALIFORNIA
26	26	rezku-name	zio-firstname	19080	nvt-city	1982/07/14	wwkd-street	91	CA	CALIFORNIA
27	27	mjlsk-name	ecx-firstname	10800	yxu-city	1950/12/11	vttb-street	195	MO	MISSOURI
28	28	wdjsi-name	aoq-firstname	13660	smo-city	1954/02/01	kako-street	7	NV	NEVADA
29	29	mwfnd-name	nyb-firstname	19760	bbu-city	1986/09/23	apdi-street	91	MS	MISSISSIPPI
30	30	vtuoz-name	jhh-firstname	17620	vad-city	1982/05/05	kzup-street	79	GA	GEORGIA
31	31	rhbyk-name	ndr-firstname	16760	fub-city	1978/11/12	reqd-street	55	OK	OKLAHOMA

Check to see that all rows are there.

Appendix:

Watch this 5-minute introductory youtube video. It illustrates how to join two tables in Spoon.
https://www.youtube.com/watch?v=RGmm_xXUwrM

User guide for Spoon -
http://wiki.pentaho.com/display/EAI/Spoon+User+Guide

A comprehensive documentation of various steps you can use in Spoon-
-https://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Steps

A comprehensive documentation of various jobs you may want to use -
https://wiki.pentaho.com/display/EAI/Pentaho+Data+Integration+Job+Entries