



中国 R 会  
The China-R Conference



中國人民大學  
RENMIN UNIVERSITY OF CHINA

统计学院  
SCHOOL OF STATISTICS



中国人民大学应用统计科学研究中心  
Center for Applied Statistics of Renmin University of China



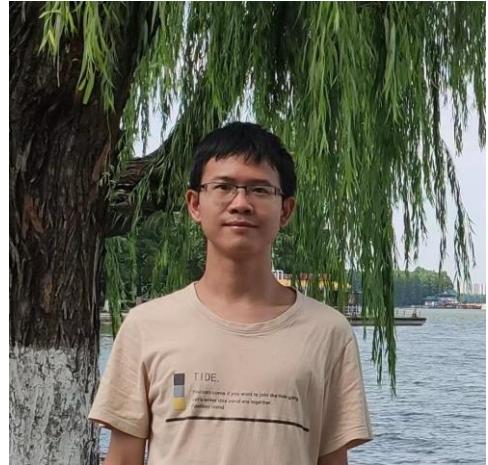
CAPITAL OF STATISTICS  
PROFESSION, HUMANITY & INTEGRITY

# Understanding the Semantics of Data Wrangling Scripts

Kai Xiong

2022.11.22

# Introduce Myself

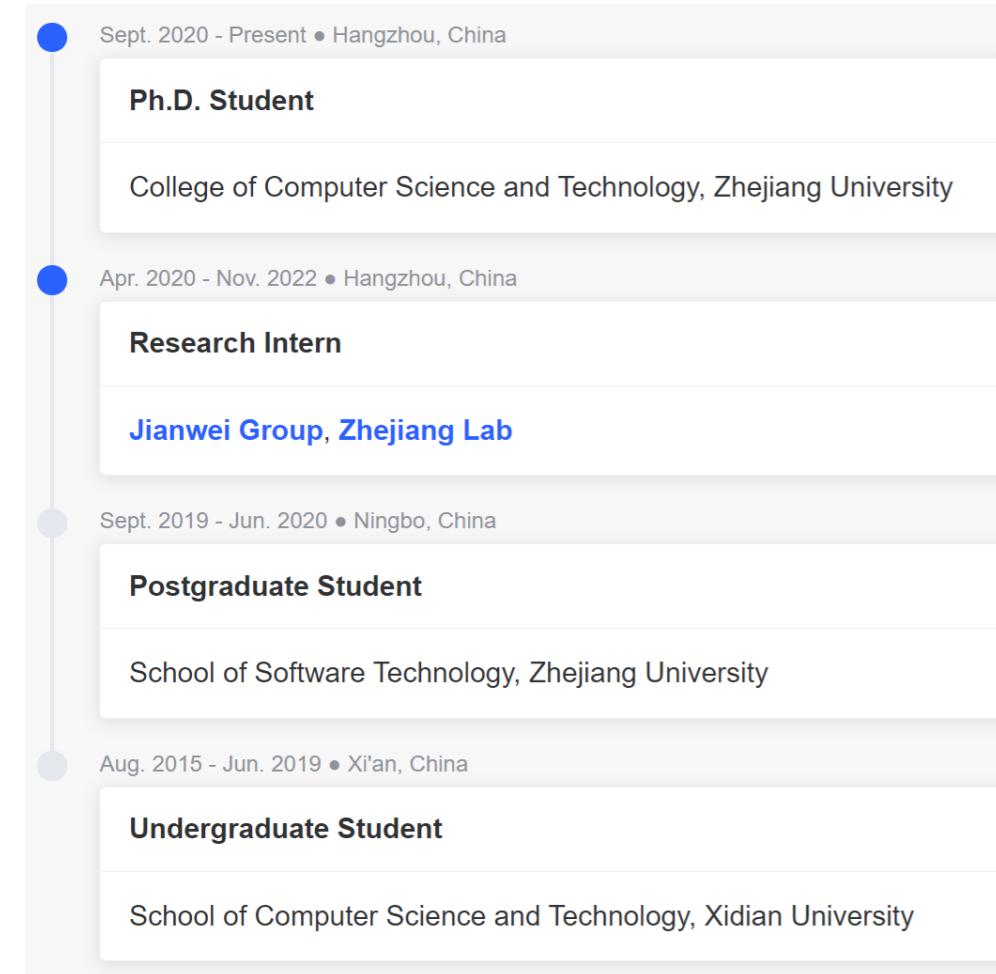


Kai Xiong (熊凯)  
kaixiong@zju.edu.cn

ZJU-IDG  
Zhejiang University

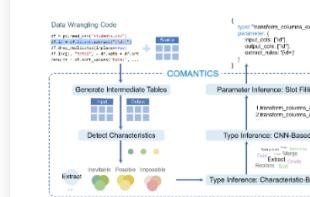


Tutor: Yingcai Wu (巫英才)  
Homepage: xkkevin.github.io



## Research Interests:

- data wrangling and visual analytics
- how to assist data workers in understanding the process of data transformations



### Revealing the Semantics of Data Wrangling Scripts With COMANTICS

Kai Xiong, Zhongsu Luo, Siwei Fu, Yongheng Wang, Mingliang Xu, Yingcai Wu

IEEE Transactions on Visualization and Computer Graphics (IEEE VIS), 2022



### Visualizing the Semantics of Data Wrangling Scripts

Zhongsu Luo, Kai Xiong, Siwei Fu, Yongheng Wang, Hujun Bao, Yingcai Wu

Journal of Computer-Aided Design & Computer Graphics (ChinaVIS), 2022



### Visualizing the Scripts of Data Wrangling with SOMNUS

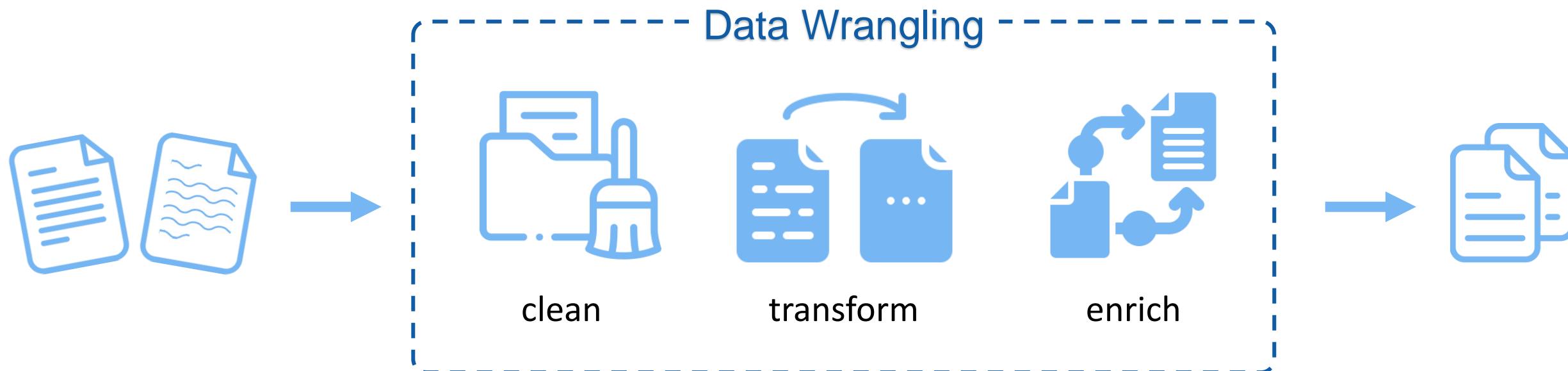
Kai Xiong, Siwei Fu, Guoming Ding, Zhongsu Luo, Rong Yu, Wei Chen, Hujun Bao, Yingcai Wu

IEEE Transactions on Visualization and Computer Graphics, 2022



# Data Wrangling

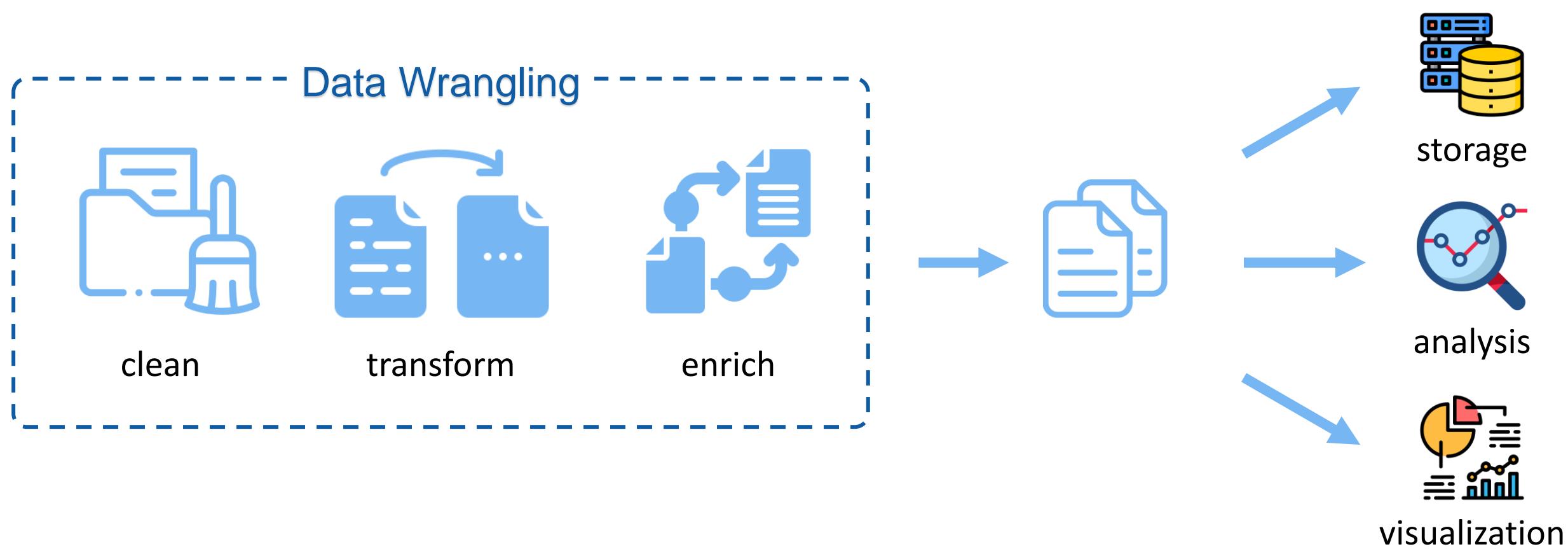
- Data wrangling is an arduous process of cleaning, transforming, and enriching messy and complex data into a desired format





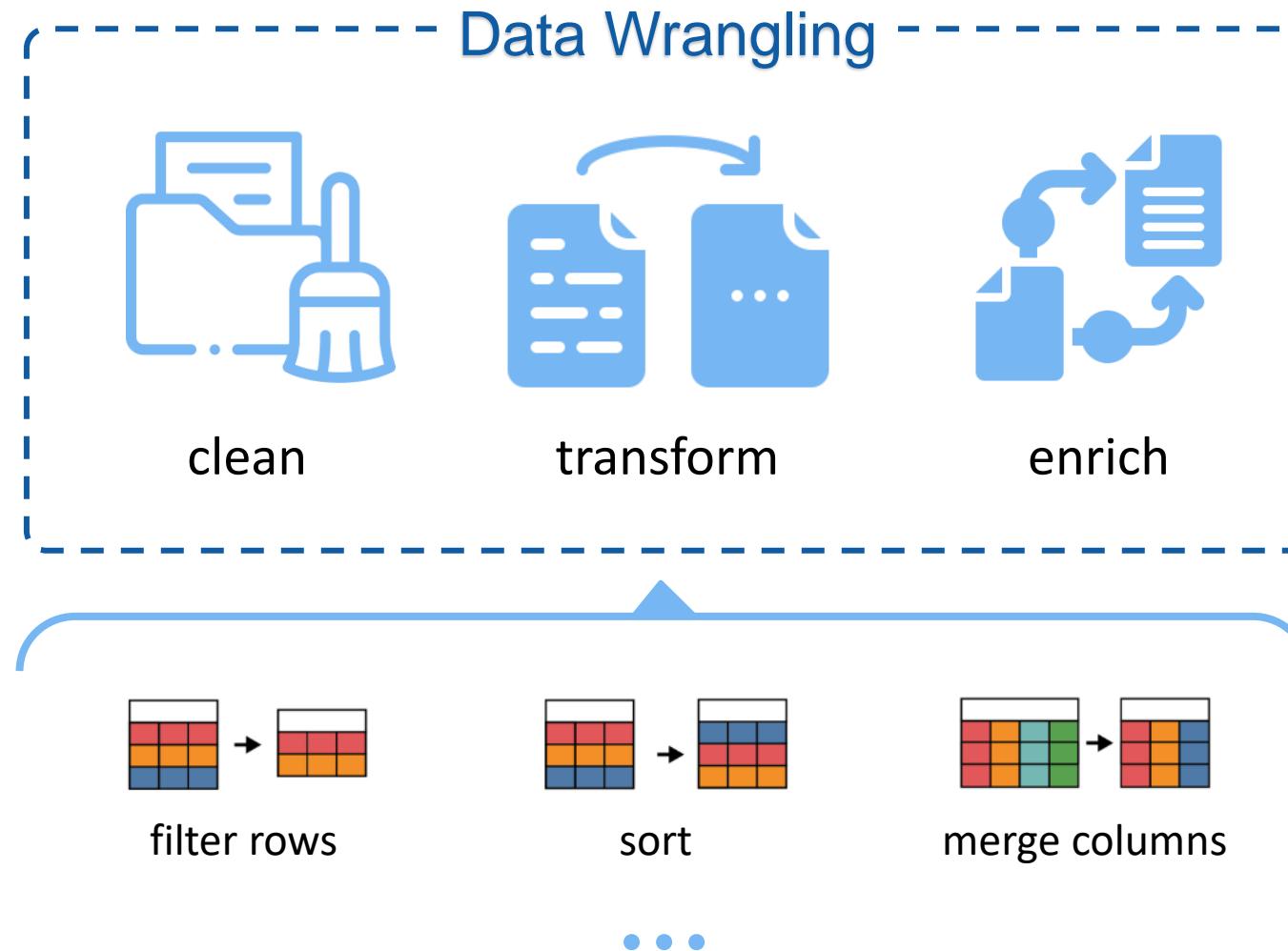
# Data Wrangling

- Data wrangling is an arduous process of cleaning, transforming, and enriching messy and complex data into a desired format





# Data Transformation



Kasica et al. 2021.

A multi-table framework for data wrangling

## Operation categories

1. Create
2. Delete
3. Transform
4. Separate
5. Combine

## Data object types

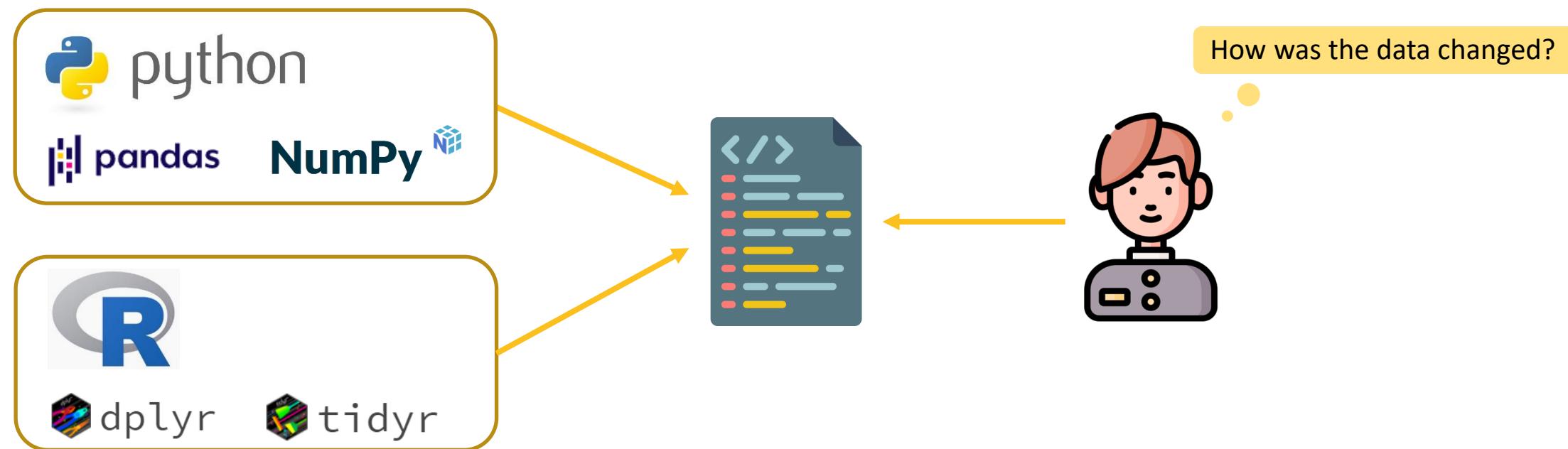
1. Tables
2. Columns
3. Rows

S. Kasica, C. Berret, and T. Munzner. Table Scraps: An actionable framework for multi-table data wrangling from an artifact study of computational journalism. IEEE TVCG, 2021.

15 high-level transformations

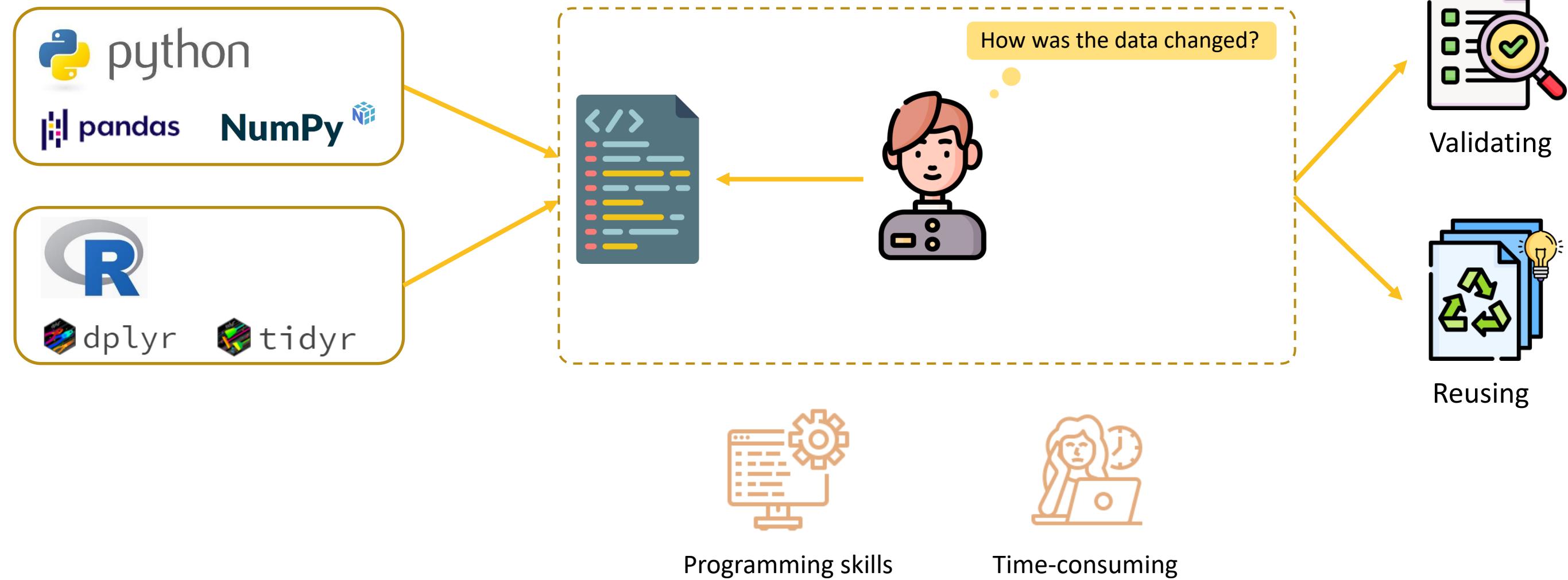


# Understanding the Semantics of Scripts





# Understanding the Semantics of Scripts





INTERACTIVE DATA GROUP

# Visualizing the Scripts of Data Wrangling with SOMNUS



Kai Xiong<sup>1, 2</sup>



Siwei Fu<sup>2</sup>



Guoming Ding<sup>1, 2</sup>



Zhongsu Luo<sup>3, 2</sup>



Rong Yu<sup>2</sup>



Wei Chen<sup>1, 2</sup>



Hujun Bao<sup>1, 2</sup>



Yingcai Wu<sup>1, 2</sup>

<sup>1</sup> State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China

<sup>2</sup> Zhejiang Lab, Hangzhou, China

<sup>3</sup> Zhejiang University of Technology, Hangzhou, China

Somnus

Script Panel

Lang: r Run

Data Panel

```

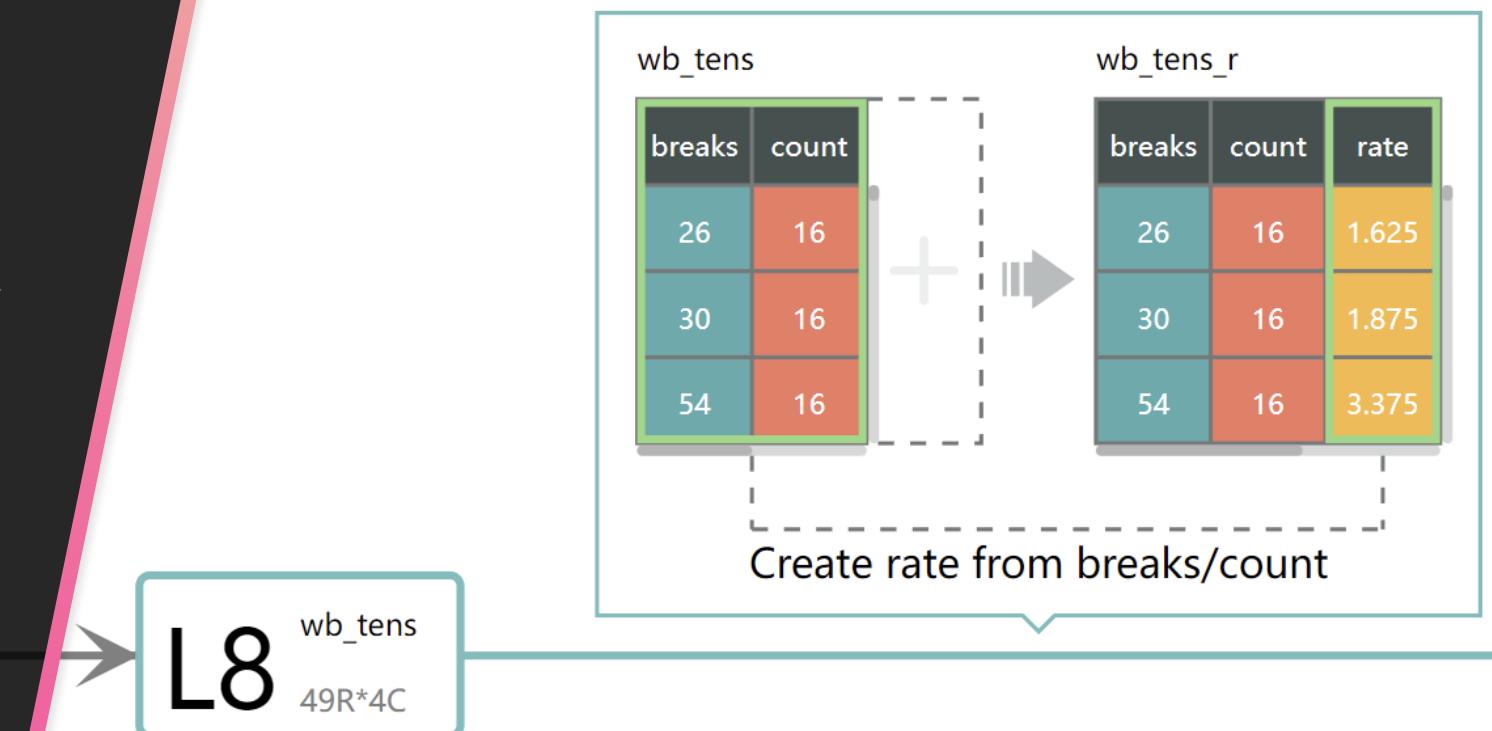
1  y(dplyr)
2
3  wb_tens <- read_csv("warpbreaks.csv")
4  wb_tens <- mutate(wb_tens, tens=tens)
5  s = group_by(wb_tens, tens)
6  s = mutate(wb_tens, count = n())
7  ungroup(wb_tens)
8
9  s_r = mutate(wb_tens, rate=breaks/
10   rbind(warpbreaks, list(70, 'A', 'L'))
11  t = arrange(wb_tens, -breaks)

```

Table Panel

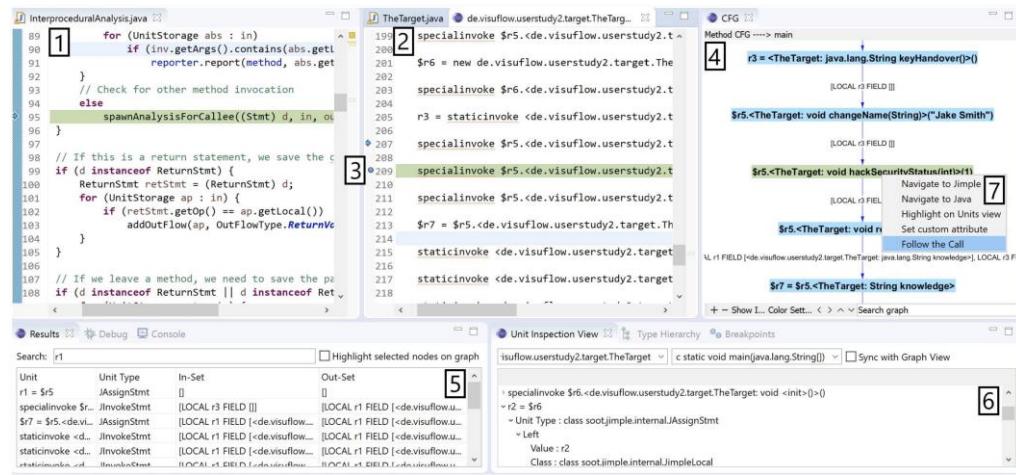
L8 (wb\_tens).csv

breaks	wool	tens	count
26	A	L	16
30	A	L	16
54	A	L	16
25	A	L	16
70	A	L	16
52	A	L	16



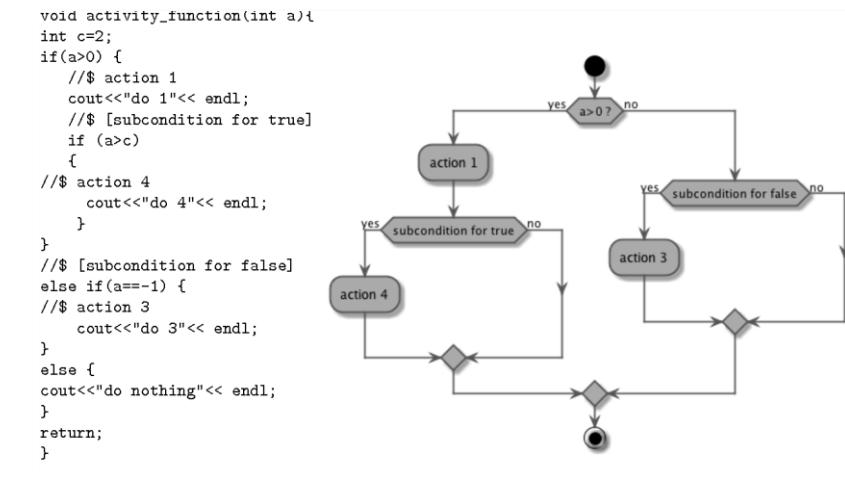
# Program Visualization

Nguyen et al. 2018



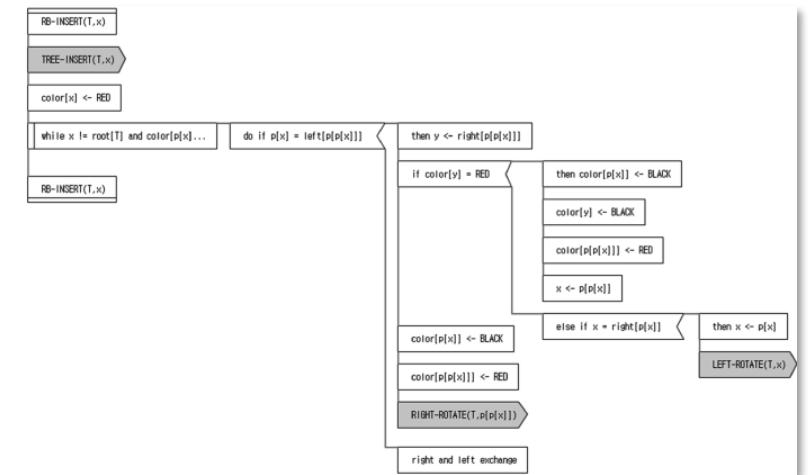
VisuFlow

Kosower et al. 2014



Not in the context of **data transformation**

Cheon et al. 2019



VizMe

# Data Provenance

Khan et al. 2017

Student ID	Last Name	Year of Enrollment	Course	GPA
1	Johnson	2012	CSE	3.47
4	Hayes	2012	CSE	3.82
2	Turner	2012	ECE	3.34
3	Jones	2012	ECE	3.54
5	Lindsay	2012	ECE	2.99
6	Smith	2013	CSE	3.12
10	Frick	2013	CSE	3.53
11	Jones	2013	CSE	3.78
12	Zelaya	2013	CSE	3.94
7	Wallace	2013	ECE	4.00
8	Jabbar	2013	ECE	3.65
9	Borg	2013	ECE	3.32

Data Tweening

Pu et al. 2021

Degree	Work	Salary
PhD	Industry	\$83,000
PhD	Academia	\$97,000
Masters	Industry	\$92,000
PhD	Academia	\$89,000
Masters	Academia	\$83,000
PhD	Industry	\$92,000

Datamations

hard to **explore** and only for **a single table**

# Somnus

## Script Panel

Language: r

Run

## Table Panel

L8 (TBL\_1).csv

### Data Panel

[Source](#)

input1.csv

input2.csv

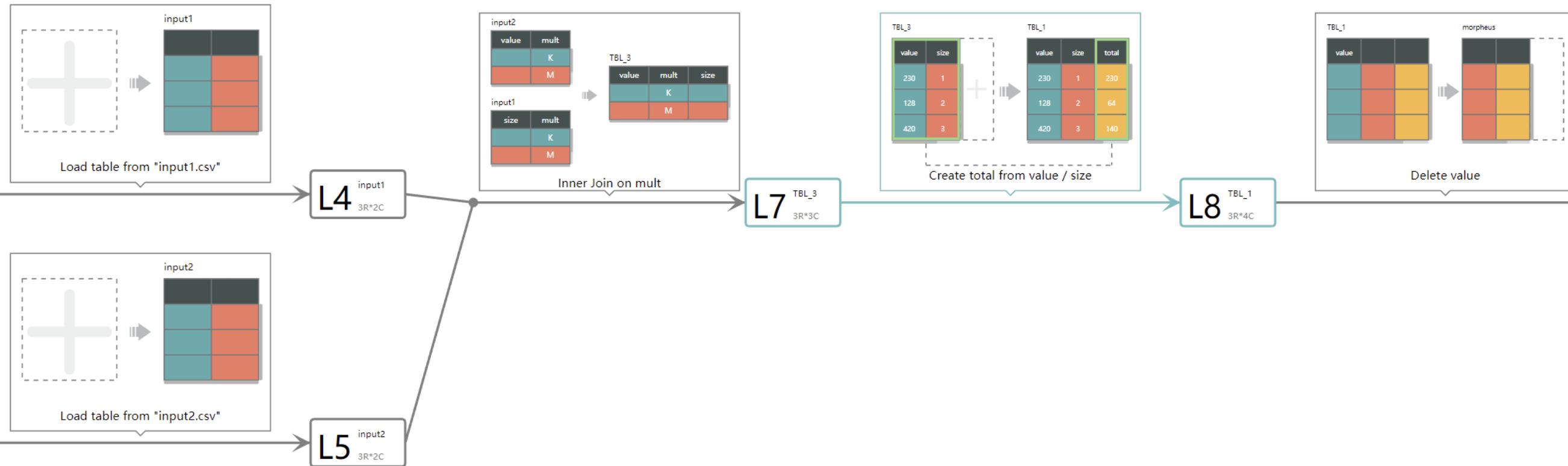
[Upload](#)

Case: r\_case1

```
1 library(tidyr)
2 library(dplyr)
3
4 input1 = read.csv("input1.csv")
5 input2 = read.csv("input2.csv")
6
7 TBL_3=inner_join(input2, input1)
8 TBL_1=mutate(TBL_3,total=value / size)
9 morpheus=select(TBL_1,-`value`)
10 morpheus=as.data.frame(morpheus)
11 morpheus=select(morpheus,2,1,3)
12 morpheus
```

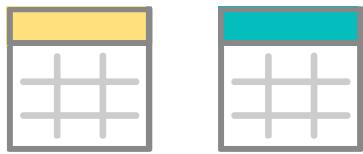
	value	mult	size	total
1	230	K	1	230
2	128	M	2	64
3	420	G	3	140

### Graph Panel



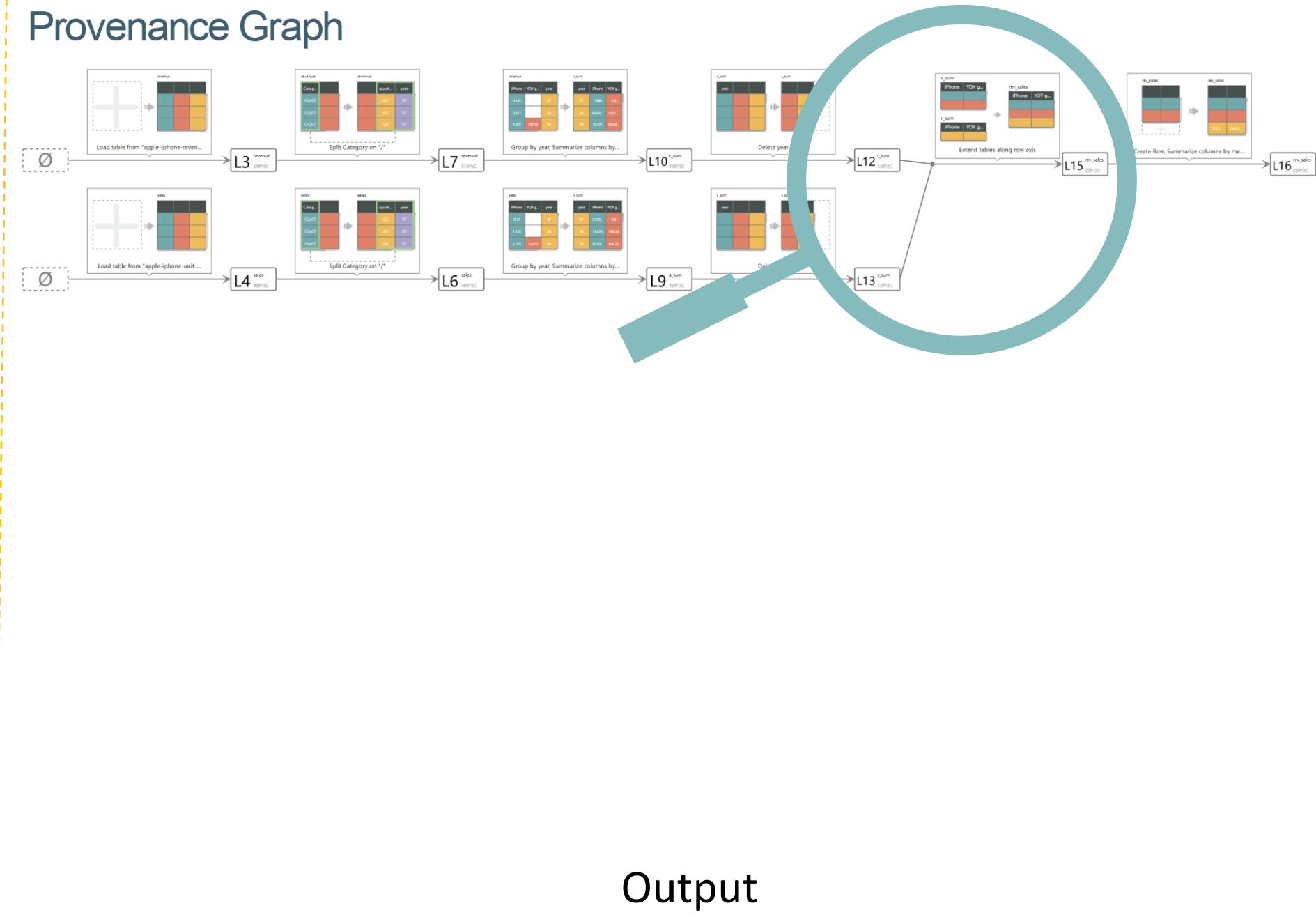
# SOMNUS Overview

## Data Tables



## Data Wrangling Script

```
....  
12 s_sum = s_sum.reset_index(drop=True)  
13 r_sum = r_sum.reset_index(drop=True)  
14  
15 rev_sales = pd.concat([s_sum, r_sum])  
16 rev_sales.loc[len(rev_sales)] = rev_sales.mean()  
....
```



# SOMNUS Overview

## Data Tables

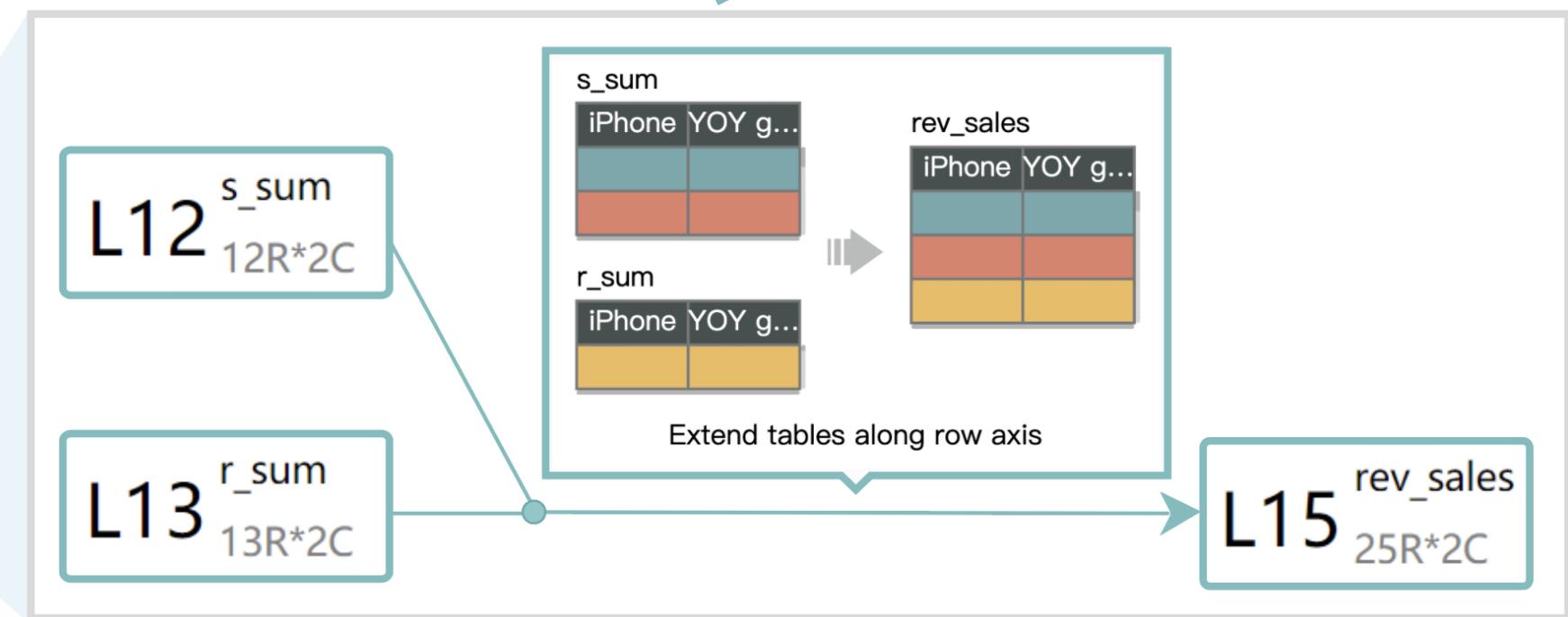
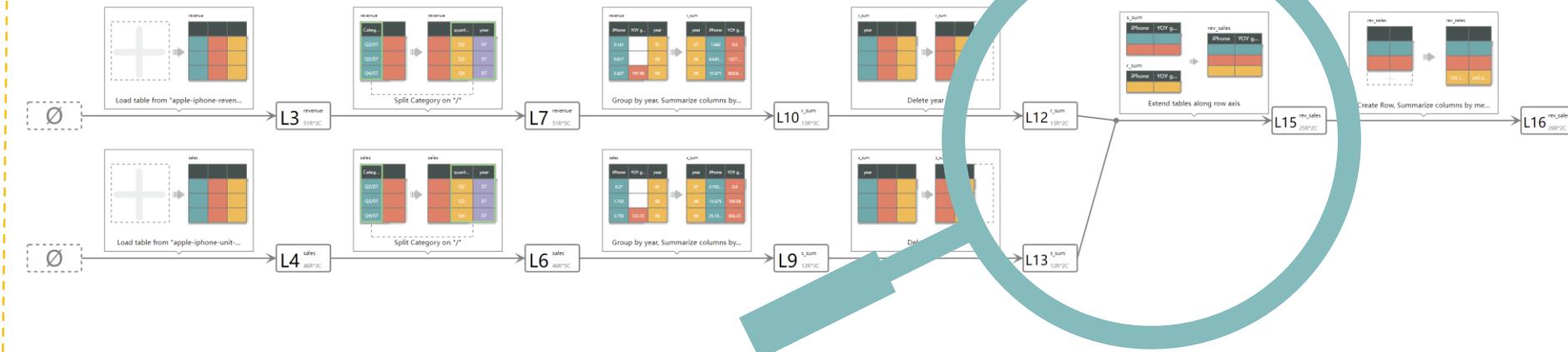


## Data Wrangling Script

```
...  
12 s_sum = s_sum.reset_index(drop=True)  
13 r_sum = r_sum.reset_index(drop=True)  
14  
15 rev_sales = pd.concat([s_sum, r_sum])  
16 rev_sales.loc[len(rev_sales)] = rev_sales.mean()  
....
```

Input

## Provenance Graph



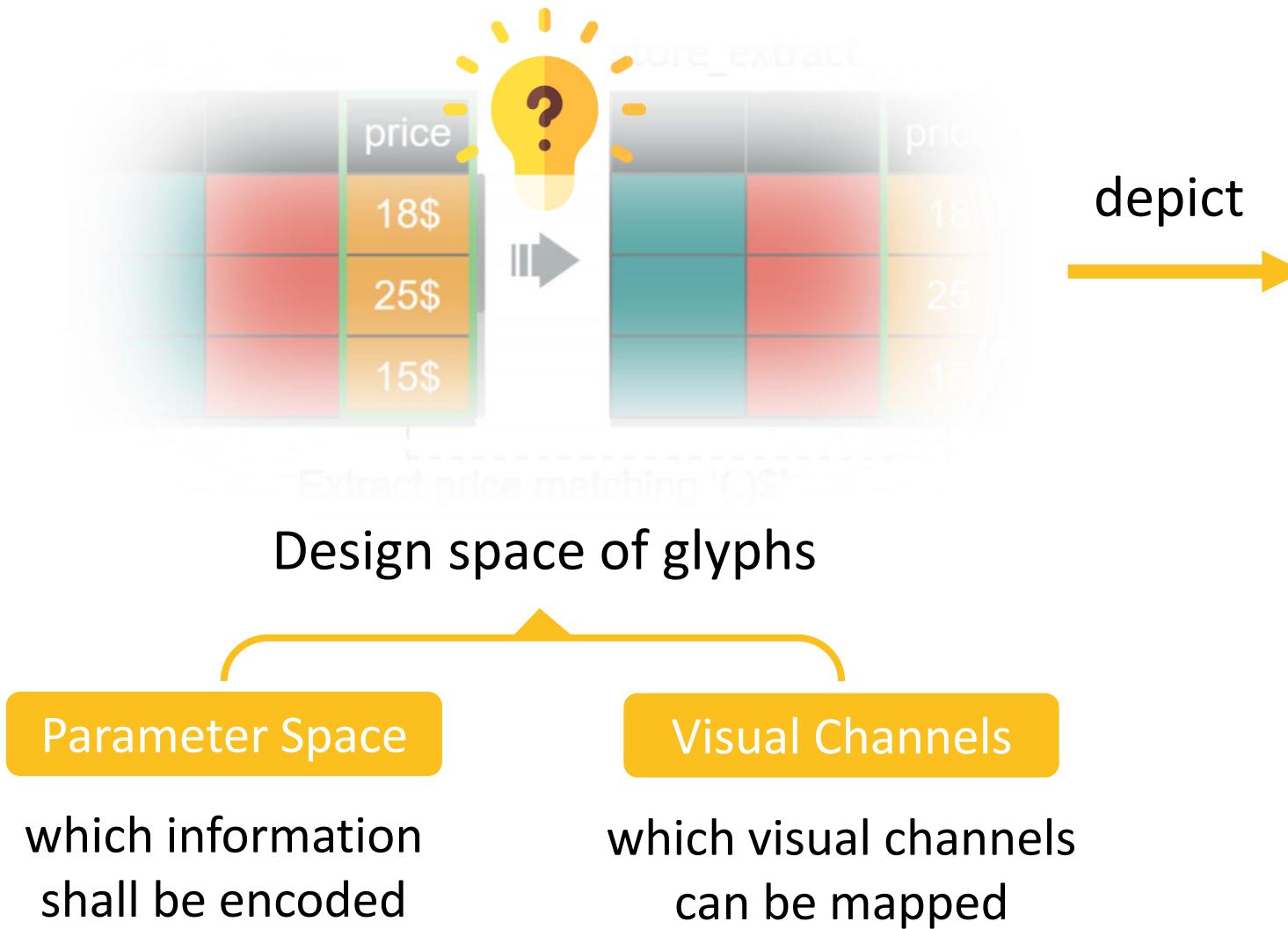
Output

# Design Requirement



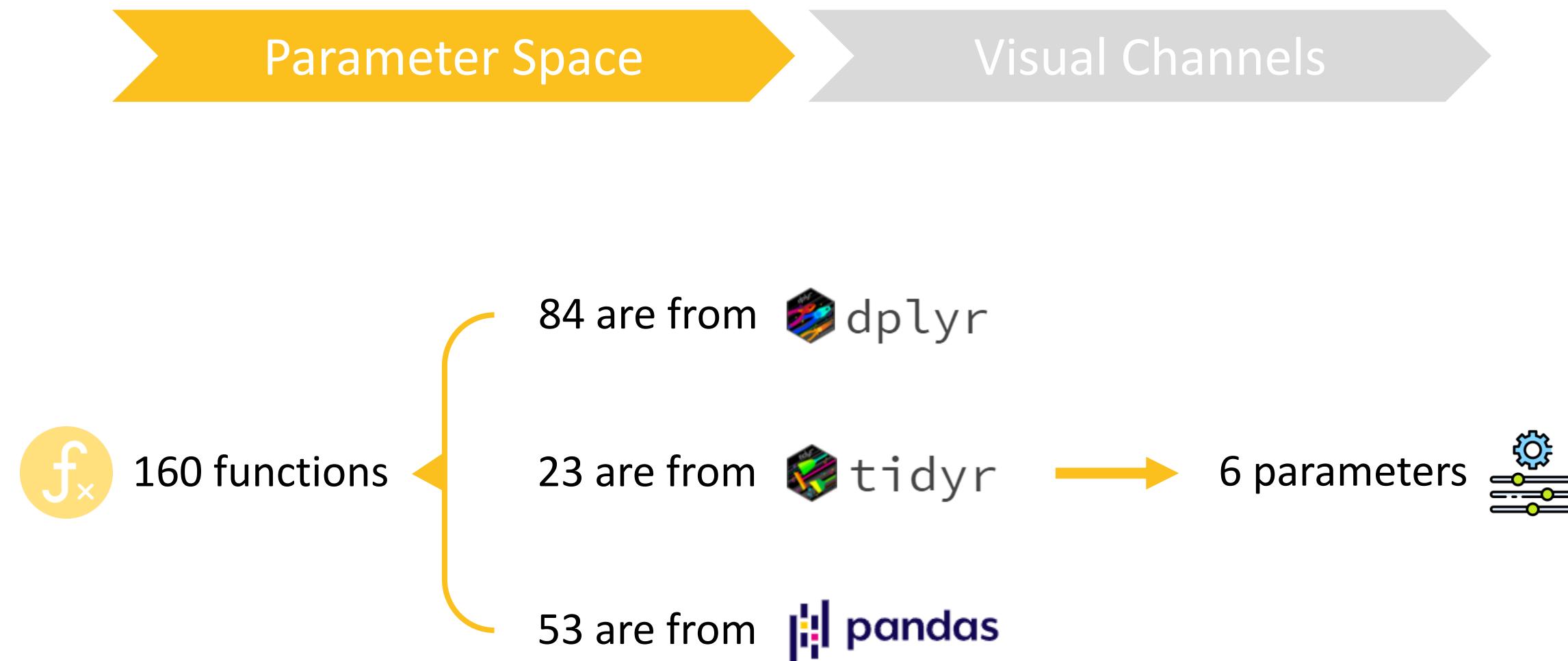
- R1: Present the Semantics
- R2: Link with Data
- R3: Depict Necessary Information
- R4: Keep Encoding Consistent
- R5: Reveal Table Provenance
- R6: Dig into Details
- R7: Independent of Programming Languages

# Design of Glyphs



Op Class	Sets	Expanded Operations
<i>Create</i>	0:1	T/C/R: Create
<i>Delete</i>	1:0	T/C/R: Delete
<i>Transform</i>	1:1	T: Rearrange, Reshape; C/R: Transform
<i>Separate</i>	1:N	T: Subset, Decompose, Split; C/R: Separate
<i>Combine</i>	N:1	T: Extend, Supplement, Match; C: Combine; R: Summarize, Interpolate

# Design of Glyphs



# Design of Glyphs

Parameter Space

Visual Channels

5 Contextual Columns/Rows

col1	col2	price
a	1	18\$
b	1	25\$
c	2	15\$

store

col1	col2	price
a	1	18
b	1	25
c	2	15

store\_extract

1 Function Name

store\_extract = extract(store, price, "price", "(.)\$") # transform price

2 Data Tables

col1	col2	col3
A	1	m
B	2	t
A	1	m

tbl3

4 Transformation Parameters

3 Explicit Columns/Rows

tbl4 = unique(tbl3) # delete duplicate rows

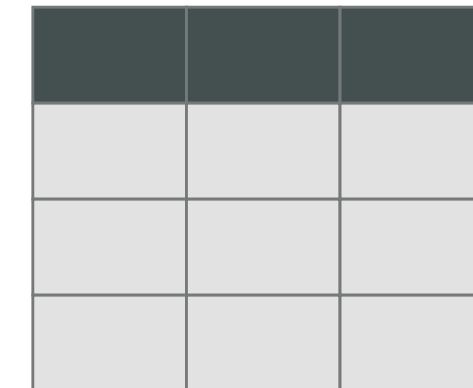
6 Implicit Columns/Rows

# Design of Glyphs

Parameter Space

Visual Channels

```
store_extract = extract(store, price, "price", "(.)$") # transform price
```



# Design of Glyphs



```
store_extract = extract(store, price, "price", "(.)$") # transform price
```

C store

		price
		18\$
		25\$
		15\$

C store\_extract

		price
		18
		25
		15

C Extract price matching '(.)\$'

A Table Metaphor



B In-table Text

price...

C Out-table Text

store...

D Cell Color



E Other Considerations

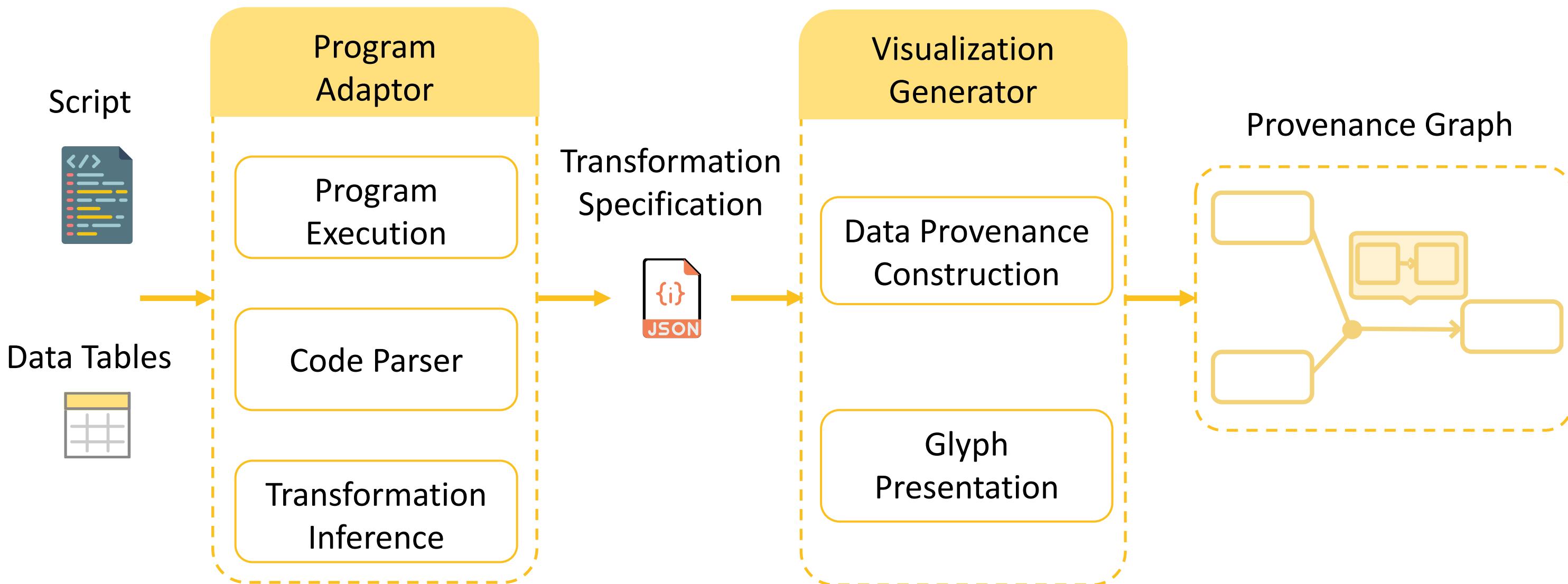


Please refer to our paper for the detailed design rationale.

# Design of Glyphs



# SOMNUS Architecture



# SOMNUS

Program Adaptor

Visualization Generator

Program  
Execution

Code Parser

Transformation  
Inference



Line

1

Input Table

Line 1

2

Line 2

...

Output Table

Line 1

Line 2

# SOMNUS

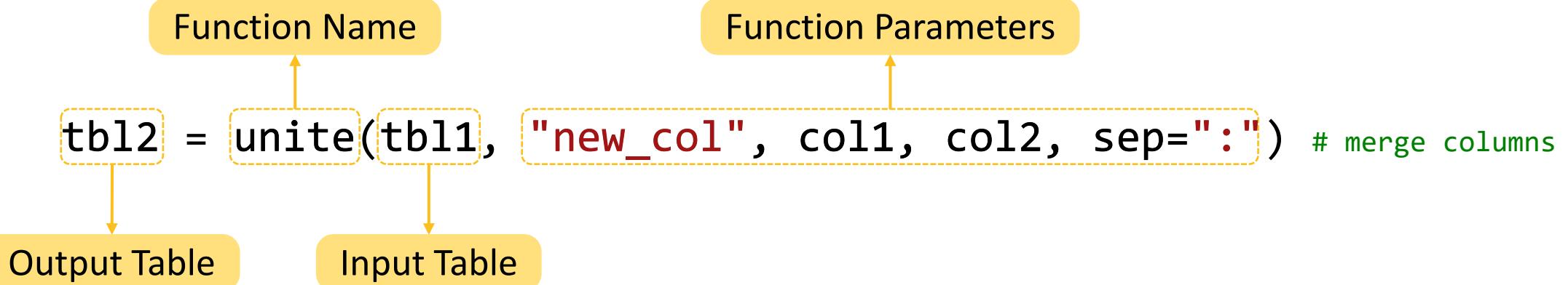
Program Adaptor

Visualization Generator

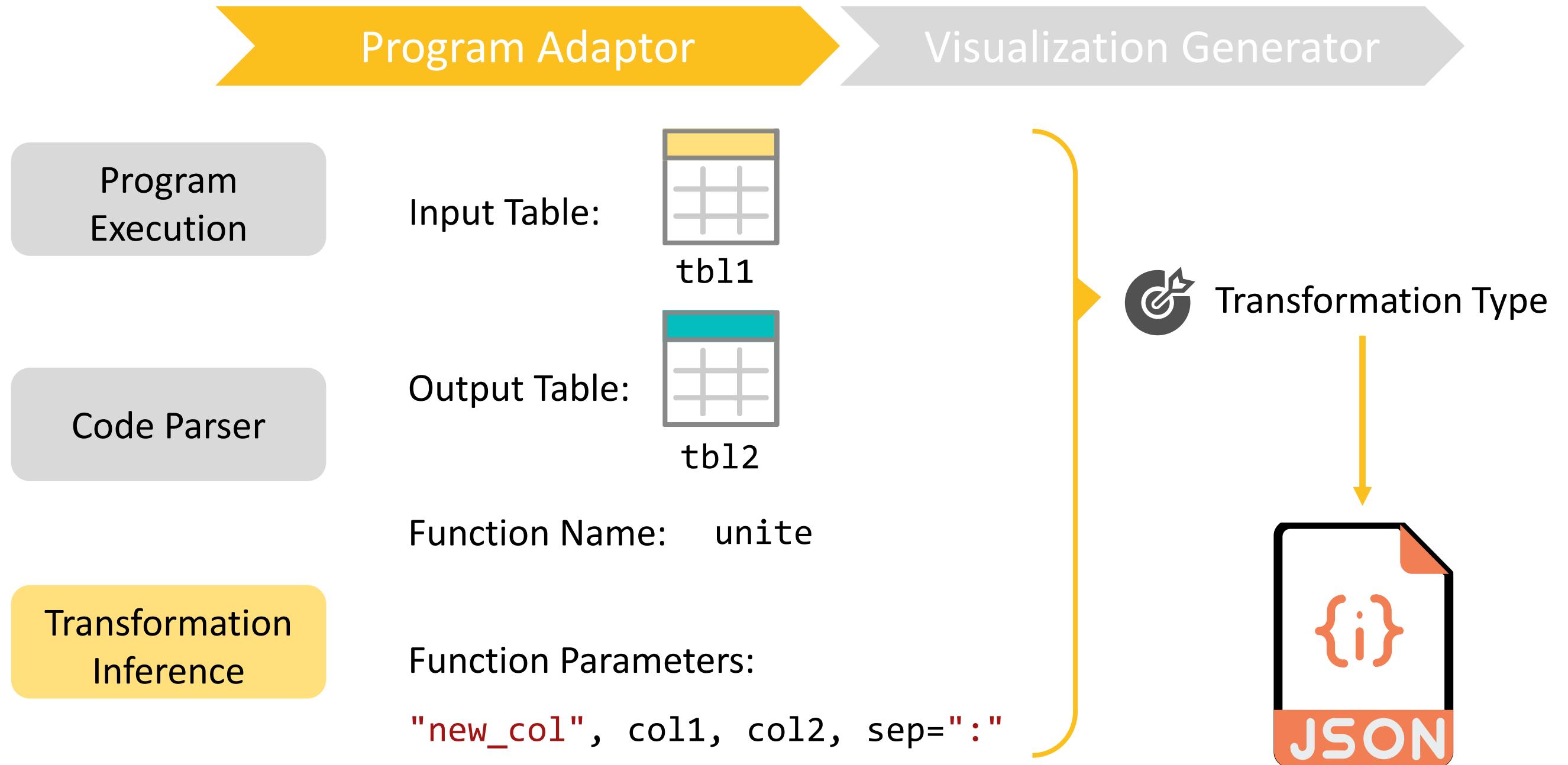
Program  
Execution

Code Parser

Transformation  
Inference



# SOMNUS



# SOMNUS

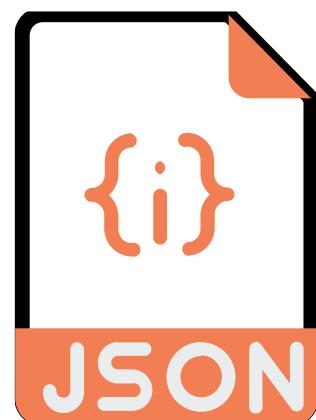
## Program Adaptor

## Visualization Generator

Program  
Execution

Code Parser

Transformation  
Inference



```
transformation specifications:  
▼ (10) [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}] ⓘ  
► 0: {operation_rule: 'Load table from "apple-iphone-revenue.csv"', output_t  
► 1: {operation_rule: 'Load table from "apple-iphone-unit-sales.csv"', output  
► 2: {input_explicit_col: Array(1), input_table_file: 'L4 (sales).csv', input  
► 3: {input_explicit_col: Array(1), input_table_file: 'L3 (revenue).csv', in  
▼ 4:  
► input_explicit_col: (2) ['YOY growth', 'iPhone']  
► input_implicit_col: ['year']  
input_table_file: "L6 (sales).csv"  
input_table_name: "sales"  
operation_rule: "Group by year, Summarize columns by sum"  
► output_explicit_col: (2) ['YOY growth', 'iPhone']  
output_table_file: "L9 (s_sum).csv"  
output_table_name: "s_sum"  
type: "combine_rows_summarize"  
► [[Prototype]]: Object  
► 5: {input_explicit_col: Array(2), input_implicit_col: Array(1), input_table  
► 6: {input_explicit_col: Array(1), input_table_file: 'L9 (s_sum).csv', input  
► 7: {input_explicit_col: Array(1), input_table_file: 'L10 (r_sum).csv', input  
► 8: {input_table_file: Array(2), input_table_name: Array(2), operation_rule  
► 9: {input_table_file: 'L15 (rev_sales).csv', input_table_name: 'rev_sales'  
length: 10
```

# SOMNUS

Program Adaptor

Visualization Generator

Data Provenance  
Construction

Glyph  
Presentation

Data Transformation

Data Tables



# SOMNUS

Program Adaptor

Visualization Generator

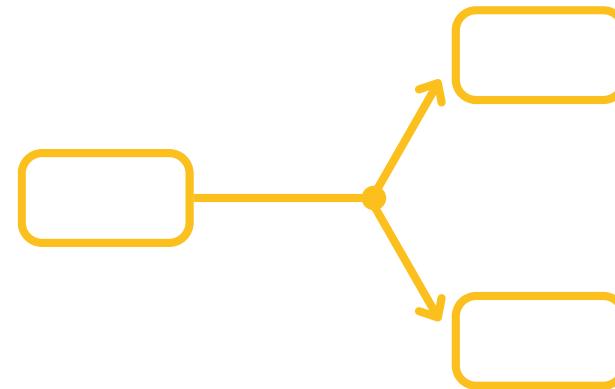
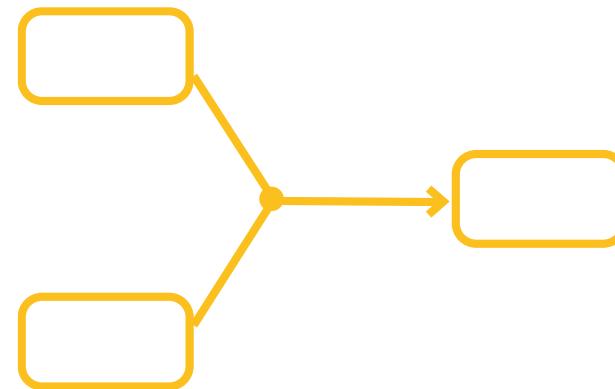
Data Provenance  
Construction

Glyph  
Presentation

Convergence edges:  
(many-to-one)

Divergence edges:  
(one-to-many)

Directed edges:  
(one-to-one, create/delete tables)



# SOMNUS

Program Adaptor

Visualization Generator

Data Provenance  
Construction

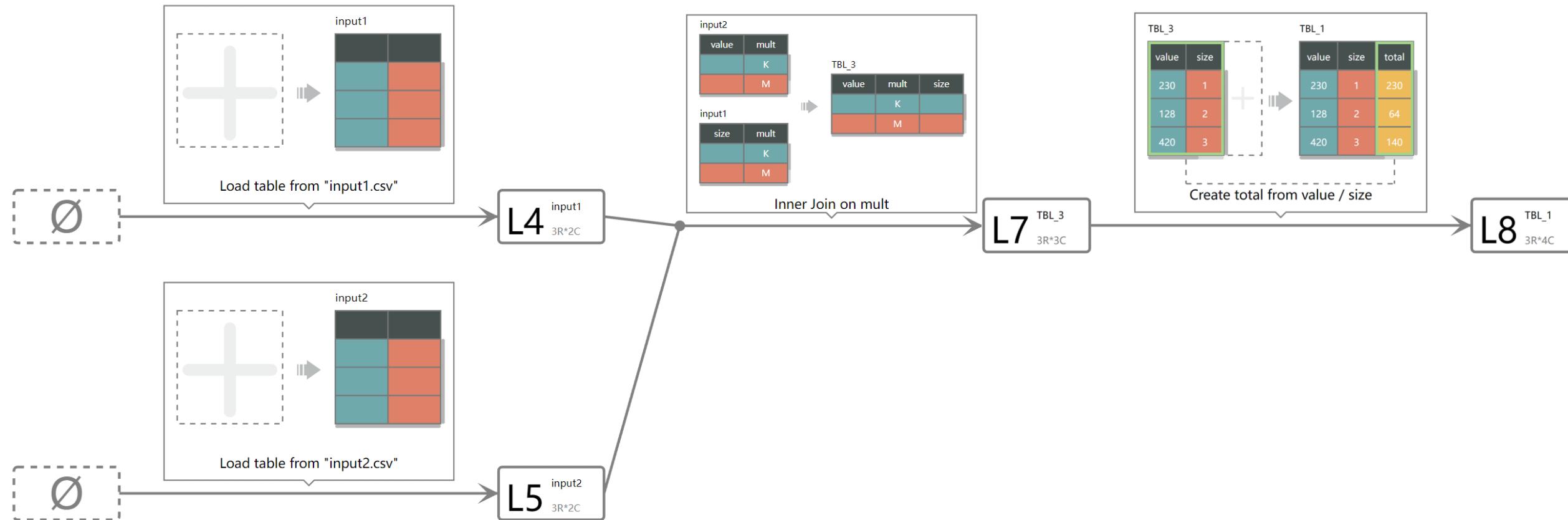
Glyph  
Presentation



# SOMNUS

Program Adaptor

Visualization Generator



# User Study

- Q1: Does the **glyph design** improve user efficiency in comprehending **the semantics of data transformation?**



5 **function** understanding tasks (*Func*)

- Q2: Does the **provenance graph** facilitate the understanding of **data dependencies?**



5 **script** understanding tasks (*Script*)

# User Study

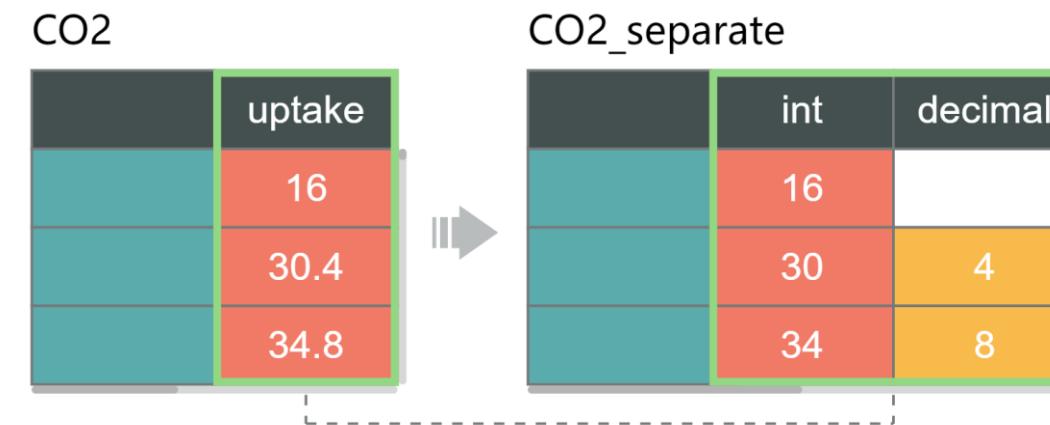
- Techniques:

Code:

```
CO2_separate = separate(CO2, uptake, into=c("int", "decimal"), sep=".")
```

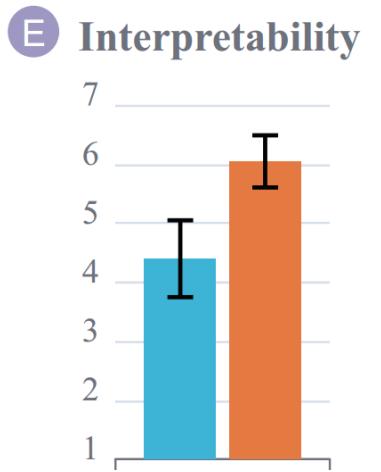
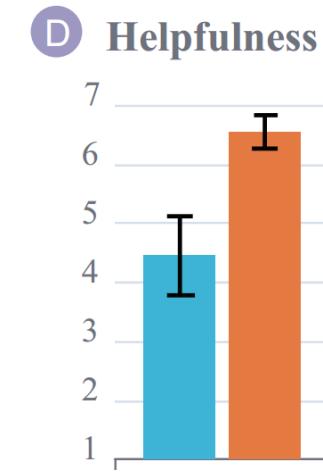
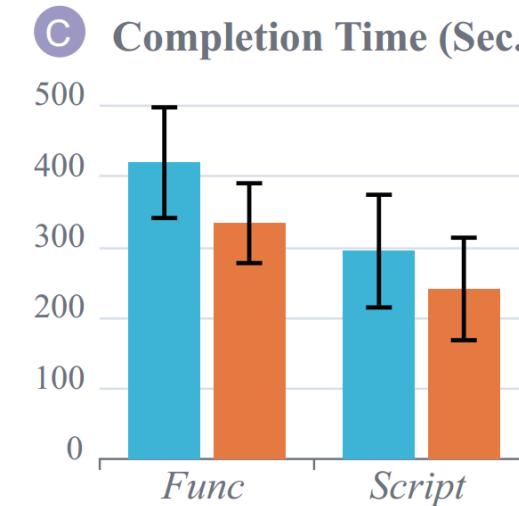
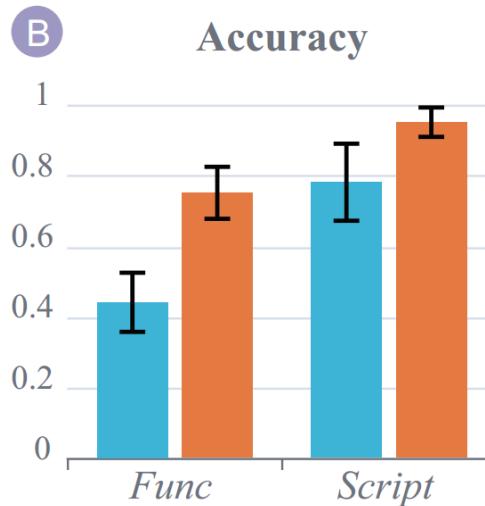
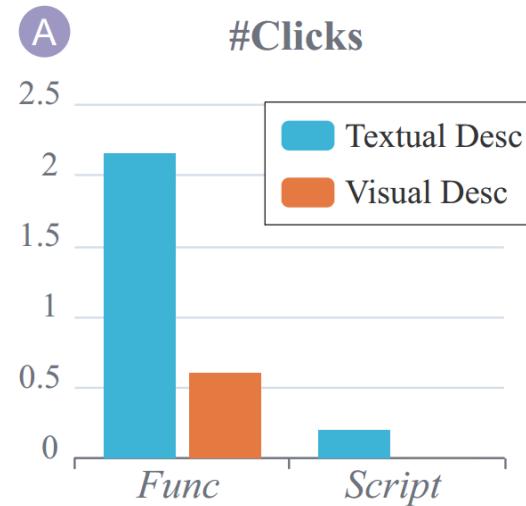
Text: Split uptake on delimiters matching '.' into "int" and "decimal" columns  TRIFACTA

Visualization:



# User Study

- Participants:
  - 20 subjects (4 females, 16 males, age: 22-35)
- Results:



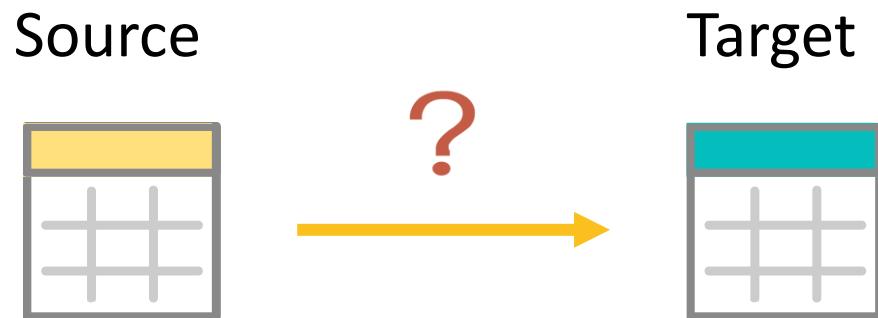
# Example Applications

## 1. Double-checking:

- help data scientists validate the procedure of data transformation

## 2. MORPHEUS Revisited:

- reveal intermediate data transformations given source and target tables



Y. Feng, R. Martins, J. Van Geffen, I. Dillig, and S. Chaudhuri. Component-based synthesis of table consolidation and transformation tasks from examples. ACM SIGPLAN Notices, 2017.

# Somnus

## Script Panel

Language: r

Run

## Table Panel

input1.csv

## Data Panel

Source



Upload

Case:

r\_case1

```
1 library(tidyr)
2 library(dplyr)
3
4 input1 = read.csv("input1.csv")
5 input2 = read.csv("input2.csv")
6
7 TBL_3=inner_join(input2, input1)
8 TBL_1=mutate(TBL_3,total=value / size)
9 morpheus=select(TBL_1,-`value`)
10 morpheus=as.data.frame(morpheus)
11 morpheus=select(morpheus,2,1,3)
12 morpheus
```

size	mult
1	K
2	M
3	G

## Graph Panel



# Conclusion

## Contributions

- a design space consisting of two dimensions that guide the design of a collection of 23 glyphs.
- a pipeline, SOMNUS, that visualizes the creation and evolution of data tables across a series of transformations.
- a controlled study that evaluates how users perform with visualization and text using comparison tasks.
- two example applications that showcase how SOMNUS can benefit different usage scenarios.

## Future Work

- enhance SOMNUS by supporting a large number of functions and parameters.
- explore how to visualize conditional statements and loops in the provenance graph.

# Data Wrangling Code



INTERACTIVE DATA GROUP

```
df = pd.read_csv("students.csv")
df.id = df.id.str.extract("(\\d+)")
df.loc[:, "total"] = df.math + df.art
```

Source

# Revealing the Semantics of Data Wrangling Scripts With COMANTICS

COMANTICS



Kai Xiong<sup>1, 2</sup>



Zhongsu Luo<sup>3, 2</sup>



Siwei Fu<sup>2</sup>



Yongheng Wang<sup>2</sup>



Mingliang Xu<sup>4</sup>



Yingcai Wu<sup>1, 2</sup>

<sup>1</sup>State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China

<sup>2</sup>Zhejiang Lab, Hangzhou, China

<sup>3</sup>Zhejiang University of Technology, Hangzhou, China

<sup>4</sup>School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China

VIS 2022

{

```
    type: "transform_columns_extract",
    parameter: {
        input_cols: ["id"],
        output_cols: ["id"],
        extract_rules: '(\d+)'
    }
}
```

Parameter Inference: Slot Filling

1. transform\_columns\_extract
2. transform\_columns\_mutate
- ...

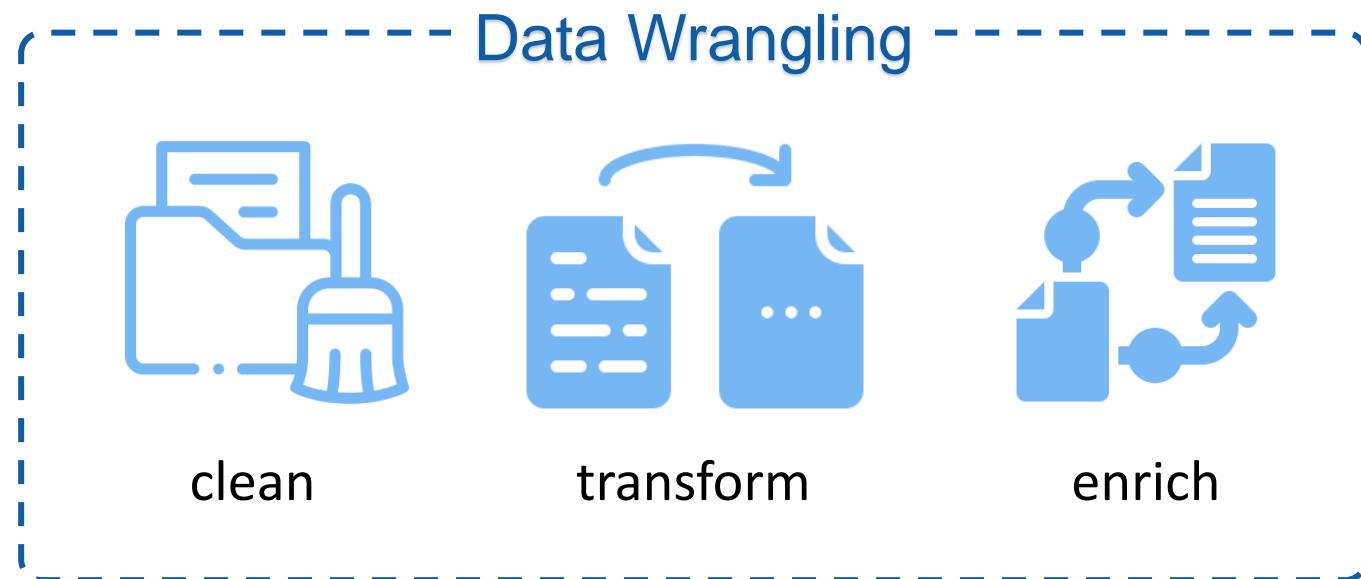
Type Inference: CNN-Based

Deduplicate Merge Summarize  
Fold Mutate Merge  
Extract Create  
Replace Sort

Type Inference: Characteristic-Based



# Semantics of Scripts



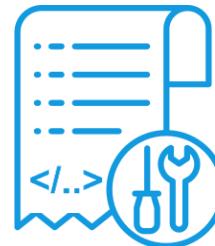
**Semantics**  
How was the data changed?



debugging



reusing

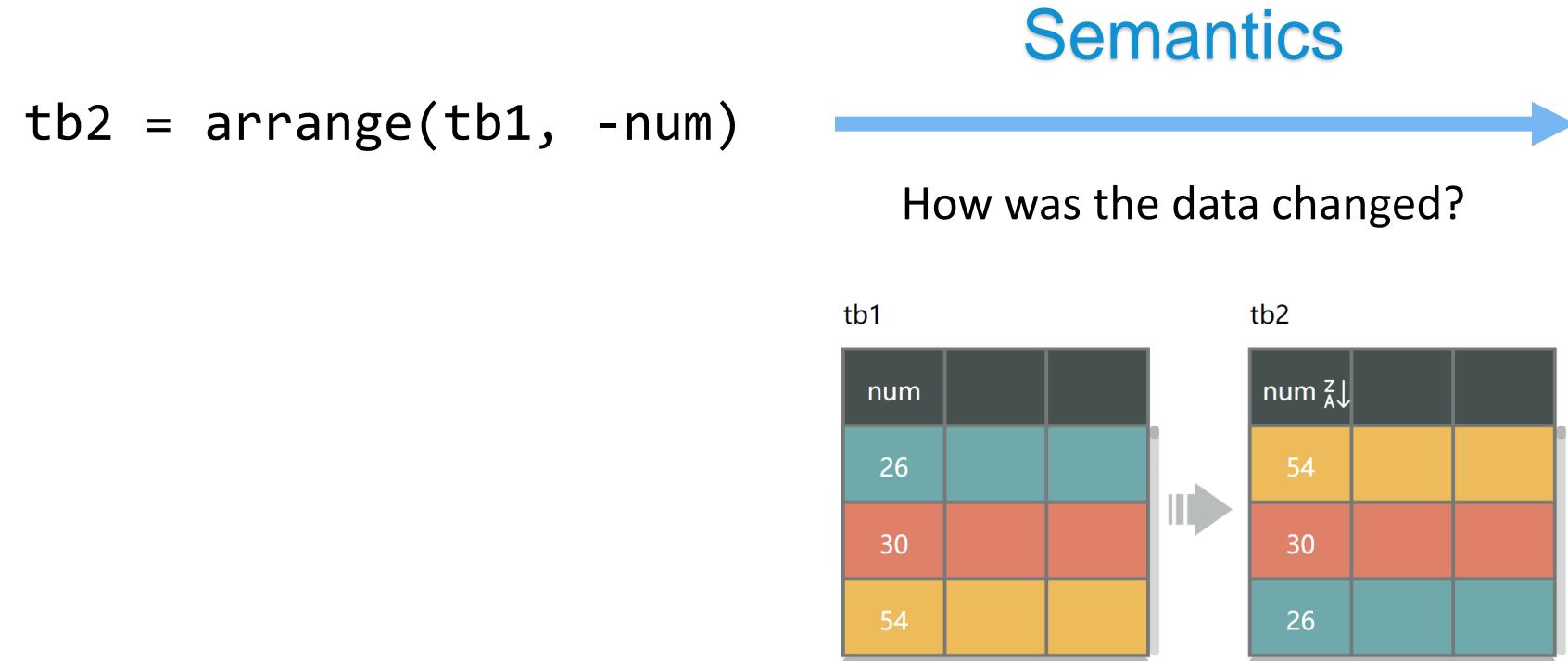


maintaining



# Semantics of Scripts

- The **semantics** of wrangling code is the **type of data transformation** and its **parameters**.



Transformation type: sort

Parameters: {  
Column: num  
Input Table: tb1  
Order: descending  
Output Table: tb2  
}



# Program Understanding

Shrestha et al. 2021

The screenshot shows the Unravel interface. At the top (A), there is an R code editor with the following code:

```
library(tidyverse)
library(babynames)

# calculate the percent of male babynames and the ratio of males to females
babynames %>%
  group_by(year, sex) %>%
  summarise(total = sum(n)) %>%
  pivot_wider(names_from = sex, values_from = total) %>%
  mutate(percent_male = round(M / (M + F) * 100, 2), ratio = M / F) %>%
  unravel()
```

Below the code editor (B) is the "Viewer" pane, which displays the execution history of the code steps:

- Step 1: `babynames %>%` (1.9M rows)
- Step 2: `group_by(year, sex) %>%` (1.9M rows)
- Step 3: `summarise(total = sum(n)) %>%` (276 rows)
- Step 4: `pivot_wider(names_from = sex, values_from = total) %>%` (138 rows)
- Step 5: `mutate(percent_male = round(M / (M + F) * 100, 2), ratio = M / F)` (138 rows)

At the bottom (C) is a data preview table showing columns: year, F, M, percent\_male, and ratio. A cursor (F) is hovering over the "percent\_male" column.

Unravel

Yang et al. 2021

The screenshot shows the WrangleDoc interface. At the top (1) is a "SUMMARY" section with "INPUTS" and "data" and "OUTPUTS" and "data".

Below it (2) is a "data -> data" section with the following text:  
changed columns: [Size]  
Size = float(str\_transform(Size))

Table (3) showing data transformations:

4	5	App	Category	Rating	Reviews
type	object→float64	object	object	float64	object
unique	413→413	8190	33	39	5990
range	→[8.5, 10000000.0]			[1.0, 5.0]	
7460	19M→19000000.0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND DESIGN	4.1	159
1637	Varies with device→22948351.235594977	Floor Plan Creator	ART_AND DESIGN	4.1	36639
263	201k→201.0	Restart Navigator	AUTO_AND VEHICLES	4	1403
5	23k→23.0	Plugin:AOT v5.0	BUSINESS	3.1	4034

WrangleDoc

Cannot support inferring **data transformations**



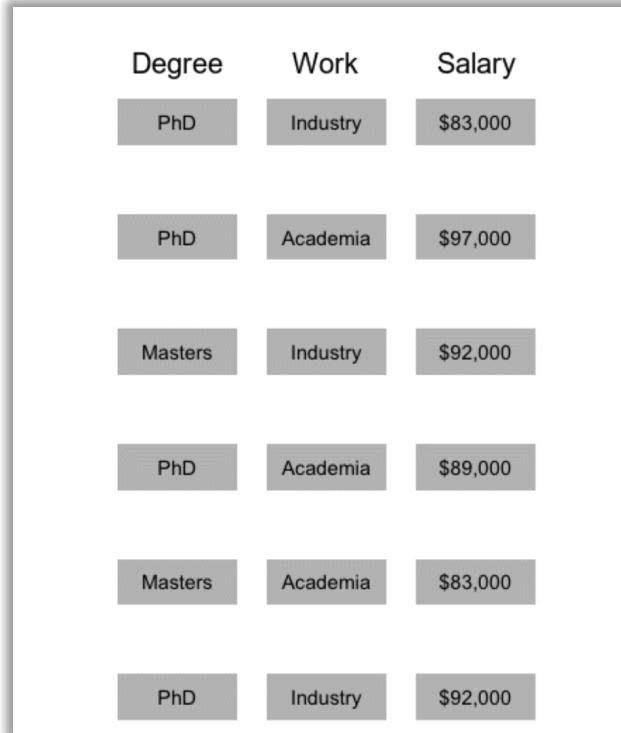
# Program Understanding

Khan et al. 2017

Student ID	Last Name	Year of Enrollment	Course	GPA
1	Johnson	2012	CSE	3.47
4	Hayes	2012	CSE	3.82
2	Turner	2012	ECE	3.34
3	Jones	2012	ECE	3.54
5	Lindsay	2012	ECE	2.99
6	Smith	2013	CSE	3.12
10	Frick	2013	CSE	3.53
11	Jones	2013	CSE	3.78
12	Zelaya	2013	CSE	3.94
7	Wallace	2013	ECE	4.00
8	Jabbar	2013	ECE	3.65
9	Borg	2013	ECE	3.32

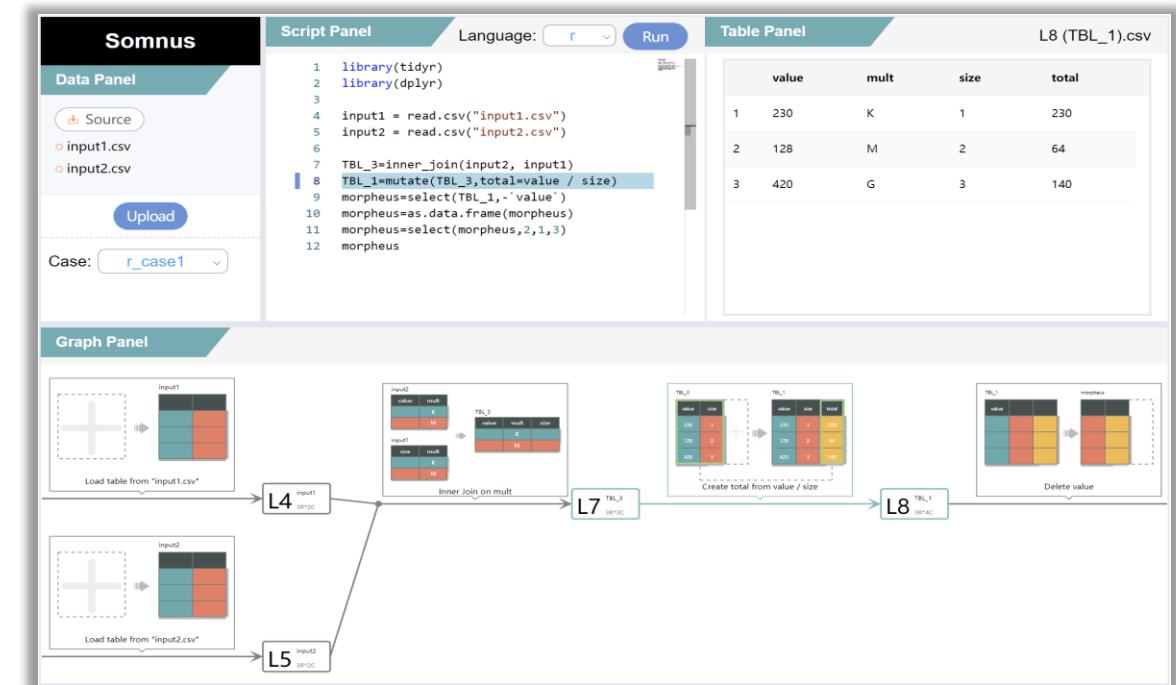
Data Tweening

Pu et al. 2021



Datamations

Xiong et al. 2022



SOMNUS

Rely on hand-crafted rules, limited in **generalizability and scalability**



# COMANTICS

## Data Wrangling Script

```
1 import pandas as pd  
2  
3 df = pd.read_csv("students.csv")  
4  
5 df.id = df.id.str.extract("(\\d+)")  
6 df.drop_duplicates(inplace=True)  
7 df.loc[:, "total"] = df.math + df.art  
8 results = df.sort_values("total", ascending = False)  
....
```



## Semantics

```
{  
    type: "transform_columns_extract",  
    parameter: {  
        input_cols: ["id"],  
        output_cols: ["id"],  
        extract_rules: '(\\d+)'  
    }  
}
```



# Challenges

Scripting Language Level

Function Level

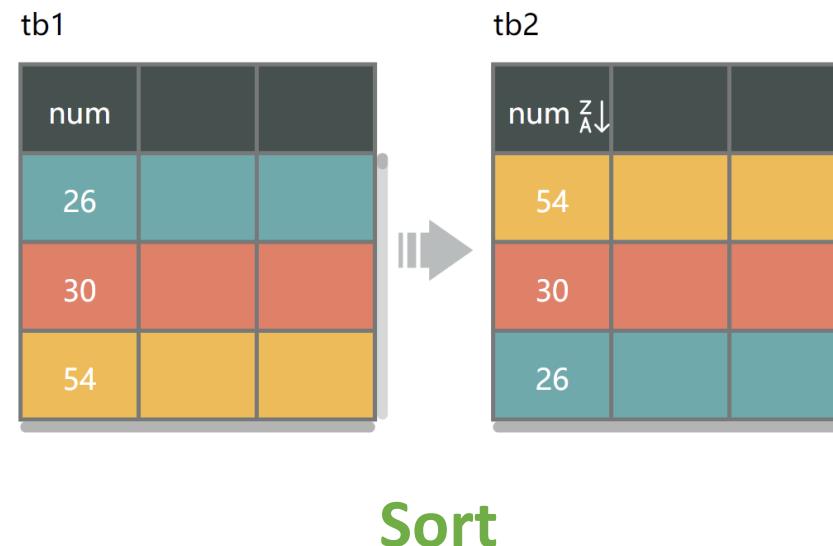
Parameter Level



# Challenges

## Scripting Language Level

- Different scripting languages have diverse regulations in implementing data transformations



R dplyr

```
tb2 = arrange(tb1, -num)
tb2 = arrange(tb1, desc(num))
```

python pandas

```
tb2 = tb1.sort_values('num', ascending=False)
tb2 = tb1.sort_values(by=['num'], ascending=False)
```



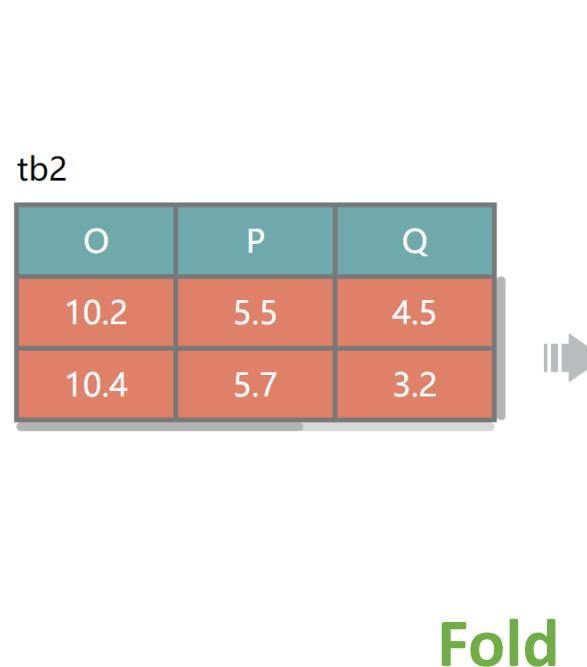
# Challenges

Scripting Language Level

Function Level

Parameter Level

- There are different functions to implement the same data transformation



`tb2 = gather(tb1, K, V, -`ID`, -`T`)`

`tb2 = pivot_longer(tb1, 0:Q, names_to='K', values_to='V')`



# Challenges

Scripting Language Level

Function Level

Parameter Level

- Functions may omit important parameters for simplicity

store	revenue		
Category	Category	Quarter	Year
Q2/07	Q2/07	Q2	07
Q3/08	Q3/07	Q3	08
Q4/09	Q4/07	Q4	09

Split Category on '/'

Split



revenue = separate(store, Category, c("Quarter", "Year"), sep="/")



# Challenges

Scripting Language Level

Function Level

Parameter Level

- Different scripting languages have diverse regulations in implementing data transformations
- There are different functions to implement the same data transformation
- Functions may omit important parameters for simplicity



# Our Solution

Qualitative research



A space of table changes

Outputs:	Inputs:	Table	Column	Row	Cell
1. $\text{blue} > 0 \& \text{orange} > 0$	6. $\text{blue} = \text{orange}$	23. $\text{blue} > \text{orange}$			
2. $\text{blue} = 0 \& \text{orange} > 0$	7. $\text{blue} < \text{orange}$	24. $\text{blue} < \text{orange}$			
3. $\text{blue} > 0 \& \text{blue} < \text{orange}$	8. $\text{blue} > \text{orange}$	25. $\text{blue} > \text{orange}$			
4. $\text{blue} > 0 \& \text{blue} < \text{blue}$	9. $\min(\text{blue}) > \text{orange}$	26. $\min(\text{blue}) > \text{orange}$			
5. $\text{blue} > 0 \& \text{blue} < \text{orange}$	10. $\min(\text{blue}) = \text{orange}$	27. $\min(\text{blue}) = \text{orange}$			
	11. $\min(\text{blue}) < \text{blue} \& \max(\text{blue}) > \text{orange}$	28. $\min(\text{blue}) < \text{blue} \& \max(\text{blue}) > \text{orange}$			
	12. $\max(\text{blue}) = \text{orange}$	29. $\max(\text{blue}) = \text{orange}$			
	13. $\max(\text{blue}) < \text{blue} \& \sum(\text{blue}) > \text{orange}$	30. $\max(\text{blue}) < \text{blue} \& \sum(\text{blue}) > \text{orange}$			
	14. $\sum(\text{blue}) = \text{orange}$	31. $\sum(\text{blue}) = \text{orange}$			
	15. $\sum(\text{blue}) < \text{blue}$	32. $\sum(\text{blue}) < \text{blue}$			
	16. $\sum(\text{blue}) < \min(\text{orange})$	33. $\sum(\text{blue}) < \min(\text{orange})$			
	17. $\sum(\text{blue}) = \min(\text{orange})$	34. $\sum(\text{blue}) = \min(\text{orange})$			
	18. $\sum(\text{blue}) < \min(\text{orange}) \& \sum(\text{blue}) < \max(\text{orange})$	35. $\sum(\text{blue}) > \min(\text{orange}) \& \sum(\text{blue}) < \max(\text{orange})$			
	19. $\sum(\text{blue}) = \max(\text{orange})$	36. $\sum(\text{blue}) = \max(\text{orange})$			
	20. $\sum(\text{blue}) > \max(\text{orange}) \& \sum(\text{blue}) < \sum(\text{orange})$	37. $\sum(\text{blue}) > \max(\text{orange}) \& \sum(\text{blue}) < \sum(\text{orange})$			
	21. $\text{order}(\text{blue}) = \text{ascend}$	38. $\text{order}(\text{blue}) = \text{ascend}$			
	22. $\text{order}(\text{blue}) = \text{sum}$	39. $\text{order}(\text{blue}) = \text{sum}$			
	40. $\text{index}(\text{blue}) = \text{index}(\text{orange})$	45. $\text{index}(\text{blue}) = \text{index}(\text{orange})$			
	41. $\text{index}(\text{blue}) \neq \text{index}(\text{orange})$	46. $\text{index}(\text{blue}) \neq \text{index}(\text{orange})$			
	42. $\text{order}(\text{blue}) = \text{disorder}$	47. $\text{order}(\text{blue}) = \text{disorder}$			
	43. $\text{order}(\text{blue}) = \text{ascend}$	48. $\text{order}(\text{blue}) = \text{ascend}$			
	44. $\text{order}(\text{blue}) = \text{descend}$	49. $\text{order}(\text{blue}) = \text{descend}$			

Infer transformation candidates

Gender	Age
Male	23
Female	
Male	31

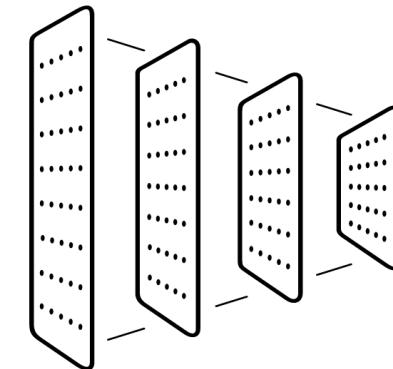
Gender	Age
Male	23
Male	31

- ?
- 1 Keep Rows Where Gender is Male
  - 2 Remove Rows with Missing Values
- ...

Machine learning



A CNN model



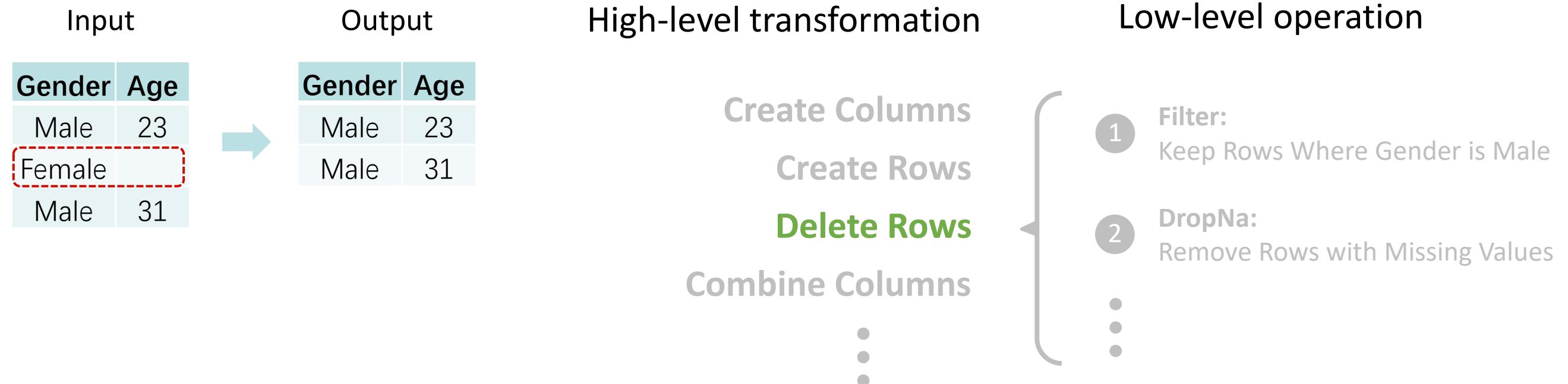
Resolve ambiguities

- 1 Keep Rows Where Gender is Male X
  - 2 Remove Rows with Missing Values ✓
- ...
- X



# Observation

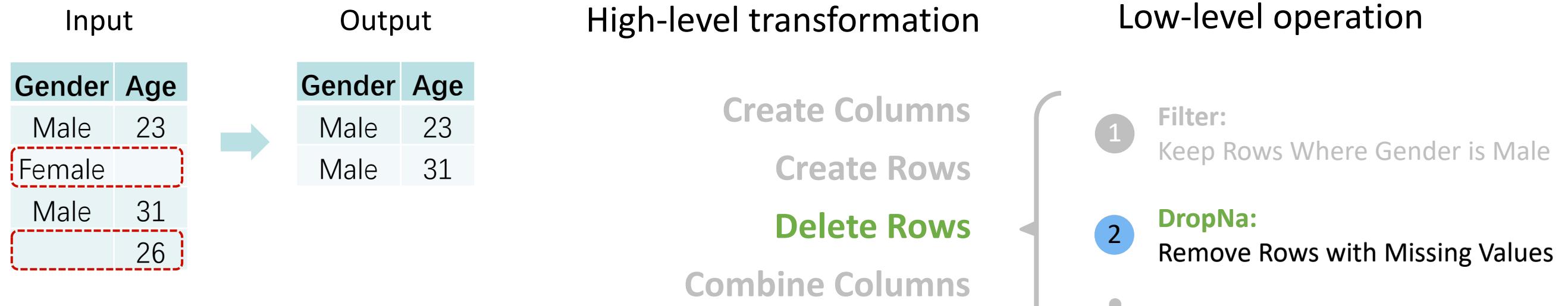
- Differences between input and output tables can reveal the transformation type





# Observation

- Differences between input and output tables can reveal the transformation type





# Space of Table Changes

- For different transformations, what changes can be made to tables?



python R

921 lines of  
wrangling code

Inst	Language	Case	Code	Input Table	Output Table	Transformation Type
1	Python	1	raw_full = pd.read_csv('full-downloaded.csv',index_col=None)		L3 (raw_full).csv	create_tables_load
2	Python	1	raw_partial = pd.read_csv('partials-downloaded.csv',index_col=None)		L4 (raw_partial).csv	create_tables_load
3	Python	1	data_raw = pd.concat([raw_full,raw_partial])	L3 (raw_full).csv L4 (raw_partial).csv	L5 (data_raw).csv	combine_tables_extend
4	Python	1	data_typed = data_raw.assign(DemoType = data_raw.apply(lambda x: x.astype(str)))	L5 (data_raw).csv	L8 (data_typed).csv	create_columns_mutate
5	Python	1	permit_status_filtered = data_typed[(data_typed['StatusCurrent'] == 'PERMIT') & (data_typed['StatusCurrent'] != 'DEMOLITION')]	L8 (data_typed).csv	L10 (permit_status_filtered).csv	delete_rows_filter
6	Python	1	filtered_residential = permit_status_filtered[permit_status_filtered['IssuedDate'] < '2018-01-01']	L10 (permit_status_filtered).csv	L11 (filtered_residential).csv	delete_rows_filter
7	Python	1	cutoff = filtered_residential[filtered_residential['IssuedDate'] < '2018-01-01']	L11 (filtered_residential).csv	L12 (cutoff).csv	delete_rows_filter
8	Python	1	demolitions_cut = filtered_residential[filtered_residential['PermitClass'] == 'DEMOLITION']	L11 (filtered_residential).csv	L13 (demolitions_cut).csv	identical_operation
9	Python	1	filtered_homes = demolitions_cut[(demolitions_cut['PermitClass'] == 'DEMOLITION') & (demolitions_cut['StatusCurrent'] == 'PERMIT')]	L13 (demolitions_cut).csv	L14 (filtered_homes).csv	delete_rows_filter
10	Python	1	demolitions_full = filtered_homes	L14 (filtered_homes).csv	L15 (demolitions_full).csv	identical_operation

914	R	32	p78_input1= read.csv("r78_input1.csv")		L4 (p78_input1).csv	create_tables_load
915	R	32	TBL_15=filter(p78_input1,'y' < 1.900000)	L4 (p78_input1).csv	L5 (TBL_15).csv	delete_rows_filter
916	R	32	TBL_7=group_by(TBL_15,'x')	L5 (TBL_15).csv	L6 (TBL_7).csv	identical_operation
917	R	32	TBL_3=summarise(TBL_7,a=mean('y'))	L6 (TBL_7).csv	L7 (TBL_3).csv	combine_rows_summarize
918	R	32	TBL_1=inner_join(TBL_3,p78_input1)	L7 (TBL_3).csv L4 (p78_input1).csv	L8 (TBL_1).csv	combine_tables_inner_join
919	R	32	morpheus=mutate(TBL_1,z=y / a)	L8 (TBL_1).csv	L9 (morpheus).csv	create_columns_mutate
920	R	32	morpheus=as.data.frame(morpheus)	L9 (morpheus).csv	L10 (morpheus).csv	identical_operation
921	R	32	morpheus=select(morpheus,1,3,4,2,5)	L10 (morpheus).csv	L11 (morpheus).csv	transform_tables_rearrange

Observation

Change Space

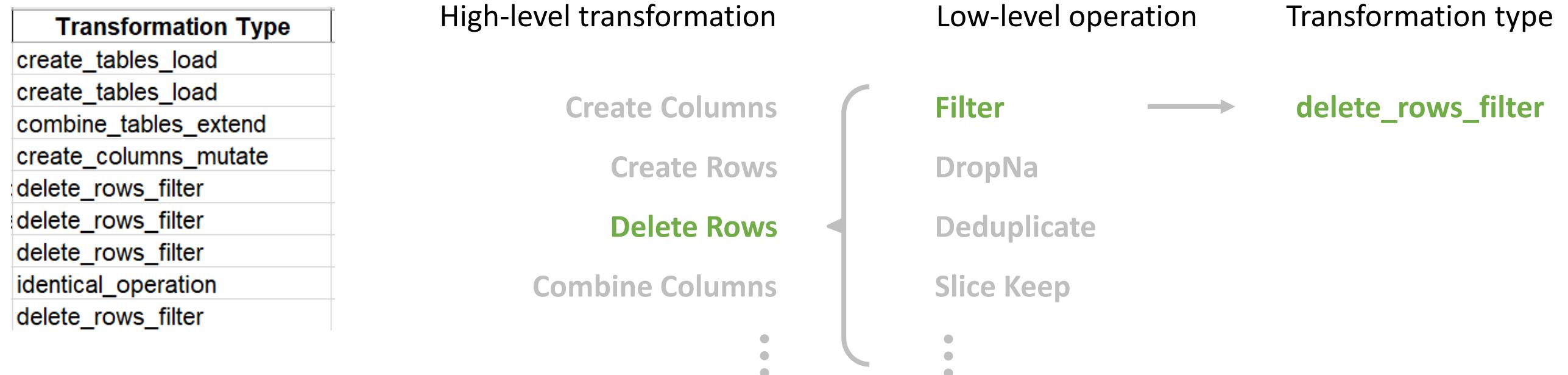
Mapping

COMANTICS



# Space of Table Changes

- For different transformations, what changes can be made to tables?

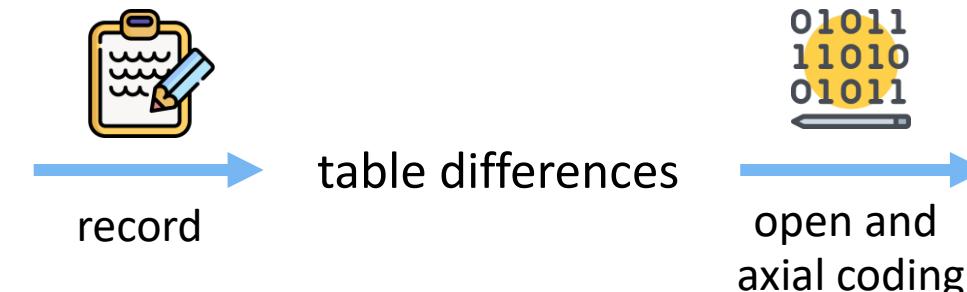




# Space of Table Changes

- For different transformations, what changes can be made to tables?

Input Table	Output Table
	L3 (raw_full).csv
	L4 (raw_partial).csv
L3 (raw_full).csv L4 (raw_partial).c sv	L5 (data_raw).csv
L5 (data_raw).csv	L8 (data_typed).csv
L8 (data_typed).csv	L10 (permit_status_filtered).c sv
L10 (permit_status_filtered).csv	L11 (filtered_residential).c sv
L11 (filtered_residential).csv	L12 (cutoff).csv
L11 (filtered_residential).csv	L13 (demolitions_cut).csv
L13 (demolitions_cut).csv	L14 (filtered_homes).csv



	Outputs :  Inputs :	Table	Column	Row	Cell
1.  > 0 &  = 0	6.  =	23.  -			
2.  = 0 &  > 0	7.  <	24.  <			
3.  > 0 &  =	8.  >	25.  >			
4.  > 0 &  >	9. min  >	26. min  >			
5.  > 0 &  <	10. min  <	27. min  =			
	11. min  <  & max  >	28. min  <  & max  >			
	12. max  =	29. max  =			
	13. max  <  & sum  >	30. max  <  & sum  >			
	14. sum  =	31. sum  =			
	15. sum  <	32. sum  <			
	16.  = min	33.  = min			
	17.  < min	34.  < min			
	18.  > min  &  < max	35.  > min  &  < max			
	19.  = max	36.  = max			
	20.  < max  &  < sum	37.  < max  &  < sum			
	21.  < sum	38.  < sum			
	22.  = sum	39.  = sum			
	40. index  = index	45. index  = index			
	41. index  > index	46. index  > index			
	42. order  > disorder	47. order  > disorder			
	43. order  = ascend	48. order  = ascend			
	44. order  = descend	49. order  = descend			
	50.  <	58.  <	66.  <		
	51.  =	59.  =	67.  =		
	52.  >	60.  >	68.  >		
	53.  <	61.  <	69.  <		
	54.  > duplicate  &  < duplicate	62.  > duplicate  &  < duplicate	70.  > duplicate  &  < duplicate	same column	
	55.  > duplicate  &  > duplicate	63.  > duplicate  &  < duplicate	71.  > duplicate  &  > duplicate		
	56.  > duplicate  &  < duplicate	64.  > duplicate  &  < duplicate	72.  > duplicate  &  < duplicate		
	57.  > duplicate  &  > duplicate	65.  > duplicate  &  < duplicate	73.  > duplicate  &  < duplicate		
				same row	
	74.  > duplicate  &  < duplicate	75.  > duplicate  &  < duplicate	76.  > duplicate  &  < duplicate		
	77.  > duplicate  &  < duplicate	78.  > missing  &  < missing	79.  > missing  &  > missing	80.  > missing  &  < missing	
		81.  < missing  &  < missing	82.  > specific val	83.  > specific val	
		84.  < missing  &  > missing	85.  < missing  &  > missing	86.  < missing  &  < missing	
		87.  < missing  &  < missing	88.  > specific val	89.  > specific val	
		90.  < missing  &  < missing	91.  < missing  &  < missing	92.  < missing  &  < missing	
		93.  < missing  &  < missing	94.  > specific val	95.  > specific val	
		96.  < missing  &  < missing	97.  < missing  &  < missing	98.  < missing  &  < missing	
		99.  < missing  &  < missing	100.  = val1 &  = val2	101.  > val1 &  = val2	
		102. type  = type	103. type  > type		

the space of table changes



# Space of Table Changes

	Table	Column	Row	Cell	
Number	1. $\text{blue} > 0 \ \& \ \text{red} = 0$ 2. $\text{blue} = 0 \ \& \ \text{red} > 0$ 3. $\text{blue} > 0 \ \& \ \text{blue} = \text{red}$ 4. $\text{blue} > 0 \ \& \ \text{blue} > \text{red}$ 5. $\text{blue} > 0 \ \& \ \text{blue} < \text{red}$	6. $\text{blue} = \text{red}$ 7. $\text{blue} < \text{red}$ 8. $\text{blue} > \text{red}$ 9. $\min(\text{blue}) > \text{red}$ 10. $\min(\text{blue}) = \text{red}$ 11. $\min(\text{blue}) < \text{red} \ \& \ \max(\text{blue}) > \text{red}$ 12. $\max(\text{blue}) = \text{red}$ 13. $\max(\text{blue}) < \text{red} \ \& \ \sum(\text{blue}) > \text{red}$ 14. $\sum(\text{blue}) = \text{red}$ 15. $\sum(\text{blue}) < \text{red}$ 16. $\text{blue} < \min(\text{red})$ 17. $\text{blue} = \min(\text{red})$ 18. $\text{blue} > \min(\text{red}) \ \& \ \text{blue} < \max(\text{red})$ 19. $\text{blue} = \max(\text{red})$ 20. $\text{blue} > \max(\text{red}) \ \& \ \text{blue} < \sum(\text{red})$ 21. $\text{blue} = \sum(\text{red})$ 22. $\text{blue} > \sum(\text{red})$	23. $\text{blue} = \text{red}$ 24. $\text{blue} < \text{red}$ 25. $\text{blue} > \text{red}$ 26. $\min(\text{blue}) > \text{red}$ 27. $\min(\text{blue}) = \text{red}$ 28. $\min(\text{blue}) < \text{red} \ \& \ \max(\text{blue}) > \text{red}$ 29. $\max(\text{blue}) = \text{red}$ 30. $\max(\text{blue}) < \text{red} \ \& \ \sum(\text{blue}) > \text{red}$ 31. $\sum(\text{blue}) = \text{red}$ 32. $\sum(\text{blue}) < \text{red}$ 33. $\text{blue} < \min(\text{red})$ 34. $\text{blue} = \min(\text{red})$ 35. $\text{blue} > \min(\text{red}) \ \& \ \text{blue} < \max(\text{red})$ 36. $\text{blue} = \max(\text{red})$ 37. $\text{blue} > \max(\text{red}) \ \& \ \text{blue} < \sum(\text{red})$ 38. $\text{blue} = \sum(\text{red})$ 39. $\text{blue} > \sum(\text{red})$	∅	
Order	∅	40. $\text{index}(\text{blue}) = \text{index}(\text{red})$ 41. $\text{index}(\text{blue}) \neq \text{index}(\text{red})$ 42. $\text{order}(\text{blue}) = \text{disorder}$ 43. $\text{order}(\text{blue}) = \text{ascend}$ 44. $\text{order}(\text{blue}) = \text{descend}$	45. $\text{index}(\text{blue}) = \text{index}(\text{red})$ 46. $\text{index}(\text{blue}) \neq \text{index}(\text{red})$ 47. $\text{order}(\text{blue}) = \text{disorder}$ 48. $\text{order}(\text{blue}) = \text{ascend}$ 49. $\text{order}(\text{blue}) = \text{descend}$	∅	
Relation	∅	50. $\text{blue} < \text{red}$ 51. $\text{blue} > \text{red}$ 52. $\text{blue} = \text{red}$ 53. $\text{blue} \neq \text{red}$  54. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 55. $\exists \text{ duplicate}(\text{blue}) \ \& \ \forall \text{ duplicate}(\text{red})$ 56. $\forall \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 57. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$	58. $\text{blue} < \text{red}$ 59. $\text{blue} > \text{red}$ 60. $\text{blue} = \text{red}$ 61. $\text{blue} \neq \text{red}$  62. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 63. $\exists \text{ duplicate}(\text{blue}) \ \& \ \forall \text{ duplicate}(\text{red})$ 64. $\forall \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 65. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$	66. $\text{blue} < \text{red}$ 67. $\text{blue} > \text{red}$ 68. $\text{blue} = \text{red}$ 69. $\text{blue} \neq \text{red}$  same column 70. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 71. $\exists \text{ duplicate}(\text{blue}) \ \& \ \forall \text{ duplicate}(\text{red})$ 72. $\forall \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 73. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$  same row 74. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 75. $\exists \text{ duplicate}(\text{blue}) \ \& \ \forall \text{ duplicate}(\text{red})$ 76. $\forall \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$ 77. $\exists \text{ duplicate}(\text{blue}) \ \& \ \exists \text{ duplicate}(\text{red})$	
Value	∅	78. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 79. $\exists \text{ missing}(\text{blue}) \ \& \ \forall \text{ missing}(\text{red})$ 80. $\forall \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 81. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 82. $\text{blue} \in \text{specific val}$ 83. $\text{blue} \notin \text{specific val}$	84. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 85. $\exists \text{ missing}(\text{blue}) \ \& \ \forall \text{ missing}(\text{red})$ 86. $\forall \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 87. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 88. $\text{blue} \in \text{specific val}$ 89. $\text{blue} \notin \text{specific val}$	90. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 91. $\exists \text{ missing}(\text{blue}) \ \& \ \forall \text{ missing}(\text{red})$ 92. $\forall \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 93. $\exists \text{ missing}(\text{blue}) \ \& \ \exists \text{ missing}(\text{red})$ 94. $\text{blue} \in \text{specific val}$ 95. $\text{blue} \notin \text{specific val}$	96. $\text{! missing}(\text{blue}) \ \& \ \text{missing}(\text{red})$ 97. $\text{missing}(\text{blue}) \ \& \ \text{! missing}(\text{red})$ 98. $\text{! missing}(\text{blue}) \ \& \ \text{! missing}(\text{red})$ 99. $\text{missing}(\text{blue}) \ \& \ \text{missing}(\text{red})$ 100. $\text{blue} = \text{val1} \ \& \ \text{red} = \text{val2}$ 101. $\text{blue} \neq \text{val1} \ \& \ \text{red} = \text{val2}$
Type	∅	102. $\text{type}(\text{blue}) = \text{type}(\text{red})$ 103. $\text{type}(\text{blue}) \neq \text{type}(\text{red})$	∅	∅	

2 dimensions

## Types of data objects

1. Tables
2. Columns
3. Rows
4. Cells

## Properties of data changes

1. Number
2. Order
3. Relation
4. Value
5. Type

103 characteristics

$C_1 - C_{103}$

### Number

#### Table

1. the number of output tables is greater than zero and the number of input tables is zero
2. the number of output tables is zero and the number of input tables is greater than zero
3. the number of output tables is greater than zero and the number of output tables is equal to the number of input tables
4. the number of output tables is greater than zero and the number of output tables is greater than the number of input tables
5. the number of output tables is greater than zero and the number of output tables is less than the number of input tables

#### Column

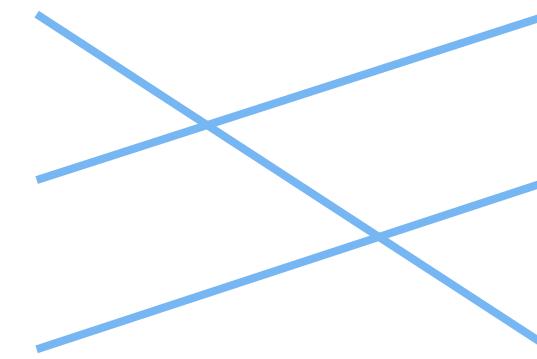
6. the number of columns in the output table is equal to the number of columns in the input table
7. the number of columns in the output table is less than the number of columns in the input table





# Build a Mapping

Transformations
• <code>create_columns_mutate</code>
• <code>delete_rows_filter</code>
• <code>delete_columns_select_keep</code>
• <code>combine_rows_summarize</code>
• <code>transform_columns_rename</code>
• <code>transform_columns_mutate</code>
• <code>transform_tables_sort</code>
• <code>transform_columns_replace</code>
⋮



Characteristics
6. $\text{[ ]} = \text{[ ]}$
7. $\text{[ ]} < \text{[ ]}$
8. $\text{[ ]} > \text{[ ]}$
9. $\min(\text{[ ]}) > \text{[ ]}$
10. $\min(\text{[ ]}) = \text{[ ]}$
11. $\min(\text{[ ]}) < \text{[ ]} \& \max(\text{[ ]}) > \text{[ ]}$
12. $\max(\text{[ ]}) = \text{[ ]}$
13. $\max(\text{[ ]}) < \text{[ ]} \& \sum(\text{[ ]}) > \text{[ ]}$
⋮

Output  
 Input

Observation

Change Space

Mapping

COMANTICS



# Build a Mapping

Will the characteristic appear after performing a transformation?

Transformations	Inevitable	Impossible	Possible
• <code>create_columns_mutate</code>	✓	✗	?
• <code>delete_rows_filter</code>		✗	?
• <code>delete_columns_select_keep</code>	✓	✗	?
• <code>combine_rows_summarize</code>			• $C_2, \dots$
• <code>transform_columns_rename</code>			• $C_3, \dots$
• <code>transform_columns_mutate</code>			• $C_1, \dots$
• <code>transform_tables_sort</code>			
• <code>transform_columns_replace</code>			
⋮	⋮	⋮	⋮



# Build a Mapping

Will the characteristic appear after performing a transformation?

Transformations	
<ul style="list-style-type: none"><li>• <code>delete_rows_filter</code></li></ul>	
Gender	Age
Male	23
Female	
Male	31

Keep Rows Where Gender is Male



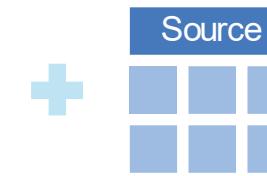
- Inevitable 
- The number of columns remains unchanged  
• • •
- Impossible 
- The number of rows has increased  
• • •
- Possible 
- The number of rows has decreased  
• • •



# COMANTICS Overview

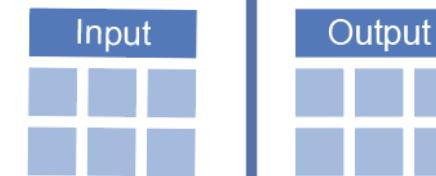
## Data Wrangling Code

```
df = pd.read_csv("students.csv")
df.id = df.id.str.extract("(\\d+)")
df.drop_duplicates(inplace=True)
df.loc[:, "total"] = df.math + df.art
results = df.sort_values("total", ...)
```



```
{  
    type: "transform_columns_extract",  
    parameter: {  
        input_cols: ["id"],  
        output_cols: ["id"],  
        extract_rules: '(\\d+)'  
    }  
}
```

## Generate Intermediate Tables



## Parameter Inference: Slot Filling

- 1.transform\_columns\_extract
- 2.transform\_columns\_mutate
- ...

## Detect Characteristics



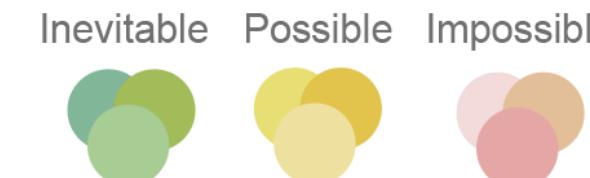
## Type Inference: CNN-Based

Deduplicate  
Fold  
Extract  
Replace

Merge  
Mutate  
Create  
Sort

Summarize

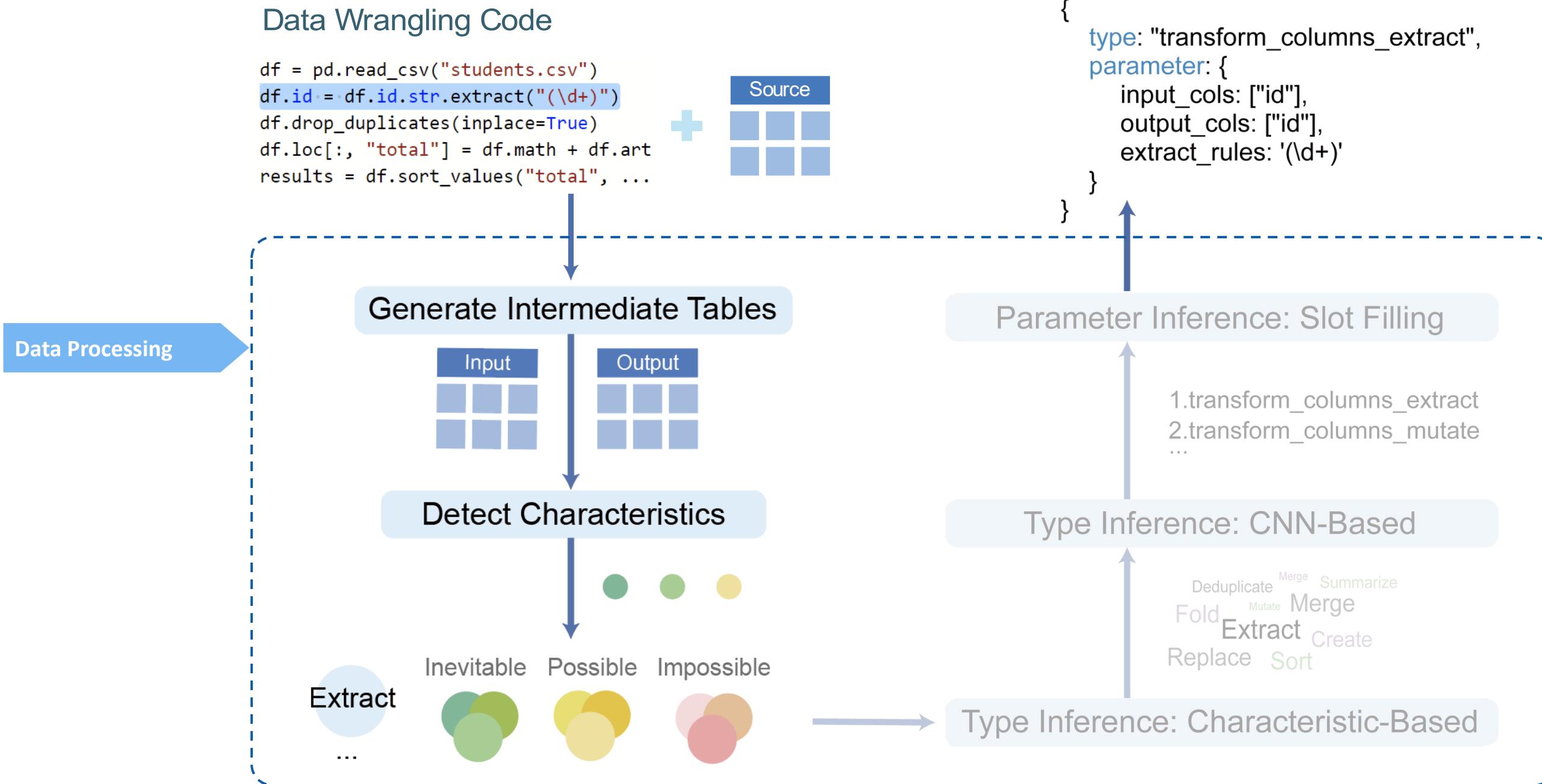
## Extract



## Type Inference: Characteristic-Based



# COMANTICS Overview

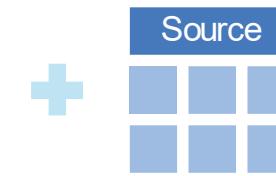




# COMANTICS Overview

## Data Wrangling Code

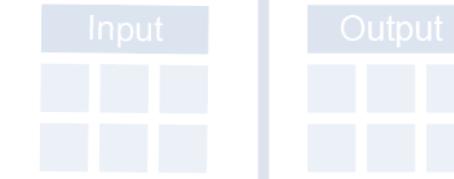
```
df = pd.read_csv("students.csv")
df.id = df.id.str.extract("(\\d+)")
df.drop_duplicates(inplace=True)
df.loc[:, "total"] = df.math + df.art
results = df.sort_values("total", ...)
```



```
{  
    type: "transform_columns_extract",  
    parameter: {  
        input_cols: ["id"],  
        output_cols: ["id"],  
        extract_rules: '(\\d+)'  
    }  
}
```

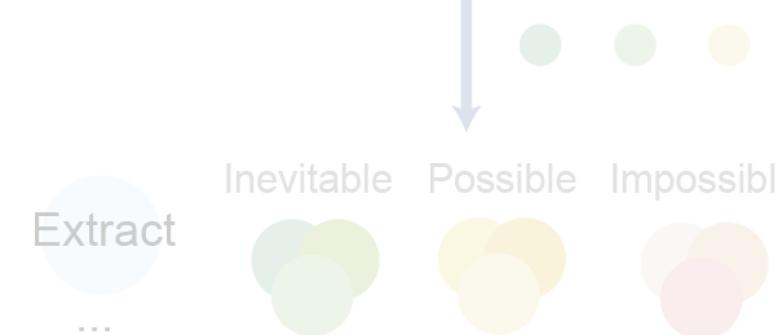
Data Processing

Generate Intermediate Tables



Type Inference

Detect Characteristics



Parameter Inference: Slot Filling

- 1.transform\_columns\_extract
- 2.transform\_columns\_mutate
- ...

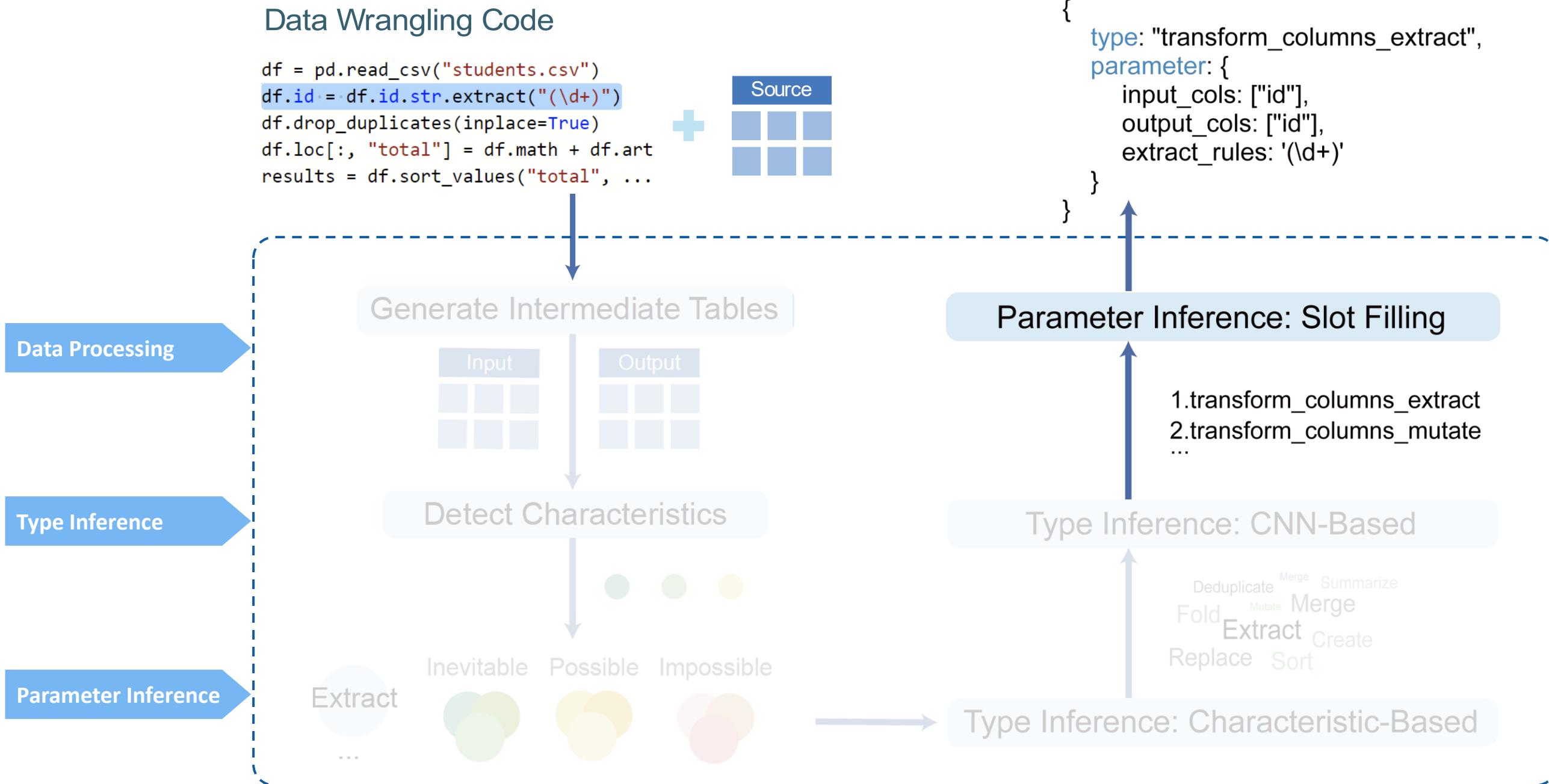
Type Inference: CNN-Based

Deduplicate   Merge   Summarize  
Fold   Mutate   Merge  
Extract   Create  
Replace   Sort

Type Inference: Characteristic-Based

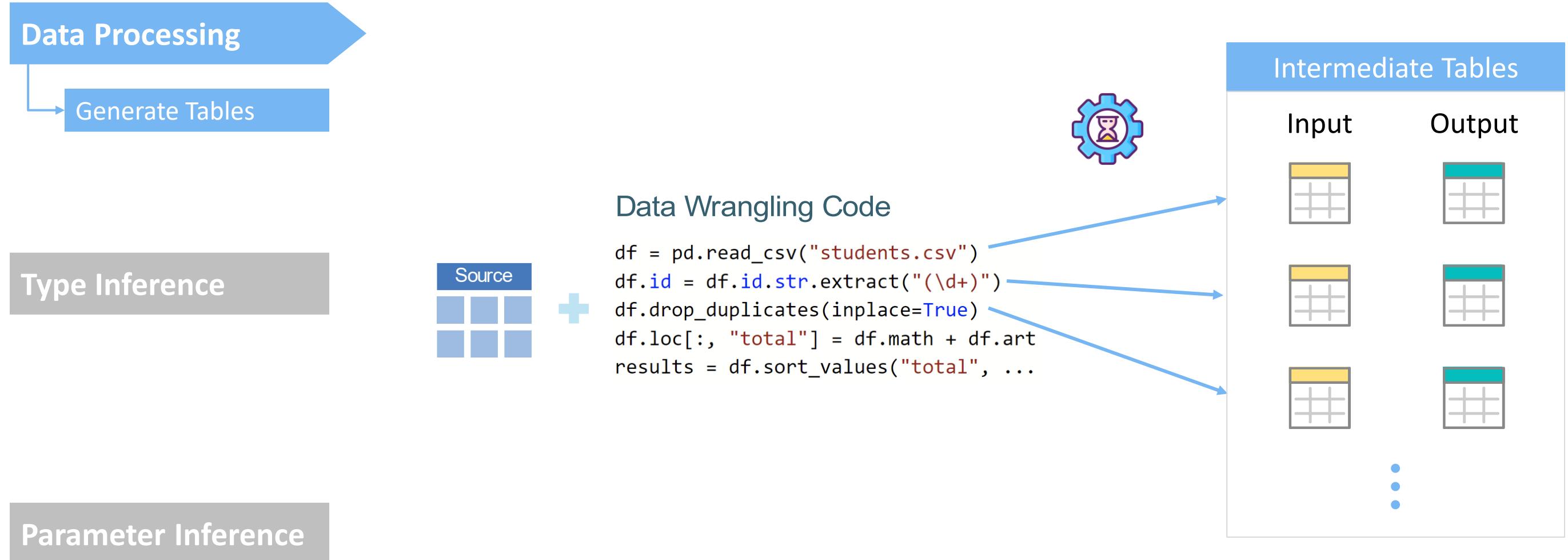


# COMANTICS Overview



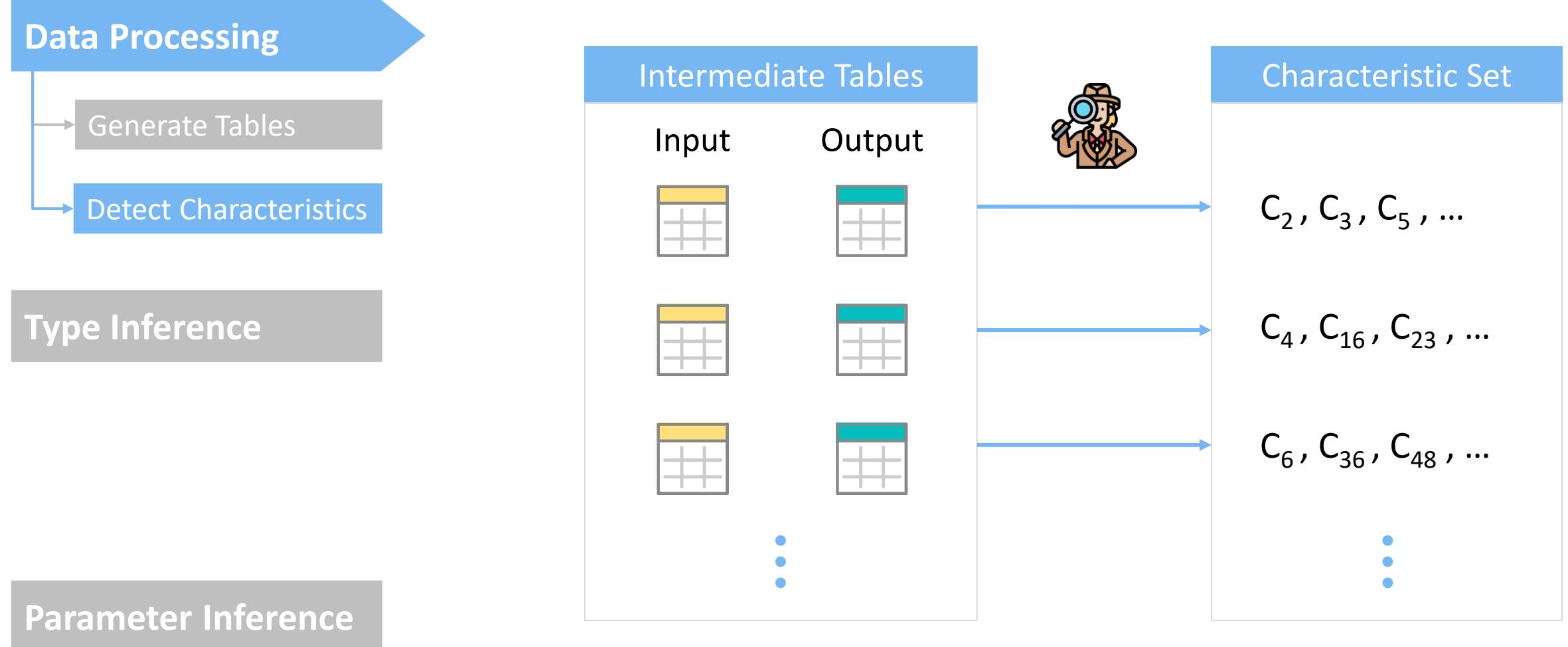


# COMANTICS





# COMANTICS





# COMANTICS

## Data Processing

- Generate Tables
- Detect Characteristics

## Type Inference

- Characteristic-Based

## Parameter Inference

## Characteristic Set

$C_2, C_3, C_5, \dots$

$C_4, C_{16}, C_{23}, \dots$

$C_6, C_{36}, C_{48}, \dots$

⋮

Two Strategies: filter out transformations whose

1. **inevitable** set is not a subset of the characteristic set
2. **impossible** set intersects the characteristic set



# COMANTICS

## Data Processing

- Generate Tables
- Detect Characteristics

## Type Inference

- Characteristic-Based

## Parameter Inference

## Characteristic Set

$C_2, C_3, C_5, \dots$

$C_4, C_{16}, C_{23}, \dots$

$C_6, C_{36}, C_{48}, \dots$

⋮

## Mapping

Transformation	Inevitable	Impossible	Possible
$T_1$	$C_1, C_3$	$C_6, C_9$	$C_2, \dots$
$T_2$	$C_2, C_5$	$C_1, C_7$	$C_3, \dots$
$T_3$	$C_3, C_5$	$C_2, C_8$	$C_1, \dots$
...	...	...	...

Two Strategies: filter out transformations whose

1. **inevitable** set is not a subset of the characteristic set
2. **impossible** set intersects the characteristic set



# COMANTICS

## Data Processing

- Generate Tables
- Detect Characteristics

## Type Inference

- Characteristic-Based

## Parameter Inference

## Characteristic Set

$C_2, C_3, C_5, \dots$

$C_4, C_{16}, C_{23}, \dots$

$C_6, C_{36}, C_{48}, \dots$

⋮

## Mapping

Transformation	Inevitable	Impossible	Possible
$\times T_1$	$C_1, C_3$	$C_6, C_9$	$C_2, \dots$
$T_2$	$C_2, C_5$	$C_1, C_7$	$C_3, \dots$
$T_3$	$C_3, C_5$	$C_2, C_8$	$C_1, \dots$
...	...	...	...

Strategy 1

Two Strategies: filter out transformations whose

1. **inevitable** set is not a subset of the characteristic set
2. **impossible** set intersects the characteristic set



# COMANTICS

## Data Processing

- Generate Tables
- Detect Characteristics

## Type Inference

- Characteristic-Based

## Parameter Inference

## Characteristic Set

$C_2, C_3, C_5, \dots$

$C_4, C_{16}, C_{23}, \dots$

$C_6, C_{36}, C_{48}, \dots$

⋮

## Mapping

Transformation	Inevitable	Impossible	Possible
✗ $T_1$	$C_1, C_3$	$C_6, C_9$	$C_2, \dots$
$T_2$	$C_2, C_5$	$C_1, C_7$	$C_3, \dots$
✗ $T_3$	$C_3, C_5$	$C_2, C_8$	$C_1, \dots$
...	...	...	...

Strategy 1

Strategy 2

Two Strategies: filter out transformations whose

1. **inevitable** set is not a subset of the characteristic set
2. **impossible** set intersects the characteristic set



# COMANTICS

## Data Processing

→ Generate Tables

→ Detect Characteristics

## Type Inference

→ Characteristic-Based

## Parameter Inference

Gender	Age
Male	23
Female	28
Male	31

Gender	Age
Male	23
Female	28
Male	31

Gender	Age
Male	23
Female	
Male	31

Gender	Age
Male	23
Male	31

Ambiguity 1

the output is identical to the input

- 1 Remove Duplicate Rows
- 2 Sort Age in Ascending Order

⋮

Ambiguity 2

more than two transformations lead to the same changes

- 1 Keep Rows Where Gender is Male
- 2 Remove Rows with Missing Values

⋮

## Mapping

Transformation	Inevitable	Impossible	Possible
X T <sub>1</sub>	C <sub>1</sub> , C <sub>3</sub>	C <sub>6</sub> , C <sub>9</sub>	C <sub>2</sub> , ...
T <sub>2</sub>	C <sub>2</sub> , C <sub>5</sub>	C <sub>1</sub> , C <sub>7</sub>	C <sub>3</sub> , ...
X T <sub>3</sub>	C <sub>3</sub> , C <sub>5</sub>	C <sub>2</sub> , C <sub>8</sub>	C <sub>1</sub> , ...
...	...	...	...

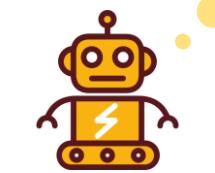


Transformation candidates:  
T<sub>2</sub>, T<sub>6</sub>, T<sub>17</sub>, T<sub>21</sub>



# COMANTICS

Which one is correct?



## Data Processing

→ Generate Tables

→ Detect Characteristics

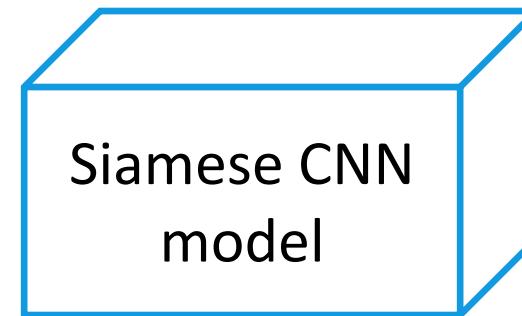
## Type Inference

→ Characteristic-Based

→ CNN-Based

## Parameter Inference

Transformation candidates:  $T_2, T_6, T_{17}, T_{21}$



suitable for

- 1 few shot problems
- 2 imbalanced data

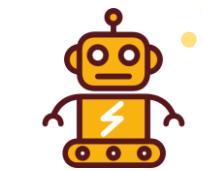


Transformation likelihood:  $T_6 > T_2 > T_{21} > T_{17}$



# COMANTICS

Which one is correct?



## Data Processing

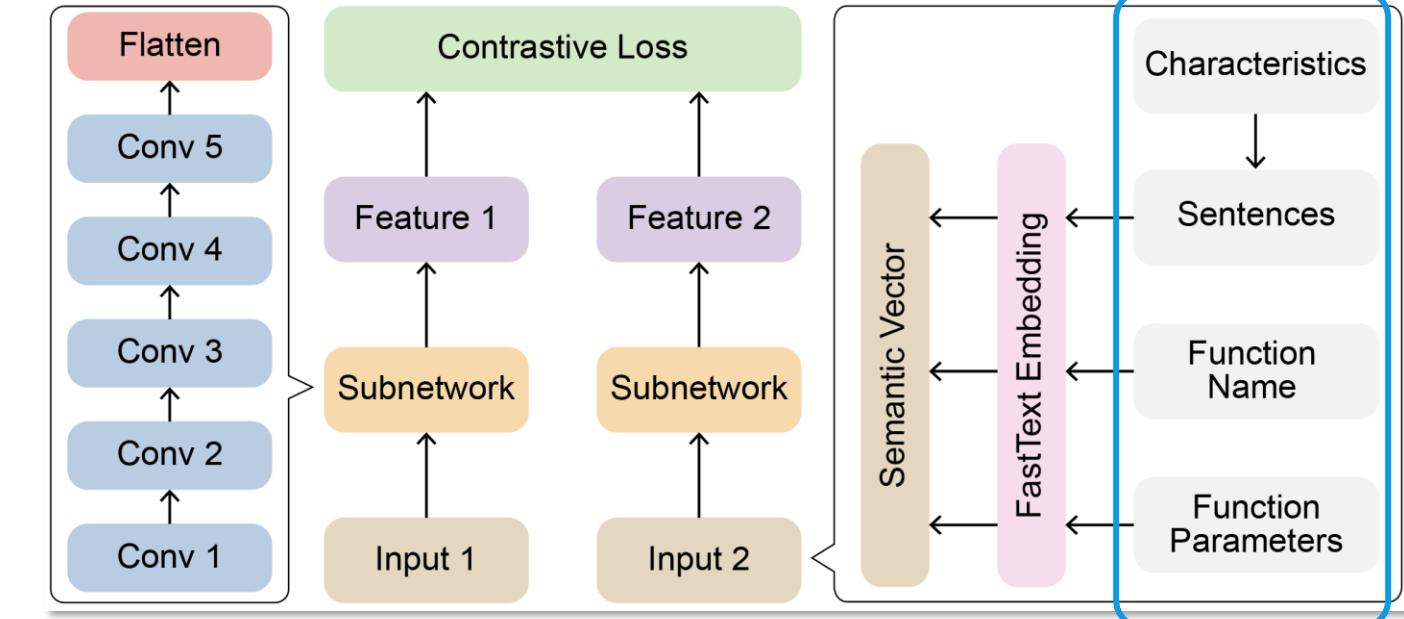
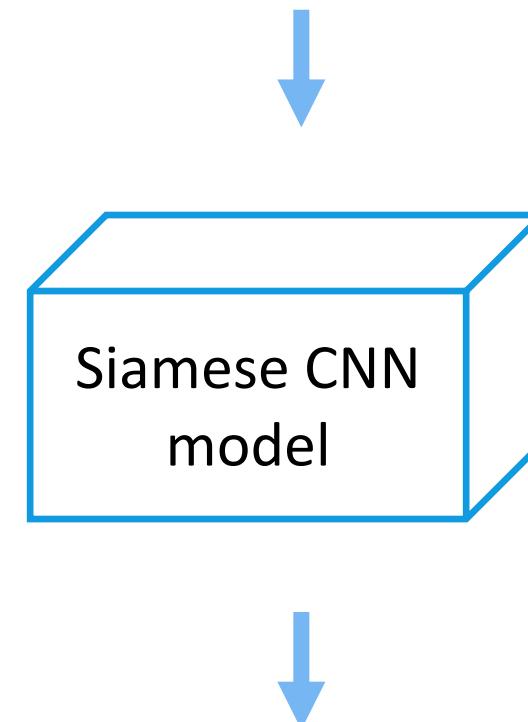
- Generate Tables
- Detect Characteristics

## Type Inference

- Characteristic-Based
- CNN-Based

## Parameter Inference

Transformation candidates:  $T_2, T_6, T_{17}, T_{21}$



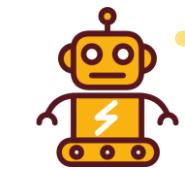
Please refer to the paper for details.

Transformation likelihood:  $T_6 > T_2 > T_{21} > T_{17}$



# COMANTICS

Which one is correct?



## Data Processing

→ Generate Tables

→ Detect Characteristics

## Type Inference

→ Characteristic-Based

→ CNN-Based

## Parameter Inference

→ Slot Filling

Transformation likelihood:  $T_6 > T_2 > T_{21} > T_{17}$

✗

✓

$T_6$

- P1: ✓
- P2: ✓
- P3: ✗
- P4:

$T_2$

- P1: ✓
- P2: ✓
- P3: ✓
- P4:

$T_{21}$

- P1:
- P2:
- P3:
- P4:
- P5:

$T_{17}$

- P1:
- P2:
- P3:
- P4:

slot



Transformation type:  $T_2$

Parameters: {

P1: xxx ,  
P2: xxx ,  
P3: xxx

}

```
df.id = df.id.str.extract("(\\d+)")
{
    type: "transform_columns_extract",
    parameter: {
        input_cols: ["id"],
        output_cols: ["id"],
        extract_rules: '(\\d+)'
    }
}
```



# Experiments

- Two aspects of evaluation:
  - How each part of COMANTICS contributes to the overall **performance**
  - How well can COMANTICS **generalize** to different programming languages
- Dataset
  - 606 lines of Python code
  - 315 lines of R code
- Results

Table 2. The training time, Top-1, and Top-3 performances of COMANTICS and its CNN-based component in different experiment settings.

	Setting		CNN		COMANTICS		Time (minutes)
	Train	Test	Top-1	Top-3	Top-1	Top-3	
1st	Python	R	19.1	30.3	53.8	76.1	19.5
2nd	R	Python	23.2	32.3	44.5	82.6	15.1
3rd	Python	Python	62.1	83.3	80.3	94.7	18.3
4th	All	Python	72.0	90.9	90.2	98.5	21.9
5th	All	All	78.0	92.2	92.2	99.0	19.6

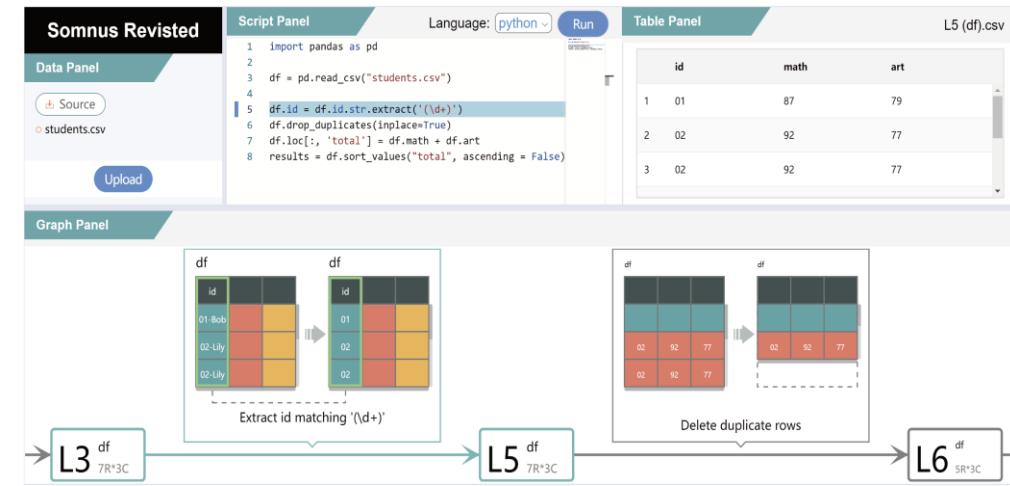


# Example Applications

jupyter

```
1 # Transform Columns: Extract (\d+) from id into id
2 df.id = df.id.str.extract('(\d+)')
3 # Delete Rows: Delete duplicate rows
4 df.drop_duplicates(inplace=True)
5 # Create Columns: Mutate math and art into total
6 df.loc[:, 'total'] = df.math + df.art
7 # Transform Tables: Sort total in descending order
8 results = df.sort_values("total", ascending = False)
9 results
```

Code Annotation



Semantics Visualization

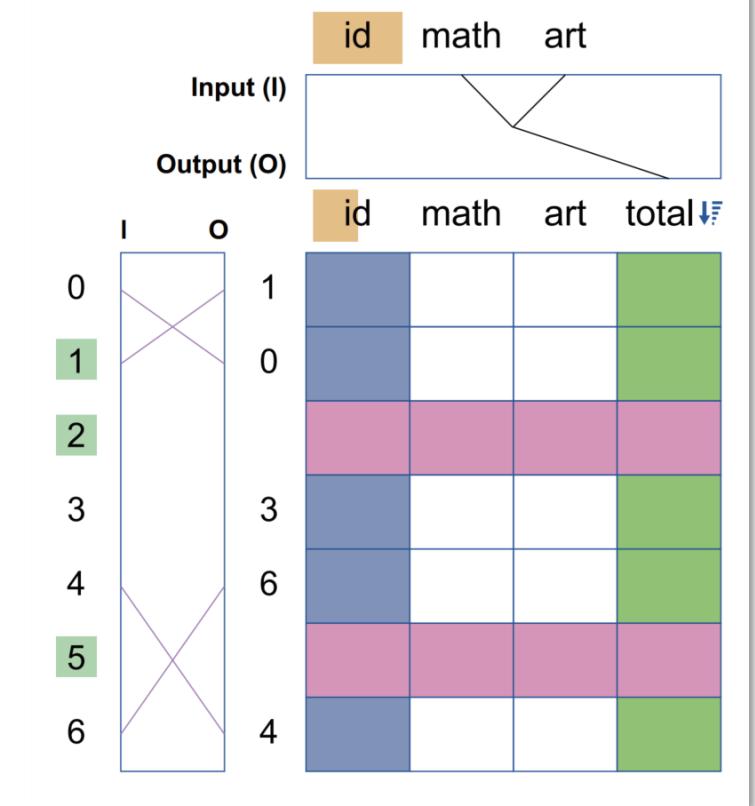


Table Comparison

File Edit View Insert Cell Kernel Navigate Help

可信 | siamese



In [1]: 1 import pandas as pd

In [2]: 1 df = pd.read\_csv("students.csv")  
2 df

Out[2]:

	id	math	art
0	01-Bob	87	79
1	02-Lily	92	77
2	02-Lily	92	77
3	04-Alan	64	93
4	05-John	68	81
5	05-John	68	81
6	07-Iris	71	84

In [3]: 1 df.id = df.id.str.extract('(\d+)')  
2 df.drop\_duplicates(inplace=True)  
3 df.loc[:, 'total'] = df.math + df.art  
4 results = df.sort\_values("total", ascending = False)  
5 results

Out[3]:

	id	math	art	total
1	02	92	77	169
0	01	87	79	166
3	04	64	93	157
6	07	71	84	155
4	05	68	81	149

This case shows that the Comantics-augmented Jupyter notebook can perform code annotation automatically.



# Conclusion

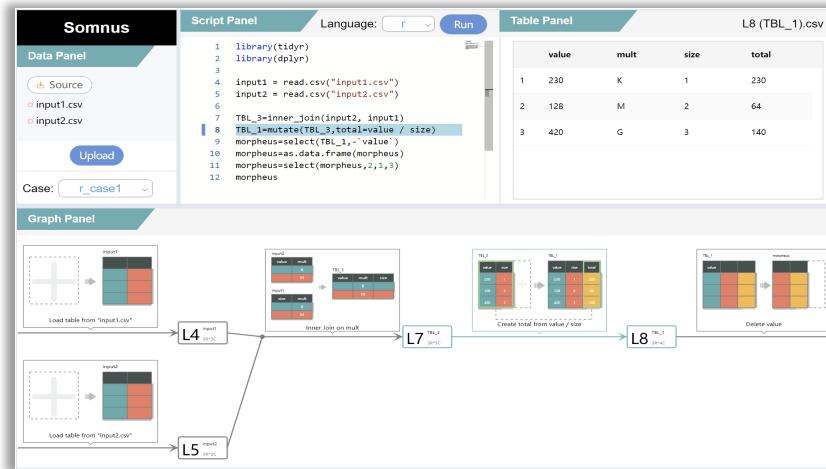
## 💡 Contributions

- summarize a space including 103 characteristics of table changes
- construct COMANTICS, a novel pipeline for inferring semantics of data wrangling scripts
- build a real-world dataset including 921 lines of wrangling code where each is annotated with a data transformation
- perform quantitative experiments to evaluate the performance of COMANTICS
- apply COMANTICS to three applications in different domains

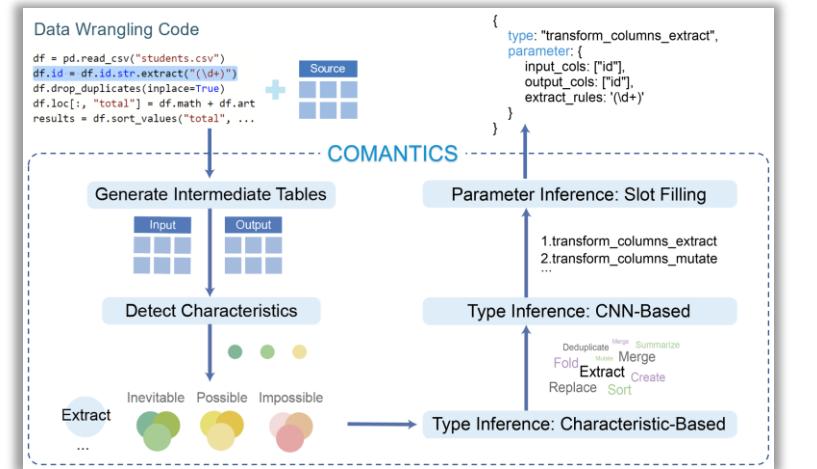
## 📋 Future Work

- improve the performance of COMANTICS by refining the Data Preprocessing step and Type Inference module
- enhance the capability to meet the need for production usage
- incorporate COMANTICS in a broad range of applications, including spreadsheet software, data wrangling tool, and source-to-source compiler

## SOMNUS



## COMANTICS



# Thank You!

IEEE TVCG 2022

IEEE VIS 2022

E-mail: [kaixiong@zju.edu.cn](mailto:kaixiong@zju.edu.cn)