Data Cleaning in the Wild: Reusable Curation Idioms from a Multi-Year SQL Workload

Shrainik Jain, Bill Howe
Computer Science and Engineering Department,
University of Washington,
Seattle, WA, USA
{shrainik, billhowe}@cs.washington.edu

ABSTRACT

In this work-in-progress paper, we extract a set of curation idioms from a five-year corpus of hand-written SQL queries collected from a Database-as-a-Service platform called SQLShare. The idioms we discover in the corpus include structural manipulation tasks (*e.g.*, vertical and horizontal recomposition), schema manipulation tasks (e.g., column renaming and reordering), and value manipulation tasks (e.g., manual type coercion, null standardization, and arithmetic transformations). These idioms suggest that users find SQL to be an appropriate language for certain data curation tasks, but we find that applying these idioms in practice is sufficiently awkward to motivate a set of new services to help automate cleaning and curation tasks. We present these idioms, the workload from which they were derived, and the features they motivate in SQL to help automate tasks. Looking ahead, we describe a generalized idiom recommendation service that can automatically apply appropriate transformations, including cleaning and curation, on data ingest.

1. INTRODUCTION

Data curation is increasingly recognized as the bottleneck to analytics. Researchers and practitioners report spending a high proportion of their time cleaning, restructuring, transforming or otherwise preparing data for analysis. Worse, the time and effort spent on these "janitorial" tasks are difficult to amortize over repeated analysis projects; requirements tend to vary widely from project to project.

Classical approaches to data integration are relevant to curation, but tend to emphasize the design of a mediated schema to subsume two or more existing schemas. Data warehouse cubes are also associated with significant up front design and engineering of a centralized schema and the ETL workloads to fill it. These heavyweight "once and for all" approaches are a poor fit in data science contexts, where small teams of analysts convert data into actionable insights in more or less real time, drawing together multiple sources to answer targeted questions using specialized methods. Recent systems aim to reduce the effort required during the data curation step (*e.g.*, format-busting and data profiling with Data Wrangler [1], enterprise integration with Tamr [18]), but scripts-and-files approaches are still dominant among data scientists: there is no time to amortize the up-front cost of warehouse design or global-as-view/local-as-view data integration exercises, and moreover, the data is rarely being extracted from a carefully engineered schema on which these methods tend to rely.

But the cost of this over-reliance on scripts and files is high: Our collaborators in the sciences report spending up to 90% of their time manipulating data [10], consistent with other anecdotal reports of the balance of time between data curation versus data analysis.

To reduce the burden of data janitorial work and improve reuse, we posit that databases can be naturally extended to support the entire data lifecycle, including preliminary data cleaning and curation from untrusted sources typically handled outside the database. That is, we argue that databases should be designed to *encourage* ingestion of dirty, weakly structured data (i.e., rows-and-columns but no engineered schema), and that curation should be performed directly in the database by writing SQL. In this paper, we provide evidence of how this approach of letting databases do the curation via SQL queries, actually worked in practice by finding and characterizing *'cleaning idioms'* in a multi-year query workload.

We see multiple benefits to letting databases do this heavy lifting: i) the data always resides at one place during the entire analysis lifecycle (Figure 1), ii) the cleanup steps become more scalable, reliable, and reusable, and iii) the raw data is always directly available for reprocessing and recleaning in new contexts.

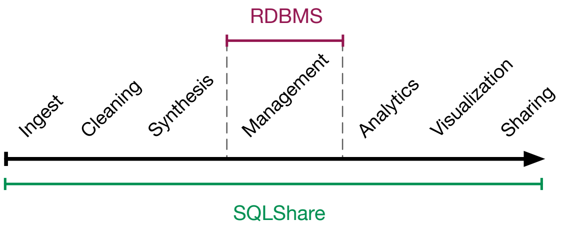

Figure 1: We find relational databases to be relevant at all stages in the scientific data lifecyle. SQLShare, a cloud-hosted database, empowers novice users by providing a system which handles use-cases across the data lifecycle.

The primary disadvantage of this approach is the SQL authorship: many common curation tasks, while expressible in SQL, are sufficiently awkward as to prevent uptake. Our hypothesis was that direct support for a set of common SQL data curation idioms can make SQL-based curation competitive with script-based curation.

To understand data curation tasks in practice, we analyzed the workload of the SQLShare system [11, 12, 10], a Database-as-a-Service system targeting scientists and engineers. SQLShare encourages users to upload uncurated datasets over the web "as is," write queries across any datasets in the system, and share the results as views. The goal is to reduce the overhead in using relational databases in ad hoc analytics scenarios by reducing or eliminating upfront costs associated with installation, configuration, schema design, tuning, and ingestion. SQLShare supports automated schema inference and tolerates dirty data; these features allowed users to switch from managing and sharing brittle, dynamic sequences of scripts to a single system where all the operations can be performed safely, reliably, and scalably.

Over the years, we collected the query logs on SQLShare and analyzed interesting use cases. One common use case, as we expected, was that of data cleanup and curation tasks. We show in this work how we can use these query logs to identify common cleanup tasks and provide them as suggestions for newer dataset uploads.

Furthermore, we envision how these cleanup *idioms* can be used to inform design of newer databases as follows:

1. Identify the clean up task from the query logs (hint: these are often the very first tasks performed on a dataset)
2. Generate templatized idioms for these cleanup tasks.
3. Upon newer dataset uploads, identify which idioms can be applied to the datasets.
4. Synthesize a clean up query from the selected idiom.

In this paper, we identify common curation patterns that appeared prominently in the SQLShare workload, we describe how they are used in practice, and how these patterns informed specific features in SQLShare to assist in data ingest and query authoring. Finally, we describe some ongoing work in semi-automatic data curation based on these idioms.

2. SQLSHARE WORKLOAD

SQLShare is a Database-as-a-Service system targeting scientists and engineers. Users upload datasets through a web interface as-is with no explicit schema, write queries across any datasets in the system, and share the results as views. The goal is to reduce the overhead in using relational databases in ad hoc analytics scenarios: installation, configuration, schema design, tuning, and data ingestion. Queries are submitted through a web interface, allowing collaborative query authoring and avoiding any software installation. The SQLShare interface facilitates and encourages the liberal use of views. Users frequently create deeply nested hierarchies of views to break down complex problems, clean and share intermediate datasets, and record provenance for complex results. SQLShare has been deployed in a number of scientific contexts and has proven to be useful even among SQL first-timers. Howe *et al.* describe the SQLShare architecture and motivation in detail in [9, 12, 11]. [10, 12] describes an in depth use cases of SQLShare view relational view sharing. SQLShare aims at reducing data management overhead in all stages of the data lifecycle shown in Figure 1. While there is no built-in support for visualization, Key *et al.* showed how the SQLShare can be extended to afford automatic visualization [14].

Analysis of this workload [3] showed that users were frequently expressing data curation and cleaning tasks directly in SQL. Since SQLShare encouraged users to upload datasets as is, a significant number of queries submitted to the system were intended to reshape the data, rename columns, remove errant values, and implement other data curation tasks. In this paper, we identify the common data cleaning idioms present in the SQLShare workload and consider how they can be generalized and applied automatically to incoming data.

Listing 1 Example queries for each idiom.

```
Vertical recompositioning:
"SELECT * from [gbc3].[sqlshare-exp.txt]
UNION ALL
SELECT * from [gbc3].[gen_sqlshare.txt]"
Horizontal recompositioning:
"SELECT * FROM [che].[m1]
FULL OUTER JOIN [che].[m3]
ON
[che].[m1].m1_loci_id=[che].[m3].m3_loci_id"
Column rename:
"SELECT column2 as sp, column3 as SPID,
column4 as Prot FROM
[userX].[uniprotolyblastx2.tab]"
NULL injection:
"SELECT CASE WHEN [400 avg NSAF] = 0
THEN NULL
ELSE [2800 avg NSAF]/[400 avg NSAF] END
FROM
[emma].[NSAFwithAve]"
```

Table 1: Frequency of observed idioms (total datasets: 4535)

Idiom	Datasets
Vertical recompositioning	100
Horizontal recompositioning	210
Column rename	720
NULL injection	420

3. CURATING IDIOMS

The ubiquity of weakly structured data in the science use cases required SQLShare to tolerate (and even embrace) upload of weakly structured data. SQLShare encourages users to write SQL queries to repair and reorganize data rather than relying on offline scripts. By mining the workload, we extract generalizable patterns used to perform these repairs and use them to design services to partially automate cleaning tasks.

The SQLShare query corpus presents rich evidence to support this hypothesis. By searching the corpus of 4535 derived datasets (views), we found specific SQL idioms that correspond to schematization tasks: cleaning, typecasting, and integration.

We focus on the following curation patterns extracted from the SQLShare logs. Along with each idiom, we present an example query from the logs in Listing 1, and a method for using the idiom to support curation-on-ingest. Table 1 shows the frequency of occurrences of these idioms.

- *Vertical recompositioning*: Datasets in SQLShare are often representative of scientific processes where one logical dataset arrives in the form of several distinct files arriving at different times. For example, one lab collected data daily from a sensor depoyed in a local estuary. The need to pre-establish a schema and load the data file-by-file makes databases unattractive in these contexts, but SQLShare helped eliminate steps during data ingest. However, users still needed to craft a UNION ALL query to assemble the results, sometimes reordering columns or casting types to align the derived schemas.

  Curation on ingest: By learning these schema alignment heuristics automatically from the data, and applying schema matching methods, these UNION ALL queries can be automatically recommended and applied by the system upon data ingest. One such approach was describe in our previous work on automatically deriving example queries from base data [8].

- *Horizontal recompositioning*: This idiom pertains to horizontally partitioned datasets. As with vertical recompositioning, scientific processes generating the data sometimes produce horizontally split data. Sometimes different labs working on same samples generate different attributes about them. These cases appear in the logs as multiway 1:1 joins.

  Curation on ingest: Suggesting queries for horizontal recompositioning can be non-trivial. However, we can again use the approach shown in [8] to find potential for joins automatically. Automatic join finding using measures like jaccard similarity has been done in the past, combining this approach with a rich hand written query log to suggest data curation idioms is something that can finally make such approaches viable.

- *Column renaming*: It is common for datasets in SQLShare to have no column names in the source files. For this user scenario, SQLShare assigns default column names. Users are encouraged to write SQL to assign semantic names. We find evidence of 1996 uploaded tables, which is about 50% of the total tables, that had at least one default-assigned column name. The number of datasets for which all columns were assigned the default value is 1691. Almost 16% of datasets involve some kind of column renaming step, suggesting that users have adopted SQL as a tool for adding semantics to their data. We find this as sufficient evidence to back our hypothesis that the SQLShare workload contains a rich set of cleanup and curation queries.

  Curation on ingest: While identifying potential columns to rename is easy (columns with the default names are obvious candidates to begin, with a few false positives), suggesting valid renames is a very ambiguous problem. However, since we do have the advantage of having the previous tables and queries written on them. One approach could be to match the range of values of the column to rename to the range of values to previously existing and renamed columns. For example, for an attribute whose range is 0 to 360 and renamed to *'Angle'*, it might make sense to suggest for columns with values in the same domain. Another possible way could to be calculate the earth mover distance [17] between the histograms of column values and suggest rename to column with which this distance is least. There are other principled approaches as in WebTables [7] which uses the *attribute correlation statistics* to suggest schema auto-complete.

- *NULL injection and Type Coercion*: Sentinel values are routinely used to mark missing or inapplicable data; we see string values of "N/A" for example embedded in an otherwise numeric column. The SQL authors can use assemblies of CASE WHEN expressions, filtering, and type casting to replace these values with NULL or otherwise repair them. These constructs are conceptually trivial ("Across all columns, replace the value X with NULL") but hand-writing the corresponding query is tedious and error-prone. After removing bad tuples and replacing missing values with NULL, we find that about 200 of derived datasets used SQL CAST to introduce new types on existing columns.

  Curation on ingest: Our current implementation automatically infers data types based on a prefix of rows, and creates two table. The first table corresponds to the predicted type, and the second table holds non-conforming rows and has every column typed as a string. Finally, a view is created to union the 2 tables and is presented to the user, along with the information about the 2 base tables. This process helps separate the numeric data from the sentinel values, but does not automatically apply the CASE expressions.

3.1 Towards Idiom-Based Data Curation

So far we have shown the evidence of curation via SQL queries in the SQLShare workload. We discussed how these queries can be characterized into common curation idioms and finally we detailed the potential algorithms for curation on ingest.

Tying it all together, the idiom recommendation algorithm would work as follows:

- Identify the common curation idioms, the very first queries on a dataset are often representative of these idioms.
- Generate a query template for each idiom as shown in Qunits [16] and also in SQLShare analysis [12].
- At the time of data ingest, we use the **curation on ingest** techniques in section §3 to identify possible idioms.
- Synthesize the curation queries from the templates and provide them as suggestion to the users. The query synthesis problem has already been solved with multiple examples already available in the literature [6, 4, 5].

This approach of suggesting queries at ingest can save a lot of user time because writing these queries by hand can be repetitive and time consuming. The false positives don't hurt a lot because the user is always in loop and chooses which curation idiom, if any, she wants to apply to her dataset.

In our current implementation, we have a working analysis pipeline [12] and idiom detection. The next steps include integrating this pipeline with the SQLShare system and implementing a query synthesis algorithm. We are actively working on a demo system and hope to present in the immediate future.

4. RELATED WORK

Parsing of complex formats (messy data) to produce weakly structured data for further processing is a problem that has been approached previously, OpenRefine [2] and Wrangler [1] being popular examples of this approach. These tools do not offer support working with multiple datasets and have been shown to have dominant costs [13].

SnipSuggest [15] is an example of system which provides auto completion of queries and has been shown to enable non-experts to write complex SQL. Our work has similar aims, but our approach is to suggest complete queries, automatically synthesized based on previous queries.

WebTables [7] is another work in similar domain, but the focus is to provide automatic schema completion for myriad of documents of the web. We approach suggests a possible use of their schema completion algorithm, but goes beyond just schema completion and provides a richer set of curation idioms.

Akbarnejad *et al.* [5] used a similar approach, *i.e.* using history and preferences to recommend queries. We hope to extend their work and use it an a setting where user is automatically suggested queries on ingest, *i.e.*, the required interaction is minimal, while the queries are still relevant. However, we have one critical advantage in our proposed system, a workload with **real handwritten** queries.

One of our previous works [8] presents the notion of suggesting automatic starter queries, or queries by example, aimed at providing novice users with example and ease the ramping up process. This paper has a similar goal, but our approach has one major difference in that we learn the idioms we are suggesting from the query logs, these idioms are proven to be useful to users, since they have already used them and have the potential to suggest very complex queries (something which the previous approach lacked).

5. CONCLUSION AND FUTURE WORK

We presented a work in progress which uses handwritten queries from a five-year corpus of Database-as-a-Service platform called SQLShare to identify data clean up queries written in SQL. The design choices in SQLShare enabled the users to use databases all stages in the scientific data lifecycle. We present evidence of clean up and curation being done via SQL queries and discuss methods in which these query **idioms** can be used to suggest curation queries to users at the time of data ingest. We talked about the analysis of SQLShare workload and how we mined the queries related to cleaning and curation. We also identified some commonly used idioms which in our opinion should be better supported in databases. Currently we have set up workload analysis pipeline as shown in [12] and have a naive way to find out possible curation queries. Since the clean up queries are usually the very first queries on a dataset, our current methods looks for common idioms amongst these. Our immediate next plan is to extend this work to:

- Incorporate an idiom recommendation process into SQLShare
- Identify other query idioms for scientific use cases. This can be done by clustering embeddings for queries in a higher dimensional space and associating idioms with these clusters.

Our final vision is to have a system which takes in a dataset (plus JBOTs) and suggests curation & scientific analysis queries that can run on it.

6. REFERENCES

[1] Data wrangler. http://vis.stanford.edu/wrangler/.

[2] OpenRefine (formerly google refine). http://openrefine.org/.

[3] Sqlshare workload data release 1. https://uwescience.github.io/sqlshare/data_release.html.

[4] S. Abdul Khalek and S. Khurshid. Automated sql query generation for systematic testing of database engines. In *Proceedings of the IEEE/ACM International Conference on Automated Software Engineering*, ASE '10, pages 329–332, New York, NY, USA, 2010. ACM.

[5] J. Akbarnejad, G. Chatzopoulou, M. Eirinaki, S. Koshy, S. Mittal, D. On, N. Polyzotis, and J. S. V. Varman. Sql querie recommendations. *Proceedings of the VLDB Endowment*, 3(1-2):1597–1600, 2010.

[6] N. Bruno, S. Chaudhuri, and D. Thomas. Generating queries with cardinality constraints for dbms testing. *Knowledge and Data Engineering, IEEE Transactions on*, 18(12):1721–1725, 2006.

[7] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, 1(1):538–549, 2008.

[8] B. Howe, G. Cole, N. Khoussainova, and L. Battle. Automatic example queries for ad hoc databases. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 1319–1322. ACM, 2011.

[9] B. Howe, G. Cole, E. Souroush, P. Koutris, A. Key, N. Khoussainova, and L. Battle. Database-as-a-service for long-tail science. In *Scientific and Statistical Database Management*, pages 480–489. Springer, 2011.

[10] B. Howe, F. Ribalet, D. Halperin, S. Chitnis, and E. V. Armbrust. Sqlshare: Scientific workflow via relational view sharing. *Computing in Science & Engineering, Special Issue on Science Data Management*, 15(2), 2013.

[11] S. Jain, D. Moritz, and B. Howe. High variety cloud databases. In *Proceedings of the 2016 IEEE Cloud Data Management Workshop.*, 2016.

[12] S. Jain, D. Moritz, B. Howe, D. Halperin, and E. Lazowska. Sqlshare: Results from a multi-year sql-as-a-service experiment. In *Proceedings of the 2016 ACM SIGMOD international conference on Management of data*, 2016.

[13] S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Wrangler: Interactive visual specification of data transformation scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3363–3372. ACM, 2011.

[14] A. Key, B. Howe, D. Perry, and C. Aragon. Vizdeck: self-organizing dashboards for visual analytics. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 681–684. ACM, 2012.

[15] N. Khoussainova, Y. Kwon, M. Balazinska, and D. Suciu. Snipsuggest: Context-aware autocompletion for sql. *Proceedings of the VLDB Endowment*, 4(1):22–33, 2010.

[16] A. Nandi and H. Jagadish. Qunits: queried units in database search. *arXiv preprint arXiv:0909.1765*, 2009.

[17] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *Computer Vision, 1998. Sixth International Conference on*, pages 59–66. IEEE, 1998.

[18] M. Stonebraker, D. Bruckner, I. F. Ilyas, G. Beskales, M. Cherniack, S. B. Zdonik, A. Pagan, and S. Xu. Data curation at scale: The data tamer system. In *CIDR*, 2013.