Sigma Workbook: A Spreadsheet for Cloud Data Warehouses

James Gale
Sigma Computing
jlg@sigmacomputing.com

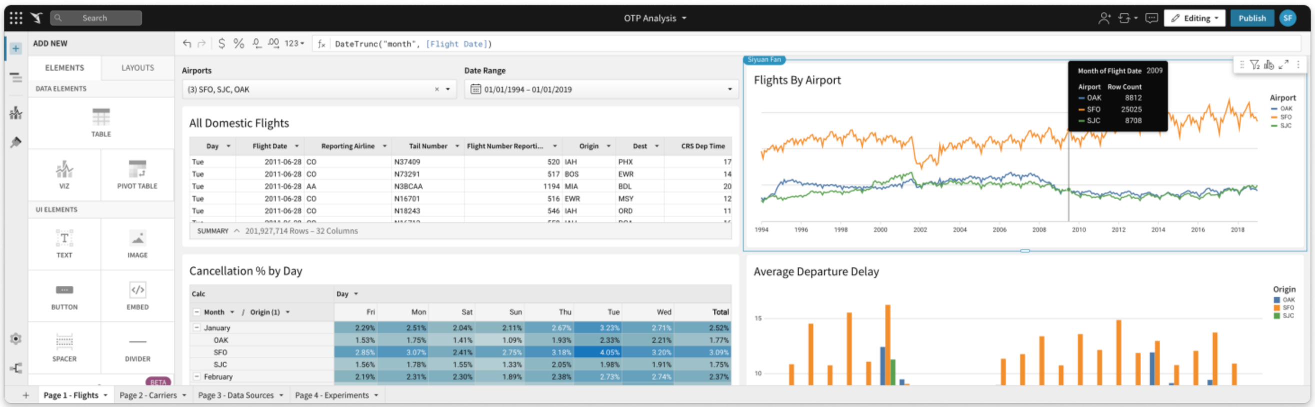
Max Seiden
Sigma Computing
max@sigmacomputing.com

Deepanshu Utkarsh
Sigma Computing
deepanshu@sigmacomputing.com

Jason Frantz
Sigma Computing
jason@sigmacomputing.com

Rob Woollen
Sigma Computing
rwoollen@sigmacomputing.com

Çağatay Demiralp
Sigma Computing
cagatay@sigmacomputing.com



```
DateTrunc("month", [Flight Date])
```

Airports: (3) SFO, SJC, OAK

Date Range: 01/01/1994 – 01/01/2019

All Domestic Flights

Day	Flight Date	Reporting Airline	Tail Number	Flight Number Reporti...	Origin	Dest	CRS Dep Time
Tue	2011-06-28	CO	N37409	520	IAH	PHX	17
Tue	2011-06-28	CO	N73291	517	BOS	EWR	14
Tue	2011-06-28	AA	N3BCAA	1194	MIA	BDL	20
Tue	2011-06-28	CO	N16701	516	EWR	MSY	12
Tue	2011-06-28	CO	N18243	546	IAH	ORD	11

SUMMARY 201,927,714 Rows – 32 Columns

Cancellation % by Day

Calc	Day							
Month / Origin (1)	Fri	Mon	Sat	Sun	Thu	Tue	Wed	Total
January	2.29%	2.51%	2.04%	2.11%	2.67%	3.23%	2.71%	2.52%
OAK	1.53%	1.75%	1.41%	1.09%	1.93%	2.33%	2.21%	1.77%
SFO	2.85%	3.07%	2.41%	2.75%	3.18%	4.05%	3.20%	3.09%
SJC	1.56%	1.78%	1.55%	1.33%	2.05%	1.98%	1.91%	1.75%
February	2.19%	2.31%	2.30%	1.89%	2.38%	2.73%	2.74%	2.37%

Month of Flight Date 2009

Airport	Row Count
OAK	8812
SFO	25025
SJC	8708

Figure 1: Sigma Workbook is an interactive workspace for analyzing enterprise-scale data in cloud data warehouses (CDWs). Its interface builds on spreadsheets while automatically compiling data operations to SQL queries and executing them on the CDW. Workbook enables users to benefit from the characteristics of SQL and CDWs using their knowledge of spreadsheets.

ABSTRACT

Cloud data warehouses (CDWs) bring large-scale data and compute power closer to users in enterprises. However, existing tools for analyzing data in CDWs are either limited in ad-hoc transformations or difficult to use for business users. Here we introduce Sigma Workbook, a new interactive system that enables business users to easily perform visual analysis of data in CDWs at scale. For this, Sigma Workbook provides an accessible spreadsheet-like interface for analysis through direct manipulation. Sigma Workbook dynamically constructs matching SQL queries from user interactions, building on the versatility and expressivity of SQL. Constructed queries are directly executed on CDWs, leveraging the superior characteristics of the new generation CDWs, including scalability. We demonstrate Sigma Workbook through 3 real-life use cases—cohort analysis, sessionization, and data augmentation—and underline Workbook's ease of use, scalability, and expressivity.

PVLDB Reference Format:
James Gale, Max Seiden, Deepanshu Utkarsh, Jason Frantz, Rob Woollen, and Çağatay Demiralp. Sigma Workbook: A Spreadsheet for Cloud Data Warehouses . PVLDB, 15(12): 3670-3673, 2022.
doi:10.14778/3554821.3554871

Proceedings of the VLDB Endowment, Vol. 15, No. 12 ISSN 2150-8097.
doi:10.14778/3554821.3554871

1 INTRODUCTION

Enterprise data is increasingly stored in cloud data warehouses (CDWs) as they enable the storage of large-scale datasets with reliability and compliance guarantees while reducing costs. Business users (non-technical domain experts such as operations associates, marketing managers, and product managers) in enterprises wish to use this data for augmenting their decision-making, ideally without going through analysts. Many of these users are comfortable doing analyses using spreadsheets. However, due to their limited scalability and expressivity [5, 14], existing spreadsheet applications aren't adequate for accessing and analyzing data residing within CDWs.

In this paper, we introduce Sigma Workbook[1] (Workbook for short), a SaaS system (Figure 1) that enables business users to perform interactive ad-hoc analysis on datasets stored in CDWs. It aims to effectively combine ease of use, expressivity, and scalability in order to support iterative visual data analysis of enterprise data. To enhance accessibility, Workbook integrates an easy-to-use, intuitive interface with affordances that have made spreadsheet applications successful [19], including a simple expression language embedded in a table of values, easy references, easy refactoring, and isolation of errors. Workbook dynamically compiles data operations interactively specified through this interface into SQL queries, building on the versatility and expressivity of SQL. This amplifies

[1] A demo video of Sigma Workbook is available at: https://tinyurl.com/sigma-workbook.

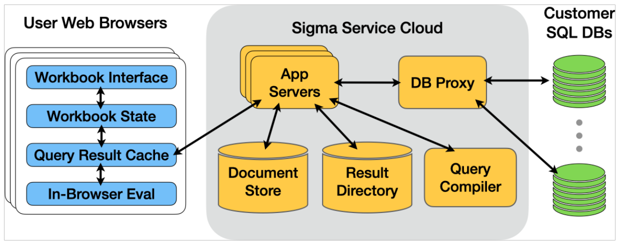


Figure 2: Sigma Workbook architecture. Web browsers running the Sigma application use Sigma's multi-tenant cloud service to interface with the user's own CDW. The app server acts as an intermediary for query requests that are compiled to SQL and a proxy performs workload scheduling and interfacing with the database. Query results are returned to the result cache in the browser app and presented to the user.

users' ability to generate complex queries that can be otherwise daunting to manually specify.

Sigma Workbook executes compiled queries on CDWs, directly leveraging their desirable properties such as scalability, security (e.g., compliance with regulations such as HIPPA and GDPR), elasticity, and reliability [7, 8]. This direct, interactive interface to the CDW differentiates Workbook from the architecture of current BI systems, which can produce beautiful visualizations and dashboards but then be limited when users want to get to the "row-level" data behind their dashboards. Unlike spreadsheet applications, Sigma Workbook enables users to explore billions of records or terabytes of data. In this sense, Sigma Workbook is an accumulation of the decades-long ideas proposing to combine the accessibility of spreadsheet-like direct manipulation with the characteristics of database systems that enterprises rely on [3, 4, 13, 16, 18, 22].

We demonstrate Workbook here with 3 use cases: cohort analysis, sessionization, and augmentation with user-created data. We use a dataset of 200M flight records for all the use cases.

2 SYSTEM OVERVIEW

Workbook is the unified interface for exploring and presenting data in Sigma, a multi-tenant Software-as-a-Service (SaaS) web application for BI. The elements of the Workbook implementation (Figure 2) support the interactive construction, composition and visualization of SQL queries through direct manipulation.

Sigma customers configure the service with access to a CDW they control. No pre-processing or ingestion is required before users can begin OLAP through Sigma. Sigma allows multiple warehouse configurations per customer, currently supporting Databricks [6], BigQuery [12, 17], PostgreSQL [20], Redshift [1] and Snowflake [7, 21]. The elastic and scalable nature of CDWs enables them to support diverse, interactive OLAP workloads from large numbers of users. Sigma benefits from these desirable characteristics by pushing computation to CDWs.

The Workbook interface enables users to find and reference the tables within their database to construct new analyses (Section 3). Workbook state can be saved and restored as a document. These documents can be named and organized in a file system within Sigma and may be shared or copied. Unnamed Workbook documents are stored as persistent, anonymous "explorations" which can be easily discarded.

Access to the customer's data warehouse by the Sigma web application is always mediated by the Sigma service. Interactive data operations expressed by a user are sent to the Sigma service as a JSON-encoding of the Workbook state. The Sigma service performs authentication, access control checks, query input graph resolution, and materialized view substitution. The validated, fully resolved query graph is compiled into a corresponding SQL query. The SQL query is then placed into a workload management queue and subsequently executed in the customer's database. The query results are fetched from the database and forwarded directly back to the web application for presentation.

3 WORKBOOK INTERFACE

The Workbook interface is designed with accessibility in mind, incorporating successful features of spreadsheets to enable business users to easily explore and analyze tables in relational databases. It provides a canvas where a user can add and arrange elements of different types. There are 3 categories of Workbook elements: (1) data elements, including tables, visualizations, pivot tables, and user-created tables; (2) user interface elements, including text, images and spacers; and (3) interactive control elements, including sliders, lists, text inputs, date pickers, and drill-downs. Users can partition the canvas into pages to organize their analysis.

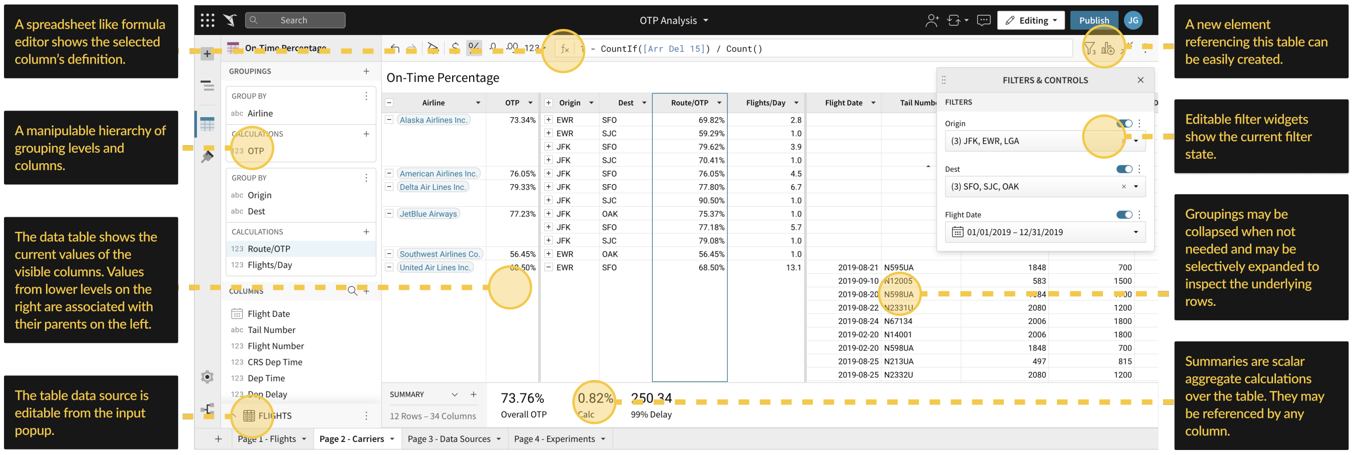
3.1 Tables

The Workbook table element (Figure 3) is an evolution of our earlier work [9] enabling users to interactively construct and manipulate database queries. Workbook tables are defined by 3 constructs: (1) the grouping levels of the table; (2) the formula definitions for columns in the table; and (3) zero or more filters applied to the table. The specification also describes the input data sources and various data formatting options.

Grouping Levels To perform aggregation and window calculations, users need a mechanism to specify grouping and sorting in the Workbook. In a Workbook table, users define a list of grouping levels (level for short) that visually arrange records in a nested fashion. A level specifies a grouping key, map of columns, and an ordering annotation—this enables expressions such as `CountDistinct` or `MovingAverage` to derive grouping and ordering properties.

Levels in the Workbook table are organized in a hierarchy. The table always has at least two levels. The lowest level is known as the *base*, and initially contains only columns that reference the input data source. It is the only level that does not have keys and is not aggregated. The highest level, known as the *summary*, has an empty key set and is used to calculate scalar aggregates. There can be zero or more levels between these two. The only restriction is that level keys must reference columns from a lower level.

Columns Each table column is defined by an expression, its visibility, and its "resident level." Column expressions, known as formulas, are written in an expression language familiar to users of spreadsheet and BI tools. Like SQL, supported functions fall into one of three categories: single row, aggregate, and window. However,


A spreadsheet like formula editor shows the selected column's definition.

A manipulable hierarchy of grouping levels and columns.

The data table shows the current values of the visible columns. Values from lower levels on the right are associated with their parents on the left.

The table data source is editable from the input popup.

A new element referencing this table can be easily created.

Editable filter widgets show the current filter state.

Groupings may be collapsed when not needed and may be selectively expanded to inspect the underlying rows.

Summaries are scalar aggregate calculations over the table. They may be referenced by any column.

```
1 - CountIf([Arr Del 15]) / Count()
```

On-Time Percentage

Airline	OTP	Origin	Dest	Route/OTP	Flights/Day
Alaska Airlines Inc.	73.34%	EWR	SFO	69.82%	2.8
		EWR	SJC	59.29%	1.0
		JFK	SFO	79.62%	3.9
		JFK	SJC	70.41%	1.0
American Airlines Inc.	76.05%	JFK	SFO	76.05%	4.5
Delta Air Lines Inc.	79.33%	JFK	SFO	77.80%	6.7
		JFK	SJC	90.50%	1.0
JetBlue Airways	77.23%	JFK	OAK	75.37%	1.0
		JFK	SFO	77.18%	5.7
		JFK	SJC	79.08%	1.0
Southwest Airlines Co.	56.45%	EWR	OAK	56.45%	1.0
United Air Lines Inc.	68.50%	EWR	SFO	68.50%	13.1

Flight Date	Tail Number		
2019-08-21	N595UA	1848	700
2019-09-10	N12005	583	1500
2019-08-20	N598UA	[illegible]84	[illegible]00
2019-08-22	N2331U	2080	1200
2019-08-24	N67134	2006	1800
2019-02-20	N14001	2006	1800
2019-02-20	N598UA	1848	700
2019-08-25	N213UA	497	815
2019-08-25	N2332U	2080	1200

FILTERS & CONTROLS

FILTERS

Origin: (3) JFK, EWR, LGA

Dest: (3) SFO, SJC, OAK

Flight Date: 01/01/2019 – 12/31/2019

SUMMARY: 73.76% Overall OTP; 0.82% Calc; 250.34 99% Delay

12 Rows – 34 Columns

Figure 3: The Workbook table element (annotated) is a query defined by grouping levels, columns, and filters.

much like a spreadsheet, there are no restrictions on how these functions are composed.

Filters The Workbook table provides specialized filter widgets that apply a predicate to a column's values to select records from the data table. As the table does not allow users to specify an explicit order of operations, Workbook implements a behavior that is explainable and predictable: filters are applied in a greedy manner, as soon as their dependencies are met.

Data Sources Every Workbook table has a data source, being either a database table, a SQL query, an uploaded CSV file, or another Workbook data element. Additional inputs can be included from the same types of sources via joins or unions.

3.2 Ad-hoc Joins with Lookup and Rollup

Column formulas can use a special function, `Lookup`, to pull-in values from other elements in a manner similar to the common spreadsheet function, VLOOKUP. `Lookup` expressions behave like a foreign-key left-join in that they never affect the cardinality of the query.

`Rollup` is similar to `Lookup`, and takes an aggregate expression to be evaluated against the join target. In fact, `Lookup` is a special case of `Rollup` with the virtual aggregate ATTR wrapping the scalar expression. Both `Lookup` and `Rollup` support self-joins.

These formulas can be input directly by the user, but Workbook also includes guided interfaces to help users construct and re-use these relationships between elements.

3.3 Visualizations and Pivot Tables

Workbook visualization elements use Vega [15] and support common visualization types. Pivot tables can also be defined. Like tables, visualization and pivot table elements include columns and filters. Similarly, both elements have a data source and may be a source for other elements.

3.4 Ad-hoc Data

Workbook supports directly adding and updating data, enabling users to augment their analysis with provisional data and run what-if scenarios. Workbook users can create free-form, editable tables, which are projected into the warehouse. When the values in these tables are changed (e.g., by editing in values or copy-and-pasting from a spreadsheet), the edits are propagated to the warehouse. Users can also add their own CSV data as sources to any workbook element. The parsed file is transparently marshaled into the user's warehouse as a database table.

3.5 Presentation and Collaboration

Workbook supports collaborative analysis. Editing of a Workbook document is "multi-player," allowing editors to observe each other's changes in real-time. User can create comments on elements in a document and view the history of edits. Workbook supports a viewer mode tailored for presentations while allowing some limited, transient exploration. Workbook also supports live document embedding into other websites.

Layout Workbooks are edited in a "desktop" web browser but may be viewed in many form factors, including mobile devices. Workbook elements are laid out as a sequence of sections, each divided into a number of columns, similar to interactive website builders. The layout is responsive to varying screen sizes so that the elements are still legible and interactable. Workbook offers the inclusion of presentation elements, such as images, text, and spacers. Text elements may include embedded formulas, rendering results of these formulas inline.

Interactive Controls Workbook elements may be wired to control elements, such as text input, date selectors, or slider controls, enabling the creation of "dashboard" style applications. The controls can be referenced by column formulas and can be set by parameters to the Workbook document URL.

4 SCALABILITY

The Workbook implementation has several components to facilitate scalable, interactive collaboration. We discuss two of them here.

Caching Workbook employs a hierarchy of caching to reduce the load on the user's database. An important constraint on our implementation is that user warehouse data is never stored within the Sigma service cloud. The first level of caching is within the browser itself. Recent query results are remembered and re-used, helping the interactivity of undoing operations or switching to a previous page. The second level is a directory of recent queries maintained by the Sigma app server. The directory points to available result sets, stored in the CDW by their query-id, which can be re-fetched as requested. It also tracks in-flight query requests, enabling multiple browsers to share results when collaboratively editing a document. Finally, the result sets of user-selected Workbook elements can be materialized into a warehouse table. The queries for elements that reference the element are automatically re-written by the Workbook compiler to use these tables. The materialization can be configured by the user to refresh on a schedule.

In-Browser Evaluation The browser query-result cache is augmented with an evaluation engine, written in C++ and compiled to WebAssembly [10], which in many cases can synthesize new results from existing rows already fetched from the CDW. These local evaluations avoid the latency of a round-trip to the database, and facilitate the interactivity of workbook editing. In some cases (e.g. lower cardinality tables), we are able to prefetch a resultset that could be used to fully evaluate all future operations on the table locally in the browser.

5 DEMONSTRATION

We demonstrate the capabilities of Sigma Workbook for common business intelligence analysis through 3 example scenarios, highlighting Workbook's ease of use, scalability, and expressivity. In all the scenarios, we use the On-Time database of the United States domestic airline carrier flights between 1987–2020 [2]. In each scenario, we also show the SQL queries generated by our compiler to produce these results.

Scenario 1: Cohort Analysis Cohort analysis is a common analysis with longitudinal datasets. It involves grouping data into subsets with similar characteristics and comparing how the groups change over time. The cohort scenario is important enough that earlier research proposed to extend SQL with new operators to support it [11]. Sigma Workbook has no optimization special to cohort analysis, but it can be expressed simply with a few basic aggregate expressions. This analysis is also possible in Power BI but requires comparatively complex DAX formulas [23].

We demonstrate this analysis in Workbook as follows: (1) Starting with the `FLIGHTS` fact table, we create a self-join using Workbook's `Rollup` function to identify the date of the first flight for each plane. This date, truncated to the quarter-year, identifies the cohort for each plane; (2) We then create a hierarchy of grouping levels, first grouping by cohort and then by flight date truncated by quarter. We compute the total population of planes in each cohort and, using cross-level references, the percentage active in each quarter; (3) Finally we create a scatter-plot over this dataset, colored by active population, presenting the synthesized result from over 200M rows of raw data.

Scenario 2: Sessionization Sessionization is an enrichment where events in time, associated with an entity, are grouped into time periods known as "sessions." This is useful in marketing, security, and other applications and is often performed by special-purpose analysis systems. Relating rows within a SQL database table requires self-joins or window expressions, both of which are difficult to express in the language and in many BI systems.

We demonstrate this analysis in Workbook: (1) Starting with the `FLIGHTS` table, we create a grouping by airplane tail number and then order the base level by flight date. We infer aircraft servicings from periods of inactivity by adding a window calculation, `Lag` of flight date, and comparing the result with the current flight date. We mark all flights with the time of service using another window calculation, `FillDown`, as a "session identifier"; (2) In a child table element we group first by these discovered sessions and then by cumulative air-time since service was done, and compute cancellation rates for flights at different times in the service lifecycle. We show that users can inspect the rows at each level of aggregation, down to the base; (3) We visualize this result with a line chart showing how cancellations change with flight hours.

Scenario 3: Augmenting Warehouse Data Workbook allows users to enrich the shared data of the warehouse with their own data sources to "contextualize" this data. It enables this in a way familiar to spreadsheet users.

We demonstrate how to use Workbook to augment warehouse data with external data sources. (1) First we inspect the FLIGHTS records in workbook and we discover that they are missing some desired dimensional data about the airports; (2) So we perform a web search and find a plausible dataset that is copied into an editable Workbook table; (3) Now we join the new values into the fact table via a `Lookup` expression; (4) Upon further inspection we notice the pasted data is "dirty" and correct it with direct editing. We show that these edits propagate to downstream queries automatically.

REFERENCES

[1] Amazon. 2013–2022. Redshift. https://aws.amazon.com/redshift.
[2] Bureau of Transportation Statistics. 1987–2020. Airline On-Time Performance Data. https://www.transtats.bts.gov/ONTIME.
[3] Bakke et al. 2011. A spreadsheet-based user interface for managing plural relationships in structured data. In *CHI*.
[4] Bendre et al. 2015. Dataspread: Unifying databases and spreadsheets. In *VLDB*.
[5] Bendre et al. 2018. Towards a holistic integration of spreadsheets with databases: A scalable storage engine for presentational data management. In *ICDE*.
[6] Behm et al. 2022. Photon: A Fast Query Engine for Lakehouse Systems. In *SIGMOD*.
[7] Dageville et al. 2016. The Snowflake Elastic Data Warehouse. In *SIGMOD*.
[8] Gupta et al. 2015. Amazon redshift and the case for simpler data warehouses. In *SIGMOD*.
[9] Gale et al. 2021. Sigma Worksheet: Interactive Construction of OLAP Queries. arXiv:2012.00697
[10] Haas et al. 2017. Bringing the Web up to Speed with WebAssembly. In *SIGPLAN*.
[11] Jiang et al. 2016. Cohort Query Processing. In *VLDB*.
[12] Melnik et al. 2010. Dremel: interactive analysis of web-scale datasets. In *VLDB*.
[13] Raman et al. 1999. Scalable Spreadsheets for Interactive Data Analysis. In *SIGMOD*.
[14] Rahman et al. 2020. Benchmarking Spreadsheet Systems. In *SIGMOD*.
[15] Satyanarayan et al. 2016. Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization. In *IEEE TVCG (Proc. InfoVis)*.
[16] Witkowski et al. 2003. Spreadsheets in RDBMS for OLAP. In *SIGMOD*.
[17] Google. 2011–2022. BigQuery. https://cloud.google.com/bigquery.
[18] Bin Liu and HV Jagadish. 2009. A spreadsheet algebra for a direct data manipulation query interface. In *ICDE*.
[19] Bonnie A. Nardi and James R. Miller. 1990. The Spreadsheet Interface: A Basis for End User Programming. In *INTERACT*.
[20] PostgreSQL. 1996–2022. PostgreSQL. https://www.postgresql.org/.
[21] Snowflake. 2015–2022. Snowflake. https://www.snowflake.com/workloads/data-warehouse-modernization/.
[22] Jerzy Tyszkiewicz. 2010. Spreadsheet as a relational database engine. In *SIGMOD*.
[23] Luca Zanna. 2019. Power BI: Chort Analysis. https://finance-bi.com/power-bi-cohort-analysis.