Class: BE 3

Batch: P3

Roll no. 41310

Name: Prem Vinod Bansod

Date: 19-8-2020

Assignment 1

Problem Definition:

For an organization of your choice, choose a set of business processes. Design star / snow flake schemas for analysing these processes. Create a fact constellation schema by combining them. Extract data from different data sources, apply suitable transformations and load into destination tables using an ETL tool.

For Example: Business Origination: Sales, Order, Marketing Process.

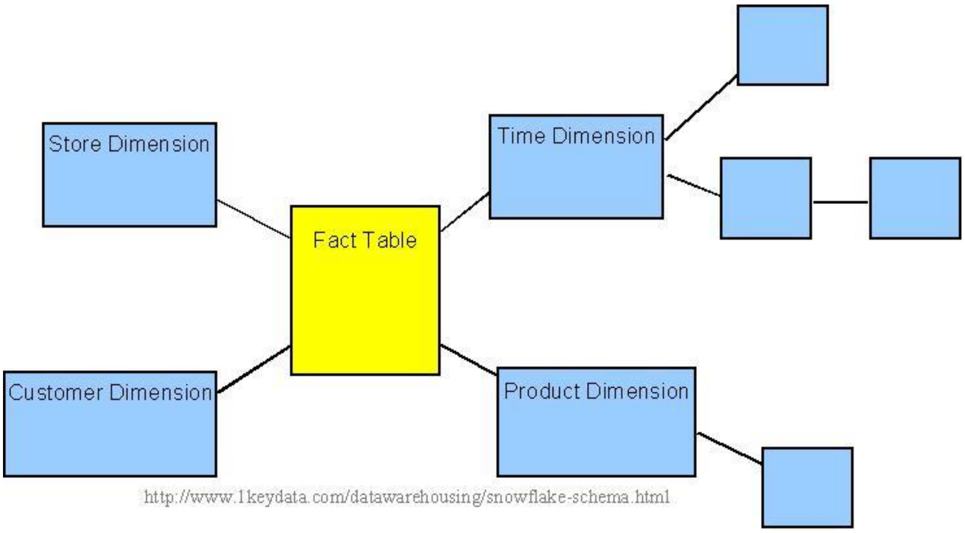
Learning Objective:

- Implementation of the problem statement using ETL Tool.
- Star / snow flake schemas for analysing processes.

Theory:

Star / snow flake schemas:

The star schema is the simplest type of Data Warehouse schema. It is known as star schema as its structure resembles a star. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. It is called snowflake because its diagram resembles a Snowflake. Star schema and the snowflake schema are ways to organize data marts or entire data warehouses using relational databases. The third differentiator in this Star schema vs Snowflake schema face-off is the performance of these models. The Snowflake model has more joins between the dimension table and the fact table, so the performance is slower. The Star model, on the other hand, has fewer joins between dimension tables and the facts table.



http://www.1keydata.com/datawarehousing/snowflake-schema.html

Characteristics of Star Schema:

1. The dimension table should contain the set of attributes.
2. The dimension table is joined to the fact table using a foreign key.
3. The dimension table are not joined to each other.
4. The schema is widely supported by BI Tools.

Snowflake Schema:

Snowflake Schema is also the type of multidimensional model which is used for data warehouse. In snowflake schema, the fact tables, dimension tables as well as sub dimension tables are contained. This schema forms a snowflake with fact tables, dimension tables as well as sub- dimension tables.

Characteristics of Snowflake Schema:

1. The main benefit of the snowflake schema it uses smaller disk space.
2. Easier to implement a dimension is added to the Schema.
3. Due to multiple tables query performance is reduced.

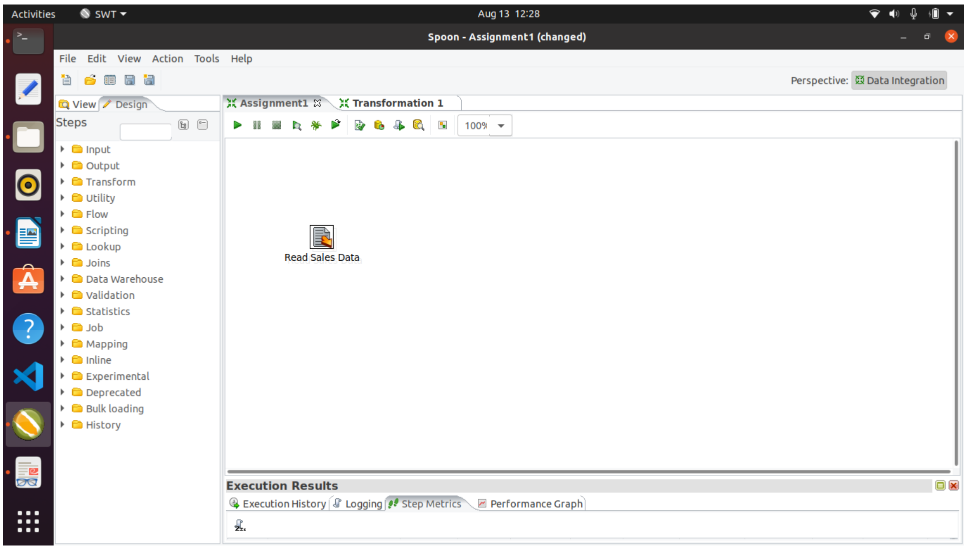
What is ETL?

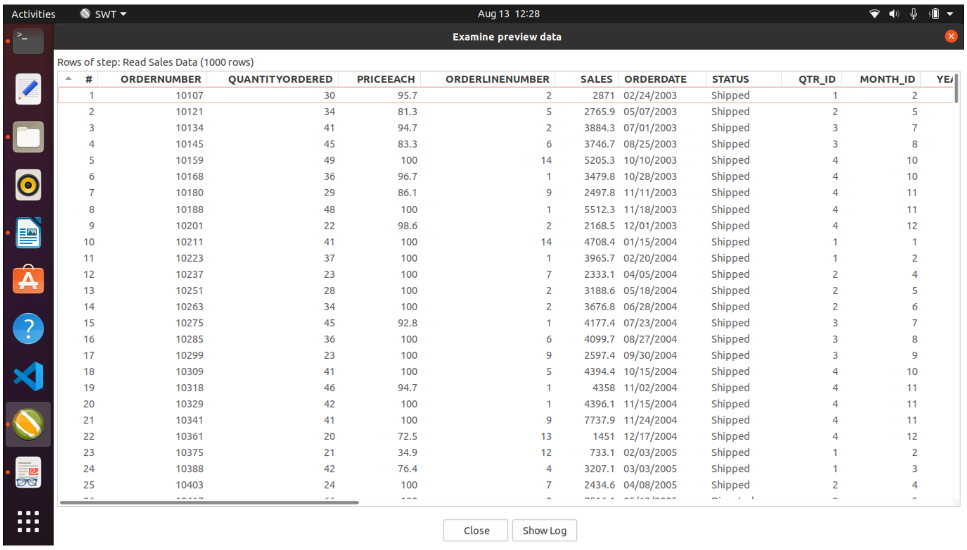
ETL is an abbreviation of Extract, Transform and Load. In this process, an ETL tool extracts the data from different RDBMS source systems then transforms the data like applying calculations, concatenations, etc. and then load the data into the Data Warehouse system. In ETL data is flows from the source to the target. In ETL process transformation engine takes care of any data changes.

List of open sources ETL Tools:

1. CloverETL
2. Jedox
3. Pentaho
4. Talend

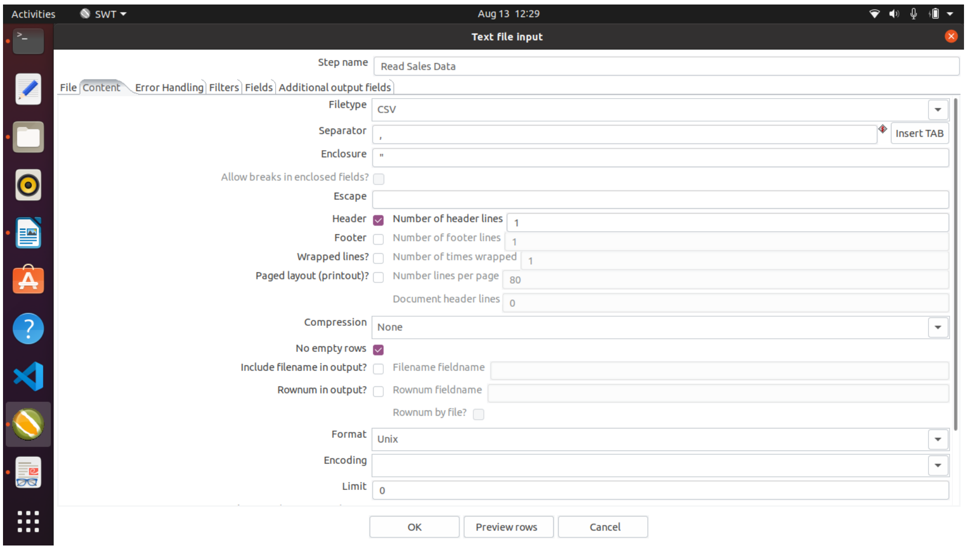
Steps:





Rows of step: Read Sales Data (1000 rows)

#	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER	SALES	ORDERDATE	STATUS	QTR_ID	MONTH_ID	YEA
1	10107	30	95.7	2	2871	02/24/2003	Shipped	1	2	
2	10121	34	81.3	5	2765.9	05/07/2003	Shipped	2	5	
3	10134	41	94.7	2	3884.3	07/01/2003	Shipped	3	7	
4	10145	45	83.3	6	3746.7	08/25/2003	Shipped	3	8	
5	10159	49	100	14	5205.3	10/10/2003	Shipped	4	10	
6	10168	36	96.7	1	3479.8	10/28/2003	Shipped	4	10	
7	10180	29	86.1	9	2497.8	11/11/2003	Shipped	4	11	
8	10188	48	100	1	5512.3	11/18/2003	Shipped	4	11	
9	10201	22	98.6	2	2168.5	12/01/2003	Shipped	4	12	
10	10211	41	100	14	4708.4	01/15/2004	Shipped	1	1	
11	10223	37	100	1	3965.7	02/20/2004	Shipped	1	2	
12	10237	23	100	7	2333.1	04/05/2004	Shipped	2	4	
13	10251	28	100	2	3188.6	05/18/2004	Shipped	2	5	
14	10263	34	100	2	3676.8	06/28/2004	Shipped	2	6	
15	10275	45	92.8	1	4177.4	07/23/2004	Shipped	3	7	
16	10285	36	100	6	4099.7	08/27/2004	Shipped	3	8	
17	10299	23	100	9	2597.4	09/30/2004	Shipped	3	9	
18	10309	41	100	5	4394.4	10/15/2004	Shipped	4	10	
19	10318	46	94.7	1	4358	11/02/2004	Shipped	4	11	
20	10329	42	100	1	4396.1	11/15/2004	Shipped	4	11	
21	10341	41	100	9	7737.9	11/24/2004	Shipped	4	11	
22	10361	20	72.5	13	1451	12/17/2004	Shipped	4	12	
23	10375	21	34.9	12	733.1	02/03/2005	Shipped	1	2	
24	10388	42	76.4	4	3207.1	03/03/2005	Shipped	1	3	
25	10403	24	100	7	2434.6	04/08/2005	Shipped	2	4	


Activities
SWT
Aug 13 12:29

Text file input

Step name: Read Sales Data

File | Content | Error Handling | Filters | Fields | Additional output fields

Filetype: CSV

Separator: ,

Enclosure: "

Allow breaks in enclosed fields? ☐

Escape:

Header ☑ Number of header lines: 1

Footer ☐ Number of footer lines: 1

Wrapped lines? ☐ Number of times wrapped: 1

Paged layout (printout)? ☐ Number lines per page: 80

Document header lines: 0

Compression: None

No empty rows ☑

Include filename in output? ☐ Filename fieldname:

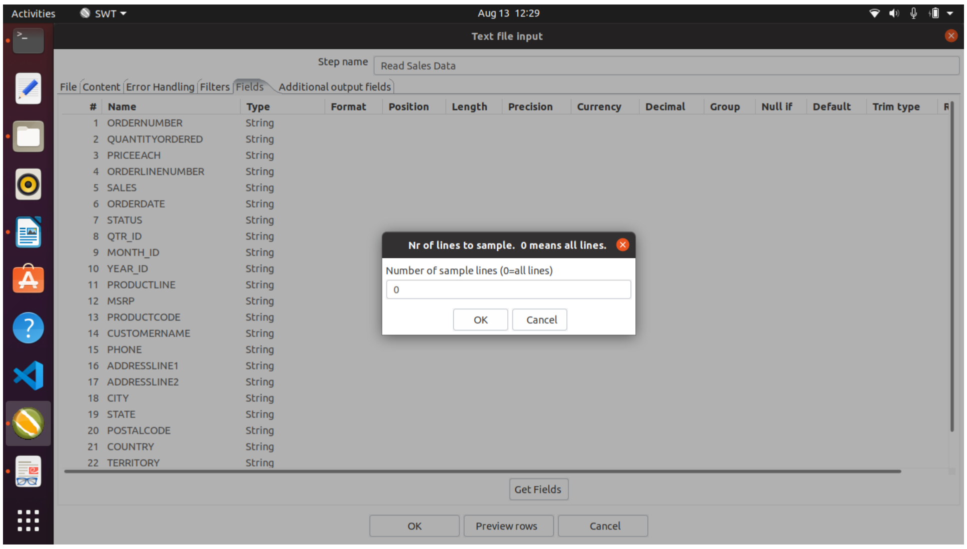
Rownum in output? ☐ Rownum fieldname:

Rownum by file? ☐

Format: Unix

Encoding:

Limit: 0


Activities
SWT
Aug 13 12:29

Text file input

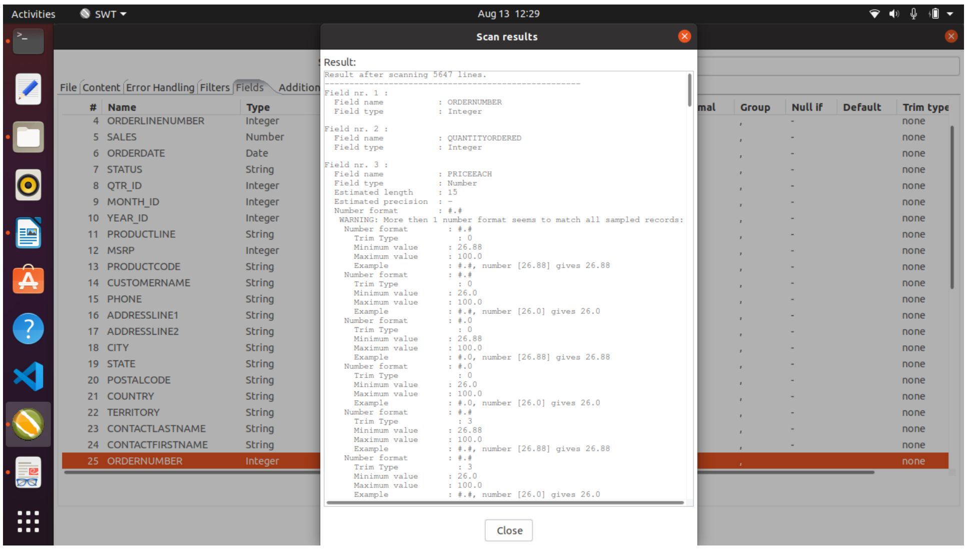
Step name: Read Sales Data

File | Content | Error Handling | Filters | Fields | Additional output fields

#	Name	Type	Format	Position	Length	Precision	Currency	Decimal	Group	Null if	Default	Trim type	R
1	ORDERNUMBER	String											
2	QUANTITYORDERED	String											
3	PRICEEACH	String											
4	ORDERLINENUMBER	String											
5	SALES	String											
6	ORDERDATE	String											
7	STATUS	String											
8	QTR_ID	String											
9	MONTH_ID	String											
10	YEAR_ID	String											
11	PRODUCTLINE	String											
12	MSRP	String											
13	PRODUCTCODE	String											
14	CUSTOMERNAME	String											
15	PHONE	String											
16	ADDRESSLINE1	String											
17	ADDRESSLINE2	String											
18	CITY	String											
19	STATE	String											
20	POSTALCODE	String											
21	COUNTRY	String											
22	TERRITORY	String											

Nr of lines to sample. 0 means all lines.

Number of sample lines (0=all lines): 0



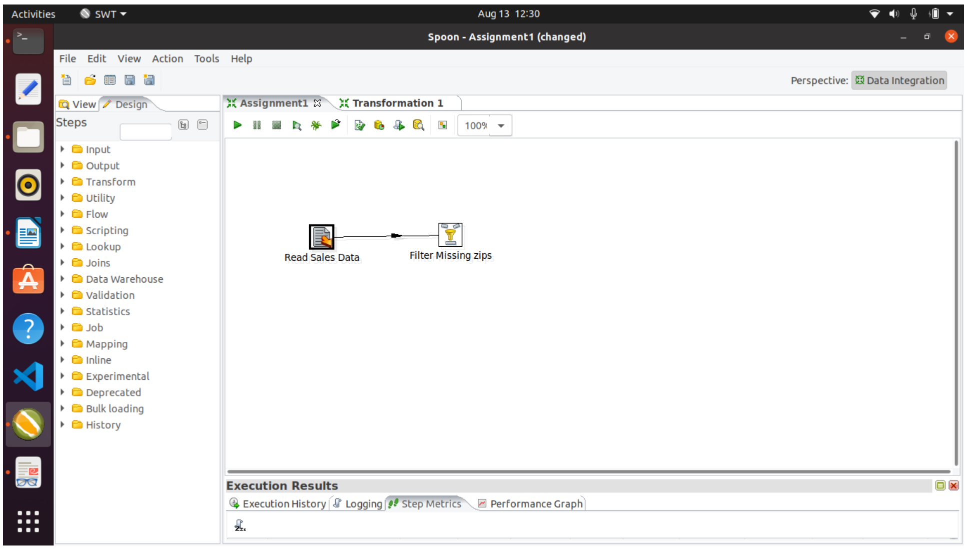
#	Name	Type	Group	Null if	Default	Trim type
4	ORDERLINENUMBER	Integer	,	-		none
5	SALES	Number	,	-		none
6	ORDERDATE	Date	,	-		none
7	STATUS	String	,	-		none
8	QTR_ID	Integer	,	-		none
9	MONTH_ID	Integer	,	-		none
10	YEAR_ID	Integer	,	-		none
11	PRODUCTLINE	String	,	-		none
12	MSRP	Integer	,	-		none
13	PRODUCTCODE	String	,	-		none
14	CUSTOMERNAME	String	,	-		none
15	PHONE	String	,	-		none
16	ADDRESSLINE1	String	,	-		none
17	ADDRESSLINE2	String	,	-		none
18	CITY	String	,	-		none
19	STATE	String	,	-		none
20	POSTALCODE	String	,	-		none
21	COUNTRY	String	,	-		none
22	TERRITORY	String	,	-		none
23	CONTACTLASTNAME	String	,	-		none
24	CONTACTFIRSTNAME	String	,	-		none
25	ORDERNUMBER	Integer	,			none

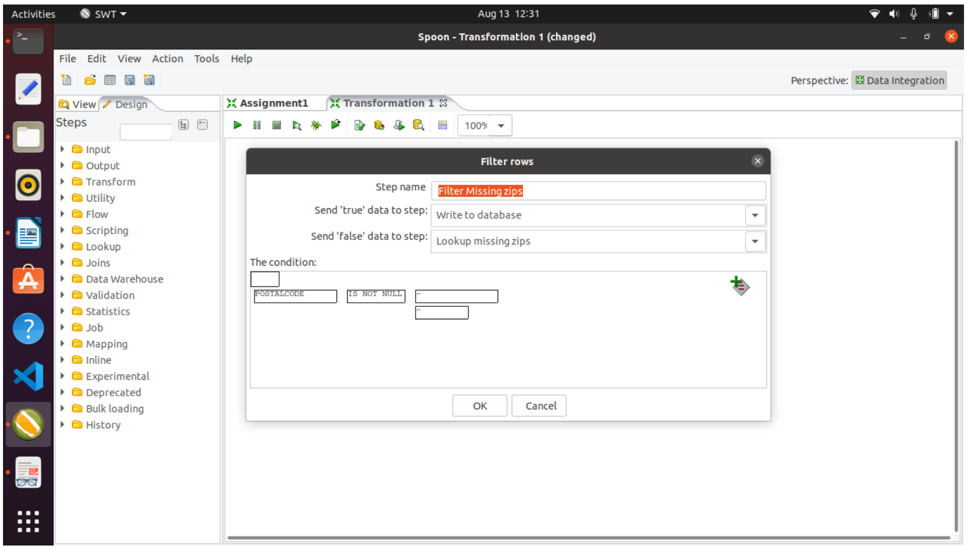
Result:

```
Result after scanning 5647 lines.
----------------------------------------------------
Field nr. 1 :
  Field name           : ORDERNUMBER
  Field type           : Integer

Field nr. 2 :
  Field name           : QUANTITYORDERED
  Field type           : Integer

Field nr. 3 :
  Field name           : PRICEEACH
  Field type           : Number
  Estimated length     : 15
  Estimated precision  : -
  Number format        : #.#
   WARNING: More then 1 number format seems to match all sampled records:
    Number format        : #.#
      Trim Type            : 0
      Minimum value      : 26.88
      Maximum value      : 100.0
      Example            : #.#, number [26.88] gives 26.88
    Number format        : #.#
      Trim Type            : 0
      Minimum value      : 26.0
      Maximum value      : 100.0
      Example            : #.#, number [26.0] gives 26.0
    Number format        : #.0
      Trim Type            : 0
      Minimum value      : 26.88
      Maximum value      : 100.0
      Example            : #.0, number [26.88] gives 26.88
    Number format        : #.0
      Trim Type            : 0
      Minimum value      : 26.0
      Maximum value      : 100.0
      Example            : #.0, number [26.0] gives 26.0
    Number format        : #.#
      Trim Type            : 3
      Minimum value      : 26.88
      Maximum value      : 100.0
      Example            : #.#, number [26.88] gives 26.88
    Number format        : #.#
      Trim Type            : 3
      Minimum value      : 26.0
      Maximum value      : 100.0
      Example            : #.#, number [26.0] gives 26.0
```





Filter rows

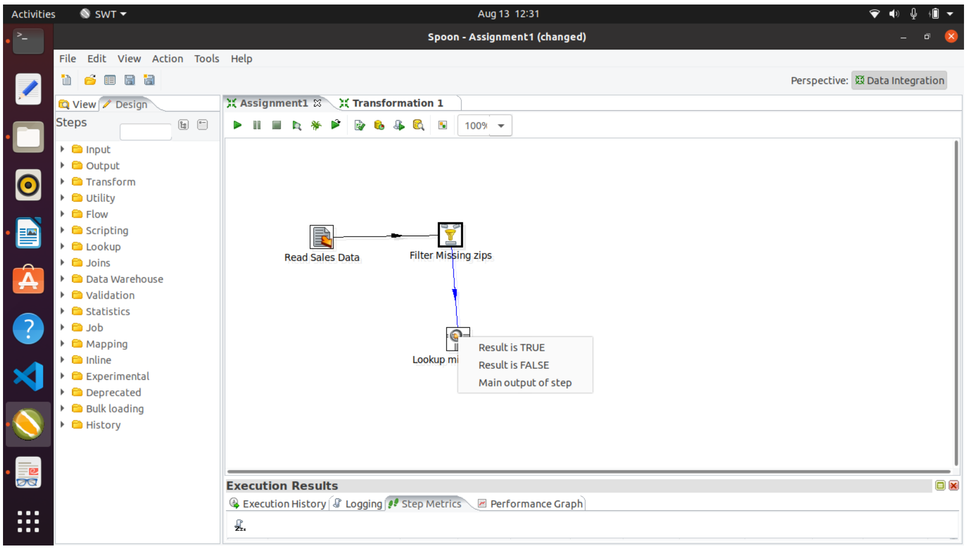
Step name: Filter Missing zips

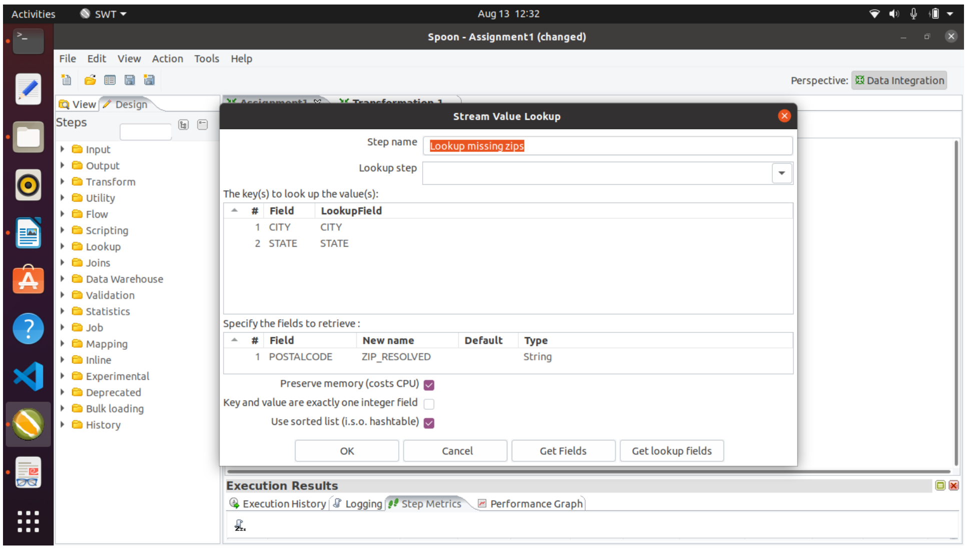
Send 'true' data to step: Write to database

Send 'false' data to step: Lookup missing zips

The condition:

POSTALCODE IS NOT NULL





Stream Value Lookup

Step name: Lookup missing zips

Lookup step:

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

#	Field	LookupField
1	CITY	CITY
2	STATE	STATE

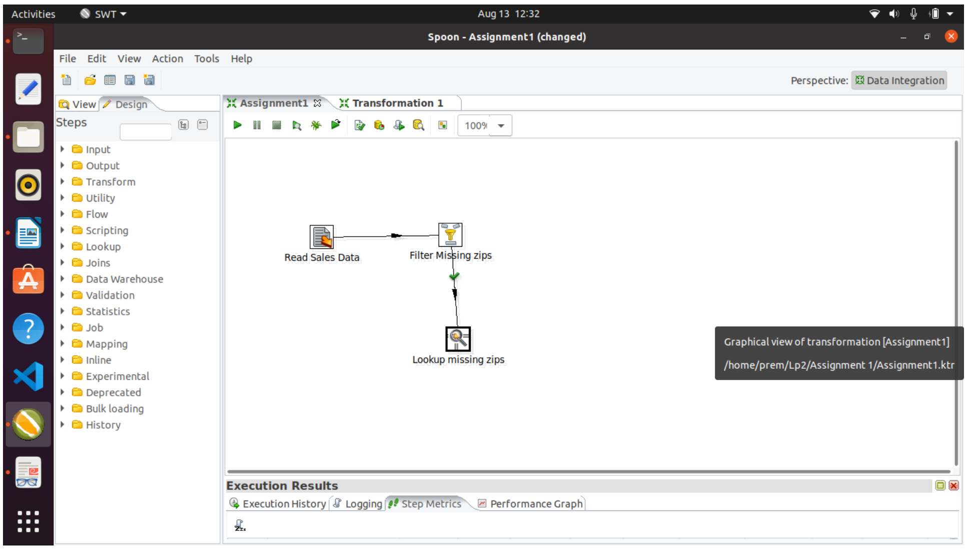
Specify the fields to retrieve :

#	Field	New name	Default	Type
1	POSTALCODE	ZIP_RESOLVED		String

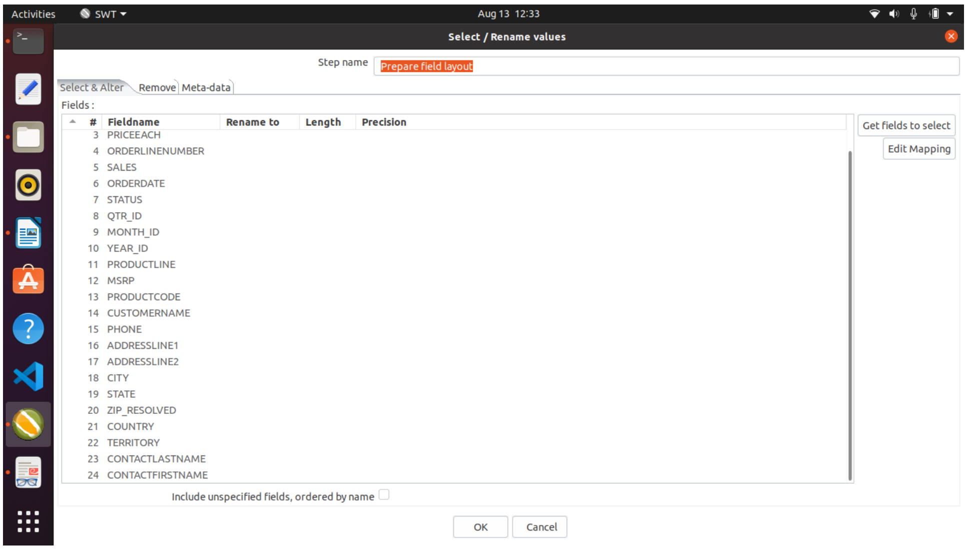
Preserve memory (costs CPU) ☑

Key and value are exactly one integer field □

Use sorted list (i.s.o. hashtable) ☑



Graphical view of transformation [Assignment1]
/home/prem/Lp2/Assignment 1/Assignment1.ktr



Select / Rename values

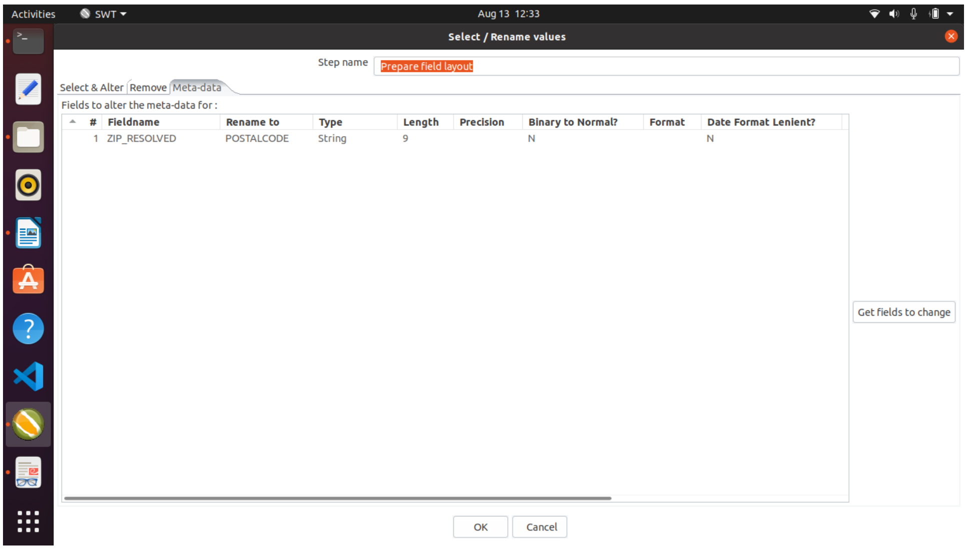
Step name: Prepare field layout

Select & Alter | Remove | Meta-data

Fields :

#	Fieldname	Rename to	Length	Precision
3	PRICEEACH			
4	ORDERLINENUMBER			
5	SALES			
6	ORDERDATE			
7	STATUS			
8	QTR_ID			
9	MONTH_ID			
10	YEAR_ID			
11	PRODUCTLINE			
12	MSRP			
13	PRODUCTCODE			
14	CUSTOMERNAME			
15	PHONE			
16	ADDRESSLINE1			
17	ADDRESSLINE2			
18	CITY			
19	STATE			
20	ZIP_RESOLVED			
21	COUNTRY			
22	TERRITORY			
23	CONTACTLASTNAME			
24	CONTACTFIRSTNAME			

Include unspecified fields, ordered by name □



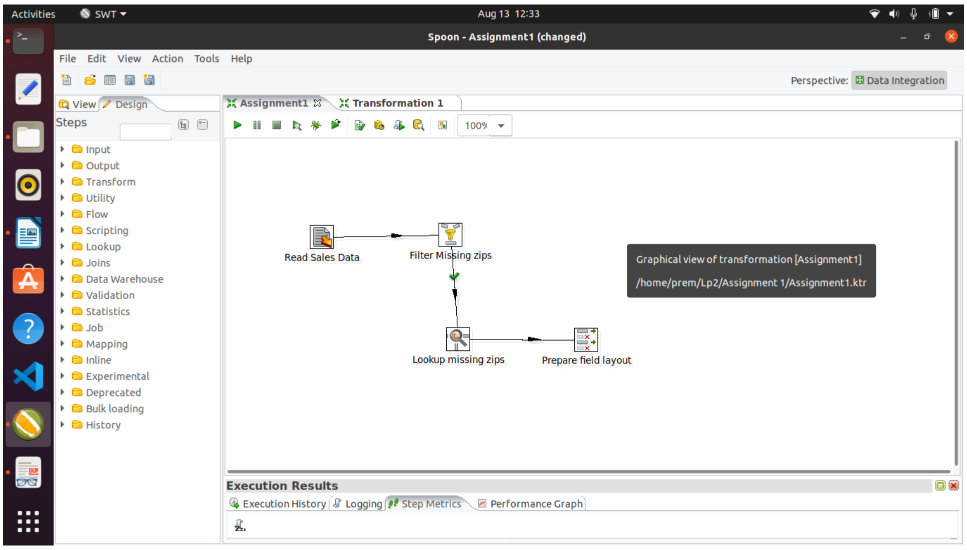
Select / Rename values

Step name: Prepare field layout

Select & Alter | Remove | Meta-data

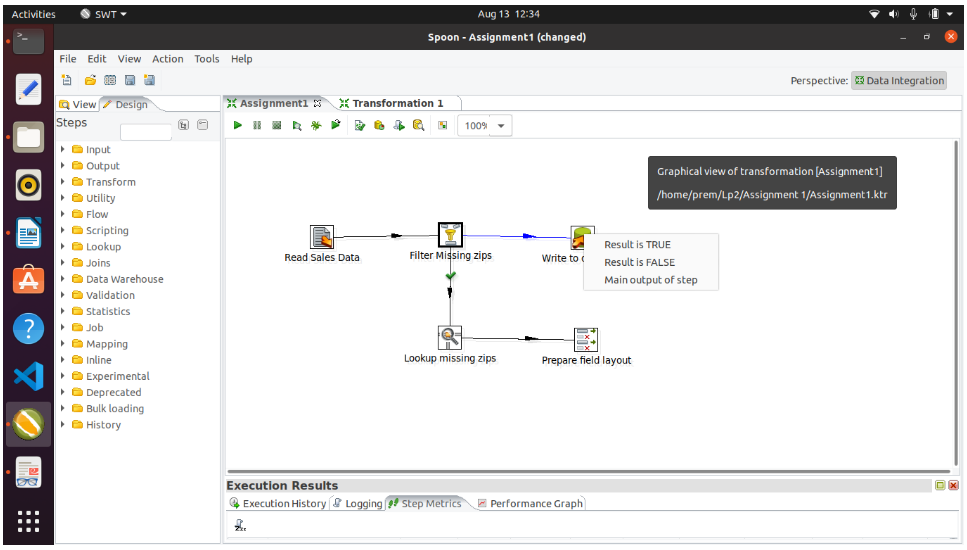
Fields to alter the meta-data for :

#	Fieldname	Rename to	Type	Length	Precision	Binary to Normal?	Format	Date Format Lenient?
1	ZIP_RESOLVED	POSTALCODE	String	9		N		N

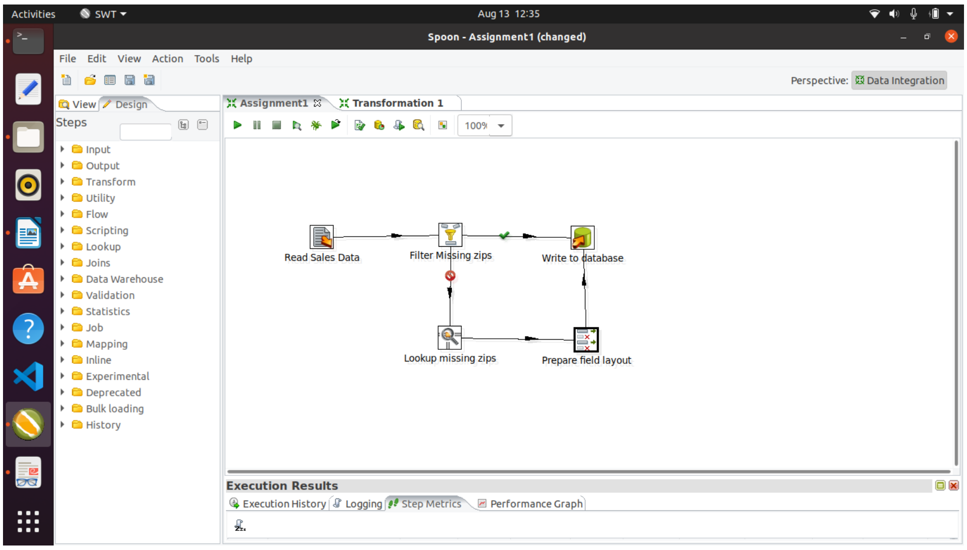


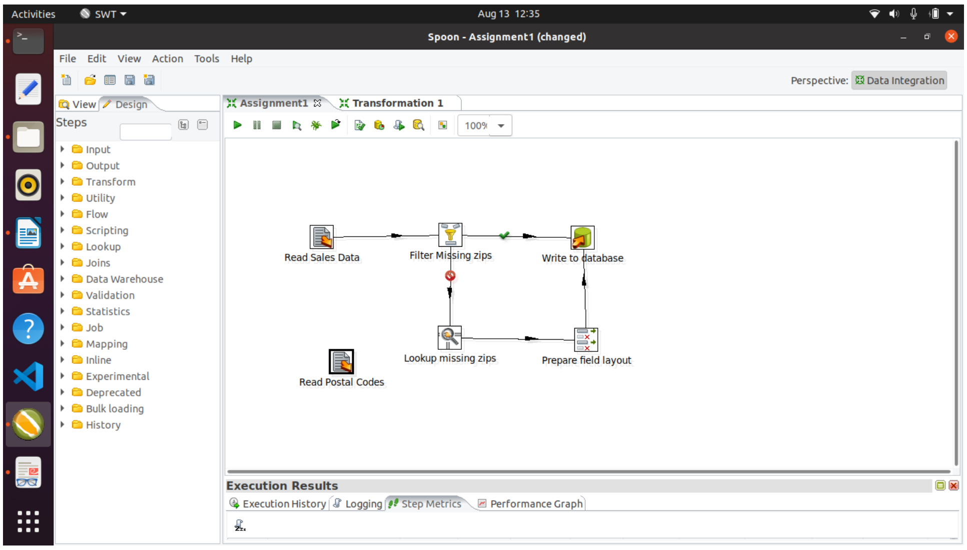
Spoon - Assignment1 (changed)

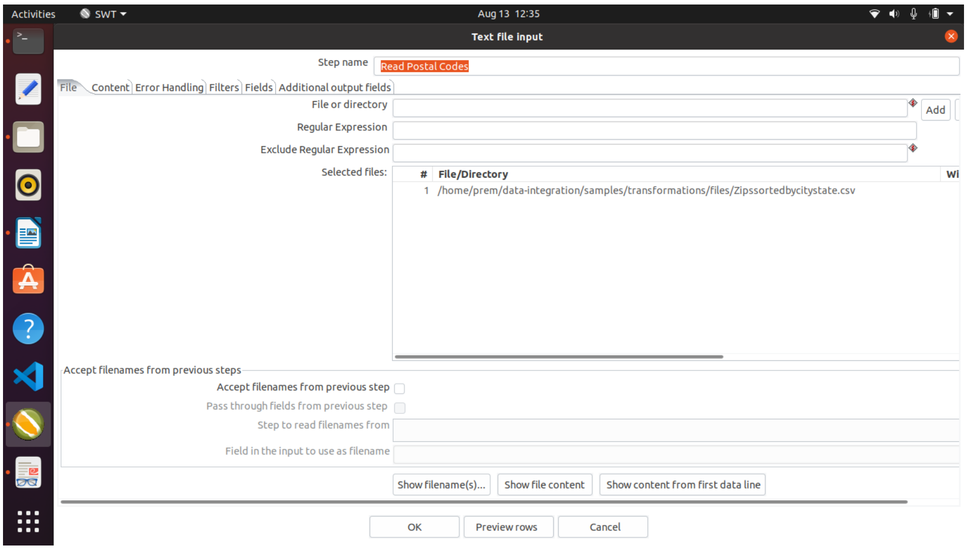
Graphical view of transformation [Assignment1]
/home/prem/Lp2/Assignment 1/Assignment1.ktr



Graphical view of transformation [Assignment1]
/home/prem/Lp2/Assignment 1/Assignment1.ktr






Text file input

Step name: Read Postal Codes

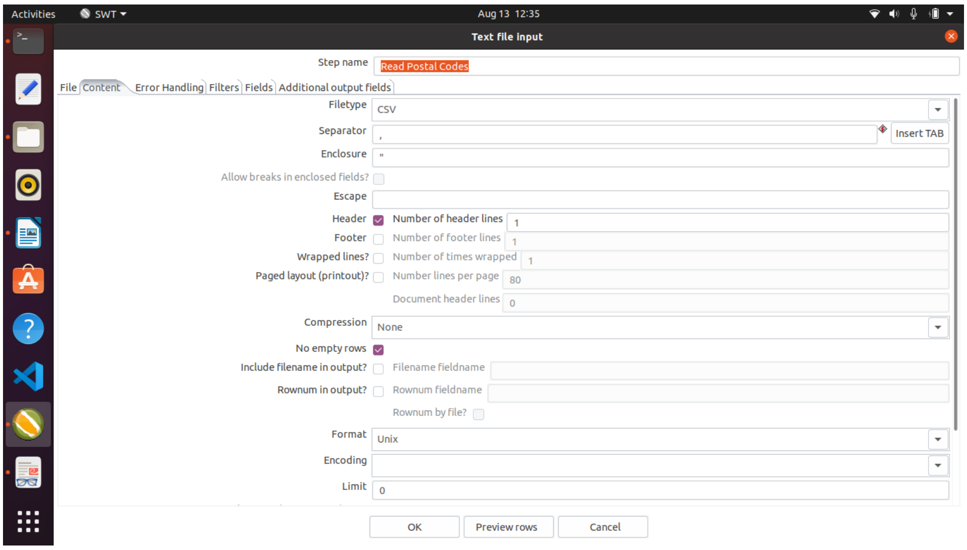
Selected files:

#	File/Directory
1	/home/prem/data-integration/samples/transformations/files/Zipssortedbycitystate.csv

Accept filenames from previous steps

Accept filenames from previous step □

Pass through fields from previous step □


Text file input

Step name: Read Postal Codes

Filetype: CSV

Separator: ,

Enclosure: "

Allow breaks in enclosed fields? □

Escape:

Header ☑ Number of header lines: 1

Footer □ Number of footer lines: 1

Wrapped lines? □ Number of times wrapped: 1

Paged layout (printout)? □ Number lines per page: 80

Document header lines: 0

Compression: None

No empty rows ☑

Include filename in output? □ Filename fieldname

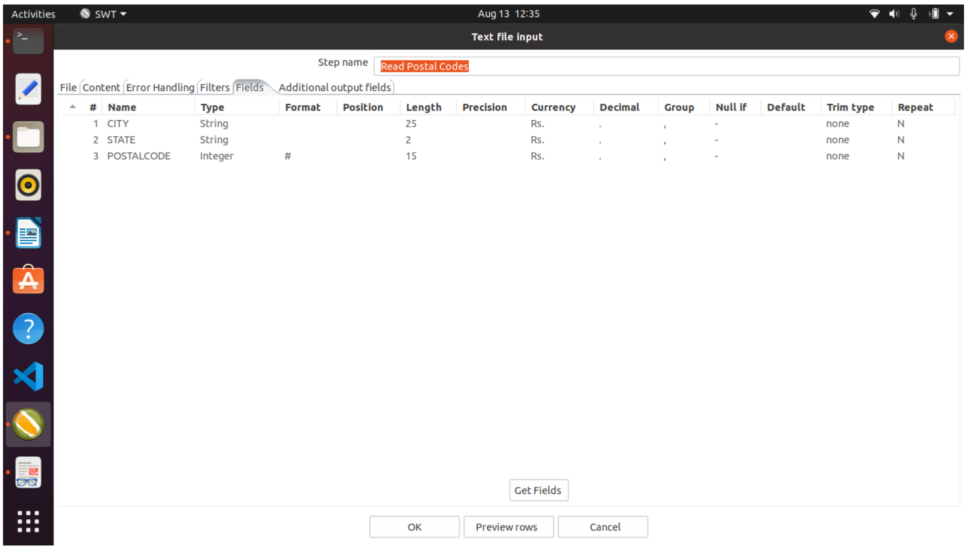
Rownum in output? □ Rownum fieldname

Rownum by file? □

Format: Unix

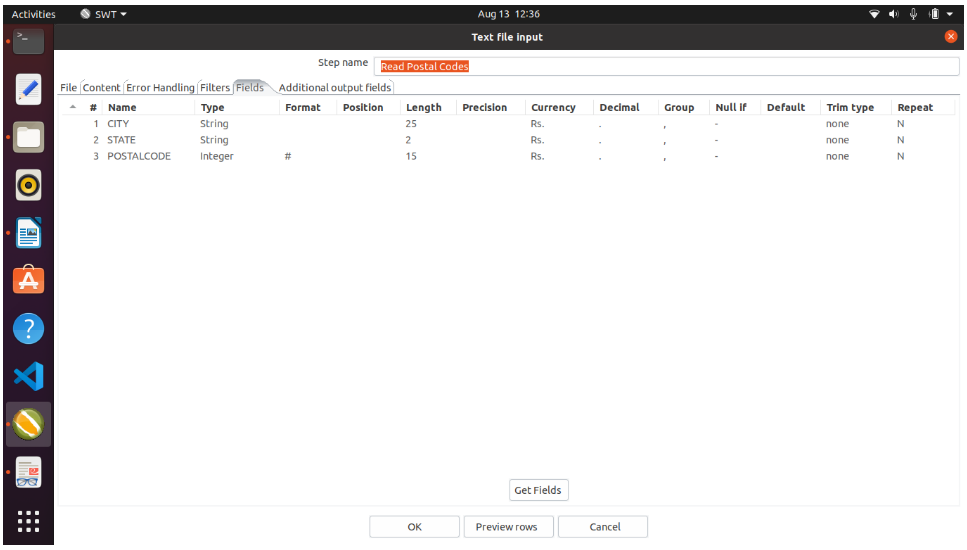
Encoding:

Limit: 0



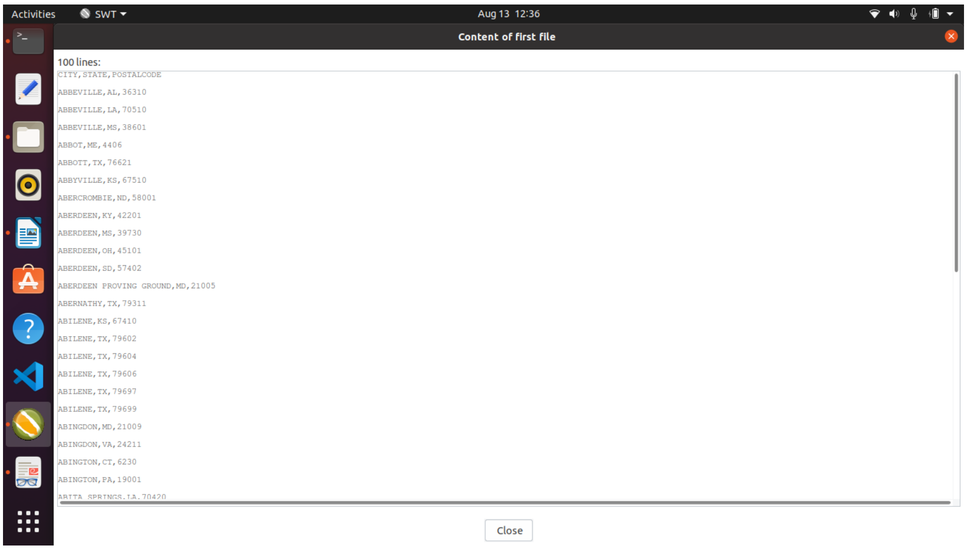
Step name: Read Postal Codes

#	Name	Type	Format	Position	Length	Precision	Currency	Decimal	Group	Null if	Default	Trim type	Repeat
1	CITY	String			25		Rs.	.	,	-		none	N
2	STATE	String			2		Rs.	.	,	-		none	N
3	POSTALCODE	Integer	#		15		Rs.	.	,	-		none	N



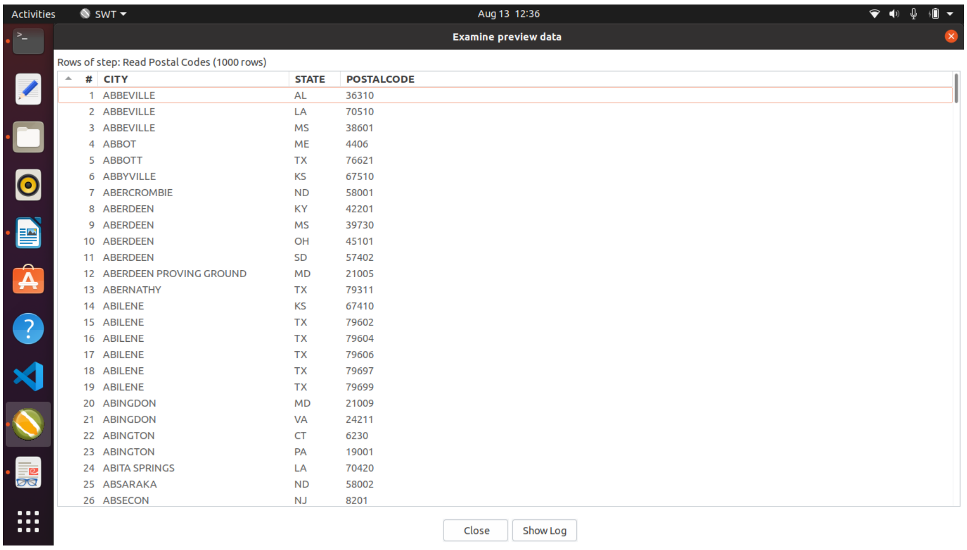
Step name: Read Postal Codes

#	Name	Type	Format	Position	Length	Precision	Currency	Decimal	Group	Null if	Default	Trim type	Repeat
1	CITY	String			25		Rs.	.	,	-		none	N
2	STATE	String			2		Rs.	.	,	-		none	N
3	POSTALCODE	Integer	#		15		Rs.	.	,	-		none	N



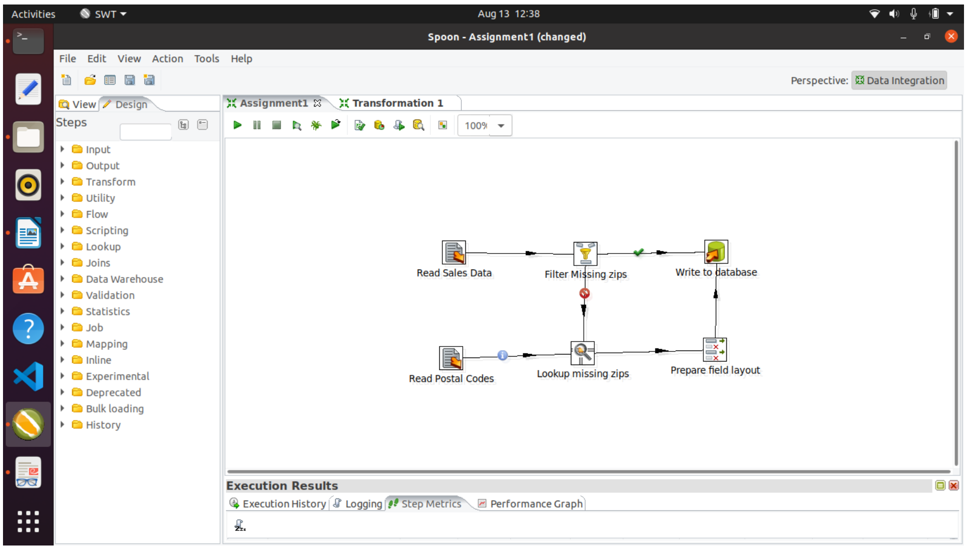
100 lines:

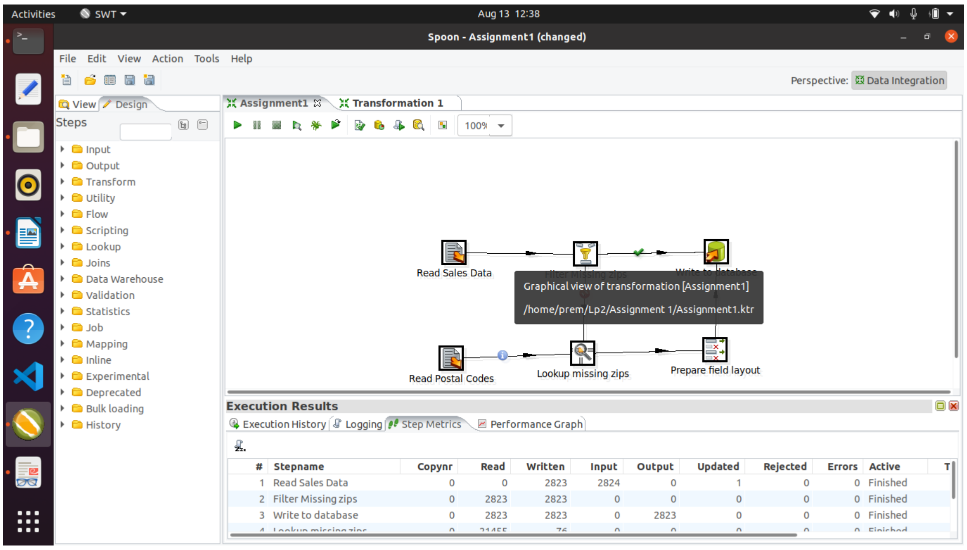
```
CITY,STATE,POSTALCODE
ABBEVILLE,AL,36310
ABBEVILLE,LA,70510
ABBEVILLE,MS,38601
ABBOT,ME,4406
ABBOTT,TX,76621
ABBYVILLE,KS,67510
ABERCROMBIE,ND,58001
ABERDEEN,KY,42201
ABERDEEN,MS,39730
ABERDEEN,OH,45101
ABERDEEN,SD,57402
ABERDEEN PROVING GROUND,MD,21005
ABERNATHY,TX,79311
ABILENE,KS,67410
ABILENE,TX,79602
ABILENE,TX,79604
ABILENE,TX,79606
ABILENE,TX,79697
ABILENE,TX,79699
ABINGDON,MD,21009
ABINGDON,VA,24211
ABINGTON,CT,6230
ABINGTON,PA,19001
ABITA SPRINGS,LA,70420
```



Rows of step: Read Postal Codes (1000 rows)

#	CITY	STATE	POSTALCODE
1	ABBEVILLE	AL	36310
2	ABBEVILLE	LA	70510
3	ABBEVILLE	MS	38601
4	ABBOT	ME	4406
5	ABBOTT	TX	76621
6	ABBYVILLE	KS	67510
7	ABERCROMBIE	ND	58001
8	ABERDEEN	KY	42201
9	ABERDEEN	MS	39730
10	ABERDEEN	OH	45101
11	ABERDEEN	SD	57402
12	ABERDEEN PROVING GROUND	MD	21005
13	ABERNATHY	TX	79311
14	ABILENE	KS	67410
15	ABILENE	TX	79602
16	ABILENE	TX	79604
17	ABILENE	TX	79606
18	ABILENE	TX	79697
19	ABILENE	TX	79699
20	ABINGDON	MD	21009
21	ABINGDON	VA	24211
22	ABINGTON	CT	6230
23	ABINGTON	PA	19001
24	ABITA SPRINGS	LA	70420
25	ABSARAKA	ND	58002
26	ABSECON	NJ	8201

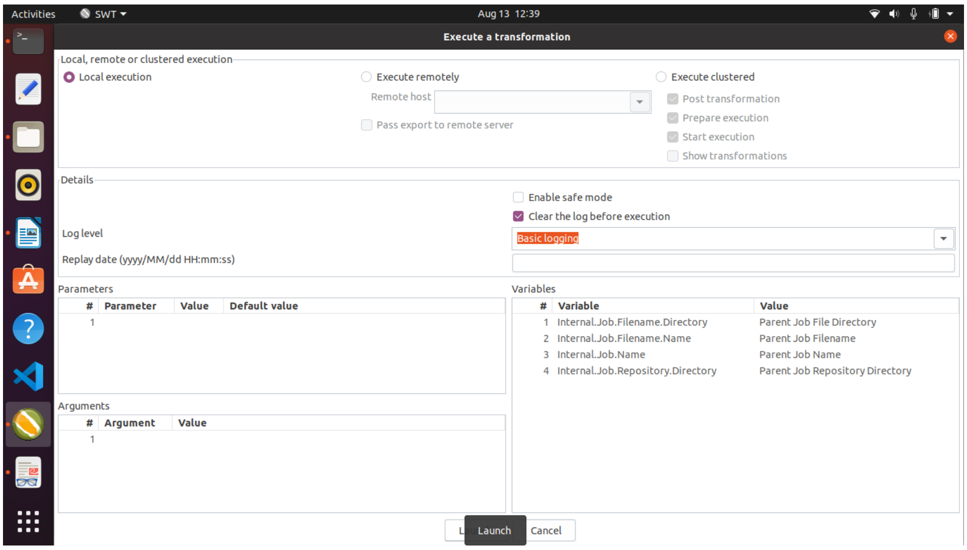




Graphical view of transformation [Assignment1]

/home/prem/Lp2/Assignment 1/Assignment1.ktr

#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	T
1	Read Sales Data	0	0	2823	2824	0	1	0	0	Finished	
2	Filter Missing zips	0	2823	2823	0	0	0	0	0	Finished	
3	Write to database	0	2823	2823	0	2823	0	0	0	Finished	
4	Lookup missing zips	0	21455	76	0	0	0	0	0	Finished	



Local, remote or clustered execution

● Local execution

○ Execute remotely

Remote host

□ Pass export to remote server

○ Execute clustered

☑ Post transformation

☑ Prepare execution

☑ Start execution

□ Show transformations

Details

□ Enable safe mode

☑ Clear the log before execution

Log level: Basic logging

Replay date (yyyy/MM/dd HH:mm:ss)

Parameters

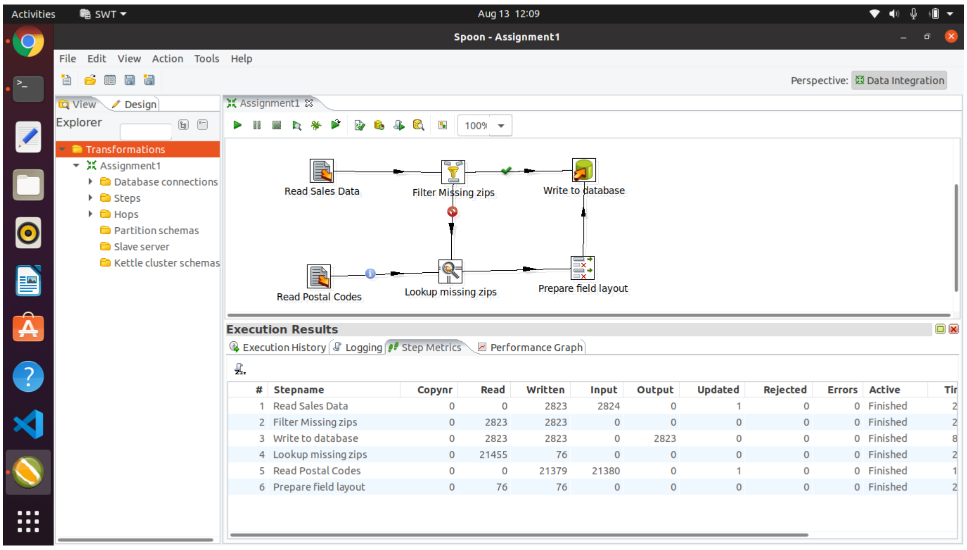
#	Parameter	Value	Default value
1			

Variables

#	Variable	Value
1	Internal.Job.Filename.Directory	Parent Job File Directory
2	Internal.Job.Filename.Name	Parent Job Filename
3	Internal.Job.Name	Parent Job Name
4	Internal.Job.Repository.Directory	Parent Job Repository Directory

Arguments

#	Argument	Value
1		



#	Stepname	Copynr	Read	Written	Input	Output	Updated	Rejected	Errors	Active	Tir
1	Read Sales Data	0	0	2823	2824	0	1	0	0	Finished	2
2	Filter Missing zips	0	2823	2823	0	0	0	0	0	Finished	2
3	Write to database	0	2823	2823	0	2823	0	0	0	Finished	8
4	Lookup missing zips	0	21455	76	0	0	0	0	0	Finished	2
5	Read Postal Codes	0	0	21379	21380	0	1	0	0	Finished	1
6	Prepare field layout	0	76	76	0	0	0	0	0	Finished	2

Conclusion:

Thus, Successfully learned how to extract data from different data sources using pentaho and apply suitable transformations and load into destination tables using an ETL tool.