# Towards an Objective Metric for Data Value Through Relevance

Boris Glavic*, Pengyuan Li*, Ziyu Liu*, Dieter Gawlick$^{\alpha}$, Vasudha Krishnaswamy$^{\alpha}$, Danica Porobic$^{\alpha}$, Zhen Hua Liu$^{\alpha}$
Illinois Institute of Technology*, Oracle$^{\alpha}$
USA
bglavic@iit.edu,{pli26,zliu102}@hawk.iit.edu,{dieter.gawlick,vasudha.krishnaswamy,danica.porobic,zhen.liu}@oracle.com

## ABSTRACT

The rate at which humanity is producing data has increased significantly over the last decade. As organizations generate unprecedented amounts of data, storing, cleaning, integrating, and analyzing this data consumes significant (human and computational) resources. At the same time organizations extract significant *value* from their data. In this work, we present our vision for developing an *objective metric for the value of data* based on the recently introduced concept of *data relevance*, outline proposals for how to efficiently compute and maintain such metrics, and how to utilize data value to improve data management including storage organization, query performance, intelligent allocation of data collection and curation efforts, improving data catalogs, and for making pricing decisions in data markets. While we mostly focus on tabular data, the concepts we introduce can also be applied to other data models such as semi-structure data (e.g., JSON) or property graphs. Furthermore, we discuss strategies for dealing with data and workloads that evolve and discuss how to deal with data that is currently not relevant, but has potential value (we refer to this as *dark data*). Furthermore, we sketch ideas for measuring the value that a query / workload has for an organization and reason about the interaction between query and data value.

**ACM Reference Format:**
Boris Glavic*, Pengyuan Li*, Ziyu Liu*, Dieter Gawlick$^{\alpha}$, Vasudha Krishnaswamy$^{\alpha}$, Danica Porobic$^{\alpha}$, Zhen Hua Liu$^{\alpha}$. 2024. Towards an Objective Metric for Data Value Through Relevance. In *Proceedings of CIDR*. ACM, New York, NY, USA, 9 pages.

## 1 INTRODUCTION

As organizations produce unprecedented amounts of data, managing that data and utilizing it to improve the operations of the organization has become a critical challenge. While data catalogs and metadata management systems like Apache Atlas [1] and Goods [15] enable organizations to track their data and its usage, these organizations still have to decide how to spend limited resources (storage, compute, human time, ...) most effectively on what data. For instance, data collection and curation require significant manual labor and, thus, should be focused on data that of high importance to the operations of a business. What is missing, is an objective metric for the **value** that data has to an organization or, more specifically, for a particular purpose within the organization (e.g., deciding what products to promote). Such a metric would allow resources to be spent where they are most needed. For example, it is sensible to spend human effort on cleaning and collecting data that is important while data that is of little value can be subjected to lossy compression, stored on slow storage media, and may not need to be curated.

In this work, we present our vision for an **objective metric** of data value. We argue that the only value of a piece of data is its contribution to producing the results of queries (and other computations). Data that is only stored, but not used in computations in any meaningful way, is unobservable to the user: deleting this data would not affect the results of computations.

Insight 1. *Data is typically accessed through some type of interface, e.g., a declarative query language. Thus, data only has value by contributing to a result returned through such an interface.*

We base our definition of data value on the notion of data relevance that we recently introduced in [19]. Data relevance uses provenance [9] to track which data is needed to produce the answer of a computation. Note that this is different from techniques that track how "hot" data is: just because data is accessed by a query that does not mean that the data is actually relevant for producing the query result.

Insight 2. *Data access is not the same as data relevance. Only data that contributes to the result of a computation is of value to this computation.*

For instance, a database system may evaluate a selection query using a full table scan as this can be more efficient than using an index, but data items not fulfilling the selection conditions of the query do not contribute to the query's result.

Example 1. *Consider the webshop database and workload shown in Figure 1. The workload consists of two queries: $Q_{top}$ (executed daily) determines products that achieve significant revenue. $Q_{problem}$ (executed once per week) identifies the product with the lowest average rating of reviews from Europe that mention the word "problem". $Q_{top}$ returns two products (with* `pid` *1 and 3). The relevant data for this query is highlighted in light blue. The product rows $p_1$ and $p_3$ and all three order items for these products are needed to produce the query's result. As the reader may verify, any other product as well as all review tuples can be deleted without changing the result. The relevant data for $Q_{problem}$ for this database is the product with* `pid`

$Q_{top}$

```
SELECT pid, sum(quantity * price) AS rev
FROM products NATURAL JOIN orders
GROUP BY pid HAVING sum(quantity * price) > 1000
```

$Q_{problem}$

```
SELECT pid, name, avg(rating) AS avg_rating
FROM products NATURAL JOIN reviews
WHERE region = 'Europe' AND review LIKE '%problem%'
GROUP BY pid, name
ORDER BY avg_rating ASC LIMIT 1;
```

**products**

| | pid | name | category | price |
|---|---|---|---|---|
| $p_1$ | 1 | Asus S10 | laptop | 499 |
| $p_2$ | 2 | Dell SP4400 | desktop | 1230 |
| $p_3$ | 3 | Samsung T6 | ssd | 199 |

**orders**

| | oid | pid | quantity |
|---|---|---|---|
| $o_1$ | 1 | 1 | 1 |
| $o_2$ | 2 | 1 | 3 |
| $o_3$ | 2 | 3 | 10 |

**reviews**

| | pid | region | rating | review |
|---|---|---|---|---|
| $r_1$ | 1 | Europe | 3.4 | machine problems … |
| $r_2$ | 1 | Asia | 4.0 | problem with … |
| $r_3$ | 3 | Europe | 1.3 | problems after … |
| $r_4$ | 3 | Europe | 3.8 | the problem is … |

**Figure 1: Example workload and relevant subsets of the database for the queries of the workload. By aggregating the relevance information we obtain an objective measure of the value of the data for the workload.**

*3 and the two ratings from Europe for this product that contain the word* "problem".

As shown in this example, only a fraction of the data in the database is needed to produce the results of the queries for the example application and, thus, only this subset of the data is of value to the organization.

Insight 3. *By tracking what data is needed to answer a query, relevance provides an objective framework for data value.*

In Section 2 we will briefly discuss how the notion of relevance can be formalized and how it relates to provenance and attribution measures for data. By aggregating the data relevance weighted by query frequency, we obtain the expected / mean relevance of data which we use as a metric of data value. Data $d$ that is not relevant for any query of the workload will have a value $\mathcal{V}(d) = 0$ while data that is needed for answering every query in the workload will have the maximum achievable value of $\mathcal{V}(d) = 1$. Note that this definition of data value assumes that all queries are of equal importance to the organization which may not be the case. Note that our relevance-based metric does help us to identify data that is irrelevant for the current workload, but that has *potential value* if the workload were to be updated to use this data. We will discuss query value and potential value in Section 1.4 and section 6, respectively.

Example 2 (Data Value). *Recall that $Q_{top}$ is executed once per day and $Q_{problem}$ once per week. Thus, their relative frequencies are $F_{Q_{top}} = \frac{7}{8}$ and $F_{Q_{problem}} = \frac{1}{8}$. Consider computing the value $\mathcal{V}(p_3)$ of tuple $p_3$ as shown below. As $p_3$ is relevant for both queries, it has value 1. As another example consider tuple $o_1$. This tuple is only relevant for $Q_{top}$ and, thus, its value is equal to $Q_{top}$'s frequency ($\frac{7}{8}$).*

$$\mathcal{V}(p_3) = \frac{7}{8} + \frac{1}{8} = 1 \qquad \mathcal{V}(o_1) = \frac{7}{8}$$

An important property of this metric of value is that under certain assumptions it induces a monetary metric of data value. For instance, assume that the monetary value of getting the correct result for query $Q$ is $\$(Q)$, the value of getting an incorrect result is \$0, and that missing any relevant input to a query will lead to an incorrect result. Under these assumptions, if we additionally weight each query by $\$(Q)$, then the value $\mathcal{V}(t)$ of a row $t$ has a precise monetary interpretation: if we delete $t$, then we loose $\$\mathcal{V}(t) \cdot \mid \mathcal{Q} \mid$ because of incorrect query answers for a workload $\mathcal{Q}$ which consists $\mid \mathcal{Q} \mid$ instances of queries.

Example 3 (Data Value). *Let us assume the following monetary benefits associated with receiving the correct answers to $Q_{top}$ and $Q_{problem}$:*

$$\$(Q_{top}) = \$200 \qquad \$(Q_{problem}) = \$300$$

*By weighting the contribution of each query $Q$ by $\$(Q)$ in the computation of the value for each row, we get the following value for rows $p_3$ and $o_1$:*

$$\begin{aligned}\mathcal{V}(p_3) &= \frac{7}{8} \cdot \$(Q_{top}) + \frac{1}{8} \cdot \$(Q_{problem}) \\ &= \frac{7}{8} \cdot \$200 + \frac{1}{8} \cdot \$300 = \$212.5 \\ \mathcal{V}(o_1) &= \frac{7}{8} \cdot \$(Q_{top}) = \frac{7}{8} \cdot \$200 = \$175\end{aligned}$$

*Over the course of a week (7 instances of $Q_{top}$ and 1 instance of $Q_{problem}$), we would loose $8 \cdot \$175 = \$1400$ if $o_1$ is deleted, e.g., we decide to save storage by deleting some rows or determine that getting a cleaned version of $o_1$ is not worth the effort.*

Of course, the above example relies on several idealized assumptions that are unlikely to hold in the real world: (i) we are able to determine the monetary value of a query result and (ii) query results have zero value unless they are 100% correct. Assumption (ii) can be relaxed if the value of partially incorrect answers is known and using attribution metrics for facts like Shapley value [3] or responsibility [6, 8] to measure the relative contribution of a row towards a query result. Nonetheless, the example illustrates the benefits of basing data value on an **objective** metric that models relevance of rows for query results based on the well-established concepts of data provenance [9] and/or responsibility notions.

Insight 4. *A data value metric based on relevance extends naturally to monetary metrics when the value of query results is known.*

## 1.1 Utilizing Data Value

An objective metric for the data value has many potential applications in data management. It can be used to improve query performance by restricting query evaluation to data that is relevant, for informed allocation of resources for data curation, integration, and collection based on how valuable data is, to enrich data discovery and data catalogs, to make value-based data life-cycle and self-tuning decisions, and to help buyers in data markets to decide how valuable a dataset is to them.

Example 4 (Applications of Data Value). *Observe that in our running example roughly half of the data has zero value (all rows without colored background). Such data could be heavily compressed to save storage space. Using the techniques from [19], that we will introduce in more detail in Section 2, we can filter irrelevant data early-on during query processing (while state-of-the-art DBMS can only filter data based on selection conditions). Furthermore, we can adapt the physical design of the data to improve access to valuable data, e.g., we could partition the data to cluster data based on their value and index only partitions with high data value.*

In summary, in any use case where decisions are based on how frequently data is accessed, we can improve the decision-making process by replacing data access frequency with our data value metric. For instance, self-tuning physical design techniques, e.g., automatic data partitioning [20], typically rely on access patterns ignoring whether the accessed data is relevant or not.

Example 5 (Self-tuning & Data Life-Cycle Management). *Continuing with our running example, let us assume that we want to implement an in-memory cache for data that is optimized wrt. our workload ($Q_{top}$ and $Q_{problem}$). Let us assume that the DBMS uses index nested-loop joins for both queries utilizing a primary key index for table* `products` *on column* `pid`*. Query $Q_{problem}$ has to scan the full reviews table even though only rows $r_3$ and $r_4$ are needed to produce the query answer. An approach that makes caching decisions based on data access would assume that caching any review row has the same benefit. However, when using provenance-based data skipping to determine what irrelevant data can be skipped, we can decide to only cache rows $r_3$ and $r_4$ as the other two rows of the review table are not needed to produce the result of $Q_{problem}$.*

While we made some simplifying assumptions in the example above, it nonetheless demonstrates how data relevance can be used to optimize self-tuning decisions like caching, index and materialize-view selection, automated partitioning and many others.

Insight 5. *Many aspects of data management can be improved using an objective metric of data value. Virtually every self-tuning approach that is based on data access can make more informed decisions by utilizing data value.*

## 1.2 Measuring and Managing Data Value

As our data value metric relies on tracking relevance instead of tracking data access only, it is more challenging to compute as it requires capturing data provenance. Extending the idea of provenance sketches introduced in [19], we discuss strategies for measuring data value at coarser granularity and outline our vision for efficient management of data value information. Furthermore, we discuss the impact of such over-approximations on data value metrics and applications of data value. Computing data value for each query in a workload is often infeasible. We will discuss potential strategies for reducing the overhead by measuring data value for a carefully selected sample of queries, e.g., grouping queries based on similar data relevance patterns.

Furthermore, we need to reason about whether the relevant data for a query $Q_1$ can be utilized to answer a different query $Q_2$. In [19], we did discuss one such technique that determines statically whether the set of data items that are relevant for $Q_1$ is sufficient for answering $Q_2$. We can exploit such information to reuse data relevance information across queries in a workload, thus, reduce the cost of managing data value.

Insight 6. *It is important to understand how the queries in a workload relate to each other with respect to what data is relevant for them. For instance, if the relevant data for one query $Q_1$ is guaranteed to be a subset of the relevant data for another query $Q_2$, then we can answer $Q_2$ over the relevant data for $Q_1$.*

## 1.3 Maintaining Data Value

Our metric for data value is specific to a particular workload and state of the database. However, data and workloads are typically not static, but evolve over time. We will discuss in Section 4.3 how (approximate) data value metrics can be maintained when the data and/or workload changes. Using standard provenance techniques [9, 19], data value can be computed using queries. Thus, standard incremental view maintenance techniques can be applied to maintain data value metrics under data changes by maintaining the results of the queries that compute these metrics. However, novel optimizations are possible if data value is computed at a coarser granularity or if we allow for approximation. To deal with evolving workloads, we may be able to exploit similarities of queries wrt. their relevance *"profiles"*.

Insight 7. *Data value metrics have to be maintained when the workload or the data evolve. Novel incremental maintenance techniques are needed for the maintenance of approximate and coarse-grained value information, and to handle changes in the workload.*

## 1.4 Query Value and Dealing with Dark Data

Initially, we make the simplifying assumption that all queries in a workload are of equal importance. However, in practice this is often not the case, e.g., queries for order processing and advertisement may be more important for the operations of an online retailer than queries that analyze the effectiveness of user support. We discuss how to extend our data value metric to incorporate information about the importance of queries and outline techniques for determining data and query value if both are unknown upfront. Furthermore, data value can identify data that is very infrequently used or not used at all (we refer to this as *dark data*). We envision the integration of data value into data discovery and recommendation systems that systematically recommend data to users to identify whether data is dark because it is of no use what data has potential value, but has been underutilized.

We make the following contributions in this work:

- We introduce an **objective metric of data value** based on what data is **relevant** for which query of a given workload.
- We discuss strategies for efficiently computing data value and maintaining it when workloads and / or data evolve.
- We discuss several applications where an objective data value metric brings significant benefits including query processing, resource allocation for data cleaning and preparation, self-tuning and data life-cycle management, enriching data catalogs, and assessment of data value in data markets
- We discuss how to extend our framework to also model the **value of queries**.
- We sketch ideas for utilizing data value to address the problem of **dark data**, i.e., data that is not utilized or underutilized, and discuss potential techniques for measuring the value of queries for an organization and how to integrate this with data value metrics.

## 2 DATA RELEVANCE & PROVENANCE

We now briefly review the concept of data relevance from [19]. Intuitively, a data item $d$ from a database $D$ is relevant for a query $Q$ if "we need $d$ to compute $D$". The notion of relevance is closely related to data provenance and indeed we will use *"belonging to the provenance of some output of query $Q$"* as one way to determine the relevance of a data item. We use $\mathcal{R}(Q; D)$ to denote the subset of the database $D$ that is relevant for answering $Q$. At the minimum, $\mathcal{R}(Q; D)$ needs to be **sufficient** for answering query, i.e., evaluating $Q$ over $\mathcal{R}(Q; D)$ yields the same result as evaluating $Q$ over the full database:

$$Q(D) = Q(\mathcal{R}(Q; D)) \qquad \textbf{(sufficiency)}$$

Obviously, sufficiency is not a strict enough condition to ensure that $\mathcal{R}(Q; D)$ contains only data that is needed to produce the result, because sufficiency does not prevent us from including irrelevant data in $\mathcal{R}(Q; D)$ that does not affect the result of $Q$. While also requiring necessity would seem to be the right approach for excluding irrelevant data, this fails in the presence of operators that are disjunctive in nature, e.g., union. Indeed, this is a well-known fact in database provenance. As standard notions of provenance for databases, e.g., provenance polynomials or lineage [13] handle this issue, we will assume here that $\mathcal{R}(Q; D)$ is computed using one of these models. We do not introduce formal definitions of provenance here, but refer the interested reader to a recent survey on the topic [9].

Example 6 (Data Relevance and Sufficiency). *Revisiting Example 1, as the reader can verify, the rows highlighted in blue (red) are sufficient for producing the same result for $Q_{top}$ ($Q_{problem}$). Indeed, these rows are the Lineage [9] for these two queries.*

Database systems already make use of relevance information in a limited way during query optimization. By analyzing the selection conditions of queries, query optimizers determine *statically* what data is relevant for answering a query. In the example above, any reasonable query optimizer will determine that only reviews from region *"Europe"* containing the word *"problem"* are relevant for answering $Q_{problem}$. The optimizer may then decide whether to make use of the database's physical design artifacts, e.g., indexes, zone maps, or partitions, to filter out irrelevant data early-on during query processing.

Insight 8. *Database systems* ***statically*** *analyze what data is relevant for answering a query to determine how to filter out irrelevant data early-on. However, static analysis often yields a coarse over-approximation of what data is relevant, because data relevance is often data-dependent which necessitates* ***dynamic*** *capture of relevance information at query runtime.*

As mentioned before, *"being accessed by a database system to evaluate a query"* in general does not imply data relevance. For instance, the database may choose a full table scan even though only a fraction of the data in a table may be actually relevant for producing a query result. To see why this is the case, note that for many provenance models, the provenance of a query is sufficient for producing the query's result and the provenance of a query is typically only a fraction of the tables accessed by the query.

Insight 9. *Data access is not the same as data relevance as typically not all data accessed to answer a query affects the result returned by the query.*

For many use cases of data relevance, it is beneficial to understand the relationship of two queries wrt. their relevant data. We write $Q_1 \sqsubseteq_{rel} Q_2$ to denote that query $Q_1$ is relevance-contained in $Q_2$ and formally define relevance containment as shown below.

$$Q_1 \sqsubseteq_{rel} Q_2 \Leftrightarrow \forall D : Q_1(\mathcal{R}(Q_2; D)) = Q_1(D) \qquad \textbf{(relevance containment)}$$

That is, relevance-containment implies that the relevant data for $Q_2$ is sufficient for answering $Q_1$. Note that for monotone queries, relevance containment is equivalent to containment of the relevant data, i.e., containment of provenance [12, 19]:

$$Q_1 \sqsubseteq_{rel} Q_2 \Leftrightarrow \forall D : \mathcal{R}(Q_1; D) \subseteq \mathcal{R}(Q_2; D) \quad \textbf{(monotone queries)}$$

In [19] we did present a static method that checks relevance containment for two instances of the same query template, i.e., queries share the same structure and only differ in constants used in selection conditions. We will explain further in Section 5.1 how this check is utilized when using relevance information for a query to answer subsequent queries. This method relies on constraint solvers and is similar in spirit to recent work for checking query equivalence [24]. Checking relevance containment for queries with different structures remains an interesting open problem.

Insight 10. *Relevance containment enables relevance information for one query to be reused for another, similar query.*

## 3 OBJECTIVE DATA VALUE METRICS

We argue that an objective metric for the value of data should not assign some intrinsic value to the data, but rather should be based on the actual benefit to the organization gained by possessing the data. For instance, a dataset that is stored by an organization but is not accessed by any query / computation has zero value to the organization as deleting the data would not change the outcome of any computations, i.e., whether the dataset exists or not has no bearing on the operations of the organization. However, as discussed in Section 2 being accessed by a computation is a necessary, but not sufficient condition for data to be relevant and, thus, of value. Put differently, data that is accessed by a query (or other type of computation) but is not relevant to that query (computation) can be deleted without affecting the outcome of the query.

So far we have argued that data that is not relevant to any computation run by an organization is of zero value. Let us start by defining a data value metric that assumes that data is immutable and that the distribution of queries evaluated over this data is known upfront. Assume a workload $\mathbb{Q}$ consisting of queries $Q_i$:

$$\mathbb{Q} = \{Q_1, \dots, Q_n\}$$

where each query $Q_i$ is each associated with a frequency $F_{Q_i} \in [0, 1]$ (out of all queries executed by the workload what is the relative frequency of $Q_i$). This workload could constitute all queries run by the organization or just the queries used by a particular set of users or applications for which we want to understand the value of data towards $\mathbb{Q}$. The value $\mathcal{V}(d)$ of a data item $d$ (e.g., we could measure value at the granularity of rows or at a coarser granularity such as fragments of a table) is defined as the fraction of queries weighted by their frequency for which the data item is relevant:

$$\mathcal{V}(d) = \sum_{Q \in \mathbb{Q}} \mathbb{1}[\mathcal{R}(Q; d)] \cdot F_Q \tag{1}$$

As already discussed in Section 1, it would be possible to replace the binary indicator $\mathbb{1}[\mathcal{R}(Q; d)]$ with a metric that measures the contribution of a data item to the result of the query. Several such attribution measures have been discussed in related work including Shapley values [3] and causal responsibility [8]. However, these techniques are in general of high computational complexity. It remains to be seen whether it is possible to develop efficient approximations of such metrics for complex computations that could be utilized for measuring data value.

## 3.1 Updates

Let us now relax the assumption that data is immutable. If data is subject to change, then the value of data will change over time too. For simplicity, we will assume a linear history $\mathcal{D}$ of database versions:

$$\mathcal{D} = D_1, \dots, D_n$$

Then the value of data is specific to a database version $D_i$, i.e., $\mathcal{V}_i(Q; d)$ denotes the value of data item $d$ wrt. query $Q$ evaluated over database version $D_i$. For transactional histories executed under weaker isolation levels, e.g., read-committed snapshot isolation [22], data value becomes specific to a statement within a particular transactions. However, it is probably sensible to restrict data value measurements to committed data.

## 3.2 Changes to the Workload

Note that when we also consider an evolving workload, then the relevance of data items wrt. queries, i.e., $\mathcal{R}(Q; d)$, does not change. However, we can no longer assume that the frequency of queries is constant. One way to approach changing workloads is to weight instances of a query $Q$ based on time, e.g., we could replace $F_Q$ with a weighted sum over weights that decrease over time. Let $\mathcal{T}_Q = \tau_1, \dots, \tau_m$ denote the points in time when query $Q$ was executed and let $w()$ be is a function of the current time $\tau_{now}$ and a time point $\tau$ such that $w(\tau_{now}, \tau_1) < w(\tau_{now}, \tau_2)$ if $\tau_1 < \tau_2$ (the weight is anti-monotonic wrt. the temporal order, e.g., $w(\tau_{now}, \tau) = \frac{1}{1+(\tau_{now}-\tau)}$). We can then redefine $F_Q$ as a function of the current time $\tau_{now}$ as shown below:

$$F_Q[\tau_{now}] = \sum_{\tau \in \mathcal{T}_Q} w(\tau_{now}, \tau)$$

Of course, one can imagine many other reasonable weight definitions. Putting this together with a temporal version of relevance based on updates to the data, i.e., assuming that $D_\tau$ is the database version at time $\tau$, we get a definition of data value that takes changes to both the data and workload into account:

$$\mathcal{V}(d; \tau) = \sum_{Q} \sum_{\tau' \in \mathcal{T}_Q : \tau' \leq \tau} \mathbb{1}[\mathcal{R}(Q; d; \tau')] \cdot w(\tau, \tau')$$

That is, the value of data item $d$ at time $\tau$ is the contribution of $d$ to each query at the time of the query's execution (if that execution happened before $\tau$) weighted using $w(\cdot, \cdot)$ based on how recent that execution is. Note that here $\mathcal{R}(Q; d; \tau)$ denotes $\mathcal{R}(Q; d)$ computed for $D_\tau$.

## 4 MEASURING AND MANAGING DATA VALUE

In this section, we discuss strategies for managing data value information efficiently including capturing (approximate) data value information at different granularities, how to approximate data value for workloads through sampling and clustering of queries, and how to deal with evolving workloads and data.

## 4.1 Measuring (Approximate) Data Value

For now consider a static workload and data that is not subject to updates. A naive way to measure data value according to Equation (1) would be to capture provenance information for every query evaluated over the data and maintain a counter for each tuple that tracks the number of queries for which the tuple is relevant. However, this would result in significant runtime overhead and require additional storage that is linear in the size of the database.

**Data Granularity.** Our value metric for data can be applied at any level of granularity, e.g., rows, pages, or larger data chunks. Thus, an obvious optimization is to record data value at a coarser granularity at the cost of loosing some fidelity of the value information. In addition to reducing storage consumption, this also reduces the overhead of capturing provenance for computing data value. For instance, in [19] we did capture relevance information at the granularity of fragments of a horizontal partitioning of a table resulting in *provenance sketches* which can be encoded compactly as bitvectors (one bit per fragment that indicates whether the fragment is relevant or not). Another alternative is to use approximate data structures representing sets such as bloom filters and store row-level provenance using such a data structure. To give a concrete example, in [19] we did evaluate the storage requirements of provenance sketches against materialized views for a given workload. The total amount of storage required for all materialized views required to support the workload efficiently ranged between ~2GB and ~44GB while the total size of provenance sketches was no more than ~200KB. To capture data value instead of relevance we can simply replace the bit vector with an array of counters (integer values). Whenever a data item is found to be relevant for a query, its counter is increased and we also maintain a global counter for queries.

*Example 4.1 (Provenance Sketches for Approximating Value).* Consider the provenance (relevant rows) in table `products` for our running example query $Q_{top}$. Furthermore, consider the horizontal partition $F = \{f_{laptop}, f_{desktop}, f_{ssd}, \ldots\}$ of this table based on the value of attribute `category` (a so-called range-partitioning):

$$f_{laptop} = \{p_1\} \qquad f_{desktop} = \{p_2\} \qquad f_{ssd} = \{p_3\}$$

Using this partition of table `products`, we can create a provenance sketch $\mathcal{P}_{Q_{top},F}$, for $Q_{top}$ by recording which of the three fragments contain relevant rows (provenance). In our example, these are fragments $f_{laptop}$ and $f_{ssd}$. Thus,

$$\mathcal{P}_{Q_{top},F} = \{f_{laptop}, f_{ssd}\}$$

The data in the sketch is sufficient for producing the result (it contains all relevant rows). However, in general such a sketch will contain some irrelevant data as typically not all data in a fragment will belong to the provenance of the query. For instance, for a larger database instance, there may be some laptops and ssds that do not have a sufficiently high revenue (`HAVING sum(quantity * price) > 1000`).

Provenance sketches are very storage-efficient: a provenance sketch requires one bit of storage per fragment (is the fragment in the sketch or not) in addition to storing information about the partitioning, e.g., the ranges of domain values for a range-based horizontal partitioning. However, note that partition information can be shared across sketches that are based on the same partitioning.

While most provenance models have been shown to produce provenance that is sufficient, this often only holds for the granularity (most often rows) for which the model was designed for. Thus, one challenge with recording data relevance at coarser granularity is that we need to ensure that the over-approximation of row-level provenance encoded by a sketch is sufficient which is guaranteed for monotone queries, but requires more care for non-monotone queries (e.g., negation or aggregation). We envision an approach that statically analyzes queries to determine whether an over-approximation is safe (preserves sufficiency). In [19], we did introduce a sound safety check that can determine statically whether a provenance sketch based on range-partitioning a table on a set of attributes $A$ is sufficient for the query it was produced for. We plan to extend this technique to other types of partitioning of input tables. One challenge with using provenance sketches to measure data value is that the accuracy of the over-approximation of relevant data for a query is quite sensitive to the choice of partition. That is, to get an accurate estimate of data value we may have to choose different partitions for different queries. It not clear immediately how to combine sketches that are based on different partitions without loosing accuracy.

Insight 11. *Tracking approximate data value at a coarser granularity can significantly improve the cost of measuring data value.*

**Sampling & Clustering Queries.** We demonstrated in [19] that capturing provenance sketches is significantly faster than capturing full provenance, but still has non-trivial overhead compared to regular query execution. Piggy-backing provenance capture onto regular query processing [9] can further reduce the overhead, but requires extensions to the query engine. To further reduce the cost of measuring data value, we can avoid capturing relevance information for every query of a workload. A straight-forward strategy would be to just compute relevance for a randomly sampled subset of queries to approximate data value for a workload. The problem with this approach is that it ignores that some queries may be quite similar in terms of their relevance profiles and that the amount of relevant data per query may be skewed, e.g., it is common that heavy analytical queries are executed less frequently than OLTP queries, but access significantly more data. Such infrequent queries can easily be missed by a naive sampling mechanism.

One possible strategy for addressing this problem is to cluster queries in a workload based on having similar data relevance profiles. Prior work on clustering queries could be of use here, e.g., see [17] for a comparison of similarity metrics for clustering queries. However, most prior work tries to cluster queries based on similar structure while for our use case we need to cluster queries based on what subset of the data is relevant for the query. Assuming such a similarity metric, we could cluster queries in the workload and then sample data relevance stratified by cluster to get a good approximation of data value for the whole workload. An alternative to using clustering is to train embeddings for queries and data [23] and use similarity in the embedding space instead. A generalization of the notion of relevance containment discussed in Section 3 to a relevance overlap measure could be the basis of a distance metric for queries.

## 4.2 Non-relational Data

While we have only discussed data value for relational data so far, the concept of data value extends to non-relational data models such as semi-structured data models like JSON, graph data, and potentially vector databases. Specifically, for semi-structured data it is common to ingest raw data produced by applications, e.g., logs of web API requests, directly into a database or key-value store. Relevant parts of the data are then extracted at query time. Such applications would benefit significantly from data relevance tracking to prune irrelevant data early-on during query processing. Furthermore, data relevance information could be used to enrich structure-based data guides [11] to guide users in formulating queries based on summarized information about data content in addition to information about the data's structure.

## 4.3 Dealing with Evolving Workloads and Data

For many applications, data and workloads are not static, but evolve over time. Thus, data value information has to be maintained to reflect these changes to the workload and the data. As relevance information can be computed using queries, existing incremental view maintenance techniques [14] are applicable to maintain data value metrics when data changes. However, as we would typically want to approximate data value for performance reasons, the main difference to classical incremental maintenance is that we may be able to trade accuracy of the data value metric for improved performance. For instance, we may optimistically assume that data that is inserted contributes to a query result without actually incrementally maintaining the result.

*Example 4.2 (Approximate Incremental Maintenance of Data Value).* Continuing with Example 4.1, assume that a new product of category `ssd` with `pid` 4 got inserted into the table products. As the fragment $f_{ssd}$ for this category value is already part of the provenance sketch we have the choice to compute the aggregation result for the group `pid = 4` if the new row has any join partners in table orders. Alternatively, we can skip this step as $f_{ssd}$ is already part of the sketch. Choosing the second option would result in an approximate sketch on which we can no longer process deletions accurately: if the group $pid = 3$ is deleted, then we do not know whether $f_{ssd}$ will be still part of the sketch.

Insight 12. *Data value needs to be incrementally maintained when data or workloads evolve. If tracked through provenance sketches, novel optimizations are possible that trade maintenance performance for size of the maintained sketches.*

To maintain data value under evolving workloads, we have to maintain the clustering of queries taking also the changes to data into account as cluster membership may change depending on the data distribution.

### 4.4 Legal Requirements

Organizations may be subject to legal requirements about how long they have to retain data. As we will discuss in Section 5, data value will be used to make data management decisions, e.g., we may delete or compress irrelevant data. To ensure that legal requirements about data retention are fulfilled we could assign infinite value to data that should be retained to ensure it will not be deleted. However, such data may no longer be in use and, thus, could be of low value otherwise. An alternative is to track retention requirements in addition to tracking data value and always ensure that data management decisions do not violate retention requirements.

## 5 UTILIZING DATA VALUE

We now discuss how data value information can be leveraged to improve many aspects of data management for improving query performance by filtering irrelevant data, over informed allocation of resources (both human and computational), enriching metadata management, and improving data markets.

### 5.1 Filtering Irrelevant Data

An important application of data value is provenance-based data skipping (PBDS), i.e., filtering irrelevant data early-on during query processing to improve query performance. We have explored this use case of data relevance / value in [19]. In this work, we capture provenance sketches, compact over-approximations of what data is relevant for a query $Q$. A provenance sketch $\mathcal{P}$ encodes a subset of the database $D_{\mathcal{P}}$ that is guaranteed to be *sufficient*, i.e., evaluating the query over $D_{\mathcal{P}}$ yields the same result as evaluating the query over the full database. For example, consider the case of our running example query $Q_{top}$ from Figure 1. Without a provenance sketch, evaluating this query requires computing the revenue for each product in the database. However, only data for products with a revenue above \$1,000 are relevant for producing $Q_{top}$'s result. A provenance sketch records data relevance at the granularity of fragments of a horizontal partitioning of the input tables. For instance, we may build a sketch on the product table's `category` attribute. As explained in Example 4.1, only the fragments for categories *laptop* and *ssd* do contain relevant data and, thus, belong to the sketch. Capturing a provenance sketch for a query requires instrumenting the query to generate the sketch: we spend time to create the sketch. Once a sketch for a query $Q$ has been created it can be used to speed-up future executions of $Q$ and queries similar to $Q$. For example, the sketch for $Q_{top}$ based on product category could be used to answer a variant of $Q_{top}$ with a more restrictive **HAVING** clause condition, say `HAVING sum(quantity * price) > 2000`. To use a sketch $\mathcal{P}$ to answer a query $Q$, we add additional **WHERE** clause conditions to the query to filter out data that does not belong to the sketch. For instance, to use the sketch for $Q_{top}$ on product category we would add the following filter condition:

```
WHERE product.category IN ('laptop', 'ssd')
```

We present in [19] techniques for generating sketches, using sketches, and discuss how to determine when a sketch can be reused. Furthermore, we demonstrated that query performance can be significantly improved with PBDS, even for standard benchmark workloads for which database systems are heavily optimized for. By capturing relevance information, PBDS is also a main enabler for many of the advanced applications of data value discussed in the following.

### 5.2 Information Life-cycle Management & Adaptive Physical Design

Information life-cycle management decides what where to store data (what device) and what format to use as well as what data to cache when and in which format. Typically, information life-cycle management decisions are based on tracking data access. For instance, infrequently accessed data can be heavily compressed to save storage space without much effect on query performance. We argue that decisions on where to place data and whether to compress it should be based on data value instead. This has the potential to significantly improve life-cycle management as only data that is actually needed to answer queries will be placed on faster storage or will be cached in-memory. For example, consider again the running example query $Q_{top}$. We explained above that without data value, $Q_{top}$ has to compute the revenue for every product in the database. However, as only products with a revenue above 1,000 are relevant for producing the result of this query, it is more efficient to prune irrelevant data before computing aggregation results. Furthermore, data that is not relevant for the two queries of our example workload could be compressed (or even deleted) while data that is relevant for these two queries should be cached in memory. Additionally, data value information can be utilized for adaptive physical design and other self-tuning tasks. For example, instead of indexing full tables, we can only build indexes over data that is relevant for a significant fraction of queries executed as part of a workload to reduce index storage size and improve look-up performance.

### 5.3 Intelligent Resource Allocation

Data collection and curation requires extensive human and computational resources, but is critical for maintaining data quality and, thus, trustworthy analysis results. As the resources of an organization are limited, it is important to allocate resources where they are most needed. While this idea has been explored in on-demand data curation / integration, we argue for the use of data value as a means for selecting how to spend resources most effectively. Specifically, resources should be allocated to the most valuable data. However, additional challenges have to be overcome for this to be effective as the impact of spending a given amount of resources may not be uniform across different parts of the data and the improvement in data quality gained may not be linear in the amount of resources spent on curating a subset of the data. We envision that curation efforts can be monitored to learn over time how much effort is required for certain curation tasks. Furthermore, techniques like reinforcement learning could be applied to explore the impact of data curation efforts and then optimize for curation efforts that effectively improve the quality of the most valuable data.

### 5.4 Enriching Metadata Management

Metadata management systems like Goods [15] and Apache Atlas [1] enable users to keep track of data in their data lakes, record provenance information, and facilitate dataset discovery. We argue that data value information should be also managed by such systems. For facilitating data discovery, data value should be recorded for several representative workloads to enable users to select data that is relevant for their tasks. For instance, an organization may define a set of use cases (e.g., HR, advertizing, or logistics) each associated with a workload for which data value is tracked. When a user is searching for data that may be used for a particular task, we could then determine which workload is most similar to the user's task and then use the value of a dataset for this workload as an estimate for the relevance of the data for the user's task. Query similarity metrics and clustering of workload queries as explained in Section 4.1 could be utilized to identify which workload is most similar to the user's task.

### 5.5 Data Markets

Data markets [2, 7] enable data owners to generate revenue from their data by making it available for purchase to buyers who benefit by gaining access to the data they need. The price of data in data markets is subject to economic forces and considerations (e.g., ensuring arbitrage-free pricing [16]). Nonetheless, we argue that our data value metric could be critical for buyers and sellers to make informed decisions. Assuming that data value is tracked for representative workloads as outlined above, data buyers can take the value of data for workloads similar to the workload they want to purchase data for into account to decide how valuable the data would be to them and, thus, how much they are willing to spend on the data. Furthermore, organizations running data marketplaces can measure data value for representative workloads on the data uploaded to their platform to enable their buyers to make informed choices which increases the utility of the market place.

## 6 DARK DATA AND QUERY VALUE

### 6.1 Dealing with Dark Data

With growing amount of data, it also becomes harder for users in an organization to identify which data is relevant for their use. While our data value metrics can identify which data is most relevant for an application, this only works under the assumption that the datasets containing the data are already used by this application. That is, some data may have currently no value as it is not relevant in the context of a workload, but has *potential value*, i.e., it has intrinsic value for an application, but this value has not been realized yet as the application did not make use of the data. We refer to data that currently has no / low value, but may have significant value if utilized for the right application as *dark data*. We envision a dataset discovery system that helps users to distinguish between not valuable and potentially valuable dark data through controlled recommendations [18]. That is, a dataset discovery system would recommend dark data to users and then measure data value of this data for the queries run by users over the recommended data. This would enable the system to discover over time which data is useful and which data is of no potential value.

### 6.2 The Value of Computations / Queries

So far we assumed that the results of all queries / computations have equal value to an organization. Of course, this assumption may not be realistic. In this section we consider how data value metrics are affected by dropping this assumption. Furthermore, we outline strategies for measuring the value of a query either by assuming it as input given by the user (if there is a feasible way for the users to assign value to their queries which may or may not be realistic) or how to automatically compute it.

*6.2.1 Modelling Query Value.* To model the value of a query for an organization we associate a numeric value $\mathcal{V}(Q) \in [0,1]$ to each query $Q$ of a workload $\mathbf{Q}$. Intuitively, a value of zero indicates that the query's result is of no value to the organization while a value of one indicates that the result of the query is of maximal importance. These values are used as weights to adjust the value of a data item $d \in D$ for a database $D$. Thus, the definition of the value of $d$ is updated as follows:

$$\mathcal{V}(d) = \sum_{Q \in \mathbf{Q}} F_Q \cdot \mathcal{V}(Q) \cdot \mathcal{R}(Q; d)$$

**Query Value as User Input.** In the simplest possible case, the value for queries in a workload $\mathbf{Q}$ is provided as input by the user. In this case we can directly apply the formula shown above to compute the value of a data item $d$.

*6.2.2 Automatically Determining Query Value.* Even if the value of a query is not known apriori, we can still make progress based on the following observations: (i) query value depends on valuable data being relevant. When a highly valuable data item is relevant to a query $Q$, then this increases the value of the query; and (ii) being relevant to valuable queries increases data value. If data is relevant to an important query, then this increases the value of the data as this data is used to compute query results that are important to the organization. To summarize, data and query value are intrinsically connected. However, in the absence of ground truth information about either query or data value, we are facing the following dilemma: to compute data value we need to know the value of queries and vice versa! To overcome this dilemma, we envision to adapt techniques, e.g., PageRank [4], which compute (approximate) solutions for fixpoint equations similar to ours.

## 7 RELATED WORK

Data value is also an important concept in data markets [2, 7, 16]. However, that line of work has focused more on economic aspects of data pricing rather than on providing an objective metric for how valuable data is for an organization. The approximation techniques we propose for approximating data value in addition to their basis in provenance sketches [19] are also closely related to data summarization techniques for explanations [10]. The problem of determining query and data value that are mutually depend is similar to other problems for computing fixpoints for recursive equation systems such as PageRank [4] and related algorithms such as detecting spam reviews based on trust [21] (both the trustworthiness of reviewers and quality of items is not known upfront). Automated techniques for physical design and data life-cycle management [5, 20] are often based on data access patterns. Any such technique can be improved by using data value to ignoring data that is not relevant for generating query results. To selectively measure data value for only some queries, we need techniques for clustering queries and computing similarity between queries. Existing techniques [17] for measuring query similarity could be applied to this problem. However, for our use case, query similarity should be based on what data is relevant.

## 8 CONCLUSIONS

In this work, we have introduced an *objective metric for data value* based on provenance [9] and the recently introduced concept of data relevance [19]. We argue that data is only of value by being used to generate results of queries (or more generally computations). These results, and not the raw data, are what is observed by applications and users of a database (or data lake). As data relevance records what data is required to answer a query, relevance provides us with the means to determine the value of data based on its role of generating query results. We discuss the significant benefits of objective data value metrics in improving various aspects of data management like data life-cycle management, pruning of irrelevant data during query execution (we explored this aspect in detail in [19]), and for self-tuning. We also present possible strategies for measuring and managing data value and how to approximate value to reduce the overhead of measuring and maintaining these metrics. In future work, we plan to integrate data value management into database (and data lake) systems, investigate techniques for measuring query value, and explore additional use cases of data value to improve data management operations.

## REFERENCES

[1] Apache. [n.d.]. http://atlas.apache.org/.

[2] Santiago Andrés Azcoitia and Nikolaos Laoutaris. 2022. A Survey of Data Marketplaces and Their Business Models. *SIGMOD Rec.* 51, 3 (2022), 18–29.

[3] Leopoldo E. Bertossi, Benny Kimelfeld, Ester Livshits, and Mikaël Monet. 2023. The Shapley Value in Database Management. *SIGMOD Rec.* 52, 2 (2023), 6–17.

[4] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput. Networks* 30, 1-7 (1998), 107–117.

[5] Surajit Chaudhuri and Vivek Narasayya. 2007. Self-tuning database systems: a decade of progress. In *VLDB*. 3–14.

[6] H. Chockler and J.Y. Halpern. 2004. Responsibility and blame: A structural-model approach. *Journal of Artificial Intelligence Research* 22, 1 (2004), 93–115.

[7] Raul Castro Fernandez, Pranav Subramaniam, and Michael J. Franklin. 2020. Data Market Platforms: Trading Data Assets To Solve Data Problems. *PVLDB* 13, 11 (2020), 1933–1947.

[8] Cibele Freire, Wolfgang Gatterbauer, Neil Immerman, and Alexandra Meliou. 2015. The complexity of resilience and responsibility for self-join-free conjunctive queries. *PVLDB* 9, 3 (2015), 180–191.

[9] Boris Glavic. 2021. Data Provenance - Origins, Applications, Algorithms, and Models. *Foundations and Trends® in Databases* 9, 3-4 (2021), 209–441.

[10] Boris Glavic, Alexandra Meliou, and Sudeepa Roy. 2021. Trends in Explanations: Understanding and Debugging Data-driven Systems. *Foundations and Trends in Databases* 11, 3 (2021), 226–318.

[11] Roy Goldman and Jennifer Widom. 1997. DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. In *VLDB*, Matthias Jarke, Michael J. Carey, Klaus R. Dittrich, Frederick H. Lochovsky, Pericles Loucopoulos, and Manfred A. Jeusfeld (Eds.). 436–445.

[12] T.J. Green. 2011. Containment of conjunctive queries on annotated relations. *Theory of Computing Systems* 49, 2 (2011), 429–459.

[13] Todd J Green and Val Tannen. 2017. The Semiring Framework for Database Provenance. In *PODS*. 93–99.

[14] Ashish Gupta, Inderpal Singh Mumick, et al. 1995. Maintenance of materialized views: Problems, techniques, and applications. *IEEE Data Eng. Bull.* 18, 2 (1995), 3–18.

[15] Alon Y. Halevy, Flip Korn, Natalya Fridman Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, and Steven Euijong Whang. 2016. Goods: Organizing Google's Datasets. In *SIGMOD*. 795–806.

[16] Paraschos Koutris, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. 2013. Toward practical query pricing with QueryMarket. In *SIGMOD*. 613–624.

[17] Gökhan Kul, Duc Thanh Anh Luong, Ting Xie, Varun Chandola, Oliver Kennedy, and Shambhu J. Upadhyaya. 2018. Similarity Metrics for SQL Query Clustering. *TKDE* 30, 12 (2018), 2408–2420.

[18] Fatemeh Nargesian, Erkang Zhu, Renée J. Miller, Ken Q. Pu, and Patricia C. Arocena. 2019. Data Lake Management: Challenges and Opportunities. *PVLDB* 12, 12 (2019), 1986–1989.

[19] Xing Niu, Ziyu Liu, Pengyuan Li, Boris Glavic, Dieter Gawlick, Vasudha Krishnaswamy, Zhen Hua Liu, and Danica Porobic. 2021. Provenance-based Data Skipping. *PVLDB* 15, 3 (2021), 451–464.

[20] Stratos Papadomanolakis and Anastassia Ailamaki. 2004. AutoPart: Automating Schema Design for Large Scientific Databases Using Data Partitioning. In *SSDBM*. 383–392.

[21] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2012. Identify Online Store Review Spammers Via Social Review Graph. *ACM Trans. Intell. Syst. Technol.* 3, 4 (2012), 61:1–61:21.

[22] Maysam Yabandeh and Daniel Gómez Ferro. 2012. A critique of snapshot isolation. In *Proceedings of the 7th ACM european conference on Computer Systems*. 155–168.

[23] Li Zhang, Shuo Zhang, and Krisztian Balog. 2019. Table2Vec: Neural Word and Entity Embeddings for Table Population and Retrieval. In *SIGIR*. 1029–1032.

[24] Qi Zhou, Joy Arulraj, Shamkant B. Navathe, William Harris, and Jinpeng Wu. 2022. SPES: A Symbolic Approach to Proving Query Equivalence Under Bag Semantics. In *ICDE*. 2735–2748.