
# NiceT: Named Entity Cleaning and Enhancement with Human-in-the-loop

Garima Gaur
Inria Saclay, Ecole Polytechnique
Palaiseau, France
garima.gaur@inria.fr

Ioana Manolescu
Inria Saclay, Ecole Polytechnique
Palaiseau, France
ioana.manolescu@inria.fr

María Mellado Tenorio
Universidad de Chile
Santiago, Chile
maria.mellado@ug.uchile.cl

Madhulika Mohanty
Inria Saclay, Ecole Polytechnique
Palaiseau, France
madhulika.mohanty@inria.fr

Moritz Sommer
RWTH Aachen, Ecole Polytechnique
Palaiseau, France
m.sommer@iat.rwth-aachen.de

## ABSTRACT

Named Entities (NEs, in short) are frequent in many datasets, e.g., journalistic investigations (people, places, companies), market analysis (companies and officers), etc. NEs often appear under varied forms, due to spelling variants or mistakes. To leverage NE-rich datasets, the NEs need to be *clean* (error-free), and possibly *enriched* with information from external sources. Existing data cleaning solutions tools only partially answer the challenges raised by NE cleaning (lack of regular-structured attributes, need for interactive human input) and/or they do not exploit the opportunities it raises (semantic signals). We propose to demonstrate NiceT, a dedicated NE cleaning tool. Through a *visual workflow interface*, NiceT allows *comparing, clustering, and enriching* based on Knowledge Bases and LLMs, and also *interacting with the NE cleaning process*, down to the granularity of an attribute in a record. The latter is crucial in order to capture *advanced knowledge that only domain experts have*, and which may be absent from all other sources of information (KB, LLM, etc.) For explainability, NiceT gathers *how-provenance* that traces the ways in which information contributes to clean NEs. We will showcase it on a variety of real-life datasets inspired by our collaborations with journalists.

**PVLDB Reference Format:**
Garima Gaur, Ioana Manolescu, María Mellado Tenorio, Madhulika Mohanty, and Moritz Sommer. NiceT: Named Entity Cleaning and Enhancement with Human-in-the-loop. PVLDB, 19(12): 4646 - 4649, 2026.
doi:10.14778/3827998.3828087

**PVLDB Artifact Availability:**
The source code, data, and/or other artifacts have been made available at https://gitlab.inria.fr/cedar/nicet-tool.

## 1 INTRODUCTION

Named entities (or NEs), such as People, Locations, Organizations (which can be further specialized into Companies, NGOs, Government Agencies, etc.) appear in many applications' datasets: local tax or school population datasets mention geographic locations, market analysis involves companies and their chief executives, journalistic investigations are often about politicians, parties, companies, precincts and electoral areas, etc. The effectiveness of tools used by investigative journalists [3–5, 8], that extract NEs from text, structured, or semi-structured data, correlates positively with the quality of NEs, i.e., the cleaner the NEs, more effective the tools are. Data cleaning is a well-known database problem, covered by a rich literature, e.g., [1, 13]. In particular, cleaning records (tuples) describing people is a classical task in Customer Relationship Management (CRM). However, recent developments, especially, in the area of Information Extraction (IE, in short), based on trained Language Models (LMs), lead to **new challenges in the NE cleaning task**, as follows:

**Table 1: Performance comparison of OpenRefine and NiceT NE cleaning operators.**

| Tool | Methodology | Accuracy |
|---|---|---|
| OpenRefine | Clustering (Key collision Metaphone3) | 0.18 |
| | Reconciliation | 0.85 |
| NiceT | Clustering | 0.95 |

First, IE outputs NEs that consist of a short sequence of tokens, together with a *NE type*, e.g., Person, Location, Organization, etc.; this differs from the setting where a person is described by a multi-attribute record, e.g., first name, last name, street, city, and country. *NEs resulting from extraction are "poorer"* (lack associated attributes). More attributes can be *acquired* and *added* to the NEs from external resources such as Knowledge Bases (KBs) and/or Large Language Models (LLMs); however, for long tail entities, information is lacking even in such resources, thus NE attributes are not always available. Second, IE may output very diverse NEs for a given entity type. For instance, Organization entities, a frequent "catch-all" class used in many NE Recognition (NER) tools, can cover very broad things: companies, public administrations, NGOs, universities, etc. Thus, there is *even same-type NEs may be more heterogeneous* than records to be cleaned in typical relational contexts. This is challenging because the NEs may lack multiple common attributes based on which to compare them. Third, NEs are not just strings; they carry contextual and semantic information. Thus, *existing data cleaning tools fall short of providing a comprehensive set of NE-specific cleaning operations.*

This work is licensed under the Creative Commons BY-NC-ND 4.0 International License. Visit https://creativecommons.org/licenses/by-nc-nd/4.0/ to view a copy of this license. For any use beyond those covered by this license, obtain permission by emailing info@vldb.org. Copyright is held by the owner/author(s). Publication rights licensed to the VLDB Endowment.
Proceedings of the VLDB Endowment, Vol. 19, No. 12 ISSN 2150-8097.
doi:10.14778/3827998.3828087

To illustrate this gap, we attempted to clean a collection of Organization NEs using the state-of-the-art data cleaning tool **OpenRefine** [10]. We extracted these NEs out of scientific articles published in Toxicology journals, TAP and RTP, in prior work [11]. We randomly selected 10 organizations and their multiple surface forms, resulting in a total of 40 extractions. We then measured how many extracted NEs OpenRefine correctly mapped to their canonical clean form. For example, we expect that *UC Berkeley* and *University of California Berkeley* are mapped to the same organization. For OpenRefine, we used its two main features for string cleaning: Clustering[1] which relies primarily on syntactic similarity, and Reconciliation[2] which attempts to deduplicate by linking an input string to its corresponding Wikidata entity. Using NiceT, we designed a pipeline that clusters NE embeddings, obtained from a small language model [9], via agglomerative clustering. We report the accuracy of these methods in Table 1. The strong performance of this relatively simple NE-specific cleaning workflow highlights the need for a dedicated, NE-tailored cleaning tool.

We view NE cleaning as a collaborative process between a technical user (**TU**) who is familiar with technical cleaning methods, and an non-technical user (**NTU**) with domain knowledge. We propose **NiceT**, an interactive, visual tool for cleaning sets of NEs of the same type. NiceT combines (*i*) traditional data cleaning primitives such as string transformation, clustering, etc. (**workflow designing**); (*ii*) controlled acquisition of information from Knowledge Bases or LLMs; this may on one hand help cleaning by adding context attributes to NEs, and on the other hand, makes them more complete and thus more useful (**NE enrichment**); (*iii*) input from users (NTU) that may inspect and modify a field, a record, or a group of records, at any point in the process; this enables them to inject precious knowledge unavailable in the other sources (**human intervention**); (*iv*) reproducibility and explainability through extensive provenance keeping (at the task or record granularity) (**result provenance**). During the demo, audience can play the role of TU and design their own workflows or can intervene as a domain expert using datasets derived from our fact-checking [6] and investigative projects [3, 11]. The tool is available online[3] along with a video of the demo[4].

## 2 RELATED WORK

**Data cleaning tools** can be broadly categorized into (*i*) data repair tools, and (*ii*) workflow platforms. Data repair tools [7, 14, 16] employ a probabilistic or ML model to detect data errors and generate repair suggestions. In contrast, workflow tools give users explicit control over the cleaning process, and NiceT belongs to this class. Workflow platforms such as [10, 15, 17–19] provide cleaning operators that handle string values solely based on their syntactic signals. For example, OpenRefine [10] includes edit distance (nearest neighbor clustering), n-grams (fingerprinting), and phonetic features (Metaphone3 fingerprinting). For NE cleaning, support for semantic features is essential to bridge their significant surface variations, introduced in Section 1. In principle, OpenClean [15] can be extended with an LM to support semantic encoding of textual data, but it lacks support for human intervention, which is important in the context of NE cleaning. NiceT is designed specifically to accommodate both these requirements that are crucial in the scenarii journalists proposed to us [3, 11].

**Table 2: Actor classes.**

| Actors Class | Pre-defined Action Templates |
|---|---|
| *Clean* | `Replace, Trim, Case Change, ...` |
| *Cluster* | `Syntactic clustering, Semantic clustering, ...` |
| *Filter* | `Attribute-based, Value-based, Type-based` |
| *Project* | `User-specified column(s)` |
| *Annotate* | `Attribute expansion, Classification` |
| *Fusion* | `Merging similar entities` |

**Collaborative Cleaning** Repair-based cleaning tools [7, 16] involve humans as *validators* (to approve or reject auto-generated data repair suggestions) whereas in the workflow platforms [18], humans are *proactive spectators*, incrementally modifying the workflow to attain the desider results. In contrast, NiceT considers the humans as *domain experts* and also enables them to *augment* the knowledge, thus, supporting a gamut of permissible actions, notably, user-defined data fusion and splitting policies for a set of NEs.

## 3 DATA AND PROCESSING MODEL

We formally model data and processing in NiceT as follows.

**Data** We consider a data model reminiscent of the nested relational model [2], also very close to regular-structure JSON collections. Let $\mathcal{V}_a$ denote an alphabet of *a*tomic *v*alues. For simplicity, we assume all strings, numbers, dates, etc. are part of $\mathcal{V}_a$. Let $\mathcal{V}$ be the set of *values*, containing $\mathcal{V}_a$, as well with any set, list, or bag (multiset) of elements from $\mathcal{V}_a$.

A **tuple** $t$ is of the form $(a_1 : v_1, \ldots, a_n : v_n)$ where each $a_i$ is an attribute name (a string) and $v_i \in \mathcal{V}$) is the attribute value. The *schema* of tuple $t$ is the attribute name list $(a_1, \ldots, a_n)$. A **dataset** $D$ is a set of tuples with the same schema.

**Workflow** A workflow $W(S, E)$ is a DAG where $S$ is the set of nodes representing *steps* of the workflow, and an edge $(s_i, s_j) \in E$, where $s_i, s_j \in S$, denotes that the output of step $s_i$ serves as the input to the step $s_j$. A **step** $s_i \in S$ is a unit of computation applied to an input dataset $D_i^{in}$ and producing the output dataset $D_i^{out}$. We denote the first step of the workflow by $s_0$, and thus, the input dataset as $D_0^{in}$. Typically, $D_0^{in}$ is a dataset of tuples having just one field, each tuple containing exactly an NE.

**Actors** are computational modules that can be instantiated to implement, each, a specific kind of step in a workflow. The different classes of actors supported in NiceT are listed in Table 2, which also illustrates a few typical actions from each actor class. For instance, a `cluster` actor implements a specific clustering method, e.g., k-means, hierarchical agglomeration clustering with semantic similarity as distance metric.

**Human Edit Operations** These allow injecting human expertise that no other source or processing module can provide. Ideally edits are rare, but they are needed to address the challenges raised by NE cleaning. We do not model edits by actors, because the latter are dedicated to computations. Instead, we allow human users to *intervene on any dataset* (input of a workflow, or output of a workflow actor), at the granularity of *one attribute in one tuple*, or *a tuple*, or *a set of tuples*. Specifically, users can *update*, *delete*, *merge*, and *split* tuples (see Sec. 4.2). Following each edit, the actors which depended upon the edited data are recomputed. We **log** all human

[1] https://openrefine.org/docs/technical-reference/clustering-in-depth
[2] https://openrefine.org/docs/technical-reference/reconciliation-api
[3] https://gitlab.inria.fr/cedar/nicet-tool
[4] https://www.youtube.com/watch?v=XF463ggknpM

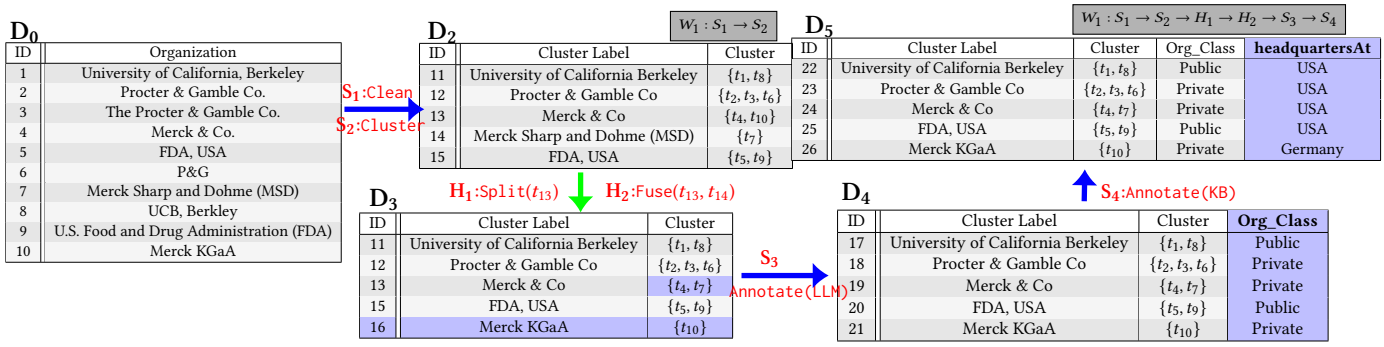


**Figure 1: With input dataset $D_0$, first the analyst designs workflow $W_1 = S_1 \rightarrow S_2$, followed by human edits ($H_1$ and $H_2$) performed by the journalist (domain expert) to correct the clusters. With entity enhancement feature of NiceT dataset $D_2$ is appended with two additional attributes *Org_Class* and *headquartersAt* using annotate actors that query GPT3.5 and Wikidata, respectively.**

edit operations and include them in the workflow description, for traceability and reproducibility.

## 4 USER INTERACTION WITH NICET

We illustrate the features of NiceT through a running example. Consider a journalist (NTU) investigating the role of organizations in funding pharmaceutical research. She collects articles from toxicology journals and collaborates with a data analyst (TU) to extract relationships between authors and funding organizations. The analyst first applies an NER tool to extract named entities (NEs) and their types, resulting in an initial set of organizations, denoted as $D_0$ in Figure 1.

### 4.1 Iterative workflow design

To clean the extracted set of Organization type NEs, the analyst (TU) choses the following steps: (*i*) *clean* the data by removing punctuation marks and extra spaces, (*ii*) *cluster* organization names such that the names referring to the same organization are in the same cluster. She does this with a workflow consisting of a *clean* actor, for the string manipulation operations, followed by a *cluster* actor. For the latter, she choses a language model for encoding strings into embeddings, a clustering method and its parameters. Out of the box, NiceT supports k-means, hierarchical agglomerative clustering, and DBSCAN. The output of this workflow is shown as $D_2$ in Figure 1. Here, each tuple in $D_2$ represents a cluster. For legibility, we omit the trivial output (dataset $D_1$) of the clean actor. Designing workflows is an iterative process. NiceT supports **selective undo-redo** of actor instances. In our example, the analyst may want to re-cluster the organizations if the quality of clusters is poor. She can *undo* the clustering step by removing the cluster actor from the workflow without impacting the previous *clean* actor. Then, she adds a new cluster actor instance with a different clustering method. If the undone step has subsequent steps in the workflow, the analyst can choose the steps that she wants to redo.

### 4.2 Human intervention

The data analyst presents the output of the clustering to the journalist (NTU). While inspecting the result with a user-friendly searchable cluster inspection interface of NiceT, she noticed that *Merck & Co.* and *Merck Sharp and Dohme (MSD)* are in different clusters ($t_{13}$ and $t_{14}$ in $D_2$). Ideally, they should be clustered together as these are two regional names of the same organization: *Merck Sharp and Dohme (MSD)* is used outside the United States. Further, the NTU notes that *Merck KGaA* and *Merck & Co.* are clustered together (tuple $t_{13}$). However, as the NTU explains, these two organizations should be in different clusters as they are two separate organizations: while the former is a German multinational science and technology company, the latter is a US-based pharmaceutical company. The NTU uses the graphical cluster interface to drag & drop the members of one cluster into the other. These actions are recorded as a sequence of human edit operations in the workflow. The final effect of human edits is shown as $D_3$ in Figure 1.

Recall that NiceT supports human intervention via 4 edits operations – *fuse*, *update*, *delete*, and *split*. In our example, the journalist (NTU) splits (recorded as $H_1$) the Merck & Co cluster (tuple $t_{13}$) by creating a new tuple *(ID:16, Cluster Label: "Merck KGaA", Cluster:* $\{t_{10}\}$*)* and updating the cluster attribute of $t_{13}$ to $\{t_4\}$ (shown in $D_3$ in Figure 1). Next, to make sure *Merck Sharp and Dohme (MSD)* and *Merck & Co* belong to the same cluster, the journalist fuses ($H_2$) $t_{14}$ to the updated $t_{13}$.

### 4.3 NE enrichment

To enrich this dataset, the journalist wants to attach some attributes to these organizations: (a) the ownership structure (*Public* vs *Private*) and (b) their headquarters' locations. To pull in this external information, the analyst adds two *annotate* actors. These actors acquire external information about NEs from external sources; specifically, they chose to rely on an LLM for organization status, and a KB (Wikidata) for headquarters information. These actors' output are shown as $D_4$ and $D_5$ in Figure 1.

**Enrichment via LLMs:** The LLM enrichment actor is defined by an LLM to use, a (predefined, customizable) prompt, and the attribute whose value is asked from the LLM, e.g., *is_private*. In this case, we get a boolean (true if the LLM considers the organization to be private, false otherwise). **Enrichment via KBs:** The KB enrichment actor works as follows. Given an NE $e$, we query a SPARQL endpoint of the KB to find a resource whose *rdfs:label* is $e$. If no result is found, then *Web search* is used to fetch the Wikidata page URL corresponding to $e$. This URL contains the Wikidata ID associated with the NE. In details, we query the DuckDuckGo search engine via the library Duckduckgo_search[5] with the query string: "$e$ Wikidata". For many NEs, this results in a Wikidata subject URI. Next, NiceT

[5]https://pypi.org/project/duckduckgo-search/

employs a *KB property harvester* that extracts properties specific: (*i*) to the *NE type*; (*ii*) to the *NE* itself. For instance, NEs of type Organization generally have *headquarters location* and *inception year*. The analyst can choose a property from the harvested list *NE type (Organization)-specific properties*, e.g., *headquarters location* (Wikidata P159) property; the harvester fetches its value for all the NEs in the dataset. For harvesting NE-specific properties, the harvestor first retireves all the properties associated with a given NE, and ranks them based on their contextual relevance using an LLM. For instance, in the example, property *ranking* (P1352) is relevant for *University of California* more than for any other organizations. Similarly, *stock exchange* (P414) is relevant for organization like *Mercek & Co,*, *Procter & Gamble Co*, and *MSD*.

### 4.4 Result provenance for result analysis

The analyst notices the cluster label "*Merck KGaA*" (tuple $t_{26}$) in $D_5$ which did not exist in the output ($D_2$) of the cluster actor. To investigate further, she queries the provenance of $t_{26}$. NiceT generates a graph-shaped representation of queried tuple's *how provenance* [12]. The provenance graph of a tuple $t$ (in our example, $t_{26}$), is a graph $G_{prov}(t)$ where a node is either a *tuple*, a *workflow step* (actor or human edit), or a special *combination operator* (labeled AND, respectively, OR). Usually, an actor node has an incoming edge from the input tuple that the actor operated on to produce $t$, and an outgoing edge to the node corresponding to $t$. When multiple tuples are involved in the generation of $t$, the input tuples are connected to the actor node via combination nodes. When several tuples together lead to a result, an AND is used, e.g., the tuple $t_{11}$ in $D_2$ is produced together by $t_1$ and $t_8$. An *OR* combination node is used when multiple tuples *individually* produce the same tuple $t$. For instance, the projection of *headquartersAt* attribute of $D_5$ (using a `project` actor) leads to a tuple with attribute value *USA*; this results from each of the four tuples $t_{22}$ to $t_{25}$.

## 5 IMPLEMENTATION

NiceT follows an MVC architecture, using FastAPI for the backend and React.js for the frontend. The system's core is an extensible Python library that handles data processing and workflow execution. For workflow storage and provenance management, we use the lightweight key–value store Redis[6]. To support efficient and scalable execution over large datasets, the system incorporates a DAG-based workflow engine that enables partial re-execution, thereby avoiding redundant full-pipeline recomputation. NiceT optimizes interactions with external services by batching LLM requests and employing an extensible caching layer for LLM and Wikidata responses. Furthermore, it employs output pagination at the UI layer ensuring bounded network overhead. The NE enrichment via LLMs feature supports both querying Chat Completion API-enabled LLMs like OpenAI's GPT models, and open-models via Ollama. For the KB based enrichment, the SPARQL endpoint of Wikidata is queried using the qwikidata.sparql[7] API. In addition to the graphical interface, NiceT provides a native Python library. It enables users to define data cleaning workflows as Python configuration files, facilitating integration with existing ETL pipelines, e.g., NiceT has been integrated with the ConnectionLens system [4].

[6]https://redis.io/docs/latest/develop/

[7]https://qwikidata.readthedocs.io/en/stable/index.html

## 6 DEMONSTRATION SCENARIO

**1.** Using NiceT's drag-and-drop actor palette, we demonstrate **interactive workflow construction** for cleaning person names extracted from fact-checking articles[6]. We first apply standard cleaning operations (e.g., removing honorifics and punctuation) via the Clean actor, followed by Fuse and Cluster to merge syntactically or semantically similar named entities (NEs). We then inspect the provenance graph of a cleaned NE to explain whether syntactic or semantic signals contributed to its final form.

**2.** We simulate **domain expert intervention** using an evaluation set of 100 organization name pairs (raw and cleaned) derived from acknowledgment sections of toxicology journals [11]. A journalist (domain expert) reviewed the set, suggesting corrections for 22% of the cleaned entities, primarily due to missing contextual information, like country names. We demonstrate how NiceT addresses these issues via **entity enrichment**. The expert also proposed clustering refinements due to mergers and acquisitions, which we incorporate using Merge and Split operations, illustrating the system's flexibility in human-in-the-loop cleaning.

## ACKNOWLEDGMENTS

This work is supported in parts by the EU project ELIAS (101120237) grant and Hi!PARIS Center.

## REFERENCES

[1] Ziawasch Abedjan, Lukasz Golab, Felix Naumann, and Thorsten Papenbrock. 2018. *Data Profiling.* Morgan & Claypool Publishers.
[2] Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of databases.* Vol. 8. Addison-Wesley Reading.
[3] Angelos Anadiotis, Oana Balalau, Théo Bouganim, et al. 2021. Empowering Investigative Journalism with Graph-based Heterogeneous Data Management. *IEEE DEBull.* 45 (2021).
[4] Angelos Anadiotis, Oana Balalau, Catarina Conceiçao, et al. 2022. Graph integration of structured, semistructured and unstructured data for data journalism. *Information Systems* 104 (2022).
[5] Angelos Christos Anadiotis, Ioana Manolescu, and Madhulika Mohanty. 2023. Integrating Connection Search in Graph Queries. In *ICDE*. 2607–2620.
[6] Oana Balalau, Pablo Bertaud-Velten, Younes El Fraihi, et al. 2024. FactCheckBureau: Build Your Own Fact-Check Analysis Pipeline. In *CIKM*. 5185–5189.
[7] Xianchun Bao, Zian Bao, Bie Binbin, et al. 2024. Rock: Cleaning Data by Embedding ML in Logic Rules. In *SIGMOD*. 106–119.
[8] Nelly Barret, Simon Ebel, Théo Galizzi, et al. 2023. User-Friendly Exploration of Highly Heterogeneous Data Lakes. In *CoopIS*. 488–496.
[9] Lihu Chen, Gael Varoquaux, and Fabian Suchanek. 2024. Learning High-Quality and General-Purpose Phrase Representations. In *EACL*. 983–994.
[10] Antonin Delpeuch, Tom Morris, David Huynh, et al. 2025. *OpenRefine/OpenRefine: OpenRefine 3.9.2.* https://doi.org/10.5281/zenodo.15089184
[11] Garima Gaur, Oana Balalau, Ioana Manolescu, et al. 2025. The Search for Conflicts of Interest: Open Information Extraction in Scientific Publications. In *EMNLP 2025*. 13922–13936.
[12] Todd J. Green, Grigoris Karvounarakis, and Val Tannen. 2007. Provenance semirings. In *PODS (PODS '07)*. 31–40.
[13] Ihab F. Ilyas and Xu Chu. 2019. *Data Cleaning.* ACM.
[14] Sanjay Krishnan, Jiannan Wang, Eugene Wu, et al. 2016. ActiveClean: interactive data cleaning for statistical modeling. *VLDB* 9 (2016), 948–959.
[15] Heiko Müller, Sonia Castelo, Munaf Qazi, et al. 2021. From papers to practice: the OpenClean open-source data cleaning library. *VLDB* 14 (2021).
[16] Mashaal Musleh, Mourad Ouzzani, Nan Tang, et al. 2020. CoClean: Collaborative Data Cleaning. In *SIGMOD*. 2757–2760.
[17] Wei Ni, Kaihang Zhang, Xiaoye Miao, et al. 2024. IterClean: An Iterative Data Cleaning Framework with Large Language Models. In *ACM-TURC*. 100–105.
[18] João L. M. Pereira, Manuel J. Fonseca, Antónia Lopes, et al. 2024. Cleenex: Support for User Involvement during an Iterative Data Cleaning Process. *J. Data and Information Quality* 16 (2024).
[19] Theodoros Rekatsinas, Xu Chu, Ihab F. Ilyas, et al. 2017. HoloClean: holistic data repairs with probabilistic inference. *VLDB* 10 (2017).