Table Illustrator: Puzzle-based interactive authoring of plain tables

Yanwei Huang
Zhejiang University
Hangzhou, Zhejiang, China
huangyw@zju.edu.cn

Yurun Yang
Zhejiang University
Ningbo, Zhejiang, China
yurunyang@zju.edu.cn

Xinhuan Shu
Newcastle University
Newcastle upon Tyne, United Kingdom
xinhuan.shu@gmail.com

Ran Chen
Zhejiang University
Hangzhou, Zhejiang, China
chenran928@zju.edu.cn

Di Weng
Zhejiang University
Ningbo, Zhejiang, China
dweng@zju.edu.cn

Yingcai Wu
Zhejiang University
Hangzhou, Zhejiang, China
ycwu@zju.edu.cn

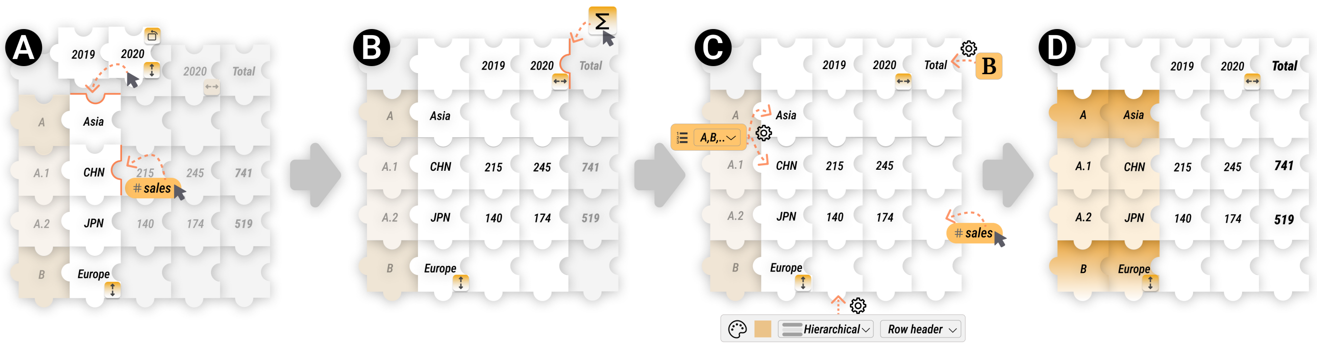


Figure 1: Table Illustrator leverages the puzzle metaphor to visualize table cells of the same data entity, data functions, or data values. Users can drag data entities, existing puzzles, or data functions to the edges of a puzzle to efficiently construct a table. For instance, (A) is derived by combining puzzles corresponding to *continents* and *countries*. To build a crosstab, users can drag the entity *sales* and the puzzle of *years* to it (A), and further drag an aggregation function to the results to append a total column (B). Users can also elaborate it by configuring local structural parameters like *key* to add indexes and stylistic parameters like *font weight* to bold a column, while also adopting global templates such as hierarchical background colors for the row header (C). A compact version of the result table is shown in (D), and users can choose to expand specific puzzles or switch to the preview mode to view the complete table.

ABSTRACT

Plain tables excel at displaying data details and are widely used in data presentation, often polished to an elaborate appearance for readability in many scenarios. However, existing authoring tools fail to provide both flexible and efficient support for altering the table layout and styles, motivating us to develop an intuitive and swift tool for table prototyping. To this end, we contribute Table Illustrator, a table authoring system taking a novel visual metaphor, puzzle, as the primary interaction unit. Through combinations and configurations on puzzles, the system enables rapid table construction and supports a diverse range of table layouts and styles. The tool design is informed by practical challenges and requirements from interviews with 10 table practitioners and a structured design space based on an analysis of over 2,500 real-world tables. User studies showed that Table Illustrator achieved comparable performance to Microsoft Excel while reducing users' completion time and perceived workload.

CCS CONCEPTS

• **Human-centered computing** → **Interactive systems and tools**; **Graphical user interfaces**; **User studies**.

KEYWORDS

Plain tables, data presentation, design study, interaction design

ACM Reference Format:
Yanwei Huang, Yurun Yang, Xinhuan Shu, Ran Chen, Di Weng, and Yingcai Wu. 2024. Table Illustrator: Puzzle-based interactive authoring of plain tables. In *Proceedings of the CHI Conference on Human Factors in Computing*

Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3613904.3642415

1 INTRODUCTION

Plain tables are one of the most versatile formats for data organization and communication. By arranging plain texts in a tabular form, plain tables empower the delivery of data facts and insights in many documents, reports, and slide decks [3, 12, 21, 24, 41]. As the "purest form" of data, plain tables are straightforward and ideal for displaying data details, making them popular among data practitioners [3, 29].

However, authoring plain tables can be a challenging task. Authors often need to carefully organize and annotate the cells in the plain tables to enhance the tables' readability and support the sensemaking of data facts [1]. For instance, the following four cases, as illustrated in Figure 2, are common in practice:

- adding annotations (e.g., numbered keys, spacings, or styles) to highlight hierarchies or specific cells (Figure 2 (B1));
- creating a sparse layout to enhance aesthetics and readability (Figure 2 (B2));
- juxtaposing divided sub-tables vertically to fit page width (Figure 2 (B3));
- merging values with identical labels to facilitate comparisons (Figure 2 (B4)).

Existing tools, including data analysis tools and spreadsheet software, fall short in facilitating plain table authoring, particularly in supporting these cases. Data analysis tools, such as Tableau Prep [43], Trifacta [45], and Wrangler [28], provide diverse data operations (e.g., folding and unfolding) to transform tables into different shapes. However, since these tools primarily focus on data analysis, they lack the support for altering table layout and styles flexibly to create elaborate plain tables. By contrast, spreadsheet software, such as Microsoft Excel and Google Sheets, enables cell-level manipulation for users to edit cells' contents, move cells around, and change cells' styles. Despite such a high degree of flexibility, it remains laborious and time-consuming to create complex plain tables with spreadsheet software, which lacks the integration of table semantics and does not scale well with data sizes.

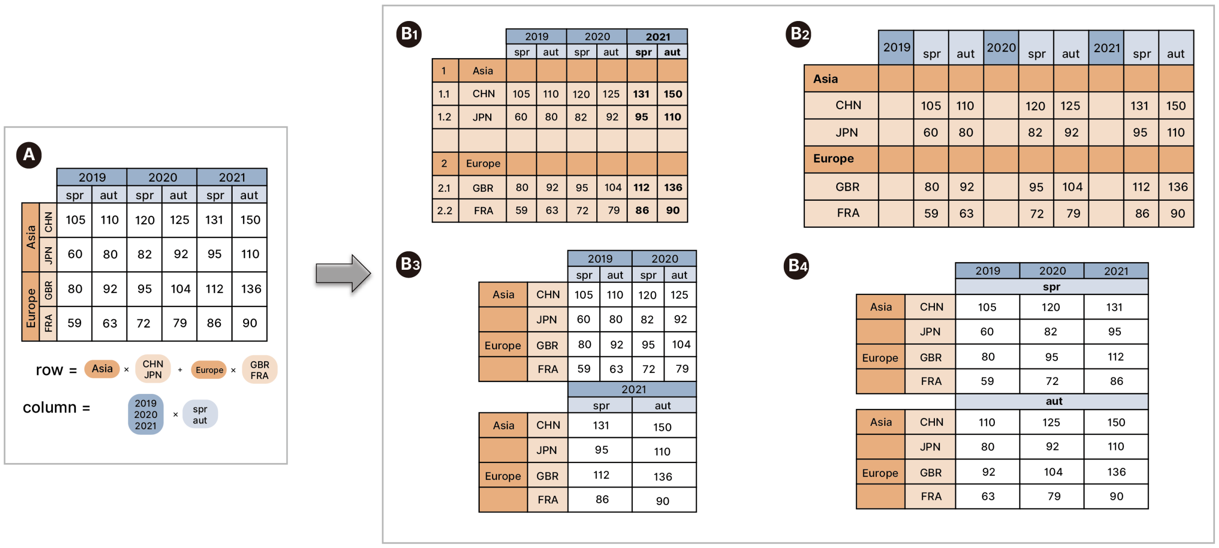
In view of the limitations found in the existing tools, we were motivated to design an interactive table authoring tool to support rapid prototyping of elaborate tables, bridging the gap between flexibility and efficiency. We first conducted a pilot study by interviewing 10 data practitioners from various domains to identify practical challenges and their requirements regarding plain table authoring. In addition, we have analyzed more than 2,500 tables in TableBank [34], a real-world table dataset, to summarize the common structural and stylistic patterns in these tables. The following two challenges were identified:

Accommodate diverse table layouts and styles. Our survey of the table dataset revealed that real-world tables exhibit a high level of diversity and complex cell arrangements. It is challenging to concisely define the designs of these tables with the simple types (e.g., horizontal, vertical, and relational tables [44]) or limited catalogs of structural and stylistic patterns concluded in the prior studies [1, 5, 19, 23, 30, 52]. To fully understand the designs of plain tables and facilitate their creation, a comprehensive survey of real-world plain tables is required to build a systematic taxonomy that properly classifies the layout and styles of these tables and summarizes commonly-used design patterns.

Support semantics-driven customization. Our pilot study revealed that authoring tables with the existing tools may involve repetitive and trivial operations. For instance, users reported that changing cell styles was highly repetitive, often resulting in a lengthy and tedious process of manually applying similar configurations to multiple cells. Prior studies have also shown that many table layout changes require modifying data hierarchies or schemas and may affect many cells simultaneously [7, 33]. We hypothesize that one fundamental reason of these limitations is the GUI of fixed grids adopted by many spreadsheet tools, potentially making semantically related cells (e.g., cells of the same data entity) spatially separated and forcing users to repeat interactions for each cell. This presents opportunities for developing novel interaction strategies that detach from the traditional spreadsheet paradigm, where interactions should be designed on the basis of semantically related cells to facilitate semantics-driven customization beyond the direct manipulation of cells, rows, and columns.

In this paper, we present an analysis of over 2,500 real-world tables and develop a design space of plain tables in terms of structures and styles. This design space is among the first to characterize the detailed features of diverse plain tables and provide a solid theoretical foundation for implementing comprehensive and effective authoring tools. Furthermore, we introduce Table Illustrator, a novel plain table authoring system that utilizes *puzzles*, an intuitive visual metaphor of jigsaw puzzle pieces, as the primary interaction units. Each puzzle is a semantic collection of table cells, representing a data entity (e.g., a column like *Country* in a relational table), the result of an aggregation function, or a specific selection of data values. By combining puzzles with simple drag-and-drop interactions on a canvas-based interface, users can rapidly construct diverse plain tables akin to completing a jigsaw puzzle. Moreover, users can adjust structures and styles easily for each puzzle with the parameters defined by our design space, which provides flexibility for batch changes and accommodates a wide range of table designs. To evaluate the system, user studies were conducted comparing Table Illustrator with Microsoft Excel, one of the most popular table authoring tools. The results showed that Table Illustrator significantly reduced the time cost, mouse clicks, and perceived workload of users in most study tasks compared to the baseline. User feedback also highlighted the intuitiveness and usability of the system as well as the expressiveness of supported table configurations. The contributions of this paper are summarized as follows:

- A requirement study with 10 table practitioners on the design process of plain tables, revealing the design challenges.
- A design space of plain tables based on a survey of more than 2,500 tables from a real-world dataset.
- Table Illustrator, a novel table authoring tool enabling users to rapidly prototype plain tables by combining puzzles with drag-and-drops and configuring parameters of table layout and styles.
- Within-subject user studies comparing the system with Microsoft Excel, demonstrating its usability and effectiveness for reducing user effort in various table authoring tasks.


A

		2019		2020		2021	
		spr	aut	spr	aut	spr	aut
Asia	CHN	105	110	120	125	131	150
	JPN	60	80	82	92	95	110
Europe	GBR	80	92	95	104	112	136
	FRA	59	63	72	79	86	90

row = Asia × CHN JPN + Europe × GBR FRA
column = 2019 2020 2021 × spr aut

B1

		2019		2020		**2021**	
		spr	aut	spr	aut	**spr**	**aut**
1	Asia						
1.1	CHN	105	110	120	125	**131**	**150**
1.2	JPN	60	80	82	92	**95**	**110**
2	Europe						
2.1	GBR	80	92	95	104	**112**	**136**
2.2	FRA	59	63	72	79	**86**	**90**

B2

	2019	spr	aut	2020	spr	aut	2021	spr	aut
Asia									
CHN		105	110		120	125		131	150
JPN		60	80		82	92		95	110
Europe									
GBR		80	92		95	104		112	136
FRA		59	63		72	79		86	90

B3

		2019		2020	
		spr	aut	spr	aut
Asia	CHN	105	110	120	125
	JPN	60	80	82	92
Europe	GBR	80	92	95	104
	FRA	59	63	72	79

		2021	
		spr	aut
Asia	CHN	131	150
	JPN	95	110
Europe	GBR	112	136
	FRA	86	90

B4

		2019	2020	2021
		spr		
Asia	CHN	105	120	131
	JPN	60	82	95
Europe	GBR	80	95	112
	FRA	59	72	86
		aut		
Asia	CHN	110	125	150
	JPN	80	92	110
Europe	GBR	92	104	136
	FRA	63	79	90

Figure 2: Different layouts (B1-B4) of the example table (A) used in HiTailor [33] without changing the underlying logical model.

2 RELATED WORK

2.1 Foundations of plain tables

Plain tables refer to a tabular collection of plain text that contains mostly meta data and is without data visualizations beyond basic stylings (e.g., font, indent, and border). Widely used in spreadsheets and web pages, they have been long studied by the HCI community [1, 5–7, 24, 31, 40, 42, 51]. Fundamentally, they can be understood as the combination of a main table containing cells with a shared structure, and contextual information including titles, footers, and legends [31, 51]. Zhang et al. [51] have presented a comprehensive analysis of their compositions and have surveyed existing taxonomies pertaining to these tables. However, despite studies examining the overall table, the majority of studies focus on the main table, among which most study the logical organizations of data. Some works start from the table composition, dividing tables into row/column headers and cells and mapping data entities to these channels, with headers sometimes exhibiting a hierarchical structure [7, 8, 33, 47]. Others follow a cell-based method, classifying the roles of individual cells in tables, either based on their positions or their semantic meanings [14, 18, 20, 48]. On the contrary, fewer research effort is paid to how data is visually organized for presentation, which often includes principles of typography and annotations. While a large proportion of existing literature on table presentation draws upon design guidelines of enterprises and organizations [4, 25, 38], these guidelines often fail to account for the practical usages of designers [5, 41]. To address these issues, several researchers statistically analyze table structures using general metrics like average row number [19, 23], overlooking the intricate design choices and visual arrangements that contribute to their structures. Others classify tables using a few general terms such as *vertical*, *horizontal*, or *relational* to provide an overall abstraction [9, 32, 44]. Besides structures, some researchers also highlight the stylistic issues in tables, most of them focusing on cell styles [16, 30, 52] and only a few propose formatting rules applicable to the whole table [46, 47]. Recently, Bartram et al. [1] further examine the structural and stylistic patterns on a few use cases by interviewing 12 data workers. However, these methods merely touch upon a handful of general table features without in-depth analysis of each characteristic's detailed patterns. Moreover, without systematically surveying real-world tables, they are not comprehensive enough to encompass the wide variety of real-world tables. Our work fills this research gap by proposing a design space of plain tables based on a survey of over 2,500 tables.

2.2 Authoring plain tables

Recent years have witnessed the emergence of numerous tools and systems to facilitate table authoring. We classify these systems into three categories: content-driven, model-driven, and layout-driven.

Content-driven systems focus on obtaining accurate values for each cell or populating tables with relevant data. One line of research studies converting datasets into desired forms, or data wrangling [27]. For instance, tools like Wrangler [28], Trifacta [45] and Tableau Prep [43] and open-source libraries like Pandas [37] incorporate various data operations or functions for data transformation. The other line of research studies enriching tables with retrieved data. For example, given an input table, SmartTable [50] is able to suggest candidate rows and columns by referring to knowledge databases like Wikipedia. Similarly, Dong et al. [15] introduce a data selection process to populate the input table with data relevant to a given machine learning task. KTabulator [49] further eliminates the need for input tables and enables users to directly organize collected data from databases in tables.

By contrast, *model-driven* systems use tailored models to represent tables and are equipped with model-related interactions. One typical example is the modeling of data hierarchies, which is common in nested data structures such as JSON and XML. Gneiss [7] represents hierarchical data as trees and proposes a syntax for manipulating them. Similarly, HiTailor [33] introduces a declarative grammar for hierarchical data and studies transformation types on such data. In addition, Rigel [8] proposes a general grammar of tables that divides tables into three channels: *row*, *column* and *cell*. Entities embraced with data functions can be mapped to these channels to derive the result table.

On top of the previous systems, *layout-driven* systems further focus on the visual appearances of tables. Prototype system [2] and spreadsheet tools such as Microsoft Excel and Google Sheets provide comprehensive support on cell-based management including adjusting styles and spatial arrangements. However, since layout changes often involve multiple cells, it is inefficient to edit the cells when the data size grows [5]. To address the issue, Xtable [47] builds on the tree model of hierarchical data and provides eight formatting rules for styling. While Table Illustrator draws much inspiration from this work, the limited number of proposed rules is hardly expressive enough to cover the diverse space of real-world cases. Besides, it is also difficult for users to explore potential table layouts without tailored strategies from the system.

To summarize, existing table authoring systems lack efficient and expressive support for manipulating and exploring table layout and styles. Consequently, Table Illustrator aims at bridging this gap by allowing users to drag puzzles to construct tables in a fast and intuitive manner, and meanwhile providing a comprehensive set of layout and style options to help users efficiently elaborate and prototype their tables.

3 PILOT STUDY

To figure out the process of designing and authoring plain tables by practitioners and collect requirements for our interface, we conducted a pilot study following the design study methodology [39].

3.1 Study design

3.1.1 Participants. We recruited 10 participants (5 males and 5 females, aged 21-29 and 24.4 on average) for our interview by advertising on a campus forum and our social media. During the selection, we made sure participants had a diverse range of professional background and designed mainly for presentational purposes. Participants were from various domains including education (4), finance (3), medicine (1), administration (1) and data journalism (1). According to their claims, they had been working on table design for 3-9 years, with an average of 5.6 years.

3.1.2 Procedure. Before the formal study, we conducted a pilot interview with a senior manager from a local enterprise who worked with tables for 20 years to decide the initial study protocol. The protocol was then revised by three authors of this work through two rounds of discussion.

Participants were asked to prepare some tables that they independently authored before the interview session and remove sensitive information if necessary. Before the interview, they first completed a consent form and a demographic questionnaire about their background information. The interview sessions were conducted either online or in person and lasted for around 30 minutes. During the session, the interviewer raised questions about two themes:

- **Authoring workflow.** What is the workflow of authoring a plain table? If there are multiple ones, which one is chosen more often and why?
- **Design process.** What tools or software are used for table design? What classes of changes are involved when designing a table? What principles do users follow during the design process? What are the design considerations? What motivate users to attempt different designs and how do they conduct this process? What challenges or obstacles do they find in the design process? What perspectives of existing tools can be improved to better facilitate their work?

Throughout the interview, participants were encouraged to illustrate their points with the assistance of specific cases they provided. At the end of the interview, they were paid $15 per hour. The interviews were video-recorded and we transcribed the audio to perform a thematic analysis to derive the insights.

3.2 Interview results

3.2.1 The general workflows of table authoring. According to the participants' feedback, the authoring process can be broadly categorized into three tasks: data preparation, table construction, and table configuration. However, there was no consensus on the specific working styles, resulting in two distinct workflows.

In the first workflow, participants conducted the three tasks sequentially. Starting from datasets from various sources such as collaborators, software, or databases, they would then use data wrangling tools or scripts to transform the source data into a format that closely matched their desired results. The transformed data served as input for table authoring systems, where users performed basic aggregations and made adjustments to table styles. For example, P7, an experienced data analyst, stated a preference for completing most data transformations using Python. Afterwards, she would utilize Excel to add annotations such as measurement units and table titles, while modifying background colors and fonts. P8 followed a similar process, explaining that "*despite the power of Python, the unbearable plainness of the exported table compelled me to beautify it in Excel.*"

In contrast, the second workflow involved users who engaged in an iterative process of the three tasks. While some started with multiple existing tables, many began with implicit data sources like textual documents or images. The authoring process typically occurred in a separate workspace, where tables were gradually constructed through manual input while simultaneously extracting required entries from the source data. In this workflow, the table schema could be initially designed with headers filled before entering the records, but it was subject to alteration during the construction process, especially when new insights were discovered. Layout changes were also made at any point during the process. For example, P6 preferred to fill in the records after completing all necessary layout configurations, such as grid merges and styles. Similarly, P1 and P9 directly built tables using templates with predefined settings. By contrast, P2 and P3 followed a different approach, adjusting table styles after completing all the entries. P10 said she was used to altering the layout as long as she found it necessary.

At the end of both workflows, there existed a proofreading step where participants went over the result table and fixed potential errors. This process was conducted by either the participant or their collaborators. Besides, there was also a great variance in the effort involved, ranging from "*a glimpse*" (P6) to "*about one-sixth of the total time*" (P2).

3.2.2 The design process. Our insights on participants' design practices are as follows.

Tools. All participants had used Microsoft Excel for designing plain tables. They also sometimes used other software including Microsoft PowerPoint (6/10), Google Sheets (5/10), Microsoft Word (2/10), GraphPad (1/10), and Tableau (1/10).

Design practices. All participants claimed that they would add various styles to selected cells, rows, and columns, which include borders, fonts, backgrounds, textual aligns, and indents. Global changes such as adjusting the table size and setting conditional formatting were also conducted. Fewer cases involved changes to the table structure. For instance, P5 provided a case where he was asked to make a table for a poster: "*The records were originally vertically arranged and there were tens of rows. This exceeded the height limitation... And I ended up dividing the rows into groups and combining them horizontally through copy-and-paste.*" P1, P5 and P8 also mentioned appending new columns for aggregations, and reshaping the table by transposing or building a crosstab to find the best layout. Besides, there also existed practices for marginalia, such as adding table titles and other textual annotations (P2, P10).

Nearly all participants (8/10) reported the repetitive interactions. For example, P8 complained about merging the grids cell-by-cell in Excel: "*Some cells need to be merged, while others should not be, and some even require division. It is really arduous.*" In addition, some participants also mentioned the endless trivial adjustments. Drawing from his experience in data visualization, P3 said he preferred to have most configurations automatically done with minimum customization as in some visualization tools, as opposed to building from scratch. Similarly, P10 highlighted the significant number of interactions and advocated for a panel of shortcuts, where each button could replace multiple operations.

We further asked participants for their attitudes towards table templates as a potential solution. However, participants found the default templates offered by Microsoft Excel and PowerPoint useful in only a limited number of cases or altogether unhelpful, mostly due to low aesthetic appeal (4/10) and diversity (2/10). Additionally, some conveyed their frustration with the arduous task of searching for templates online, highlighting the substantial effort it demanded.

Design principles and considerations. Participants mentioned various thoughts during their design process. Four participants said they crafted their layouts by adhering to established rules or instructions from public standards (e.g., annual reports), their organizations, or leaders. Five participants utilized templates from table authoring software or online for table authoring. Additionally, four participants sought inspiration from multiple existing tables. Furthermore, we received numerous responses where participants relied on their intuition. Within this group, some individuals depended on personal habits or opted for a random design approach. Conversely, others approached the task with thoughtful consideration, taking into account various factors such as value features, usage, and semantics. For instance, P5 assigned distinct colors to cells containing positive and negative values, while utilizing different font weights for aggregations. Similarly, P1 categorized columns into semantical groups and applied diverse colors accordingly.

Design attempts. The interview revealed that table designing is highly iterative, with participants frequently experimenting various designs before reaching a final decision. While most of them explored designs by performing cell interactions in table authoring tools, alternative approaches were also observed. For instance, P9 was accustomed to sketching different designs on draft papers before implementing them digitally. P1, P7 and P8 preferred to write scripts to explore different table structures, although they still found it necessary to adjust styles in spreadsheet tools. Besides, many participants complained about the effort within design iterations when no external assistance is provided. "*My leader often provides suggestions for improving the table that I can hardly think of on my own. Without such guidance, it becomes challenging for me to determine, or even realize, the direction for refinement.*" P4 noted.

Meanwhile, there also existed participants who rarely tried different designs. P5 highlighted the significance of the intended audience and stated: "*The design depends on who will be viewing the table. For my colleagues, a few adjustments are sufficient. However, when tables are intended for official project reports or our customers, our team invests days or even weeks in meticulously refining the tables.*" Meanwhile, P2 acknowledged the time-consuming nature of table elaboration and mentioned, "*I only design the table carefully during my spare time... In most cases, I focus on basic enhancements such as bolding the headers to ensure moderate readability.*" Nevertheless, they all agreed on the importance of exploring various layouts and expressed a desire for targeted approaches. "*Unfortunately, I did not have the skills and time to try different designs though I really wanted to.*" P7 said.

3.3 Conclusion

Our interview revealed that participants adopted distinct workflows and tools in their design process, where they performed structural and stylistic changes to tables. They also had diverse design considerations, ranging from referring to external sources such as templates to thoughtfully taking table semantics into account. However, the absence of clear guidance, the iterative nature of the task, and the excessive reliance on repetitive and trivial interactions all hindered users' ability to efficiently prototype tables.

4 THE DESIGN SPACE OF PLAIN TABLES

4.1 Study protocol and space overview

To understand plain table designs in practice, we conducted a content survey on TableBank [34], a recently released dataset of over 300,000 tables collected from webpages and documents on the Internet, covering a diverse range of usages, domains, and designs. During the study, we first randomly sampled 3,000 tables from it due to the large dataset. Two graduate students majored in computer science were then recruited to filter the sampled dataset, removing invalid items according to the following rules:

- Items without a tabular shape.
- Items that lack key information for understanding. (e.g. tables with unintelligible codes or many missing values)

Table 1: An overview of the proposed design space of plain tables.

Structure			Style	
Hierarchy	**Facet**	**Marginalia**	**Visual**	**Semantical**
Group-by	Division	Key	Font	Headers and cells
With group-by	2 divisions	Pattern	Background	Hierarchy
Without group-by	3 divisions	Position	Border	Line parity
Cell merge	>=4 divisions	Nested		Selection
Merged	Cell merge	Aggregation		Aggregation
All values		Function		Conditional formatting
First-value only		Position		
Cellular parent		Label		
Spanned parent		Spacing		

- Items that are not intended for data presentation. (e.g. calendars, matrices)
- For consecutive items that share the same layout and only differ in data values, only one item will be retained.

This process resulted in a collection of 2,533 tables. We then began to explore the design space based on this filtered dataset. The specific process is as follows.

- **Build a preliminary framework.** Based on a survey of existing literature [1, 9, 19, 23, 30, 44, 47, 52], we determined two primary classes of the framework, namely *structure* and *style*. To determine the sub-classes, two authors of this paper reviewed 500 tables from the dataset, coded design patterns for each table, merged similar codes into categories, and discussed the results with 3 experienced visualization researchers (also co-authors of this paper). Based on their feedback, we eliminated some general patterns (table titles, footers, and context) that had been well discussed in prior work [31, 51], resulting in the basis framework.
- **Refine the framework iteratively with the dataset.** Based on the initial framework, the two graduate students reviewed the complete dataset in multiple rounds, checking for patterns aligned with the initial framework and collecting patterns that are not included. The framework was refined by two authors at the end of each round, until no new patterns emerged after three rounds. The authors then organized the patterns into the final design space.

An overview of the result design space is illustrated in Table 1. The design space consists of two primary dimensions: *structure*, which incorporates patterns regarding the spatial arrangement, and *style*, which shows factors affecting the visual appearance of tables. We discuss the detailed classification within these dimensions in Section 4.2 and 4.3 respectively. Moreover, we also observed a few tables that were inherently heterogeneous, consisting of various subtables with a diverse range of semantics, structures or styles. Therefore, we briefly discuss the patterns of combining subtables into an integral table in Section 4.4.

4.2 Structural patterns

We decompose the structural features of plain tables into three sub-classes: *Hierarchy*, *Facet*, and *Marginalia*. Quantitative results for each proposed sub-class have been provided in the appendix.

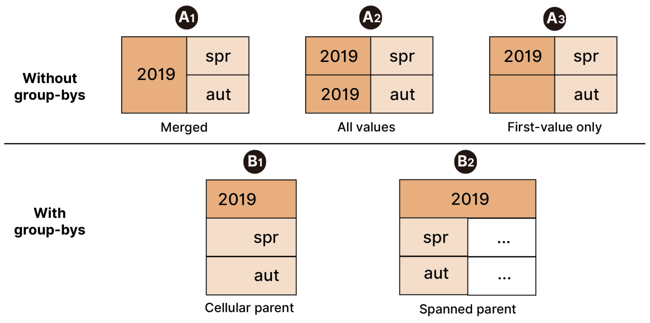


Figure 3: Visual arrangement patterns for *hierarchy*. We define *group-by* as whether the parent and child entity occupy separate columns, resulting in two classes of visual representations (A, B). Each class can be further divided based on whether identical adjacent values are merged, resulting in five sub-classes.

Hierarchy. It is common for data tables to exhibit hierarchical structures, such as the relationship between *years* and *seasons* in Figure 2 (A). In fact, for every two entities in table headers that show a parent-child relationship, there exist multiple visual arrangements that can be classified based on the following two criteria:

- **Group-by:** When *group-by* is not applied, the parent entity and the child entity occupy separate columns (or rows for entities in the column header). By contrast, they are merged into a single column when *group-by* applies. Figure 3 (A) and (B) illustrate examples of both cases.
- **Cell merge:** The cells of the parent entity can be merged with adjacent grids. For instance, when *group-by* is not applied, one can choose to merge all cells for the parent entity (Figure 3 (A1)). Alternatively, they can leave all cells or the first one filled with values (Figure 3 (A2, A3)). On the other hand, in tables with *group-bys*, the cells of the parent entity typically have the same width as the header (or height for entities in column headers), or even occupying the whole line in a few cases, as shown in Figure 3 (B1, B2).

Facet. Another frequently observed pattern is dividing the values of a header entity and their subsequent records into sections and reorganizing them by juxtaposition, often used for accommodating

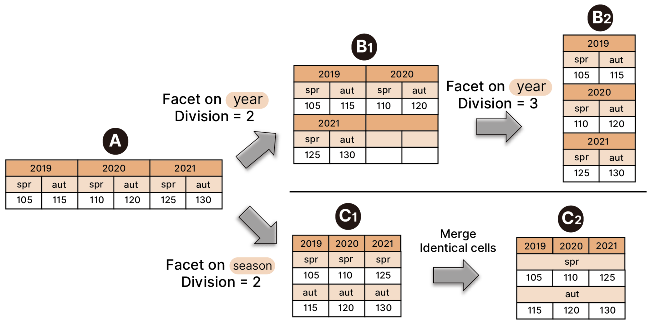


Figure 4: Visual arrangement patterns for *facet*, which refers to dividing all values of an entity into *divisions* and juxtaposing them in another direction. For instance, performing *facet* operations on the entity *year* and *season* of the table (A) can lead to tables (B1, B2) and (C1). One can also merge identical adjacent cells afterwards (C2).

to different sizes or comparing specific data. Specifically, we further classify this *facet* pattern based on the following factors:

- **Division:** The division represents the number of parts that the faceted entity is divided into, with 2 divisions and 3 divisions being the most commonly employed. For instance, as shown in Figure 4, given the regular arrangement (A) of a table, (B1) and (B2) provide examples for exerting *facet* operations on *year* with 2 and 3 divisions, respectively.
- **Cell merge:** When faceting entities in hierarchical tables, chances are that cells with identical values are adjacent in the results and can be potentially merged. For instance, after dividing *seasons* into 2 groups ("*spr*" and "*aut*") and arranging them vertically (Figure 4 (C1)), one can merge the adjacent *seasons*, as shown in Figure 4 (C2).

Marginalia. The *marginalia* refers to auxiliary information in the main table to enhance readability, including aggregations, keys and spacings.

- **Aggregation:** Aggregation functions are widely adopted and can be categorized by the following dimensions. (a) *Function*: According to our survey, more than 80% of these functions are summations, with the remaining cases using subtraction and average. (b) *Position*: In most cases, aggregation functions occupy a row or column exclusively, as shown in Figure 5 (A2), while inline aggregations (Figure 5 (A1)) are also used in a few cases. (c) *Label*: A label that is often named *Total* or *Grand Total* is optionally positioned in the row or column header to indicate aggregations, except for inline ones. Moreover, the label can have the same width as the header in some cases.
- **Key:** Adding keys besides entity values is widely used to improve the table tidiness, often adopted in the contents of official documents. By observing tables with keys, three properties have been identified. (a) *Pattern*: There are various patterns of keys, such as numbers, letters, and Roman numerals. (b) *Position*: The keys can be either embedded in target cells or positioned in separate rows or columns. (c) *Nesting*: For hierarchical tables, keys can be in a nested form, such as *"1, 1.1, 1.1.1, ..."*. For instance, in Figure 5 (B1), numerical keys are used to order both *year* and *season*, placed to the left of target cells. Meanwhile, Figure 5 (B2) illustrate examples where keys are in Roman numerals and letters are used in a nested style and embedded in the target cells.
- **Spacing:** Some tables contain a few empty rows or columns that visually divide the table into groups, which is a useful way to highlight the semantic categories within data or derive a sparse layout. For example, Figure 5 (C) shows the results of inserting empty lines between the values of *year*.

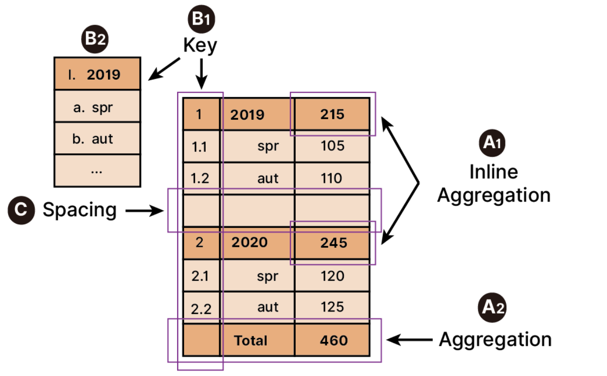


Figure 5: Visual arrangement patterns for *marginalia*, including aggregations (A1, A2), keys (B1, B2) and spacing (C).

4.3 Stylistic patterns

The design of table styles is admittedly ad-hoc and diverse, given that visual styles of cells have been well supported by spreadsheet tools and have also been discussed by existing literature [30, 52]. Therefore, rather than merely visual styles, we mainly focus on the semantics encoded by the cell styles in our framework, such as the line parity and hierarchy. We believe our exploration on the relationship between semantics and styles will lay foundation for more effective design recommendation strategies.

Specifically, we consider three classes of cell styles in our framework: font (family, weight, size, color, indent, and format), background color, and border (position and style). Notably, we exclude general styles like cell height and width since they are too universally applicable to be identified as semantic encodings. We have also summarized the following semantic elements that are encoded by the cell styles:

- **Headers and cells:** Identical styles can be applied to all cells in the row header, the column header, or the main body. One can also adopt distinct styles between channels to improve readability.
- **Hierarchy:** In hierarchical tables, cell styles can be used to represent the hierarchical structure of the data. One common practice is using different styles to indicate parent-child relationships, such as in the row header of Figure 2 (B2).
- **Line parity:** Cell styles can be modified based on the parity of the row or column ID. This is often done by setting alternating background colors for rows or columns to improve visual distinction and readability.

- **Selection:** Cell styles can be used to highlight specific rows or columns that are of particular importance.
- **Aggregation:** Cells for representing aggregations often have distinctive styles. For instance, it is common to use bold fonts in the summary row. Besides, difference of cell styles can also exist between the titles and the aggregated values in some cases.
- **Conditional formatting:** Cell styles are allowed to be dynamically adjusted based on specific conditions or rules. For example, placeholder hyphens can be used for empty cells.

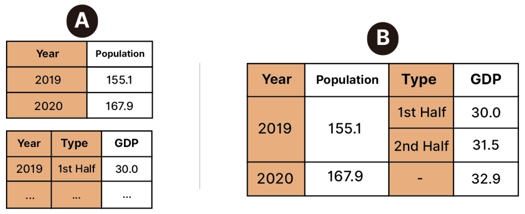
  We have statistically counted the occurrences of each style under different semantics (see the appendix for details), leading to the following findings. Overall, we observe that the styles are mostly used in column headers, partly due to the widespread employment of relational tables. Meanwhile, while some style types such as background colors are used for various encodings, most style types have their targeted usages. For instance, textual formats are predominantly used in conditional formatting and are largely ad-hoc, such as the inclusion of special symbols like brackets and digit separators. In addition, the range of design choices also varies among the encodings. For example, the styles of table headers and their hierarchies are diversely distributed, while the parity of row and column ID are typically encoded by background colors. Finally, there exist great differences between row headers and column headers in terms of the choice and frequency of style usages.

4.4 Table combination patterns

While the majority of tables in our dataset can be effectively represented using structure and stylistic patterns, there exist a few tables that lack a cohesive design, making it challenging to abstract them based on rationales for common tables. To address this issue, we propose representing these tables by combining multiple sub-tables, each one coverable by a set of structure and stylistic patterns. We thus briefly discuss the categorized dimensions of combination patterns and provide the quantitative survey results in the appendix.

- **Table arrangement:** When arranging the sub-tables spatially on a canvas, two common properties should be considered. (a) *Spacing:* the sub-tables can be placed either with a spacing or in a concatenated manner. (b) *Direction:* the sub-tables can be placed either vertically or horizontally.
- **Cell alignment:** Table combinations often include cell alignments due to their grid-based nature. These alignments can be designed either semantically or structurally, which indicates aligning by corresponding data records or adjusting the width and height of cells to equalize the sizes of sub-tables.
- **Omission of headers:** It is also observed that the combined sub-tables share some labels in a few positions, which can be omitted except for the first appearance to avoid duplications.

For example, Figure 6 (A) shows an example of vertically combining two tables side by side with a structural alignment of cells, stretching the tables to equalize their widths. By constrast, Figure 6 (B) horizontally concatenates the two tables, aligning them semantically and omitting the *years* in the header of the right table.



Year	Population
2019	155.1
2020	167.9

Year	Type	GDP
2019	1st Half	30.0
...	...	...

Year	Population	Type	GDP
2019	155.1	1st Half	30.0
		2nd Half	31.5
2020	167.9	-	32.9

Figure 6: Visual arrangement patterns for table combinations. Table (A) shows the results of vertically combining two tables with spacings and aligning them structurally by extending to the same width. By contrast, table (B) shows two tables are seamlessly combined horizontally and aligned by corresponding records, while also omitting the *years* in the header of the right table.

5 DESIGN GOALS

Based on the insights from the pilot interview, we conclude that the table authoring workflow in spreadsheet tools often involve trivial and repetitive adjustments, which is time-consuming and prevents fast table prototyping. This can get worse given that elaborate plain tables have a diverse range of layouts and styles according to our content survey, and existing tools lack sufficient support for users to swiftly explore or iterate in this large design space. Prior research [1, 5] suggests that spreadsheet tools can become less effective when handling semantical operations that involve table schemas or data entities due to their primary focus on cell-based interactions. We further hypothesize that one key reason for these limitations is the typical GUI of spreadsheet tools, which is based on fixed grids and may result in cells with semantic relevance (such as cells associated with the same data entity) not being spatially adjacent, particularly in tables with hierarchies, forcing users to repeat operations for each cell. By allowing users to operate on all relevant cells simultaneously, we believe that interactions can be reduced and design iterations can be conducted in a more efficient and systematic manner. This motivates us to explore the possibility of breaking away from the rigid, classic grid-based spreadsheet paradigm and offering an alternative, more flexible, and incremental approach to table authoring. We have therefore concluded the following design goals.

D1: Support multiple workflows. Our user interview has revealed multiple workflows adopted in users' practice. The system should be able to accommodate these requirements, specifically, exploratively constructing tables from the base data and directly starting from existing tables.

D2: Group semantically related cells. Cells that share similar semantics, such as values of the same entity or grouped aggregations, should be visualized and operated as an atomic graphical object, laying a foundation for semantical interactions that facilitate table prototyping.

D3: Support diverse semantical interactions. Based on insights from the interview, the system should assist users in two crucial subtasks of the authoring workflow: table construction and table configuration. Interactions should be developed to facilitate these subtasks. For table construction, cells should be seamlessly combined or separated through intuitive drag-and-drops to enhance the efficiency of prototyping and iterative design of tables. For table configuration, as informed by the content survey, users should be able to configure cells with tailored parameters, enabling users to make precise structural and stylistic adjustments to the table.

D4: Enable table overview. Design tools commonly provide users with the ability to zoom in or out, allowing them to switch between an overview and detailed view of their creations. This feature is equally essential in our system, given that an entity may have numerous values when dealing with huge datasets. Without a comprehensive overview of the entire table, users would face challenges in comprehending the overall structure of their constructed table. Furthermore, they would be required to scroll through extensive lengths of data to locate the boundaries of cell collections, hindering their ability to perform combinations.

D5: Visualize table configuration parameters. Our interview suggests that some users find existing spreadsheet tools expressive yet intricate due to the diverse components. While we have designed a compact list of structural and stylistic parameters based on our design space, learning them can still take non-trivial efforts. To help users understand these parameters, the potential outcome of changing these parameters should be visualized to enable users to efficiently iterate through different design variations.

D6: Support both local and global table styling. We conclude from our content survey that elaborate plain tables often involve multiple stylistic features, and some interviewees find manipulating individual parameters to be time-consuming. To address this, the proposed tool should support both local styling, changing the styles of a group of cells that belong to the same data concept including data entities or functions, and global styling, applying frequently-used style templates to different semantic table elements, such as hierarchies and table headers.

Inspired by the jigsaw puzzle metaphor, we envision visualizing a group of relevant cells as a piece of puzzle, leading us to design the Table Illustrator interface with puzzles as the primary interaction unit. With this visual hint, users are prompted to combine puzzles on a canvas for table construction, which is demonstrated to be user-friendly and provides a novel, flexible, and incremental paradigm of table authoring beyond fixed grids.

6 TABLE ILLUSTRATOR

In this section, we introduce Table Illustrator, an interactive system for rapidly prototyping plain tables. The system is currently implemented as a web application based on JavaScript. Figure 7 provides an overview of Table Illustrator. The interface incorporates three views: a data view (A), a canvas view (B), and a configuration view (C). The data view contains a data panel (A1) for importing and displaying the source data, and an entity panel (A2) visualizing the entities. In the canvas view, users can construct target tables through puzzle-based interactions with the assistance of a toolbar (B1). They can also switch between the design mode and the preview mode (B2). Finally, the configuration view is meant for manipulating parameters of table layout and styles.

We then discuss the system workflow as follows. In Section 6.1, we explain the supported data types and how they can be imported into the system. Section 6.2 introduces the interactive process through which users can construct the target tables. Meanwhile, users are also allowed to configure the table layouts in parallel with the table construction, which is discussed in Section 6.3. We finally provide necessary implementation details in Section 6.4.

6.1 Importing datasets and tables

Based on the design goal [D1], Table Illustrator supports both the input of a dataset and an existing table. In the *dataset-driven* scenario, users start by clicking on the *import dataset* button in the data panel (Figure 7 (A1)) to import a dataset (formatted in JSON or CSV) to the system. An overview of the imported dataset and incorporated entities will then be respectively shown in the data and entity panel (Figure 7 (A1, A2)). In the *table-driven* scenario, users initially import an existing table by clicking on the *import table* button in the data panel (Figure 7 (A1)), which will later be parsed and transformed into puzzles based on its organization. We adopt a heuristic algorithm to parse the input table. Currently, it supports two most common table types as input: pivot tables and vertical tables. Pivot tables refer to crosstabs where row and column headers are non-empty and the value of other cells correspond to the respective headers. By contrast, vertical tables are arranged in a way that the top few values in each column serve as the entity names for the subsequent values in that column, i.e., each column represents a different attribute or category. Before parsing, users have to select the table type in the pop-up confirm dialog of the *import table* button. Details of the parsing algorithm can be found in the Appendix.

6.2 Constructing the target table

Having imported the data, users are allowed to construct the target table in the canvas view. Note that when an existing table is imported, the parsed table will be shown on the canvas. Users can directly work on it instead of building from scratch in the dataset-driven scenario. In the following discussion, we only explain the interaction process of building tables from scratch, covering the available interactions of both scenarios. We assume the raw dataset is a relational table including four entities: *continent*, *country*, *year*, and *sales*, as illustrated in the data panel.

Table Illustrator enables users to construct the target table through intuitive drag-and-drop interactions [D3]. Specifically, users can start from creating puzzles on the canvas, which consist of three types: data entities, entity names, and aggregation functions [D2].

- To create a puzzle of a data entity, users can drag the entity (Figure 7 (A3)) to the canvas (Figure 7 (B)) to create a puzzle with its cells containing the values of the entity. For instance, when dragging the entity *continent* to the canvas, a puzzle is generated at the drop position (Figure 7 (B3)), with its cells arranged in a vertical direction by default. Therefore, it can be also viewed as a basic table with only the generated puzzle in the row header. Users can click on the rotate icon at the top right of the table (Figure 7 (B4)) to transpose it. Besides, only the first two values of the puzzle are displayed to provide an overview [D4]. To view all values, users can either click on the expansion icon at the bottom-left of the puzzle (Figure 7 (B5)), or switch to the preview mode to show

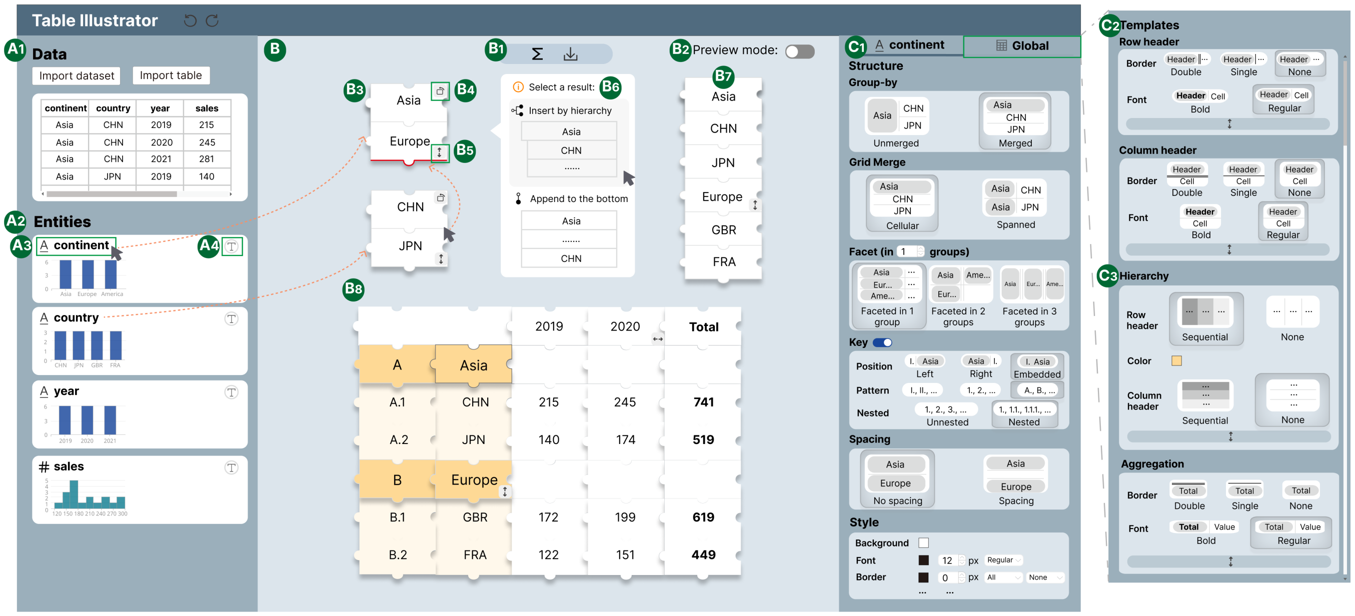


Figure 7: The Table Illustrator interface. The data view (A) consists of a data panel (A1) for importing data, and an entity panel (A2) where users can drag the values (A3) or the title (A4) of each entity to the canvas view (B) to create puzzles (B3). Users can rotate the puzzle (B4) for transposion, expand the puzzle to see the complete data values (B5), or combine the puzzles to construct tables (B6, B7, B8). The canvas view also contains a toolbar (B1) for aggregations and export, and a switch to preview mode (B2). Besides, a list of puzzle-specific parameters (C1) and global templates (C2) are provided in the configuration view to elaborate the table (C). For instance, one can set sequential background colors for header entities in the panel of *Hierarchy* (C3).

the complete result table with all collapsed puzzles expanded at a time (Figure 7 (B2)).

- To create a puzzle of an entity name, users can drag the title icon (Figure 7 (A4)) of an entity to generate a puzzle that include a single cell of the entity title (*continent* in this case).
- Additionally, for aggregation puzzles, users can drag the sigma icon in the toolbar (Figure 7 (B1)) to create a puzzle with a cell named "*Total*".

Puzzles can be freely positioned on the canvas. Note that there is no limit of puzzle numbers in the canvas, i.e. users can manipulate multiple tables in parallel.

With the puzzle shape as a visual cue, users can drag entities, entity titles, functions, or existing puzzles to the four edges of other puzzles to combine them seamlessly [D3]. The specific edge onto which users drop puzzles depends on the relative spatial position where they want to place the puzzle values. Users can also drag a puzzle to any available position on the canvas to separate it from combined puzzles. This allows users to construct tables of different structures in an intuitive and efficient way. Notably, dragging to different edges of header puzzles can yield distinct outcomes, as exemplified in Figure 8. Since there are multiple possible results when dragging to the top, bottom, and right edge, a confirm window displaying snapshots of potential result tables will be shown for selection when the mouse is released. For instance, when dragging the puzzle of *countries* to the bottom edge of the puzzle of *continents*, users are asked to select between two candidate tables (Figure 7 (B6)). However, we have decided not to implement selections when dragging to puzzles in the cell channel to avoid an overwhelming number of possible results. Consequently, it is only permitted to drag into a puzzle in the cell channel when it is empty, or drag to the top (or left) edge when it does not have a corresponding puzzle in the column (or row) header. In such cases, the source puzzle will be inserted into the dropped puzzle in the former scenario, or inserted into the corresponding header in the latter one.

Additionally, the system also supports some general interactions for usability. For example, when users right click on a puzzle, a menu will be shown with options including deleting the puzzle, filtering, sorting, and editing the data values of the puzzle. Users can also undo and redo their operations by clicking on icons at the system header, or click on the export icon in the toolbar (Figure 7 (B1)) to download the table that the selected puzzle belongs to as an Excel spreadsheet file (.xlsx).

6.3 Adjusting the table layout

Alongside the table construction, Table Illustrator allows users to make real-time changes to structural and stylistic configurations of puzzles in the configuration view (Figure 7 (C)) [D3]. Users can start by clicking on an arbitrary cell to be altered. Afterwards, all cells that belong to the same puzzle as the clicked cell will be highlighted, indicating the scope of the following changes. Meanwhile, the available structural and stylistic parameters will be displayed (Figure 7 (C1)). Adjusting these parameters will lead to changes to the corresponding fields for the selected puzzle in the table specification. Specially, to eliminate the learning barrier of the structural

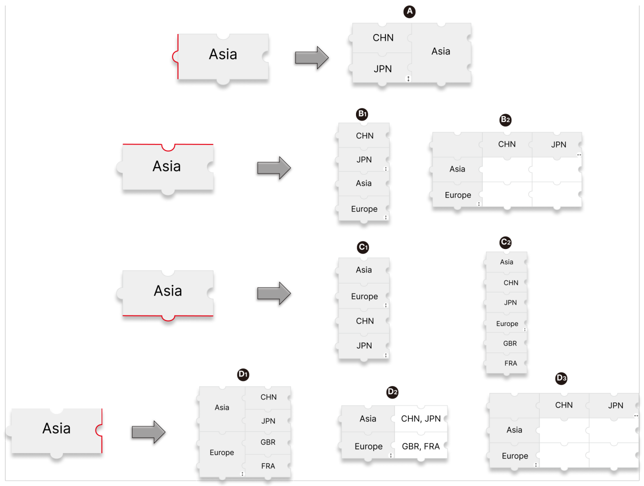


Figure 8: The possible results after dragging the puzzle of the entity *country* to the left (A), top (B1-B2), bottom (C1-C2), and right (D1-D3) edges of another row header puzzle that correspond to the entity *continent*. Results for dragging to puzzles in the column header can be inferred similarly.

parameters, we have designed a glyph for each candidate option of the structural parameters, showing a snapshot of the result table after applying the change [D5].

Furthermore, while the puzzle parameters primarily facilitate local changes, Table Illustrator complements this by supporting adjusting styles for global semantic elements, enabling simultaneously changing multiple puzzles to further accelerate the design process [D6]. Specifically, in the *global* tab of the configuration view (Figure 7 (C2)), we have designed for each semantic element several commonly used templates based on the most commonly used styles in our context survey. For instance, one can easily set hierarchical colors for the row header by simply clicking on the corresponding template glyph in the "*Hierarchy*" panel (Figure 7 (C3)). The semantic elements are generally ordered by frequency based on our survey, with relevant ones placed together. Moreover, while the interface only displays the common templates by default, users can also click on the expansion button at the bottom of each panel to show the complete detailed style parameters for customization.

6.4 Implementation

Table Illustrator provides puzzle-based interactions with various configurations. To systematically implement these features, we developed the system following a formal approach. Specifically, we designed a declarative grammar of plain tables based on the design space described in Section 4, and we provide its details in the Appendix. This technical decision was also informed by the requirement of facilitating table-puzzles parsing in the table-driven scenario [D1], where the user-imported table can be parsed to grammatical specifications before being transformed into puzzles. To achieve the puzzle-based interactions, they will be transformed into changes to grammatical specifications of tables. Afterwards, a calculation module will translate the specifications into the result table to be rendered.

7 USAGE SCENARIO

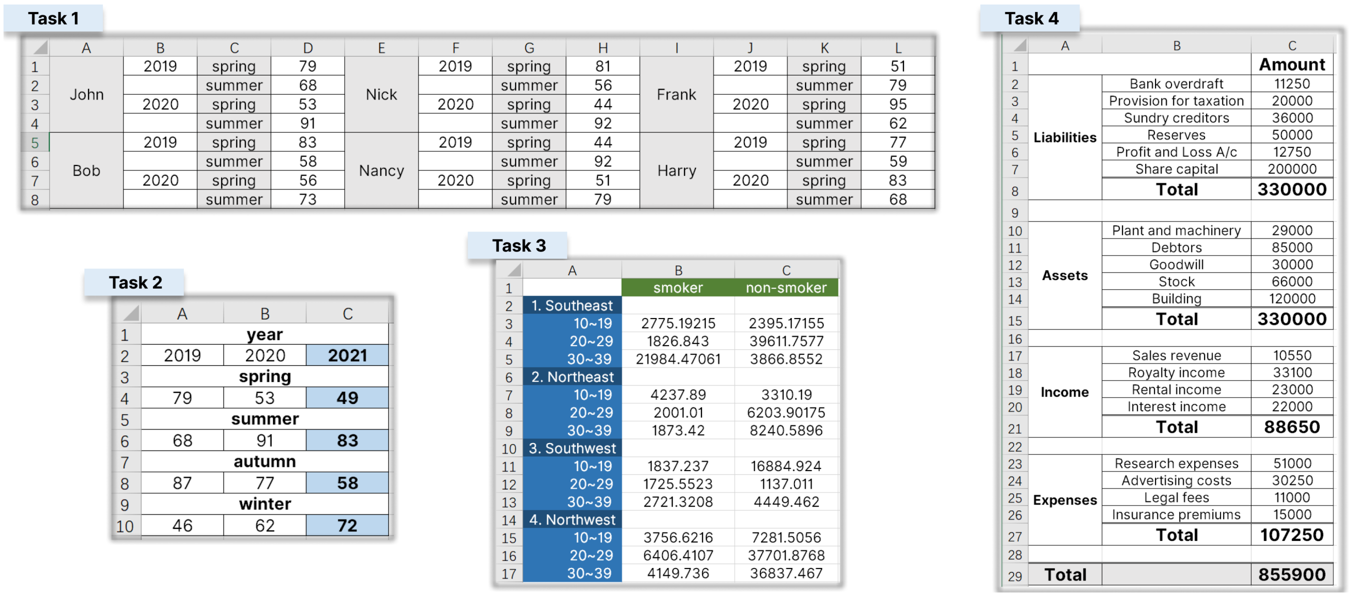
Assume an enterprise employee Alice is going to construct a table for her financial report. The input dataset is a relational table containing four entities: *continent*, *country*, *year*, and sales, as shown in Figure 7 (A1). Her goal is to show the annual and total sales for various regions. She initially drags the entities *continent* from the entity panel (Figure 7 (A3)) into the canvas (Figure 7 (B3)). Similarly, she drags the entity *country* to the canvas, and then drags the created puzzle to the bottom of the *continent* puzzle. A selection window is created (Figure 7 (B6)), prompting Alice to confirm the candidate results. Opting for the first option, she combines the puzzles hierarchically, resulting in the table shown in Figure 7 (B7).

To display the annual and total sales, Alice prefers to construct a crosstab with aggregations, as shown in Figure 7 (B8). The detailed process is illustrated in Figure 1. First, she creates a puzzle for the *year* entity by dragging it onto the canvas. Since the *years* will be arranged horizontally, she transposes the puzzle by clicking on the rotate button in the upper right corner. To position the years at the upper right of the existing table, she drags the puzzle to the top of the *Asia* cell (alternatively, dragging it to the right is also acceptable), selecting to arrange the puzzles as a crosstab. Simultaneously, she adds the *sales* entity to the right of *CHN*, filling the body cells with sales values and acquires the table in Figure 1 (B). Alice then proceeds to append a column for the total sales by dragging the *sigma* icon to the right of *2020*, leading to the table in Figure 1 (C). The *total* column is left empty by default. To fill this column, she drags the the entity *sales* again into it to display the total sales.

While the table meets Alice's goals, she finds it too plain to present to her leaders. To enhance its visual appeal, she adjusts the table by configuring puzzle and global parameters. Specifically, she adds indexes of different patterns to puzzles of *continents* and *years*, bolds the text for the *total* column, and sets hierarchical background colors for the row header. Figure 1 (D) shows the polished table.

8 EVALUATION

To assess our proposed system, we conducted two within-subject user studies comparing it to Microsoft Excel (professional plus 2019 version with default configurations), a widely used table authoring tool among practitioners. While visualization tools like Tableau are capable of providing dataset overviews and visualizing the transformation results, they fall short in accommodating a variety of table layouts and styles, such as the tables in Figure 2. By contrast, Excel is generally friendly to average users and also offers a remarkable level of flexibility in table authoring through versatile interactions, making it a valid candidate for comparison with our system. Our hypothesis posits that given a few tasks related to rapid table prototyping, users will spend less time and fewer interactions while perceiving lower cognitive load when using Table Illustrator compared to Excel. The only difference between the two studies is whether users were provided detailed tutorial and sufficient training on Excel's advanced features such as Pivot Table, leading to different usage patterns.



Task 1

	A	B	C	D	E	F	G	H	I	J	K	L
1	John	2019	spring	79	Nick	2019	spring	81	Frank	2019	spring	51
2			summer	68			summer	56			summer	79
3		2020	spring	53		2020	spring	44		2020	spring	95
4			summer	91			summer	92			summer	62
5	Bob	2019	spring	83	Nancy	2019	spring	44	Harry	2019	spring	77
6			summer	58			summer	92			summer	59
7		2020	spring	56		2020	spring	51		2020	spring	83
8			summer	73			summer	79			summer	68

Task 2

	A	B	C
1	year		
2	2019	2020	**2021**
3	spring		
4	79	53	**49**
5	summer		
6	68	91	**83**
7	autumn		
8	87	77	**58**
9	winter		
10	46	62	**72**

Task 3

	A	B	C
1		smoker	non-smoker
2	1. Southeast		
3	10~19	2775.19215	2395.17155
4	20~29	1826.843	39611.7577
5	30~39	21984.47061	3866.8552
6	2. Northeast		
7	10~19	4237.89	3310.19
8	20~29	2001.01	6203.90175
9	30~39	1873.42	8240.5896
10	3. Southwest		
11	10~19	1837.237	16884.924
12	20~29	1725.5523	1137.011
13	30~39	2721.3208	4449.462
14	4. Northwest		
15	10~19	3756.6216	7281.5056
16	20~29	6406.4107	37701.8768
17	30~39	4149.736	36837.467

Task 4

	A	B	C
1			**Amount**
2	**Liabilities**	Bank overdraft	11250
3		Provision for taxation	20000
4		Sundry creditors	36000
5		Reserves	50000
6		Profit and Loss A/c	12750
7		Share capital	200000
8		**Total**	**330000**
9			
10	**Assets**	Plant and machinery	29000
11		Debtors	85000
12		Goodwill	30000
13		Stock	66000
14		Building	120000
15		**Total**	**330000**
16			
17	**Income**	Sales revenue	10550
18		Royalty income	33100
19		Rental income	23000
20		Interest income	22000
21		**Total**	**88650**
22			
23	**Expenses**	Research expenses	51000
24		Advertising costs	30250
25		Legal fees	11000
26		Insurance premiums	15000
27		**Total**	**107250**
28			
29	**Total**		**855900**

Figure 9: Screenshots of the output tables for the tasks in the user study.

8.1 Study A: A comparative study with Excel's basic features

8.1.1 *Study design.*

Participants. Altogether, 15 participants (P1-P15, 5 females, 10 males, aged 24.3 on average) were recruited, including 3 undergraduate, 1 master, and 11 Ph. D students. All participants indicated they had used Excel for creating tables with average proficiency (M=3.07/5, SD=0.96). In addition, some of them also had experience of using Microsoft PowerPoint (11), Google Sheets (8), and Tableau (4) in table authoring. Participants also reported a moderate level of skill in making tables (M=3.73/5, SD=0.96).

Tasks. We designed four tasks for the study, each comprising a raw data table and a corresponding target table. To minimize cognitive load and save time, we ensured that the raw datasets were presented as relational tables with 16-24 records and 3-4 entities. Our task design encompassed three structural classes in our design space (*hierarchy*, *facet*, and *marginalia*), along with most sub-classes. Additionally, the tasks covered over half of the visual styles and semantic elements within the design space. Furthermore, we made sure all system interactions were included, such as basic drag-and-drop operations, parameter configuration, aggregation, and cell edits or filters. Figure 9 has shown screenshots of the output tables we designed for the four tasks. The tasks were arranged in ascending order of complexity, with the expected number of interactions ranging from approximately 20 to 100. Besides, we based the task design on common table authoring scenarios, drawing from our experience. These scenarios included table reshaping (T1), item comparison (T2), crosstab construction (T3), and categorized subtotal calculation (T4).

Procedure. The study was conducted in person. Following an introduction to the study background and instructions, participants were required to complete a pre-study questionnaire providing their basic information. Subsequently, they were assigned four table authoring tasks to be completed using both Table Illustrator and Excel, with the order determined by the parity of their ID. Before commencing the Table Illustrator session, participants received a 7-minute tutorial on system usage, followed by an additional 3-minute warm-up period to familiarize themselves with the interface. However, tutorials were not provided during the Excel session, as we had made sure they possessed basic knowledge including editing, styling, and performing basic calculations like aggregation and sorting. Nonetheless, participants were given 3 minutes to freely use Excel and recall its usage. Note that this duration could be adequate, as no participant required additional practice time during the session. Subsequently, participants were instructed to sequentially complete the four tasks, each within 10 minutes. They were informed that the tables they created should match the given output exactly, except that minor differences in colors and font sizes were deemed acceptable. Additionally, participants were not permitted to directly copy values from the given output table, which was merely for reference. There were no additional requirements regarding the authoring process, i.e., participants could either choose to directly edit the table or use advanced features such as pivot tables and subtotal tools in Excel to complete the tasks. A 2-minute break was provided between the two sessions. Upon completion of the sessions, participants were asked to fill out two post-study questionnaires and provide feedback through a survey. The overall duration of the study was approximately 70 minutes, and participants were compensated $15 each.

Measures. We video-recorded the interaction process of participants and quantitatively calculated for each task the overall time cost and the number of mouse clicks. Their interaction patterns were also coded and analyzed. In addition, we assessed the workload of participants for the two systems by adopting two sets of NASA-TLX [22] questions (rated on a 7-point Likert Scale) respectively in the post-questionnaires.

8.1.2 Quantitative results.

Task completion number. Participants succeeded in completing the tasks using the two systems except for two failure attempts: P3 failed on T2 when using Table Illustrator, and P9 failed on T3 when using Excel. While both failures were due to time out, we observed that P3 had not considered changing the *facet* parameter due to unfamiliarity with the system. Nevertheless, when told the correct operations after the attempt, she said she could really understand the method and would not hesitate to adopt it if given a second chance. On the other hand, P9 struggled with using Excel and came across numerous unexpected results. For example, he chose to build the target table right beside the input, and the table often accidentally got changed when performing filters on the input.

Time cost and mouse clicks. During our analysis of the task completion time and mouse clicks in the study results, we observed a few outliers that deviated from the mean by more than 3 standard deviations. We handled these outliers by employing three approaches: a) directly trimming, b) outlier replacement, and c) applying logarithmic transformations to all data [36]. These methods aimed to ensure the result completion time and mouse clicks associated with each task and system were normally distributed ($p > 0.05$, Shapiro–Wilk test) and thus suitable for parametric tests. The analysis results for the three approaches yielded similar conclusions. Specifically, the results presented as follows are based on the first approach, where 5 out of 120 pairs of data points (15 users × 4 tasks × 2 metrics) were trimmed for both systems. Results for other approaches can be found in the Appendix.

The mean and standard deviation of the time cost and mouse clicks for two systems are presented in Figure 10 and Table 2. We test our hypothesis by performing paired student's t-test on the data. Note that 4 pairs of data points associated with the two failure attempts were not included in the tests. We conclude from the results that users spent significantly less time when using Table Illustrator than Excel in T1 ($t(12) = -3.59, p < 0.05$), T3 ($t(13) = -9.89, p < 0.05$), and T4 ($t(14) = -3.29, p < 0.05$). There were also fewer mouse clicks for Table Illustrator compared to Excel in T1 ($t(11) = -11.52, p < 0.05$), T3 ($t(13) = -11.07, p < 0.05$), and T4 ($t(14) = -7.74, p < 0.05$). An interesting finding is that in T2, our hypothesis that participants spent less time in Table Illustrator than Excel is rejected ($t(13) = 4.12, p > 0.99$), and so is the same hypothesis for mouse clicks ($t(13) = 0.13, p = 0.55$). We explain that T2 required users to have a good understanding of the *facet* parameter. Accordingly, most participants spent the first few minutes finding the right parameter through trial-and-error. By contrast, direct editing in Excel was more straightforward for this task since the target table only contained fewer than 20 cells.

Workload. In general, the median of the overall score was 14 for Table Illustrator and 28 for Excel, supporting the hypothesis that users perceived less burden when using Table Illustrator. Figure 11 shows participants' detailed ratings on their workload. We performed a Wilcoxon signed rank test on the results and observed significant differences in terms of physical demand

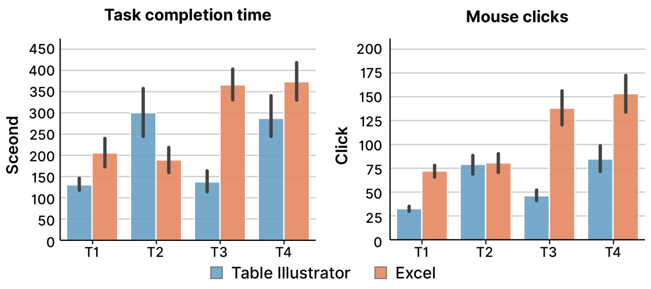


Figure 10: Bar charts of participants' time cost and mouse clicks in Study A for the four tasks (T1-T4) using Table Illustrator and Excel. Error bars indicate 95% confidence intervals.

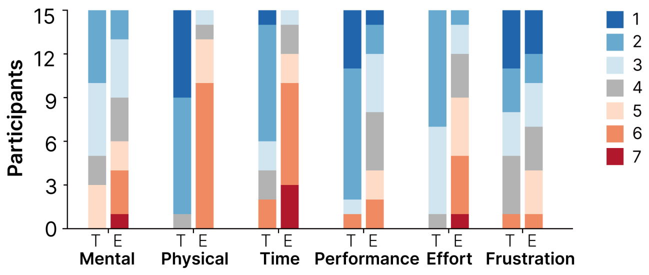


Figure 11: Participants' ratings on their workload for Table Illustrator (T) and Excel (E) in Study A, using a 7-point Likert scale based on NASA-TLX [22]. A lower rating indicates a lower level of perceived workload or a better level of performance.

($Z = -3.41, p < 0.05$), time ($Z = -3.06, p < 0.05$), performance ($Z = -2.25, p < 0.05$), and effort ($Z = -3.19, p < 0.05$). No significant differences were found in mental demand ($Z = -1.52, p = 0.06$) and frustration ($Z = -0.96, p = 0.17$). According to their report, this improvement could be mostly attributed to the reduction in repetitive interactions compared to Excel and the low learning curve brought by the puzzle metaphor and drag-and-drop interactions.

8.1.3 User feedback.

Feedback on the learning curve. While most participants agreed that Table Illustrator was overall easy to learn and use, some also mentioned the cell editing interactions in Excel were also intuitive and more familiar to people. Nevertheless, many participants said they were more willing to use Table Illustrator after a longer time of usage: "*It is super efficient to use the system as long as you understand how it works.* (P2)" P3 further said the system has a low "*reuse cost*", indicating she would have full confidence in future reuses. In addition, since Excel incorporates various features and components, we informed users about several advanced features in Excel that could efficiently solve some tasks (e.g., the tools in the *formulas* and *data* panel) and queried their attitudes. We concluded that participants appreciated the lightweightness of Table Illustrator configurations and perceived them as more natural. For instance, P8 said, "*Even if I found it somewhat difficult to understand some parameters, such as group-by and facet, it was easy to try them one-by-one through mouse clicks... In Excel, you had to search for tutorials as*

		T1		T2		T3		T4	
		μ	σ	μ	σ	μ	σ	μ	σ
Table Illustrator	Time	130.4	27.88	300.2	110.56	137.3	50.38	287.0	103.28
	Mouse clicks	32.6	4.78	79.1	20.55	46.2	11.08	84.7	27.59
Excel	Time	205.5	70.03	189.2	64.52	365.9	77.56	373.2	97.36
	Mouse clicks	72.1	11.10	80.7	20.59	138.1	34.38	153.3	39.40

Table 2: Mean and standard derivation of the task completion time and mouse clicks associated with each system and each task in Study A.

a beginner." P6, who already had knowledge of these Excel features, commented: "*Most properties in Excel only had a textual description, which was far from enough... The glyphs [in Table Illustrator] helped me understand the parameters to some extent.*" To summarize, the learning curve of Table Illustrator is comparable or slightly steeper than that of Excel's basic features (e.g., cell editing and tools in the *Home* panel) while smoother than the advanced ones (e.g., complex formulas and data tools).

Feedback on the interaction design. We asked participants on their attitudes toward the interaction patterns of both tools in the post-interview. Overall, all participants agreed that the interaction units utilized by the two systems (puzzles v.s. cell, rows, and columns) were both useful in table authoring. They also mentioned the following pros and cons of the Table Illustrator interactions when compared to Excel:

Pros. First, all participants appreciated the puzzle-based interaction manner and spoke highly of its role in helping them get familiar with the system. They also agreed that the interactions of drag-and-drops and selecting configurations on puzzles were very intuitive (9/15), saving repetitive and trivial work (5/15) and also enhancing enjoyment (3/15). Some participants (4/15) further mentioned that this benefit would become more pronounced given a bigger data size, as noted by P11: "*I love the expandable puzzles and I'm sure they'll be even helpful given bigger tables.*" Besides, many participants reported that reliability was also a major advantage. For instance, P12 said, "*I often got values wrong by accident when copying values in Excel and that's why I had to check the table every few operations. However, I fully trusted Table Illustrator as it never got me wrong.*" Moreover, P3 and P14 highlighted that the interactions helped them explore useful tables and would perform much better when given only instructions or nothing at all, as opposed to receiving explicit output tables.

Cons. About half of the participants (8/15) found the interactions of Table Illustrator lacking a comprehensive control of customizing specific cells compared to Excel. For instance, P5 explained that while he was aware of the high interaction numbers in Excel, he did not perceive the tasks as arduous because he knew exactly what the results would be after each operation. Another commonly mentioned drawback was the gap between the interactions of Table Illustrator and user habits. Six participants said they wished to frame select multiple cells at a time or directly select a whole row or column as in Excel.

Feedback on the design of table paremeters. In general, most participants were satisfied with the usability and expressiveness of the structural and stylistic parameters in Table Illustrator. P5 appreciated the novelty of the *facet* parameter, saying it would be helpful to have this parameter in his daily work. P10 also pointed out that the system matched her usage habits: "*It's great to have the identical cells merged and centered by default*". However, we also received some negative feedback. P8 regarded some parameters as useless and urged for a more flexible design: "*The spacing parameter was not that useful... It's better to change the current selection to an option in the context menu, where you can choose to add an empty line before or after each cell in a puzzle.*" Some participants also complained about the learning curve of the *facet* parameter. Moreover, opinions were divided on adjusting table styles by semantic scope in the *global* panel. Although P12 and P13 found this design reasonable and actively use this function in their practice, some participants perceived it as too informative and requested for a better navigation strategy. As noted by P5, "*Searching for the correct scope was arduous and lacked the WYSIWYG sense. I hope the scopes can be classfied into groups, with the selected scope explicitly highlighted on the canvas.*"

8.2 Study B: A comparative study with Excel, including advanced features

In Study A, we found that the majority of users (13/15) completed tasks using basic editing operations. However, a few users (5/15) tried to utilize advanced features of Excel, specifically Pivot Tables and Subtotals. Unfortunately, about half of these users (3/15) became discouraged and abandoned their attempts due to difficulties in grasping their usages within the limited time frame. Therefore, we were motivated to conduct a follow-up study to investigate whether providing participants with detailed tutorials and sufficient training on these advanced tools would lead to different user performance.

8.2.1 Study design.

Participants. We recruited a new pool of 15 participants (P16-P30, 5 females, 10 males, aged 22.9 on average), including 3 undergraduate, 7 master, and 5 Ph. D students. Similar to Study A, they had all used Excel for creating tables with average proficiency (M=2.93/5, SD=1.03). Besides, some of them also had experience of using Microsoft PowerPoint (11), Google Sheets (5), and Tableau (2) in table authoring, with an average level of skill (M=3.47/5, SD=0.92). Furthermore, three participants had experience of using Pivot Table though none of them claimed to be proficient.

Tasks, procedure, and measures. The study procedure, tasks, and measures for Study B closely mirrored those of Study A, with one key difference: participants in Study B were given a 7-minute tutorial and an 6-minute training session specifically focused on using Excel. In the tutorial, we quickly went through the basic Excel operations (editing, styling, and calculations) and introduced the

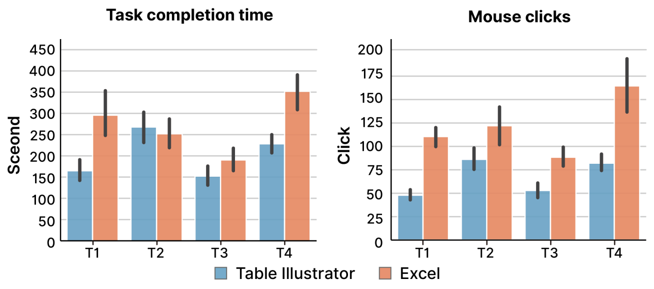


Figure 12: Bar charts of participants' time cost and mouse clicks in Study B for the four tasks (T1-T4) using Table Illustrator and Excel. Error bars indicate 95% confidence intervals.

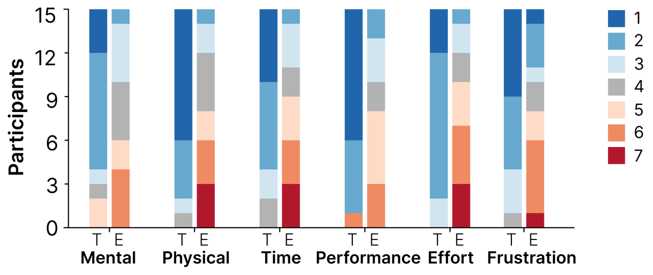


Figure 13: Participants' ratings on their workload for Table Illustrator (T) and Excel (E) in Study B, using a 7-point Likert scale based on NASA-TLX [22]. A lower rating indicates a lower level of perceived workload or a better level of performance.

usage of Pivot Table and Subtotals in a detailed manner. To ensure a fair comparison, we also extended the Table Illustrator training period to 6 minutes. As a result, the total study duration became approximately 85 minutes and each participant was compensated $18 accordingly.

8.2.2 *Quantitative results.*

Task completion number, time cost, and mouse clicks. All participants accomplished all the tasks successfully. Similar to Study A, we analyzed the time cost and mouse clicks by employing three different outlier handling approaches so that the data is distributed normally and suitable for paired student's t-test. We present the analysis results based on the outlier trimming approach as follows, where 5 out of 120 pairs of data points were trimmed for both systems. Results for other approaches can be found in the Appendix.

Figure 12 and Table 3 present participants' time cost and mouse clicks in Study B. We performed a paired student's t-test on the results. Participants spent significantly less time when using Table Illustrator than Excel in T1 ($t(14) = -5.52, p < 0.05$), T3 ($t(13) = -2.65, p < 0.05$), and T4 ($t(13) = -5.76, p < 0.05$). There were also fewer mouse clicks in T1 ($t(13) = -12.2, p < 0.05$), T2 ($t(14) = -2.93, p < 0.05$), T3 ($t(12) = -7.19, p < 0.05$), and T4 ($t(14) = -5.97, p < 0.05$). However, no statistically significant difference was found in the task completion time in T2 ($t(14) = 0.63, p = 0.73$).

Additionally, in Excel, the mean value of participants' time cost in T1 and T2 increased by 43.9% and 33.1% respectively compared to study A. Mouse clicks also showed a respective surge of 53.3% and 51.3%. However, the time cost for Excel decreased by 48.0% in T3 while the mouse clicks decreased by 35.9%. There were no significant differences observed for T4. According to our observations, we speculate that this can be attributed to the participants' increased willingness to attempt Pivot Table with the additional tutorials. The shapes of the target tables in T1 and T2 deviated from standard pivot tables and were more efficiently completed through direct editing rather than Pivot Table. Consequently, many participants switched to direct editing after attempting Pivot Table, resulting in prolonged time cost and increased mouse clicks. In contrast, the target table in T3 was a pivot table and proved to be more easily accomplished using Pivot Table, leading to improved performance. Finally, T4 was similarly difficult for both approaches due to the high number of design details, which likely explained the small performance difference.

Workload. In general, the median of the overall score was 10 for Table Illustrator and 26 for Excel. Figure 13 shows participants' detailed ratings on their workload. We performed a Wilcoxon signed rank test on the results and observed significant differences in all subscales, namely mental demand ($Z = -2.44, p < 0.05$), physical demand ($Z = -3.30, p < 0.05$), time ($Z = -3.18, p < 0.05$), performance ($Z = -3.10, p < 0.05$), effort ($Z = -3.28, p < 0.05$), and frustration ($Z = -2.88, p < 0.05$). The results supported the hypothesis that users perceived less burden when using Table Illustrator.

8.2.3 *User feedback.*

Comments from participants in Study B were consistent with Study A in various dimensions. They further commented on the differences between Pivot Table and Table Illustrator, which are discussed as follows.

Intuitiveness. All participants agreed that the learning curve of Table Illustrator was smoother than that of Pivot Table. The improvement could be mostly attributed to the puzzle-based drag-and-drop interaction for table creation, which was perceived more natural and intuitive by participants compared to dragging entities between shelves in Pivot Table. P19 said that "*I felt Table Illustrator easier to learn as I only needed to consider the positional relationships of puzzles... And the previews [of possible combination results] made it effortless to follow along.*" P22 said similarly by complaining about the Pivot Table: "*The abstract tabular model of the Pivot Table really took me a while to understand. And even when I understood it, I had to spend more time and effort trying to figure out what was happening after dragging.*" Additionally, half of the participants (7/15) indicated that configuring table parameters was somewhat arduous and hard to learn in both systems. However, three of them found the parameters in Table Illustrator had higher usability. For instance, P27 and P30 thought the parameters "*had clearer semantic meanings*" and the results were thus "*more predictable*". P21 appreciated the structural parameters were configured on puzzles rather than the whole table and therefore more flexible. Four participants also mentioned Table Illustrator's snapshot-like semantic glyphs played a better role in previewing the resulting table and facilitating navigation and selection. Besides, two participants said that the automatic subtotals or aggregations in tables generated by Pivot Table brought some cognitive overhead, while the incremental style of adding table elements in Table Illustrator was easier to follow.

		T1		T2		T3		T4	
		μ	σ	μ	σ	μ	σ	μ	σ
Table Illustrator	Time	164.9	50.15	268.0	72.32	152.4	44.58	228.3	41.05
	Mouse clicks	47.9	10.62	86.2	25.58	53.0	15.02	82.1	19.15
Excel	Time	295.8	105.83	251.9	70.70	190.4	54.00	351.9	82.34
	Mouse clicks	110.5	20.33	122.1	40.23	88.5	19.98	164.5	55.44

Table 3: Mean and standard derivation of the task completion time and mouse clicks associated with each system and each task in Study B.

Comprehensiveness. Most participants (9/15) found the table configurations in Table Illustrator were more expressive than those of Pivot Table, especially regarding merging cells (*cell merge*), dividing tables (*facet*), and adding indexes (*key*). Besides, while participants found the templates for table styling in both systems were equally expressive, two of them mentioned that the templates in Table Illustrator were "*more compact*" (P21, P29), indicating that they were more concise and efficient in their design.

8.3 Study Limitations

We have identified a few limitations in our studies. First, while we did not disclose that we developed Table Illustrator, it was hard to entirely eliminate participant response bias [10] since Excel is a well-known commercial spreadsheet software. Second, the tables in the study tasks had relatively small sizes. The systems' performances confronted with large datasets deserve further investigation. Third, though we attempted multiple ones and got similar results, the outlier handling methods per se had limitations and could potentially lead to loss of information. Fourth, it could have been more rigorous to divide participants into three groups, each group using basic Excel features, advanced Excel components, and Table Illustrator. We leave this to future work. Finally, we evaluated the system only with well-defined tasks and under the dataset-driven scenario. While a few users did find our system helpful in exploring table layouts and styles, conducting open-ended studies under different scenarios is likely to strengthen this point and lead to more interesting insights.

9 DISCUSSION

9.1 Implications

We present Table Illustrator, an interactive table authoring system to facilitate rapid prototyping of elaborate plain tables through puzzle-based combinements and configurations. To this end, we conducted a pilot interview on real-world table authoring practices and proposed a design space of plain tables for data presentation. We believe the interview, design space, and the interface we contribute have the following potential to inspire future research and implementations:

9.1.1 Puzzles as an effective metaphor for complex table models. Table Illustrator visualizes semantic groups of table cells as puzzles and enables users to combine puzzles through drag-and-drops to construct tables on a canvas-like GUI. As demonstrated by our user study, not only is this design perceived as highly intuitive and efficient, but also it acts as a bridge between users and the complex table model. Besides, compared to spreadsheet tools with fixed grids on the interface, we show that taking data as malleable graphical objects is perceived natural to some users, while also facilitating some transformations such as operations regarding hierarchy and facet. We plan to incorporate more puzzle-based interactions in the future, e.g., using a "scissors" tool to separate puzzles or tables. Additionally, while our current design is only capable of managing puzzle-wise relationships, we would like to explore approaches to make our interactions compatible with cell-based ones to better accommodate user habits.

9.1.2 Support plain tables for data presentation as new opportunities. Similar to prior work [1, 5], this work highlights the role of plain tables in data presentation in addition to a medium for data analysis which has been widely studied. Through diving into the authoring process of real-world practitioners, we identified their challenges and requirements and analyzed the usage patterns of participants in our user study. Throughout these processes, we have learned the following lessons. Firstly, creating tables for presentation caters to a wide range of users from various fields. Considering the diverse habits and skill levels of users, it is crucial to account for different workflows and design choices during the development of technical tools. Secondly, the table authoring process often involves multiple tools. For example, users may extract data from a website using an explorer, perform data wrangling and cleaning in an IDE, and adjust table layouts and styles using spreadsheet tools. We suggest that a promising direction is to provide integrated or in-situ support of table creation in these tools to eliminate tool switchings. Finally, creating plain tables for presentation is fundamentally an extension of data wrangling in that one should consider structural and stylistic issues apart from data logics. We consequently hypothesize that expanding current wrangling theories and approaches to this domain will likely yield favorable outcomes, such as empowering users to generate tables from examples using the programming-by-example theory [17, 26, 35].

9.1.3 Representing tables for downstream tasks. In this work, we have proposed a design space of plain tables as well as a formal representation, which can potentially serve as the foundation for many downstream tasks. For instance, existing machine learning models for table understanding primarily focus on predicting cell-based features or general table components (captions, headers, metadata, etc.) [11, 14, 20, 30, 48]. Our syntax provides opportunities for novel models that capture relationships between entities and global patterns of table structure and styles. In addition, the syntax can be implemented as libraries for tabular data transformation, enabling data analysts to polish tables in coding environments and avoid switching to spreadsheet tools. Finally, given the recent surge of intelligent agents and tools based on large language models (LLMs), our method can also facilitate human-AI collaboration in table authoring by assisting LLMs in comprehending intricate tables.

9.2 Limitations

We have identified the following limitations for this work. First, while we aim at supporting the workflows of both the dataset-driven and table-driven scenarios through the system, the current design and implementation are somewhat conservative. For the dataset-driven workflow, one of the weaknesses is that users were not allowed to alter the source data, which can become problematic when dealing with dirty data. For the table-driven workflow, we currently adopt a heuristic algorithm to parse the logical organization of imported data, which can potentially be improved to support more table types and the extraction of structure and style patterns. We suggest that a promising solution is to construct labeled datasets based on our table representation approach and use them to fine-tune existing pretrained table understanding models [13]. Moreover, the table-driven scenario assumes users will import a table after finishing the editing process. To further improve the user experience, recommendation strategies should be developed to provide authoring assistance in parallel with table editing, and we leave this as future work.

Furthermore, as a prototype system, the implemented system only covers a subset of the classes in our design space and lacks several common features in spreadsheet tools (e.g., functions apart from *sum*). While we attempted to develop it as an Excel plug-in to better integrate with existing spreadsheet programs, we eventually opted for a standalone web application. This was because our proposed new interaction paradigm preferably required a canvas-like interface rather than fixed grids, and existing spreadsheet programs did not provide enough engineering flexiblity for us to implement such features. What's more, although it seems promising to have a single spreadsheet environment that supports both a cell-based and a puzzle-based mode, additional design issues would arise that are beyond this paper's scope and deserve further investigation. For instance, since cell-based interactions allow users to create randomly unstructured tables, switching between modes may lead to compatibility issues and require users to spend extra efforts in explicitly structuring tables. We encourage future researchers to delve deeper into this direction.

10 CONCLUSION

We propose Table Illustrator, an interactive table authoring system that visualizes table cells as puzzles and enables table construction by combining puzzles based on drag-and-drops, while also providing a diverse list of structural and stylistic configurations for elaboration. Throughout the design process, we conducted a pilot interview with 10 table practitioners to understand the real-world authoring process, revealing their challenges and requirements. We have also proposed a design space for plain tables after surveying more than 2,500 tables. In addition, two within-subject user studies (N=15 for each study) were performed to evaluate the system, demonstrating its usability, expressiveness, and effectiveness in reducing user effort and the learning curve.

ACKNOWLEDGMENTS

Di Weng is the corresponding author. The work was supported by NSF of China (U22A2032) and the Collaborative Innovation Center of Artificial Intelligence by MOE and Zhejiang Provincial Government.

REFERENCES

[1] Lyn Bartram, Micheal Correll, and Melanie Tory. 2022. Untidy Data: The Unreasonable Effectiveness of Tables. *IEEE Transactions on Visualization and Computer Graphics* 28, 01 (2022), 686–696. https://doi.org/10.1109/TVCG.2021.3114830

[2] Mihai Bilauca and Patrick Healy. 2011. Building table formatting tools. In *Proceedings of the 11th ACM Symposium on Document Engineering*. ACM, 13–22. https://doi.org/10.1145/2034691.2034696

[3] Matthew Brehmer and Robert Kosara. 2022. From Jam Session to Recital: Synchronous Communication and Collaboration Around Data in Organizations. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2022), 1139–1149. https://doi.org/10.1109/TVCG.2021.3114760

[4] Karl W. Broman and Kara H. Woo. 2018. Data Organization in Spreadsheets. *The American Statistician* 72, 1 (2018), 2–10. https://doi.org/10.1080/00031305.2017.1375989

[5] George Chalhoub and Advait Sarkar. 2022. "It's Freedom to Put Things Where My Mind Wants": Understanding and Improving the User Experience of Structuring Data in Spreadsheets. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/3491102.3501833

[6] Chris Chambers and Chris Scaffidi. 2010. Struggling to Excel: A Field Study of Challenges Faced by Spreadsheet Users. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*. IEEE Computer Society. https://doi.org/10.1109/VLHCC.2010.33

[7] Kerry Shih-Ping Chang and Brad A. Myers. 2016. Using and Exploring Hierarchical Data in Spreadsheets. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/2858036.2858430

[8] Ran Chen, Di Weng, Yanwei Huang, Xinhuan Shu, Jiayi Zhou, Guodao Sun, and Yingcai Wu. 2023. Rigel: Transforming Tabular Data by Declarative Mapping. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2023), 128–138. https://doi.org/10.1109/TVCG.2022.3209385

[9] Eric Crestan and Patrick Pantel. 2011. Web-Scale Table Census and Classification. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*. ACM. https://doi.org/10.1145/1935826.1935904

[10] Nicola Dell, Vidya Vaidyanathan, Indrani Medhi, Edward Cutrell, and William Thies. 2012. "Yours is Better!": Participant Response Bias in HCI. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 1321–1330. https://doi.org/10.1145/2207676.2208589

[11] Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. TURL: Table Understanding through Representation Learning. *ACM SIGMOD Record* 51, 1 (2022), 33–40. https://doi.org/10.1145/3542700.3542709

[12] Evanthia Dimara, Harry Zhang, Melanie Tory, and Steven Franconeri. 2022. The Unmet Data Visualization Needs of Decision Makers Within Organizations. *IEEE Transactions on Visualization and Computer Graphics* 28, 12 (2022), 4101–4112. https://doi.org/10.1109/TVCG.2021.3074023

[13] Haoyu Dong, Zhoujun Cheng, Xinyi He, Mengyu Zhou, Anda Zhou, Fan Zhou, Ao Liu, Shi Han, and Dongmei Zhang. 2022. Table pre-training: A survey on model architectures, pre-training objectives, and downstream tasks. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 5426–5435. https://doi.org/10.24963/ijcai.2022/761

[14] Haoyu Dong, Shijie Liu, Zhouyu Fu, Shi Han, and Dongmei Zhang. 2019. Semantic structure extraction for spreadsheet tables with a multi-task learning architecture. In *Proceedings of the Workshop on Document Intelligence at NeurIPS*.

[15] Yuyang Dong and Masafumi Oyamada. 2022. Table Enrichment System for Machine Learning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. https://doi.org/10.1145/3477495.3531678

[16] Wensheng Dou, Shi Han, Liang Xu, Dongmei Zhang, and Jun Wei. 2018. Expandable Group Identification in Spreadsheets. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*. ACM. https://doi.org/10.1145/3238147.3238222

[17] Ian Drosos, Titus Barik, Philip J. Guo, Robert DeLine, and Sumit Gulwani. 2020. Wrex: A Unified Programming-by-Example Interaction for Synthesizing Readable Code for Data Scientists. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 1–12. https://doi.org/10.1145/3313831.3376442

[18] Lun Du, Fei Gao, Xu Chen, Ran Jia, Junshan Wang, Jiang Zhang, Shi Han, and Dongmei Zhang. 2021. TabularNet: A Neural Network Architecture for Understanding Semantic Structures of Tabular Data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/3447548.3467228

[19] Julian Eberius, Katrin Braunschweig, Markus Hentsch, Maik Thiele, Ahmad Ahmadov, and Wolfgang Lehner. 2015. Building the Dresden Web Table Corpus: A Classification Approach. In *Proceedings of the IEEE/ACM 2nd International Symposium on Big Data Computing*. IEEE Computer Society. https://doi.org/10.1109/BDC.2015.30

[20] Majid Ghasemi Gol, Jay Pujara, and Pedro Szekely. 2019. Tabular Cell Classification Using Pre-Trained Cell Embeddings. In *Proceedings of the IEEE International Conference on Data Mining*. Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/ICDM.2019.00033

[21] Brian D. Hall, Lyn Bartram, and Matthew Brehmer. 2022. Augmented Chironomia for Presenting Data to Remote Audiences. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. ACM. https://doi.org/10.1145/3526113.3545614

[22] Sandra G. Hart and Lowell E. Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in psychology*. Vol. 52. North-Holland, 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9

[23] Felienne Hermans and Emerson Murphy-Hill. 2015. Enron's Spreadsheets and Related Emails: A Dataset and Analysis. In *Proceedings of the 37th IEEE/ACM International Conference on Software Engineering*. IEEE Computer Society. https://doi.org/10.1109/ICSE.2015.129

[24] Felienne Hermans, Martin Pinzger, and Arie van Deursen. 2011. Supporting Professional Spreadsheet Users by Generating Leveled Dataflow Diagrams. In *Proceedings of the 33rd International Conference on Software Engineering*. ACM. https://doi.org/10.1145/1985793.1985855

[25] Micheal Izza. 2018. *Twenty principles for good spreadsheet practice*. The Institute of Chartered Accountants in England and Wales. https://www.icaew.com/-/media/corporate/files/technical/technology/excel-community/20-principles-of-good-spreadsheet-practice-2018.ashx

[26] Zhongjun Jin, Michael R. Anderson, Michael J. Cafarella, and H. V. Jagadish. 2017. Foofah: Transforming Data By Example. In *Proceedings of the ACM International Conference on Management of Data*. ACM, 683–698. https://doi.org/10.1145/3035918.3064034

[27] Sean Kandel, Jeffrey Heer, Catherine Plaisant, Jessie Kennedy, Frank van Ham, Nathalie Henry Riche, Chris Weaver, Bongshin Lee, Dominique Brodbeck, and Paolo Buono. 2011. Research Directions in Data Wrangling: Visualizations and Transformations for Usable and Credible Data. *Information Visualization* 10, 4 (2011), 271–288. https://doi.org/10.1177/1473871611415994

[28] Sean Kandel, Andreas Paepcke, Joseph M. Hellerstein, and Jeffrey Heer. 2011. Wrangler: interactive visual specification of data transformation scripts. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/1978942.1979444

[29] Eser Kandogan, Aruna Balakrishnan, Eben M. Haber, and Jeffrey S. Pierce. 2014. From Data to Insight: Work Practices of Analysts in the Enterprise. *IEEE Computer Graphics and Applications* 34, 5 (2014), 42–50. https://doi.org/10.1109/MCG.2014.62

[30] Elvis Koci, Maik Thiele, Oscar Romero, and Wolfgang Lehner. 2016. A Machine Learning Approach for Layout Inference in Spreadsheets. In *Proceedings of the International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Science and Technology Publications, Lda, 77–88. https://doi.org/10.5220/0006052200770088

[31] Andrea Kohlhase. 2013. Human-Spreadsheet Interaction. In *Human-Computer Interaction – INTERACT 2013*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-40498-6_47

[32] Larissa R. Lautert, Marcelo M. Scheidt, and Carina F. Dorneles. 2013. Web Table Taxonomy and Formalization. *ACM SIGMOD Record* 42, 3 (2013), 28–33. https://doi.org/10.1145/2536669.2536674

[33] Guozheng Li, Runfei Li, Zicheng Wang, Chi Harold Liu, Min Lu, and Guoren Wang. 2022. HiTailor: Interactive Transformation and Visualization for Hierarchical Tabular Data. *IEEE Transactions on Visualization and Computer Graphics* 29, 1 (2022), 139–148. https://doi.org/10.48550/arXiv.2208.05821

[34] Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. 2019. TableBank: A Benchmark Dataset for Table Detection and Recognition. arXiv:1903.01949 [cs.CV]

[35] Henry Lieberman. 2001. *Your wish is my command: Programming by example*. Morgan Kaufmann.

[36] Apra Lipi, Kishan Kumar, and Soubhik Chakraborty. 2022. *Outliers: Detection and Analysis*. Nova Science Publishers, Inc.

[37] pandas 2023. Pandas, a Python data analysis and manipulation tool. Retrieved August 31, 2023 from https://pandas.pydata.org

[38] Alexander Prutzkow. 2023. Principles of Data Design in Spreadsheets. *International Journal of Open Information Technologies* 11, 04 (2023), 102–105.

[39] Michael Sedlmair, Miriah Meyer, and Tamara Munzner. 2012. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2431–2440. https://doi.org/10.1109/TVCG.2012.213

[40] Justin Smith, Justin A. Middleton, and Nicholas A. Kraft. 2017. Spreadsheet practices and challenges in a large multinational conglomerate. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*. IEEE Computer Society. https://doi.org/10.1109/VLHCC.2017.8103463

[41] Spreadsheet Engineering Research Project (SERP) of Tuck School of Business at Dartmouth College. 2015. Results of On-line Survey of Spreadsheet Usage. Retrieved August 31, 2023 from https://faculty.tuck.dartmouth.edu/images/uploads/faculty/serp/serp_results.pdf

[42] Sruti Srinivasa Ragavan, Advait Sarkar, and Andrew D Gordon. 2021. Spreadsheet Comprehension: Guesswork, Giving Up and Going Back to the Author. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/3411764.3445634

[43] Tableau Inc. 2023. Tableau Prep. Retrieved August 31, 2023 from https://www.tableau.com/products/prep

[44] Ricardo Teixeira and Vasco Amaral. 2016. On the Emergence of Patterns for Spreadsheets Data Arrangements. In *Software Technologies: Applications and Foundations*. Springer International Publishing. https://doi.org/10.1007/978-3-319-50230-4_25

[45] Trifacta, Inc. 2023. Trifacta. Retrieved August 31, 2023 from https://www.trifacta.com

[46] Xinxin Wang. 1996. *Tabular abstraction, editing, and formatting*. Ph. D. Dissertation. CAN. AAINN09397.

[47] Xinxin Wang and Derick Wood. 1996. *Xtable: a tabular editor and formatter*. Technical Report. The Hong Kong University of Science and Technology.

[48] Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021. TUTA: Tree-Based Transformers for Generally Structured Table Pre-Training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/3447548.3467434

[49] Siyuan Xia, Nafisa Anzum, Semih Salihoglu, and Jian Zhao. 2021. KTabulator: Interactive Ad Hoc Table Creation Using Knowledge Graphs. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM. https://doi.org/10.1145/3411764.3445227

[50] Shuo Zhang, Vugar Abdul Zada, and Krisztian Balog. 2018. SmartTable: A Spreadsheet Program with Intelligent Assistance. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. https://doi.org/10.1145/3209978.3210171

[51] Shuo Zhang and Krisztian Balog. 2020. Web Table Extraction, Retrieval, and Augmentation: A Survey. *ACM Transactions on Intelligent Systems and Technology* 11, 2 (2020), 1–35. https://doi.org/10.1145/3372117

[52] Yakun Zhang, Xiao Lv, Haoyu Dong, Wensheng Dou, Shi Han, Dongmei Zhang, Jun Wei, and Dan Ye. 2021. Semantic table structure identification in spreadsheets. In *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM. https://doi.org/10.1145/3460319.3464812

Received 14 September 2023; revised 12 December 2023; accepted 19 January 2024