Credit Distribution through Data Provenance in Relational Scientific Databases

Dennis Dosso[a], Susan B. Davidson[b], Gianmaria Silvello[a]

[a]*Department of Information Engineering, University of Padua, Italy*
[b]*Department of Computer and Information Science, University of Pennsylvania, USA*

Abstract

Digital data is an important form of research product for which citation, and the generation of credit or recognition for authors, is still not well understood. The notion of *data credit* has therefore recently emerged as a new metric, defined and based on data citation theory.

Data credit is a real value that represents the importance of data cited by a paper or by another research entity. Credit can be used to annotate data contained in a curated scientific database, and then used as a measure for the importance and impact of that data in the research world. As such, it is a new method that, together with traditional citations, helps recognize the value of data and its creators.

In this paper we explore the problem of Data Credit Distribution, the process by which credit is distributed to the database parts responsible for the production of data being cited by a research entity.

We adopt as use case the IUPHAR/BPS Guide to Pharmacology (GtoPdb), a widely-used curated scientific relational database. We define four new distribution strategies, the first two based on two forms of data provenance, why-provenance and how-provenance, the third based on the concept of responsibility, the fourth on the Shapley value.

Using these distribution strategies we show how credit can highlight frequently used database areas and how it can be used as a new bibliometric measure for data and their corresponding curators. In particular, credit rewards data and authors based on their research impact, not merely on the number of citations. We also show how these distribution strategies vary in their sensitivity to the role of an input tuple in the generation of the output data, and reward input tuples differently.

Keywords: Data Citation, Data Credit

1. Introduction

Citations are an essential component of scientific research, enabling research products to be found as well as the relationships between them to be created and understood. They form a basis on which to give credit to authors, papers, and venues [20, 21, 63]. Citations are used, among other things, to decide on tenure, promotion, hiring, and funding of grants for researchers [22, 36, 42, 46].

Science and research are increasingly digital, and there are numerous curated databases that are at the core of scientific research efforts [12]. It is therefore generally accepted that data must be cited and citable [15, 43], and that data citations should contribute to the scientific reputation of researchers, scientists, data curators, and creators [4, 58]. It is also accepted that data citations should be counted alongside of traditional citations, and contribute to bibliometrics indicators [7, 50].

A central problem in the data citation process is how to attribute credit to data creators and curators [11]. How to handle and count the credit generated by data citation, and how it contributes to traditional and new bibliometrics, are long-standing research issues [9, 31]. However, even when correctly applied, data citations and the bibliometrics computed using them do not always fully reward the creators of data used in a database. Data, in fact, is often cited at the "database level" or the "webpage level". In the first case, the whole database is cited and therefore all credit goes to the key personnel of the database. In the second case, the database has a website with webpages that can be individually cited. The webpages use data extracted from the database, which is aggregated by topic and built to resemble a traditional research paper. Often the creators and curators of the webpage's data are not credited or only marginally credited for their work [3].

Recently, the idea of *Data Credit Distribution* (DCD) [30, 41, 62] has emerged, built on top of methodologies for data citation. Data credit is a value that is computed based on the importance of the data being cited in a paper, and is a proxy for the impact of the data on the citing paper. The DCD problem consists of distributing this credit to elements in the databases in the citation graph that are responsible for the generation of the data being cited. The goal of DCD is to improve and expand the reach of data citation, rather than being an alternative to it.

In this paper, we consider data credit as a measure of value for data in a (curated) scientific database. Credit is a real value that can be assigned to data of any kind and at any level of granularity. Therefore the concept of "data" is left intentionally vague, although in this paper we focus on relational databases. Credit acts as a proxy for the value of data based on the measure of citations, accesses, clicks, downloads, or other surrogates for data use.

We define DCD as *the process, method, or algorithm used to assign credit to a given datum or dataset*. It differs from the traditional citation setting since:

1. When a paper p_1 cites another paper p_2, a +1 citation "credit" is given to p_2, and to all its authors. It does not matter why or how paper p_1 cites paper p_2[1], the result is always +1 to the citation count of p_2 and of its authors. A different credit distribution strategy can assign a quantity of credit to p_2 and its authors that is *proportional* to the role played by p_2 in p_1. Hence, we can weight the importance of the cited entities and assign credit according to their role.
2. Traditional citations are *atomic*: a citation from p_1 to p_2 can never be broken into pieces and assigned in part to p_2 and in part to other papers or data that contributed to p_2. In contrast, with data credit, we use a *non-atomic* real value, which can be divided and distributed to multiple components of a database.
3. Credit can be *transitive*, that is, it can be propagated through one cited entity to other entities cited by it that contributed to its content. Citations, traditionally, are not.

We study the DCD problem in the context of relational databases (RDBs) since they are widely used[2] and are the main focus of current work in data citation methods [12, 14, 51]. RDBs are also frequently a test-bed for new methods that can be adapted to other databases, e.g., graphs or document databases. The "portions" of data in an RDB that can be credited can be defined at different levels of granularity, in particular: (i) the whole database, (ii) tables, (iii) tuples, and (iv) attributes. The ability to specify different

[1] Note that there is vast research on this topic and many alternative proposals, but none of them currently work at a large scale.

[2] The "relational database market alone has revenue upwards of $50B" [1].

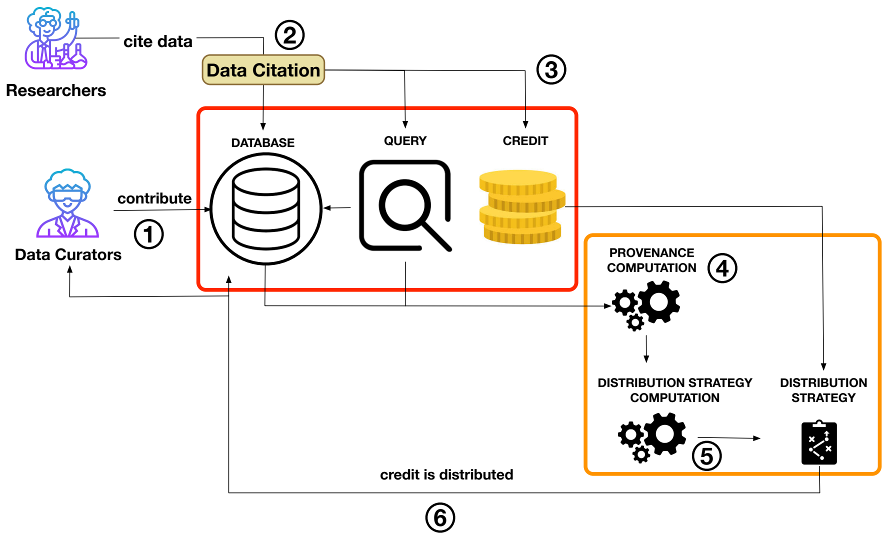

Figure 1: Overview of the credit distribution pipeline.

levels of granularity in a relational database allows us to define the DCD problem at a particular level of granularity. In this paper, we focus on DCD at the tuple level.

The DCD process that we use is summarized in Figure 1:

Step 1 Scientists and experts contribute the curated information contained in a scientific database. These are called the "Data Curators".

Step 2 Other researchers use the data in their research, and when possible, cite them.

Step 3 The citation to the data generates credit, that can be used as a proxy for the impact of the data on the citing paper. This credit is represented as a real value $k \in \mathbb{R}_{>0}$.

Step 4 Given the database instance I and the query Q, the *data provenance* of $Q(I)$ is computed. The data provenance of $Q(I)$ is a form of metadata that captures how Q used I to generate the output [17].

Step 5 Provenance is input to the *Credit Distribution Strategy* (CDS, also referred only as Distribution Strategy, DS). CDS is a function f that takes as input the credit value k, divides it and distributes it to the data in the input database I, and is defined on the basis of citation policies decided at the database administration level or at the domain community level.

Step 6 Once the CDS is computed, it is used to distribute the given credit k to the parts of the database that are responsible for the generation of $Q(I)$. Transitively, this credit is also divided and given to the corresponding authors of those data.

This paper expands the work in [27] where we first defined the problem of DCD in relational databases, and proposed a viable Distribution Strategy (DS) based on *lineage* – the simplest form of *data provenance*. The lineage of a tuple t in the output $Q(I)$ is defined as the set of all and only the tuples in the database instance I that are "relevant" to the production of t. The corresponding strategy equally redistributes the credit k to the tuples in the lineage set, thus each tuple receives credit $k/|L_t|$, where L_t is the lineage set of t.

One may argue that this DS is too simplistic, since lineage does not convey any information about the role or importance of input tuples in the query. Therefore, one may desire to give more credit to the tuples that are more *important* to the production of the output, i.e. those tuples that, if removed, would prevent the output tuple from appearing in the final result, or those tuples used more than once by the query.

Therefore, in this paper, we expand the ideas in [27] by proposing new DSs based on two other forms of data provenance: why-provenance [13] and how-provenance [33]. Also, we propose other two DS based on the concepts of responsibility [47], and the Shapley value[25, 44]. We show how these DS differ from each other as well as the one based on lineage, and discuss why one may be preferred to another depending on the application and its goals. In particular, we show that the new proposed DSs are more sensitive than the one based on lineage to the *role* of a tuple in a query, i.e. how many times the tuple is used and how it is used. We also show that the DSs based on why-provenance and responsibility give more credit to tuples that are essential to the production of the result set, whereas the how-provenance-based DS takes into consideration the different ways in which a tuple is used. Finally, the DS based on the Shapley value sees the process of distribution as a competitive game where tuples that contribute more to the generation of the output are correspondingly rewarded more.

The evaluation is based on a well-known curated database, the IUPHAR/BPS[3] Guide to Pharmacology [35], also known as GtoPdb[4], which contains expertly curated information about diseases, drugs, cellular drug targets, and their mechanisms of action. We chose GtoPdb for two main reasons: (i) it is a widely-used and valuable curated relational database, (ii) many papers in the literature use, and cite, its data (i.e., families, ligands, and receptors). Real queries used in papers can therefore be seen as data citations which, in turn, can be used to assign data credit.

We perform four sets of experiments. In the first, real queries are extracted from papers published in the British Journal of Pharmacology (BJP), that represent data citations to GtoPdb, and are used to distribute credit in the database using the three different provenance-based DSs. In the second and third experiment we analyze the behavior of the different DS when complex citation queries are employed. In the fourth set of experiments we use both real and synthetic queries to assess the difference between traditional citation and the notion of credit distribution in terms of rewarding those responsible for the data, e.g. data curators.

Contributions of this work include:

- Four new Distribution Strategies based on why-provenance, how-provenance, responsibility and the Shapley value.
- An in-depth analysis of the effects of credit distribution on real-world curated data and of the differences between the five proposed Distribution Strategies.
- A comparison between the behavior of traditional citations and data credit in rewarding data curators.

Outline. The rest of the paper is organized as follows: Section 2 presents background material and related work. Section 3 describes the GtoPdb use case. Section 4 briefly presents the forms of provenance used in the paper. Section 5 describes the credit distribution problem and the proposed distribution strategies. In Section 6 we present the experimental evaluation, followed by a discussion of our design decisions in Section 7. Section 8 draws some conclusions and outlines future work.

[3] International Union of Basic and Clinical Pharmacology/British Pharmacology Society

[4] `https://www.guidetopharmacology.org/`

2. Background

Data in Research. The world of research is rapidly transitioning towards the *fourth paradigm of science* [37], that is, data-intensive scientific discovery, where data are important for scientific advances as well as for traditional publications [6].

The scientific community is promoting an *open research culture* [49], founded on methods and tools to share, discover, and access experimental data. The community has identified the FAIR principles (Findable, Accessible, Interoperable, and Reusable) [60], that should be enforced by every database. In particular, data should be accessible from the articles, journals, and papers that cite or use them [20]. Aspects such as the need for the *reproducibility* of experiments through the used data; the *availability* of scientific data; the *connections* between data and the scientific results are all needed aspects for the fourth paradigm, and are all relevant to the domain of *data citation* [38].

Data Citation: Principles and Motivations. Data Citation principles were proposed in [19], and later summarized and endorsed by the Joint Declaration of Data Citation Principles (JDDCP) [45]. The principles are divided into two groups [56]. The first group contains principles concerning the role of data citation in scholarly and research activities such as the (i) *importance* of data (why data citation is important and why data should be considered as first-class citizens); (ii) *credit* and *attribution* to the creators and curators of the data; (iii)*evidence*; (iv) *verifiability*; and *interoperability*, with these last three requiring data citation methods to be flexible enough to operate through different communities. The second group defines the main guidelines to establish a data citation systems, and contains principles such as the (i) *unique identification* of the data being cited; (ii) *(open) access* to data; (iii) guarantee of *persistence* and *availability* of citations even after the lifespan of the cited entity; the (iv) *specificity* of a citation, i.e. it must lead to the data set originally cited.

*** SBD: Is the next paragraph necessary? Could we just say "The main motivations for data citation are outlined in [56]." ***

It is possible to outline six main motivations for data citation [56]:

- *Data attribution*: identify the individuals that should be credited for data with variable granularity.

- *Data connection*: connect papers to the data being used.
- *Data Discovery*: citations helps to find data records and subsets that would be otherwise not findable via search engines.
- *Data Sharing*: share data obtained by researchers within the whole community.
- *Data Impact*: highlight the results obtained in writing papers using specific data, the frequency and modality data were used.
- *Reproducibility*: data citation greatly impacts the reproducibility of science [5]. Many authoritative journals ask to share data and provide valid methodologies to reproduce experiments.

2.1. Data Citation in Relational Databases

Relational databases have been the target of data citation methods since the surge of the data-centric research paradigm. The RDA "Working Group on Data Citation: Making Dynamic Data Citable"[5] [52] has developed guidelines for citing large, dynamic, and changing datasets which have now moved on into adoption phase. The datasets considered by the Working Group are often relational.

In one of its most recent sessions [53], the Working Group (WG) on Data Citation reported that there are various implementations of its guidelines for Data Citation on MySQL/Postgres relational databases. Some of these databases are: DEXHELPP[6] (Social Security Records); NERC (ARGO Global Array); EODC (Earth Observation Data Centre) [32]; LNEC (River dam monitoring); MDS (Million Song Database) [8]; CBMI[7] (Center for Biomedical Informatics); VMC (Vermont Monitoring Cooperative); CCA[8] (Climate Change Center Austria); VAMDC (Virtual Atomic and Molecular Data Center) [28, 64].

More examples of work on data citation in relational databases are [2, 12, 24, 61]. The website `https://fairsharing.org/` keeps an updated list

[5]`https://www.rd-alliance.org/groups/data-citation-wg.html`

[6]`http://www.dexhelpp.at/`

[7]`https://medicine.missouri.edu/centers-institutes-labs/center-for-biomedical-informatics`

[8]`https://ccca.ac.at/startseite`

of curated and scientific databases (many of which are relational or graph-based) following FAIR guidelines. These databases are citable since they are compliant with the most recent guidelines, and they are in the vast majority of cases accessible via dynamically created Webpages. In all these databases it is, therefore, possible to implement DCD on top of the existing infrastructures for citing data.

Data citation techniques are primarily applied to relational databases because of their pervasiveness as well as the "identifiability" of the portions of data that are to be cited: the whole database, a relation, a tuple, or even an attribute. Many papers [2, 10, 12] consider more complex citable units, recognizing that often the *views* of a database are the ones to be cited. Generally, a *view* is a query on the database. To this end, [61] suggested decomposing the database into a set of views, where each view is associated with its citation.

At present, the most common practices to cite databases include:

1. A database cited as a whole, even though only parts of the databases are used in the papers or datasets. Alternatively, the so-called "data papers" are cited, being traditional papers that describe a database [16]. In this case, all the credit from the citations goes to the database administrators or to the authors of the data papers.
2. Subsets of data, obtained by issuing queries to a database, are individually cited. This is the solution adopted by the *Resource Data Alliance* (RDA) working group on Data Citation [52]. In this case, the credit generated from citations is distributed among the contributors of the portions of data being cited, and/or to the database administrators.
3. The database is accessible via a series of Webpages that arrange the content of the database by topic or theme. Examples in the life science domain include the Reactome Pathway database [40], the GtoPdb [35], and the VAMDC [64]. Every single Webpage is unequivocally identifiable and can be individually cited.

2.2. Data Credit

Data credit is related to data citation: they both aim to recognize the work of data creators and curators. Data credit can therefore also be seen as a by-product of data citation, since credit attribution is impossible without the presence of data citations.

Katz [41] suggests the need for a *modified citation system* that includes the idea of *transient* and *fractional credit*, to be used by developers of research products as software and data. Two considerations are made: (i) research objects such as data and software are currently not formally rewarded or recognized by the community; (ii) even in traditional papers, the contribution of each author to the work is hard to understand, unless explicitly specified in the paper. This is even more true for data, where different groups of people work on the same database.

In [41] credit is defined as a "quantity" that describes the importance of a research entity, such as papers, software, or data, mentioned in a citation. It also proposed the idea of a *distribution* of credit from research entities, such as papers or data, to other research entities through citations. Therefore, when discussing data credit, we need to consider *credit computation* – i.e., the process to compute the quantity of credit generated by the citation – and *credit distribution* – i.e., the process to distribute credit and to assign it to the entities that contributed to the creation/curation of the cited data. In this paper we focus on the latter.

These two processes are done by exploiting the structure of the *citation graph*, a directed graph whose nodes are publications and edges are citations. This graph is the model at the core of systems such as Google Scholar and the Web of Science. We add to this that the concept of credit can be built on top of the existing infrastructure handling traditional and data citations.

Katz [41] further explores the idea of a *distribution* of credit from research entities (i.e., papers and data) to other research entities through citations that connect them. Thanks to traditional citations and now also to data citations, this distribution is finally possible, at least between papers and data. Some problems related to traditional citations can thus be solved by citations:

1. Credit rewards research entities that to date are not (formally) recognized (a goal shared with data citation).
2. Credit can reward authors *proportionally* to their role in generating the entity. The more an author contributes to a paper, the more credit is given to him. Zou and Peterson [63] work on something similar with their zp-index, which includes in its formulation the position (and thus the role) of a publication author to represent its impact in the work itself.
3. Credit can be *transitively* channeled through a chain of papers citing each other, thus enabling the rewarding of older papers that are no more cited, since other papers summarize or report their content but

are nevertheless crucial in a research area for the influence of their content.

Fang [30] presents a framework to distribute the credit generated by a paper to its authors and to the papers in its reference list in a transitive way. Let us consider the *citation graph* as the graph where the nodes are papers and the links are the citations among them. In this graph, every paper is a source of credit, which is then transferred to the neighboring nodes. The quantity of credit received by each cited paper depends on its impact/role in the citing paper. So far, this theoretical framework is limited to papers, but it can be easily extended to a citation graph including both papers and data.

Zeng et al. [62] proposes the first method to compute credit within a network of papers citing data. Adopting a network flow algorithm, they simulate a random walker to estimate a score for each dataset, leveraging real-world usage data to compute the credit. This is the first step towards an automatic credit computation procedure. This proposal is, however, limited to assigning credit to whole datasets, and it does not deal with the granularity of data. It does not work to assign credit to a single research entity within a dataset. Differently from Zeng et al. [62], we do not treat the credit computation process, but we focus on the distribution process.

2.3. Data Provenance

To distribute credit, we base our methods on *data provenance*. Data provenance is information that describes the origin and the process of creation of data. It can also be seen as metadata pertaining to the derivation history of the data. It is particularly useful to help users to understand where data are coming from, and the process they went through. Data citation and data provenance are closely linked [3] since both are forms of annotations on data retrieved through queries. Data provenance has been widely studied in different areas of data management. In this paper, we focus on provenance for database management systems (DBMS). For further details on data provenance, please refer to surveys like [17] and [57].

Cheney et al. [17] presents four main types of data citation for DBMS: *lineage* [23], *why-provenance* [13], *how-provenance* [33] and *where-provenance* [13].

Let us start with the first three provenances. Given a database instance I, a query Q, and the result $Q(I)$, consider one tuple t of the output. Its provenance is information about its generation through the tuples of the input that are used by Q. Different types of provenance convey different levels of information. Since these three provenances are computed for each tuple of the output, they are also referred to as *tuple-based*.

Where-provenance, differently from the other three, is *attribute-based*, so we do not take it into account in this work since we consider the tuple as the finest citable unit.

2.4. Causality and Responsibility

We also consider the notions of causality and responsibility, as defined in [47]. Causality is an enrichment of lineage, and it is the attribution of a certain degree of importance to the tuples of the lineage based on their role in the generation of the output. Responsibility is a value given to the tuples of the lineage to rank them based on their degree of causality (the more important the role of a tuple in generating the output, the higher its responsibility).

While computing responsibility for general queries is hard [18], Meliou et al. [47] proved a dichotomy result for conjunctive queries: for each query without self-joins, either its responsibility can be computed in PTIME in the size of the database or checking if it has a responsibility below a given value is NP-hard.

2.5. Shapley value

The Shapley value is named after Lloyd Shapley, who introduced it for the first time in his 1952 work [55]. He considered a *cooperative fame* played by a set A of players, defined by a *wealth function* v that assigns to each coalition set $B \subseteq A$ the wealth $v(B)$. The question behind the Shapley Value is how to quantify the contribution of each player to the overall wealth. Informally, the Shapley value is defined as follows [44]: assume that we select players randomly one by one and without replacement, starting with the empty set. Every time a player a is selected, its addition to the coalition B produces a change in the wealth of the coalition from $v(B)$ to $v(B \cup \{a\})$. The Shapley value of a is the expectation of change that a causes in this probabilistic process.

The Shapley value can be used in different research areas beyond cooperative games, such as economics, law, environmental science, and network analysis, and it has strong theoretical justifications. However, its use in databases as a metric for quantifying the influence of a tuple on the output of a query (thereby presenting an alternative to responsibility) has only

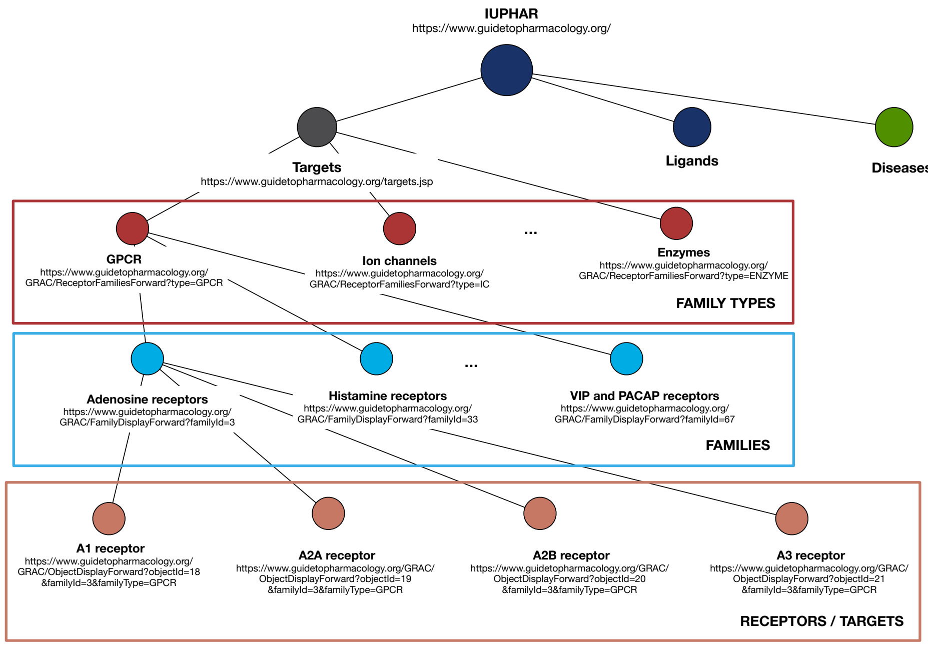

Figure 2: Partial map of the GtoPdb hierarchical structure grouping the targets into families and family types.

recently been proposed [44]. The initial theoretical analysis in [44] showed mainly lower bounds on the complexity of the problem, and did not suggest a feasible implementation. However, very recently, an efficient implementation for Boolean queries (queries that output true or false, or 1 or 0) has been provided [25], both in terms of an exact computations (which in practice works well for most queries) and in inexact one (which is extremely fast and provides the same ranking of tuples as the exact computation, but not necessarily the same values).

3. Use Case: GtoPdb

The IUPHAR/BPS Guide to Pharmacology [35] (GtoPdb[9]) is a well-known and well structured scientific relational database that contains expertly curated information about diseases, drugs in clinical use, their cellular targets, and the mechanisms of action on the human body. It is curated and

[9] `https://www.guidetopharmacology.org/`

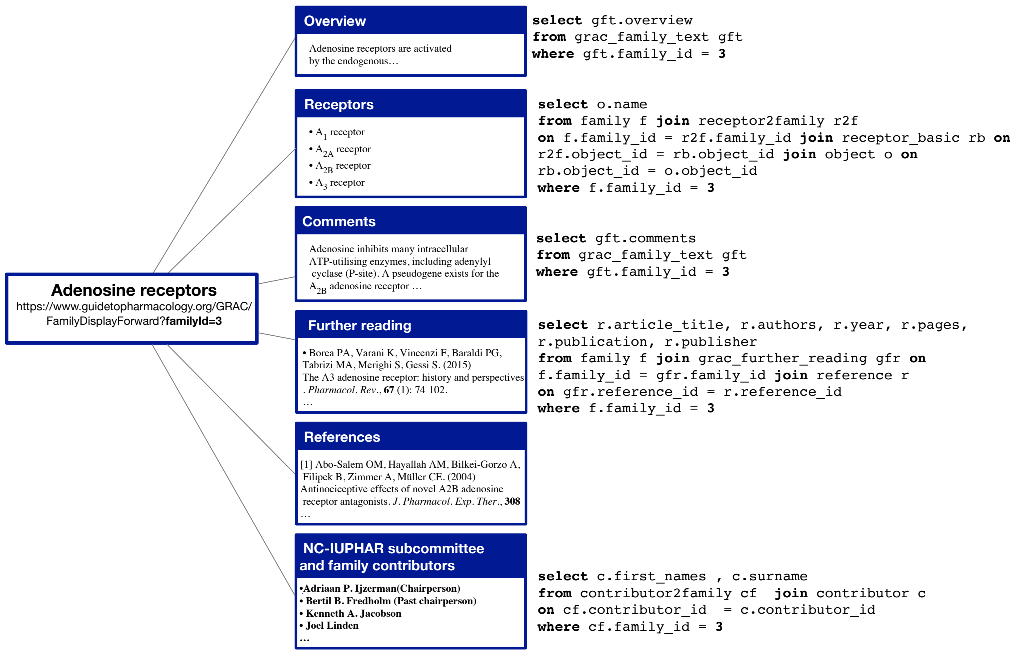

Figure 3: Basic web-page structure of "Adenosine receptors" family (ID 3), with queries used to retrieve the information contained in every section, except references.

maintained by the GtoPdb Committee and 96 subcommittees, comprising 512 scientists collaborating with in-house curators who draw the information contained in the database from high-quality pharmacological and medicinal chemistry literature. Roughly 1000 researchers from all over the world have contributed to the database, and the curators wanted to give recognition to these contributors. This led to some early work on data citation [10].

GtoPdb is relational, but its logical structure is hierarchical as shown in Figure 2. The information contained in the database is also organized into webpages focused on specific diseases, targets or ligands, and families for easier access by users. As depicted in Figure 2, the database can be thought of as a tree where the root is the database; the first level consists of all targets, ligands, and diseases; and the lower levels consists of specific targets, ligands and diseases. In this paper, we focus on targets; thus the figure at the third level shows examples of family types, at the fourth level of specific families of targets (a finer level of granularity), and finally, at the last level, the single targets (also known as receptors).

GtoPdb provides access to the webpages corresponding to all these nodes through URLs. The webpages corresponding to target families all present a similar structure, as shown in Figure 3 for the "Adenosine receptors" family. Each page has an *Overview*, a brief text describing the content of the page; a list of *Receptors* comprising the family; a section of *comments* about the family; the *References*, a list of the papers consulted by the curators of the page, similar to a reference list of a paper; the *further reading* list, reporting papers that an interested reader may want to consult to obtain more insight on the family; and a final section called *How to cite this family page*, containing text snippets useful to cite the specific page or the whole database. Figure 3 shows the SQL code that retrieves the information used to build the corresponding sections (apart from the References section). Therefore, each family page can be considered a full-fledged traditional publication, consisting of title, authors, abstract (the overview), content, and references.

In practice, many papers in the literature only reference GtoPdb (the root) without including a reference to the specific page being cited. That is, they only cite a paper describing GtoPdb as a whole (e.g., [35]) and refer to targets, ligands, diseases, etc. only by name. Thus, citations to specific families are *de-facto* "hidden" to citation systems such as Google Scholar, and useless for the computation of bibliometrics.

In certain "lucky" cases, as with papers available in PDF and published in the British Journal of Clinical Pharmacology [10] (BJCP), when a family, ligand, receptor name, etc. are used, they have a hyperlink pointing to the corresponding webpage in GtoPdb. Therefore, the citations to the families can be detected and counted using the URLs reported in the papers. However, these citations to GtoPdb webpages are not counted as such by citation systems, so they are not converted into credit for curators and collaborators.

For our running example, consider Table 1. This simplified version of GtoPdb contains three tables: `family`, `contributor` and `contributor2family`. The first table, `family`, has tuples representing families with three attributes: the id of the family, its name, and type. Table `contributor` contains people who have helped generate the data in the database. The third table, `contributor2family`, serves as a link between the families and the people who contributed to them. For instance, "John Smith" ($\mathtt{c}_1$) contributed to "Dopamine Receptors" ($\mathtt{f}_1$) as well as to the "YANK Family" ($\mathtt{f}_4$). Throughout the rest of the paper, we will use the `id` attribute of these tables as the

[10] https://bpspubs.onlinelibrary.wiley.com/journal/13652125

family

id	name	type
f_1	Dopamine Receptors	gpcr
f_2	Bile Acid Receptor	gpcr
f_3	FAK Family	enzyme
f_4	YANK Family	enzyme

contributor2family

id	family_id	contributor_id
$c2f_1$	f_1	c_1
$c2f_2$	f_1	c_2
$c2f_3$	f_2	c_3
$c2f_4$	f_4	c_1

contributor

id	Name	Country
c_1	John Smith	UK
c_2	Jim Doe	UK
c_3	Hans Zimmerman	Germany
c_4	Roberta Rossi	Italy

Table 1: Example of a database consisting of three tables. `family` contains receptor families; `contributor` contains the name and country of contributors; `contributor2family` connects contributors to the families they contributed to.

provenance token of its corresponding tuples, that is, as a symbol that serves to identify a tuple when talking about provenance.

4. Data Provenances

We now describe the three types of provenance used in this paper – lineage, why-provenance, and how-provenance – as well as the notion of Causality and Responsibility, and the Shapley value function.

4.1. Lineage

Lineage is the simplest form of provenance. It was first introduced by Cui et al. [23], and can be thought of as the set of all tuples that are used by the query to generate the output [17].

As an example, consider the following SQL query `Q1`, applied to the database described in Table 1, asking for the names of families curated by researchers based in the United Kingdom (UK):

```
Q1: SELECT DISTINCT f.name
FROM family AS f JOIN contributor2family AS c2f
ON f.id = c2f.family_id
JOIN contributor AS c ON c2f.contributor_id = c.id
WHERE c.country = 'UK'
```

I	database instance
L	lineage set of an output tuple
Γ	contingency set
ρ_t	responsibility of tuple t
Q	a query
I^n	set of endogenous tuples
I^x	set of exogenous tuples
$\mathcal{W}$	witness basis
W	a witness set
$\gamma(\mathcal{W}, t)$	set of witnesses in $\mathcal{W}$ containing t
$\mathcal{H}$	provenance polynomial
M_i	a monomial in $\mathcal{H}$
t_j	a tuple in M_i
$c(\mathcal{H})$	sum of $\mathcal{H}$'s coefficients
$e(M_i)$	sum of M_i's exponents
$mc(M_i)$	M_i's coefficient
$te(t_j, M_i)$	exponent of t_j in M_i
$\gamma(t_j, \mathcal{H})$	set of monomials in $\mathcal{H}$ containing t_j

Table 2: Notations used in this paper.

id	name	lineage
o_1	Dopamine Receptors	$\{f_1, c2f_1, c_1, c2f_2, c_2\}$
o_2	YANK Family	$\{f_4, c2f_4, c_1\}$

Table 3: Result of `Q1` over the database instance in Table 1 with the lineage of each output tuple. Attribute `id` is not part of the output, and was added to identify each tuple.

Table 3 shows the query output, which consists of two tuples. We add an extra attribute `id` so that we can easily refer to each result tuple. The lineage for tuple o_1 is the set $\{f_1, c2f_1, c_1, c2f_2, c_2\}$, since the tuple f_1 was joined with $c2f_1$ and then with c_1, and was also joined with $c2f_2$ and c_2. No other tuple is used in the database to produce o_1. For tuple o_2 the lineage is $\{f_4, c2f_4, c_1\}$. Lineage is defined for each tuple of the output, and can differ between tuples.

4.2. Why-Provenance

Why-Provenance was first defined in terms of a deterministic semistructured data model and query language [13]. We use here its definition in terms of the relational model [17].

While lineage aims to find all and only the tuples in the input relevant to the production of an output tuple, why-provenance aims to find sub-instances of the input that "witness" a part of the output. Given a tuple t in the query's output $Q(I)$, a *witness* is any sub-instance of the database that produces t, i.e., a set that guarantees the existence of t in $Q(I)$. In particular, the whole database and the lineage of t are both examples of witnesses of t. Since the definition of witness allows for the presence of "irrelevant" tuples, the set of all witnesses is finite (since the database instance I is finite), but it is potentially exponentially large [17].

Buneman et al. [13] defined the why-provenance of an output tuple t in the result $Q(I)$ as a special *subset* of the set of witnesses called the *witness basis*. The witnesses of the basis exclude tuples that are irrelevant to t being produced by Q, and thus the basis tends to be very small compared to the set of all possible witnesses [17].

In a sense, each witness in the witness basis captures one possible way in which a tuple in the output was generated by the query. To better understand this, consider the example in Table 4, where each tuple in the result of query `Q1` is annotated with its why-provenance.

id	name	why-provenance
o_1	Dopamine Receptors	$\{\{f_1, c2f_1, c_1\}, \{f_1, c2f_2, c_2\}\}$
o_2	YANK Family	$\{\{f_4, c2f_4, c_1\}\}$

Table 4: Result of `Q1` over the database instance in Table 1 with the why-provenance of each output tuple.

The why-provenance of output tuple o_2 has only one witness, which coincides with its lineage. This happens because there is only one way this output tuple can be produced, i.e., for tuple f_4 to be joined with $c2f_4$ and c_1. On the other hand, o_1 has a witness basis of two witnesses, since there are two possible ways in which the query can generate o_1. One possibility is that f_1 is joined with $c2f_1$ and c_1 (the first witness), and the second possibility is that f_1 is joined with $c2f_2$ and c_2 (the second witness). This means that to generate o_1, it is sufficient that only one of the two witnesses is present in the input database.

4.3. How-Provenance

While why-provenance describes the source tuples that witness an output tuple in the result of the query, it leaves out information about how the source tuples are used. How-provenance was therefore defined in [33] to capture this information using a *semiring* algebraic structure. It takes the form of a polynomial, called *provenance polynomial*, where the variables are taken from the set X of identifiers of the tuples (provided that each tuple in I has an identifier) and the coefficients are drew from the set of natural numbers $\mathbb{N}$.[11]

The key idea in Green et al. [33] is to use the two operators $+$ and $\cdot$ to represent two basic transformations that source tuples undergo as a result of applying a relational query to a database [17]. Two tuples may either be joined together (a join is represented with the $\cdot$ operator) or merged via union or projection (represented with the $+$ operator).

Table 5 shows the two output tuples of our running example annotated with their respective how-provenances. Tuple o_2 was produced by a join of the input tuples $f_4, c2f_4$, and c_1. The three provenance tokens are therefore "multiplied" together. The case of o_1 is slightly more complex, as already discussed. It can be obtained by the joins of two different sets of tuples,

[11] This semiring is commonly referred as $\mathbb{N}[X]$ in the literature.

id	name	how-provenance
o_1	Dopamine Receptors	$f_1 \cdot c2f_1 \cdot c_1 + f_1 \cdot c2f_2 \cdot c_2$
o_2	YANK Family	$f_4 \cdot c2f_4 \cdot c_1$

Table 5: Result of `Q1` over the database instance in Table 1 with the how-provenance polynomial of each output tuple.

so there are two monomials combined by + representing these alternative derivations. Each monomial corresponds, in a way, to the witnesses of the why-provenance of o_1.

Provenance polynomials may also have monomials whose exponents and/or coefficients are greater than one, for example, $3f_1 \cdot c2f_1 \cdot c_1 + f_1 \cdot c2f_2^3 \cdot c_2^3$. This is a polynomial of a tuple produced by a query where the result of the join between the tuples f_1, $c2f_1$, and c_1 is produced three times and then merged (e.g. as the result of a union), and the tuples $c2f_2$ and c_2 are used three times in the operation described by the second monomial (e.g., with nested queries).

4.4. Causality and Responsibility

A formal study of causality was introduced in [18, 34] and later expanded by Meliou et al. [47] to explain the causes of answers and non-answers to queries. In the following, we refer to the definition of causality and responsibility provided in [47]. In particular, we only focus on answers to a query since non-answers are not relevant in our context.

There are two types of "cause" tuples: counterfactual and actual. Let o be a tuple in the result of query q on the database instance I, and t a tuple in its lineage. We call t a *counterfactual cause* if, by removing t from I, o is also removed from the output (i.e., t is essential for the generation of t). We call t an *actual cause* if there is a set of tuples $\Gamma \subseteq I$ called a *contingency set*, such that t is a counterfactual cause in $I - \Gamma$. In other words, t is an actual cause if, even when removed from I, there is another set of tuples of the lineage that guarantees the presence of o.

Computing the causality of tuples is NP-complete for general queries [29], but for conjunctive queries can be computed in PTIME, as showed by Meliou et al. [47].

The notion of *responsibility* measures the degree of causality as a function of the size of the smallest contingency set [18]. This allows us to rank lineage

id	name	responsibility
o_1	Dopamine Receptors	$f_1 = 1, c2f_1 = 0.5, c2f_2 = 0.5, c_1 = 0.5, c_2 = 0.5$
o_2	YANK Family	$f_4 = 1, c2f_4 = 1, c_1 = 1$

Table 6: Result of Q1 over the database instance in Table 1 with the responsibilities of lineage tuples.

tuples based on their degree of causality in generating the output.

Definition 4.1. *Responsibility [47]*

Let o be an output tuple in the result of query Q on I, and let t be a cause for o. The responsibility *of t for the answer o is:*

$$\rho_t = \frac{1}{1 + min_{\Gamma}|\Gamma|}$$

where Γ ranges over all contingency sets for t.

Note that a counterfactual cause will have the maximum responsibility of 1, and that the larger the minimum contingency of an actual cause is, the smaller its responsibility will be since there are alternatives to guarantee the presence of the answer o.

As an example, consider Table 5, where we reported the result set of Q1 and the tuples of the lineages with their responsibility values. Focusing on o_1: the lineage tuple f_1 is a counterfactual cause, since its contingency set is empty (when removed from the database, o_1 disappears from the result set). Consequently, its responsibility is 1. All the other tuples of the lineage are actual causes. c_1, for example, has as minimal contingency set $\{c2f_2\}$, thus its responsibility is 0.5. For the output tuple o_2, all the tuples of the lineage are counterfactual causes, thus their responsibility is 1.

4.5. Shapley value

To use the Shapley in the context of conjunctive queries for relational databases, we use the definitions provided in [25]: given a query $q(\bar{x})$, a database D, an input fact $f \in D$ (here seen as a player) and a tuple $\bar{t}$ of same arity as $\bar{x}$, the Shapley value of f in D intuitively represents the contribution of f to the presence (or absence) of $\bar{t}$ in the query result. Formally, the Shapley value is defined as follows:

Definition 4.2. *Shapley value [26]*

Let the database instance I be partitioned into two sets of facts: a set I^x of *exogenous facts, and a set I^n of endogenous facts. Let Q be a Boolean query and $f \in I^n$ be an endogenous fact. The Shapley value of f in I for query Q is defined as:*

$$Shapley(Q, I^n, I^x, f) = \sum_{B \subseteq I^n \setminus \{f\}} \frac{|B|!\,(|I^n|-|B|-1)!}{|I^n|!} \left(Q(I^x \cup B \cup \{f\}) - Q(I^x \cup B)\right)$$

The set I^n of endogenous facts can be thought as the set of tuples being taken into consideration, while I^x is the set of ignored tuples. The choice on I^n is usually application-dependent.

The sum in the definition of the Shapley value is performed on all possible coalitions of fact B that do not contain the player f. Thus, the value $(Q(I^x \cup B \cup \{f\}) - Q(I_x \cup B))$ is the wealth brought by f when added to B. As we see, the Boolean query is used as wealth function v: its value is 1 only when the set $I^x \cup B \cup \{f\}$ makes the query true, and the set $I_x \cup B$ makes it false, i.e., when the addition of the fact f is determinant to make the Boolean query true. The value $|B|!\,(|I^n|-|B|-1)!$ is the number of all the possible permutations over I^n where the facts in B come first, then f is added, and then all the remaining facts. Thus, the value $\frac{|B|!(|I^n|-|B|-1)!}{|I^n|!}$ can be thought as a weight for the wealth brought by the addition of f to the coalition B.

To extend this definition to non-Boolean queries, we use the same straightforward approach used in Deutch et al. [25]: the Shapley value of the fact f for the answer $\bar{t}$ to $Q(\bar{x})$ is the value $Shapley(Q[\bar{x}/\bar{t}], I^n, I^x, f)$, where $Q[\bar{x}/\bar{t}]$ is the Boolean query defined by $Q[\bar{x}/\bar{t}](I) = 1$ if and only if $\bar{t}$ is in the output of $Q(\bar{x})$ on I, and 0 otherwise. *** DD: I added this paragraph down here hoping to make things clearer. If you think it fails to do so, feel free to delete it. *** In other words, the definition of $Shapley(Q, I^n, I^x, f)$ is extended to such queries $Q(\bar{x})$ with free variables by considering the Boolean query $Q[\bar{x}/\bar{t}]$ instead as value function. This query can be seen as a function that takes as input a set of facts and returns 1 if this set is a witness for $\bar{t}$, and 0 otherwise.

As an example, consider table 7, that shows the Shapley values for the lineage's tuples of o_1 and o_2, results of query `Q1`. Since the tuples of the lineage are the only one with a role in creating the output tuples, when computing the Shapley value we can use it as the set of endogenous facts. We note moreover that, to compute the Shapley value of an input tuple fm it

id	name	responsibility
o_1	Dopamine Receptors	$f_1 = \frac{7}{15}, c2f_1 = \frac{2}{15}, c2f_2 = \frac{2}{15}, c_1 = \frac{2}{15}, c_2 = \frac{2}{15}$
o_2	YANK Family	$f_4 = \frac{1}{3}, c2f_4 = \frac{1}{3}, c_1 = \frac{1}{3}$

Table 7: Result of `Q1` over the database instance in Table 1 with the Shapley values of the tuples of the lineage. In this case D^n corresponds to the lineage.

is sufficient to compute and sum the values $\frac{|B|!(|I^n|-|B|-1)!}{|I^n|!}$ for all the possible sets B such that $B \cup \{f\}$ is a witness and B is not. Thus, suppose we want to compute the Shapley value of the tuple f_1. Let us call $\bar{Q}_{1,o_1}$ the Boolean query such that $\bar{Q}_{1,o_1}(I) = 1$ if and only if o_1 is in the output of `Q1`, and L_{o_1} the lineage of o_1. Then the Shapley value of f_1 is given by:

$$\begin{aligned} Shapley(\bar{Q}_{1,o_1}, L, I \backslash L, f_1) &= \tfrac{2!2!}{5!} + \tfrac{2!2!}{5!} + \tfrac{3!}{5!} + \tfrac{3!}{5!} + \tfrac{3!}{5!} + \tfrac{3!}{5!} + \tfrac{4!}{5!} \\ &= \tfrac{7}{15} \end{aligned}$$

Where, for the first element of the sum the corresponding B is $\{c2f_1, c_1\}$, for the second element it is $\{c2f_2, c_2\}$, for the third it is $\{c2f_1, c2f_2, c_1\}$, for the fourth it is $\{c2f_1, c_1, c_2\}$, for the fifth it is $\{c2f_2, c_2, c_1\}$, for the sixth it is $\{c2f_1, c2f_2, c_2\}$, and for the seventh $\{c2f_1, c2f_2, c_1, c_2\}$. Every other possible coalition B would make the factor equal to 0.

Similarly, for tuple c_1 (and the other tuples of the lineage), the computation is:

$$\begin{aligned} Shapley(\bar{Q}_{1,o_1}, L, I \backslash L, c_1) &= \tfrac{2!2!}{5!} + \tfrac{3!}{5!} + \tfrac{3!}{5!} \\ &= \tfrac{2}{15} \end{aligned}$$

Similarly, it can be seen that for the tuples of o_2's lineage the corresponding Shapley values are all equal to 1/3, since they are all equally responsible for the generation of the output. As we can see, the sum of the Shapley values of all the tuples in an output tuple's lineage is always equal to 1 when using a Boolean query as wealth function.

5. Credit Distribution and Distribution Strategies

We now give formal definitions of data credit and Data Credit Distribution (DCD), and present three different Distribution Strategies (DSs) based on the forms of provenance discussed earlier: Lineage-based DS, Why-Provenance-based DS, How-Provenance-based DS, responsibility-based DS, and the Shapley value-based DS. We also show how these strategies distribute credit in the IUPHAR example discussed earlier.

5.1. Data Credit and Data Credit Distribution

Given a database instance I, a *recipient of credit* is a unit of information within I. In the case of relational databases, recipients may be (i) the whole database; (ii) a table; (iii) a tuple; or (iv) an attribute.

Data credit is a value $k \in \mathbb{R}_{>0}$. Every recipient in a database is annotated with a quantity of credit as a proxy for its importance. In this paper, we focus on *tuples* as recipients of credit.

Given a *distribution strategy* (DS), *Data Credit Distribution* (DCD) takes a database instance I, a quantity of credit k, and query Q over I, and it splits k among the recipients of credit in I.

In the following, we use the notation in Cheney et al. [17]: Given a database instance I, a *tuple location* (R, t) is a tuple t in relation R. With reference to the running example, `(family,` $\langle f_1$`, Dopamine Receptors, gpcr`$\rangle$`)` is the tuple location of the first tuple in the `family` relation. The set of all tuple locations in I is called *TupleLoc*. We use this to formally define DCD at the *tuple level*.

Definition 5.1. ***Tuple Level Data Credit Distribution (DCD)*** *[27] Given a query Q over I and $k \in \mathbb{R}_{>0}$, DCD is defined by the function $f_{I,Q}$: $TupleLoc \times \mathbb{R}_{>0} \rightarrow \mathbb{R}_{\geq 0}$ such that $f_{I,Q}(t, k) = h$ where $0 \leq h \leq k$ and $\sum_{t \in TupleLoc} f_{I,Q}(t, k) = k$. The function f_{IQ} is the distribution strategy (DS).*

As we can see, the DS is a function that annotates each tuple in the database with a real value, which is a fraction of the given quantity k. The only constraint is that the sum of the credit annotations on tuples must be k, i.e. that no credit is generated or destroyed during the distribution. Given I and Q, many different DSs may be defined as long as they sum up to k.

In what follows, we use information provided by data provenance to define distribution functions. For simplicity, we assume that the credit k is distributed equally across the set of output tuples (i.e. the result of a query), and discuss how the credit of one output tuple o, k_o, is distributed across the instance I.

5.2. A Lineage-based Distribution Strategy

In the lineage-based distribution strategy, each tuple in the output of a query distributes credit equally to each input tuple that appears in its lineage. More formally:

Definition 5.2. *Lineage-based Distribution Strategy [27]*
Let I be a database instance, Q a query over I, $o \in Q(I)$ an output tuple and k_o the credit associated to o. Let L be the lineage of o and t be a tuple in I, then t receives credit equal to:

$$f_{I,Q}(t, k_o) = \begin{cases} 0 & \text{if } t \notin L \\ \frac{k_o}{|L|} & \text{if } t \in L \end{cases}$$

Note that lineage-based DS distributes credit only to input tuples that have a role in creating o by the query Q, and that each receives an equal share of credit. Thus, the more tuples in a lineage set, the less credit each tuple receives.

As an example, consider the output tuples of Table 3, and assume that each output tuple has credit $k_o = 1$. The lineage of the first tuple, o_1, is the set $\{f_1, c2f_1, c_1, c2f_2, c_2\}$. Therefore, each tuple in this set receives credit 1/5. The other tuples of the database receive zero credit. The lineage of the second output tuple is $\{f_4, c2f_4, c_1\}$, therefore each of these tuples receives credit 1/3.

At the end of the process, tuples f_1, $c2f_2$ and c_2 each receive credit 1/5, tuples f_4 and $c2f_4$ receive 1/3, while tuple c_1 receives 8/15. Note that if a tuple appears in more than one lineage set, then it will accumulate credit from the distribution associated with each one of these sets, implying that it has a more significant role in the context Q, as is the case with c_1 in this example.

Not all of the tuples in the lineage of an output tuple are necessary to be present at the same time for the output tuple to appear in the query results. For example, if the database only had the set of tuples $\{f_1, c2f_1, c_1\}$ or the set $\{f_1, c2f_2, c_2\}$, the existence of o_1 would still be guaranteed. In other words, while f_1 is always needed for o_1 to appear in the output, only one of the sets of tuples $\{c2f_1, c_1\}$ and $\{c2f_2, c_2\}$ is required. One could therefore argue that it would be more fair for f_1 to receive more credit than the other four tuples, given its role in producing o_1.

This highlights one limitation of the lineage-based DS: while able to find all and only the relevant tuples of the output, it does not distinguish the *importance* of tuples in the query computations. We therefore present four other, more sophisticated, forms of distribution strategies based on why-provenance, how-provenance, responsibility, and Shapley value.

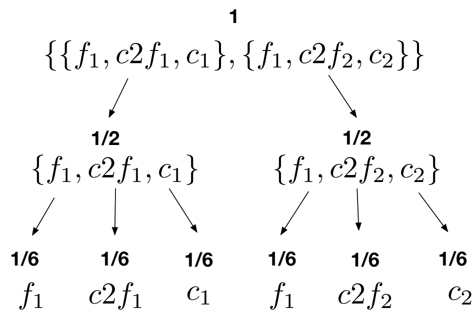

Figure 4: Distribution of credit using why-provenance-based DS for tuple o_1.

5.3. A Why-Provenance-Based Distribution Strategy

The distribution strategy based on why-provenance first equally distributes the credit k_o among the witnesses of the witness basis for o, and then equally divides the credit of a witness among the tuples in the witness. Since a tuple may appear in more than one witness, it will receive more than one portion of credit from the same distribution. More formally:

Definition 5.3. *Why-Provenance-based Distribution Strategy*
Let I be a database instance, Q a query over I, $o \in Q(I)$ an output tuple and k_o the total credit associated to o. Let $\mathcal{W} = Why(Q, I, o)$ be the witness basis of o according to Q and I, and $W \in \mathcal{W}$ be a witness.

Then tuple t in I receives credit equal to:

$$f_{I,Q}(t, k_o) = \frac{k_o}{|\mathcal{W}|} \sum_{W \in \gamma(\mathcal{W}, t)} \frac{1}{|W|}$$

where γ is a function which returns all witnesses W in which t appears:

$$\gamma(\mathcal{W}, t) = \{W \in \mathcal{W} : t \in W\}$$

Figure 4 shows the distribution of credit with why-provenance-based DS for tuple o_1. The credit is first equally divided between the two witnesses, so that both receive credit 1/2. The credit is then further divided among the tuples in each witness. Since each witness has three tuples, each tuple in a witness receives 1/6 of credit. At the end of the distribution, f_1 receives a total credit of 1/3, and the other tuples receive 1/6 each. This distribution better reflects the role of f_1 in the generation of o_1 since, as discussed earlier,

$$\mathcal{H} = \underbrace{3f_1 \cdot c2f_1 \cdot c_1}_{M_1} + \underbrace{f_1 \cdot c2f_2^3 \cdot c_2^3}_{M_2}$$

$$c(\mathcal{H}) = 4 \qquad e(M_2) = 7$$
$$mc(M_1) = 3 \qquad mc(M_2) = 1$$
$$te(c_2, M_2) = 3 \qquad \gamma(c_1, \mathcal{H}) = \{M_1\}$$
$$\gamma(f_1, \mathcal{H}) = \{M_1, M_2\}$$

Figure 5: Illustration of notation used to define the how-provenance based DS

it is the only mandatory tuple for o_1 to appear in the output; only one of the two other pairs of tuples are necessary for o_1 to appear in the result.

This example illustrates that why-provenance can better reward input tuples depending on their role. Tuples that appear in more than one witness are rewarded more than others.

5.4. A How-Provenance Based Distribution Strategy

The how-provenance-based DS first distributes the credit to the monomials of the polynomial accordingly to the weight represented by their coefficients, then to the tuples of each monomial accordingly to the weights represented by their exponents.

To define the DS more formally, we introduce some notation and illustrate it using the provenance polynomial $\mathcal{H}$ shown in Figure 5. This notation is also shown in Table 2 for easy reference.

We call c the function that, given a polynomial, returns the sum of its coefficients; thus $c(\mathcal{H}) = 3 + 1 = 4$. We call e the function that, given a monomial, returns the sum of its exponents, thus $e(M_2) = 1 + 3 + 3 = 7$. mc is the function that takes as input a monomial and returns its coefficient; thus $mc(M_1) = 3$. te is a function that takes as input a tuple and a monomial, and returns the exponent of the tuple in the monomial, if present; thus $te(c_2, M_2) = 3$. Finally, γ takes as input a tuple and the whole polynomial, and returns a set of monomials containing that tuple, if present in the polynomial; thus $\gamma(f_1, \mathcal{H}) = \{M_1, M_2\}$, $\gamma(c_2, \mathcal{H}) = \{M_2\}$.

Definition 5.4. *How-Provenance-Based Distribution Strategy*
Let I be a database instance, Q a query over I, $o \in Q(I)$ an output tuple, $\mathcal{H}$ be the provenance polynomial for o, and k_o the credit given to o. The credit

id	name
oxs_1	Dopamine Receptors

lineage	why-provenance	how-provenance
$\{f_1, c2f_1, c_1, c2f_2, c_2\}$	$\{\{f_1, c2f_1, c_1\}, \{f_1, c2f_2, c_2\}\}$	$f_1^2 c2f_1 c_1 + f_1^2 c2f_2 c_2$

Table 8: Result of query `Q2` applied on the database of Table 1 and its different provenances. The reported numbers are the credit distributed through the process.

given to tuple t in I is:

$$f_{I,Q}(t, k_o) = \frac{k_o}{c(\mathcal{H})} \sum_{M \in \gamma(t,\mathcal{H})} mc(M) \frac{te(t, M)}{e(M)}$$

Going back to the example of Table 5, consider o_1 with provenance polynomial $f_1 c2f_1 c_1 + f_1 c2f_2 c_2$. The how-provenance-based DS firstly divides the credit between the two monomials. Since the coefficients of each monomial are 1, the credit is split in half. If they were, for example, 1 and 2 respectively, 1/3 of the credit would go to the first monomial, and 2/3 to the second. Since in our example each variable has exponent 1, the credit is further divided equally among the three variables. Thus, at the end of the computation, f_1 receives 1/3, and the other tuples receive 1/6.

In this specific example, the how-provenance-based DS has the same outcome as the one based on why-provenance. We therefore consider another query over GtoPdb, `Q2`, that asks for the families of type `gpcr` that have as contributor a researcher located in the UK:

```
Q2: SELECT DISTINCT F.name
FROM family as F JOIN
(SELECT DISTINCT f.name AS name
FROM family AS f JOIN contributor2family AS c2f ON f.id = c2f.family_id
JOIN contributor AS c ON c2f.contributor_id = c.id
WHERE c.country = "UK") AS R ON F.name = R.name
WHERE F.type = "gpcr"
```

The result of `Q2` is shown in Table 8, and consists of one tuple, oxs_1, annotated with each of the three provenances. As can be seen, lineage and why-provenance are identical to those of the tuple o_1 in the previous example. The how-provenance, however, is different since tuple f_1 is used twice: first in the join of the inner query, and second in the join of the outer query. This

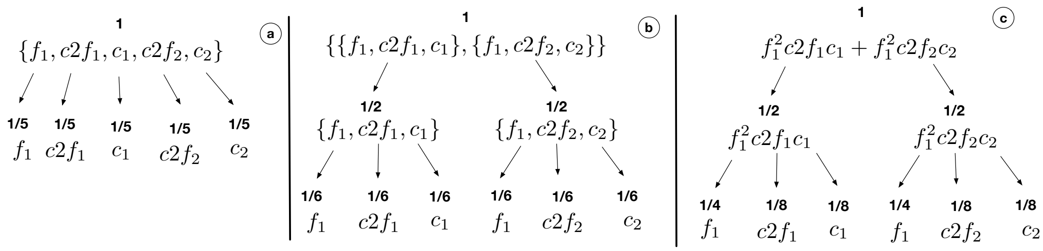

Figure 6: Comparison of different distributions strategies for tuple o_1 produced by query `Q2`.

information is lost in the first two forms of provenances since they are sets, but it is captured in how-provenance through the use of the operator '$\cdot$'.

Figure 6 shows the differences between the three DS for the tuple o_1 of Table 8. Subfigure 8.a uses lineage, sub-figure 8.b uses why-provenance, and sub-figure 8.c uses how-provenance. The DS based on the provenance polynomial gives credit $1/2$ to f_1, and $1/8$ to the other tuples. This is reasonable since `Q2` relies on f_1 even more than `Q1` does. The distribution based on how-provenance rewards f_1 more, showing that how-provenance is even more sensitive to the tuples' role in a query than why-provenance. This is a direct consequence of the fact that, as proven in [33], how-provenance is more general than why-provenance and lineage, in the sense that it contains more information.

5.5. Responsibility-based Distribution Strategy

As described in Section 4.3, causality and responsibility is new information that is added to lineage. One possible option for defining a distribution strategy using responsibility is to simply assign the responsibility of each tuple in the lineage of an output tuple as its credit. In this way, responsibility is both a way to compute credit and to distribute it. Using the example of Table 6, in the case of output tuple o_1, f_1 receives credit 1 and the other tuples receive credit 0.5.

However, we want a DS that is also a function of the input credit value k in order to be comparable with the other three strategies proposed so far. We define a new DS based on responsibility that is a function of the quantity of credit k_o that assigns to each tuple of the lineage a portion of this credit weighted by its normalized quantity of responsibility. This will give a bigger

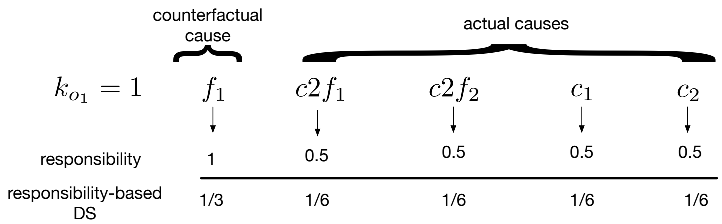

Figure 7: Example of distribution of credit using the responsibility-based DS, assuming $k_o = 1$.

portion of credit to tuples that are higher in the responsibility ranking. Formally:

Definition 5.5. *Responsibility-based Distribution Strategy*
Let I be a database instance, Q a query over I, $o \in Q(I)$ an output tuple, L the lineage of o, and k_o the credit given to o. The credit given to tuple t in I is:

$$f_{I,Q}(t, k_o) = k_o \frac{\rho_t}{\sum_{t' \in L} \rho_{t'}}$$

where ρ_j is the responsibility of tuple j as in Definition 4.1.

Note that only the tuples that belong to the lineage will receive a quantity of credit > 0. Furthermore, the more important the tuple is, i.e., the higher its responsibility, the larger the quantity of credit received.

Figure 7 shows the responsibility and credit assigned to the tuples of the lineage of the output tuple o_1 of Table 6. Assuming that $k_{o_1} = 1$, f_1 receives credit $1/3$, while the others receive credit $1/6$. As we see, the DS in this case returns the same distribution as that obtained using why-provenance as shown in Figure 6. This is not always the case though, as we show in the example of Section 6.2.

5.6. Shapley value-based Distribution Strategy

Similarly to Responsibility, the Shapley value can be seen both as a method to generate and distribute credit. Moreover, it can be seen that, using the definition of Shapley value for Boolean queries given in Section 4.3, the sum of the Shapley values of all the tuples of the lineage L of an output tuple o is 1. Thus, the definition of a Shapley value-based distribution strategy is straightforward:

Definition 5.6. *Shapley Value-Based Distribution Strategy*
Let I be a database instance, Q a query over I, $o \in Q(I)$ an output tuple, L the lineage of o and k_o the credit given to o. The credit given to tuple t in I is:

$$f_{I,Q}(t, k_o) = k_o \cdot Shapley(\bar{Q}_o, L, I \backslash L, t)$$

Where $\bar{Q}_o$ is the Boolean query such that $\bar{Q}_o(I) = 1$ if and only if o is in the output of Q on I.

As shown in Table 7, tuple f_1 in o_1's lineage takes credit $7/15$ when $k_{o_1} = 1$, while the other tuples of the lineage take credit $2/15$. This DS still rewards f_1 more than the other tuples, since it is more important than the other tuples of the lineage. This DS thus behaves differently from all the other four previous strategies. In particular, f_1 is rewarded more with this DS than with the others.

In the case of o_2 there is only one witness set, thus this DS behaves like all the others, distributing $1/3$ of credit to each tuple in the lineage.

6. Experimental Evaluation

To understand the trade-offs between these Distribution Strategies (DSs), we perform four sets of experiments using queries over target families presented on the GtoPdb website. The first set of experiments use real queries extracted from citations to GtoPdb published in the British Journal of Pharmacology. The second set uses synthetically produced provenance polynomials, corresponding to more complex queries, in order to better highlight the differences between the DSs. The third set of experiments considers the accrual of credit over time by the three strategies, again using synthetic queries. The fourth set of experiments shows how the DSs compare to traditional citations in giving credit to data curators using both real and synthetic queries.

The source code for the experiments is written in Java and supported by a PostgreSQL database. For purposes of reproducibility, the code we used for our experiments and all queries are available here: `https://bitbucket.org/dennis_dosso/credit_distribution_project`.

Figure 8: Comparison of four DS on the same table `family` using the distribution given by the queries retrieved from papers. Each cell is a tuple.

6.1. Real-world queries

Examples of real queries are drawn from papers published in the British Journal of Pharmacology (BJP) [12]. Each time a paper in this journal cites a webpage from GtoPdb, it reports the URL of the page. From this URL, the query used to obtain the webpage data can be determined. We considered all 889 papers in BJCP citing the IUPHAR/BPS Guide to pharmacology [35] as of October 2020, and extracted all webpage URLs to GtoPdb contained within the paper.[13]

The queries that we inferred are those used to build target family webpages within GtoPdb. An example was given in Figure 3, where we show how the structure of the "Adenosine receptors" family can be mapped into queries over the underlying database. In GtoPdb, all target family pages share a similar structure; the only difference is that individual sections, such as "contributors" or "further readings", may be missing. Therefore, the same queries can be used to build all of the target family pages by changing the family id used in the query (for example, in Figure 3, it is 3). Note that

[12] `https://bpspubs.onlinelibrary.wiley.com`

[13] The IUPHAR/BPS Guide is a journal that describes the structure and evolution of GtoPdb. At the time of writing, it had received more than 1200 citations on Google Scholar.

the queries are fairly simple SQL queries, and fall into a class called "select-project-join" or "SPJ" queries. A total of more than $12K$ different queries were built in this way. Without loss of generality, we give each tuple in the output of a query a credit of 1.

Results. Figure 8 shows the heat-maps obtained by the distribution of credit according to the five DS on one of the tables in the underlying database, `family`, which is often joined with other tables in the database to build the webpages. Each cell in a heat-map represents a tuple of the `family` table and the color indicates the amount of credit attributed to such tuple. It can be seen that the result of credit distribution over `family` is the same for all five strategies. The same result is also obtained with the other tables of the database used by the queries shown in Figure 3.

The reason why credit distribution is the same for all four strategies is that the queries are all simple SPJ queries, which use each table only once and do joins on key attributes. Under these conditions, each tuple of the output presents: (i) a how-provenance that is a single monomial with coefficient one and exponent one in each variable; (ii) a why-provenance with only one witness; (iii) a lineage that is the same of the witness in the basis, (iv) all tuples are counterfactual causes when considering responsibility, and (v) they all have the same importance in the production of the output tuples according to their Shapley value. Hence, for these queries, the five DSs behave in the same way: credit is uniformly distributed among the tuples of the lineage.

To illustrate this, consider one of the queries in Figure 3 which is used to build the output webpage:

```
Q3: SELECT c.first_names, c.surname
FROM contributor2family AS cf JOIN contributor AS c ON
cf.contributor_id = c.contributor_id
WHERE f.family_id = 3
```

`Q3` returned 10 tuples from the version of GtoPdb used. The first tuple, <Bertil B., Fredholm>, has $c_{939} \cdot c2f_{496}$ as its provenance polynomial. c_{939} represents the provenance token of a tuple in `contributor`, and $c2f_{496}$ the provenance token of a tuple in table `contributor2family`. The why-provenance of this tuple is $\{\{c_{939}, cf_{496}\}\}$, its lineage is $\{c_{939}, c2f_{496}\}$, both these tuples are counterfactual causes and have a responsibility of one. Therefore, the credit assigned to these tuples is 1/2 using all five DS. This happens for all the tuples in the output of each query of GtoPdb, thus making the distributions equivalent over all outputs.

However, this is not the case with more complex queries. As we showed in the previous section, when two or more tuples are merged as a result of a projection or union, the credit distributions will differ between the strategies.

6.2. Synthetic queries

To see what happens with more complex queries, we synthetically generated provenance polynomials in which the coefficients and exponents could be greater than one, and picked them at random from a uniform distribution. The queries involve three GtoPdb tables: `family`, `contributor2family`, and `contributor`. The polynomials were generated as follows: first, the number of monomials was decided by randomly choosing a number between one and six. Then, we randomly chose a tuple from the `family` table, one from the `contributor2family` table and one from the `contributor` table; these are the variables of the monomial. We then chose a coefficient for the monomial (between one and three) and an exponent for each tuple (between one and four). For the next monomial, we decided if we wanted to keep the same tuple from the table family as first tuple of the new monomial. To do so, we generated a random float number between zero and one. If the number was above 0.2, we changed the family tuple.

An example can be found in Figure 9, which shows a sample synthetic provenance polynomial (the how-provenance), the corresponding why-provenance, lineage, the causality of the tuples of the lineage, together with their responsibility, and, finally, the Shapley values of the lineage tuples. The resulting credit distribution for each DS is also shown (except for the Shapley values, that coincide with the distribution of credit).

As an example of how the distribution strategies behave with these synthetic queries, consider tuple f_5 in Figure 9. This tuple receives the highest quantity of credit using responsibility-based distribution and less credit using, in order, lineage, the Shapley value, why- and how-provenance. On the other hand, tuple f_1 is rewarded more by the Shapley value, then, in order, by why-provenance, how-provenance, responsibility, and finally lineage. This difference is explained considering the different role of the tuples in the generation of the output and the characteristics of the distributions. Generally speaking, the more complex the distribution (e.g., the how-provenance), the more credit is given to tuples that are more frequently used or more critical in the production of the output. Depending on the situation, i.e. on the syntax of the query, the distributions may differ among them. Responsibility creates a ranking among lineage's tuples describing the importance

How-provenance: $3f_1^3c2f_1^2c_1^2 + 2f_1c2f_2^3c_2^3 + 4f_5c2f_{17}^4c_{18}^3$
Credit distribution:
$f_1 = \frac{59}{315}, f_5 = \frac{1}{18}, c2f_1 = \frac{2}{21}, c2f_2 = \frac{2}{15}, c2f_{17} = \frac{2}{9}, c_1 = \frac{2}{21}, c_2 = \frac{2}{15}, c_{18} = \frac{1}{6}$

Why-provenance: $\{\{f_1, c2f_1, c_1\}, \{f_1, c2f_2, c_2\}, \{f_5, c2f_{17}, c_{18}\}\}$
Credit distribution:
$f_1 = \frac{2}{9}, f_5 = \frac{1}{9}, c2f_1 = \frac{1}{9}, c2f_2 = \frac{1}{9}, c2f_{17} = \frac{1}{9}, c_1 = \frac{1}{9}, c_2 = \frac{1}{9}, c_{18} = \frac{1}{9}$

Lineage: $\{f_1, f_5, c2f_1, c2f_2, c2f_{17}, c_1, c_2, c_{18}\}$
Credit distribution:
$f_1 = \frac{1}{8}, f_5 = \frac{1}{8}, c2f_1 = \frac{1}{8}, c2f_2 = \frac{1}{8}, c2f_{17} = \frac{1}{8}, c_1 = \frac{1}{8}, c_2 = \frac{1}{8}, c_{18} = \frac{1}{8}$

Causality: counterfactual causes: $\emptyset$,
actual causes: $\{f_1, f_5, c2f_1, c2f_2, c2f_{17}, c_1, c_2, c_{18}\}$
Responsibility:
$f_1 = \frac{1}{2}, f_5 = \frac{1}{2}, c2f_1 = \frac{1}{3}, c2f_2 = \frac{1}{3}, c2f_{17} = \frac{1}{2}, c_1 = \frac{1}{3}, c_2 = \frac{1}{3}, c_{18} = \frac{1}{2}$
Credit distribution:
$f_1 = \frac{3}{20}, f_5 = \frac{3}{20}, c2f_1 = \frac{1}{10}, c2f_2 = \frac{1}{10}, c2f_{17} = \frac{3}{20}, c_1 = \frac{1}{10}, c_2 = \frac{1}{10}, c_{18} = \frac{3}{20}$

Shapley value:
$f_1 = 0.258\bar{3}, f_5 = \frac{1}{8}, c2f_1 = 0.091\bar{6}, c2f_2 = 0.091\bar{6}, c2f_{17} = \frac{1}{8}, c_1 = 0.091\bar{6}, c_2 = 0.091\bar{6}, c_{18} = \frac{1}{8}$

Figure 9: Sample synthetic provenance polynomial (how-provenance) and corresponding why-provenance, lineage, responsibility, and Shapley values, together with the corresponding credit distributions. In the case of the Shapley value, the value is equivalent to the quantity of credit being distributed (assuming that the input credit is equal to 1.

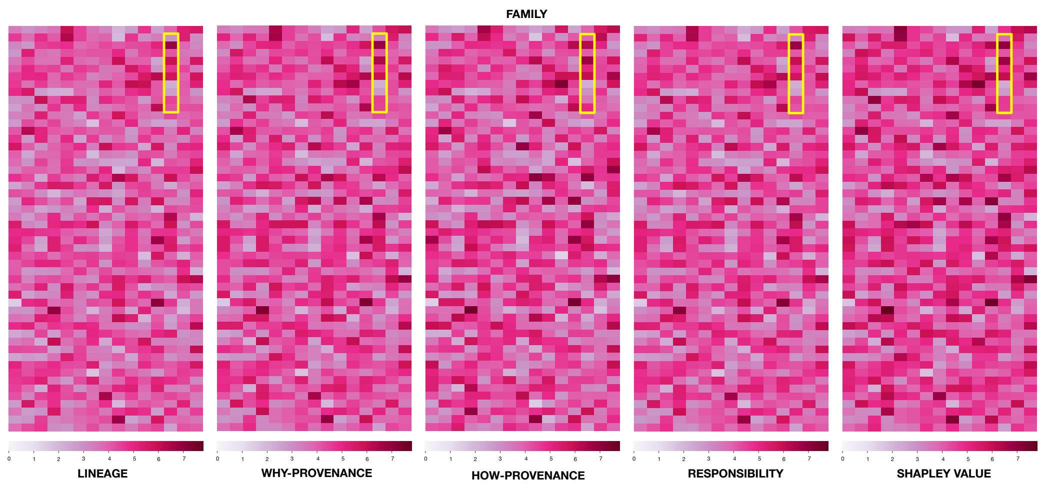

Figure 10: Comparison of three DS on the same table `family` after the distribution computed using 10K synthetic and randomly generated provenance polynomials. The tuples in the blue rectangles are used as example in the discussion connected to Figure 11.

of their role in generating the output. As such, the responsibility-based DS gives more credit to $f_1, f_5, c2f_17$ and c_18 due to their higher responsibility values. "Importance" is connected to their corresponding minimal contingency sets. For example, f_1 has a minimal contingency set (one of the many) $\{f_5\}$, with cardinality 1. On the other hand, c_1 has, as minimal contingency set (one of the many) $\{f_5, c_2\}$, with cardinality two. This means that c_1 is the "least important" amongst the tuples with minimal contingency sets of lower cardinality, and this is reflected in the different quantities of credit being distributed.

The Shapley value behaves similarly, but it rewards the most tuple f_1, then come in the ranking tuples f_5, $c2f_17$ and c_18, and the other tuples of the lineage with the lowest value. Although both Responsibility and the Shapley value create a ranking of the tuples based on their role in the generation of the output, the corresponding functions behave differently due to the syntax of the query, highlighting different aspects of the term "important".

Despite being synthetic, these provenance polynomials are realistic: they can be obtained by any nested query with join and union operations that use the same tuple multiple times (in which case the exponents are larger than one), and the same combination of operations more than once (in which case the coefficients of monomials are larger than one).

Table 9: Quantities of credit given using the 5 DSs on the first five tuples of table `family` (tuples ordered by the `family_id` attribute of the table).

lineage	why	how	responsibility	Shapley
3.3603537	3.416667	3.5928571	3.3611114	3.425758
4.4893217	5.111111	4.8620114	5.1752524	5.788059
3.1333337	3.7888894	2.9106944	3.5000005	4.200001
2.7972224	3.1111116	3.5601408	3.0055559	3.3305562
3.4670746	3.8944445	3.7216337	3.8992426	4.31758

Results. The results of credit distribution on the `family` table using 10K randomly generated synthetic provenance polynomials are shown in Figure 10. We set the maximum value in the heat maps to the highest value reached by a tuple in all five distributions (i.e., 7.7, with the Shapley value-based DS).

*** DD: is the table described in this paragraph below helpful? If not, please delete. *** As can be seen, the five strategies generate different credit distributions, indicated by the varying hues. We reported in Table 9 the values of credit assigned to the first five tuples of the table to show how these values actually differ between the five strategies. As can be seen, the strategies in these cases all behave differently. It is not even possible to identify a strategy that consistently rewards tuples more than the others, since this changes depending on the cases, reflecting the syntaxes of the polynomials being used.

However, there is a certain amount of consistency between the strategies in that tuples which are highly rewarded by one strategy are also highly rewarded by the others. This shows that the four DSs consistently reward certain tuples more than others.

Note that lineage-based DS gives the least credit to tuples in the `family` table, indicated by an overall lighter hue. This is because the DS distributes credit equally to all tuples appearing in the lineage. Since these queries also use two other tables, credit is distributed to tuples in those tables.

Moving to why-provenance-based DS, we see that more credit is given to tuples in the `family` table than with the previous strategy. This is because the DS considers the different ways that a tuple is used, e.g. in joins with other tuples. If the same tuple is present in more than one witness, it will draw more credit and take it from other tuples in the witness basis. In this case, tuples in `family` drew more credit, taking it from tuples in the other two tables, due to the role that `family` tuples played in the queries that were executed.

Consider the how-provenance-based DS heat-map. As with why-provenance, more credit is typically given to tuples in `family` compared to lineage-based DS, since it recognizes the role of these tuples in the queries, and the overall hue is deeper. The two distributions appear similar, although on closer inspection, slight differences can be seen. This is because how-provenance also considers the frequency with which tuples are used, not only the ways in which they are used. Therefore, although the overall distribution is similar, there are small differences due to the presence of exponents and coefficients in the provenance polynomials, influencing the distribution of credit.

The responsibility-based distribution strategy has a distribution that is also quite similar to the one provided by why-provenance (which is also visible from Table 9, where the values of the two distributions are different but very close). It is often the case, for example when the witnesses of the why provenance share one common tuple, that the two distributions behave similarly.

Finally, the heat-map reporting the distribution produced by the Shapley value is the one that, at a closer inspection, shows the biggest differences. Although the tuples that receive the biggest quantities of credit are the same, the hue of this tuple is different. The Shapley value in certain circumstances differs greatly from the other DSs, thus showing its ability to weight differently the roles of the tuples.

We note that the lineage-based DS gives an average credit of 3.92 to each tuple in the table, while the DS based on why-provenance assigns 4.19, how-provenance 4.18, the one based on responsibility 4.13, and the one based on the Shapley value 4.40. Moreover, lineage distributed a total of about 3121 units of credit to the `family` table, why-provenance 3333, how-provenance 3331, while responsibility assigned 3290, and the Shapley value 3505. Thus, the Shapley value is the method that accumulates the highest quantity of credit in this table.

To better understand the differences between DSs, in the next subsection we consider the accrual of credit over time. In doing so, we will focus on the ten tuples shown within the large yellow rectangles in Figure 11. Each small rectangle within a large yellow rectangle is a tuple, and we number them from 1 (top) to 10 (bottom). These ten tuples were cherry-picked because they allow us to see the evolution of the distribution of credit through time. There are other tuple sets that could have been selected driving us to the same considerations.

6.3. Credit accrual over time

Since credit accrues over time, we simulate the passage of time by varying the number of queries executed, and look at the "snapshots" of credit for each of the strategies using synthetic queries. The results are shown in Figure 11.

In this figure, four groups of heat-maps are shown. Each group represents a "snapshot" taken after 1K, 2K, 5K and 10K provenance polynomials have been considered for credit distribution. The ten tuples in each heat-map are those highlighted in the yellow boxes of Figure 10 from the `family` table.

The polynomials used are the same as the experiment of the previous section. The range of credit in each map goes from 0 (no credit) to 7 (the maximum quantity of credit reached – using how-provenance – on one of the tuples of the considered window at the "snapshot" with 10K queries). The color hue of the legend, as can be seen, still ranges from 0 to 7.7.

By the end of 1K queries, credit differentials between tuples as well as between strategies can be seen. For example, tuple 3 is usually rewarded the most credit by all five strategies. Moreover, it can be seen that tuples 1 receives a higher quantity of credit when how-provenance is adopted, showing how this form of provenance behaves differently from the others in this context. Moving to $2K$ queries, it is possible to see that tuple 3 and 7 are still the most rewarded by the strategies.

By the end of 5K queries, tuple 7 emerges with the highest value of credit with all five DSs, a position which is strengthened with 10K queries. Moreover, with the passing of time, tuple 3 ceases to be one of the most rewarded ones and new tuples, such as 6 and 9, emerge as being particularly rewarded at 5K, while at 10K tuples 6 and 7 are the most rewarded from the distributions. This is because tuple 7 is used several times within queries being executed, which is rewarded strongly by why- and how-provenance. We also note that the responsibility-based distribution confirms its trend of being similar to why-provenance, although not identical. This is more evident at step 10K, where tuple 7 is slightly less rewarded using responsibility (6.12) with respect to why-provenance (6.24). The responsibility that rewards the more tuple 7 is the one based on how-provenance (credit 7.03), followed by the Shapley value (credit 6.64). This is due to the fact that tuple 7 had, among some of the polynomials being used for the experiments, a high responsibility but it did not appear in all witnesses. This changed slightly the distribution.

While the relative value of credit "positions" of tuples within a DS strategy depends on what queries are being executed, the important thing to

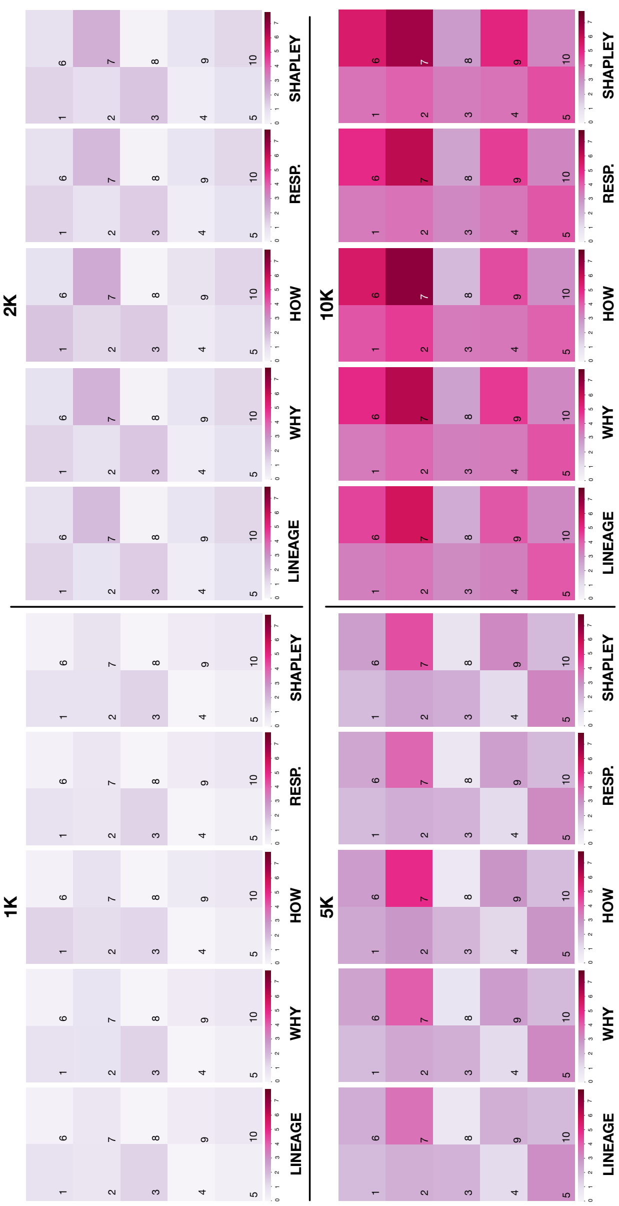

Figure 11: Comparison of the distribution of credit performed by the five DSs on a subset of 10 tuples taken from the `family` table, simulating the passing of time. The number at the top of each group of heat-maps represents the number of polynomials whose credit has been distributed.

notice is the difference between the DSs over time: overall, lineage gives less credit to tuples in the `family` table than the other strategies since credit is shared with tuples in other tables. The other strategies recognize the more important role being played by the `family` tuples than those in the other tables. The differences between why- and responsibility-based DS are, for the most times, negligible. The differences between the why- and how-provenance-based DSs are also relatively minor in most cases. However, there are certain situations in which the role of a tuple is particularly critical in a query, and in this case the difference in the value of credit assigned is notably higher for how-provenance and the Shapley value, as we saw with tuple 7 in the example of Figure 11.

To sum up, the DS based on lineage is sufficient to highlight which tuples in the database are used by a query, and distributes credit equally to these tuples. The resulting distribution rewards tuples that are used by more queries, but does not reward how many times tuples are used in the same query. However, a DS based on why-provenance, responsibility, Shapley value or how-provenance may be better if the queries are complex, since they reward more tuples that have a critical role in generating the output. In particular, these four DSs may be useful for finding "hotspots" in the database based on the role of tuples, with the how-provenance-based and Shapley value-based DSs being preferable if a higher sensitivity to the role of a tuple in queries is required.

6.4. Credit vs Citations

In the last set of experiments, we compare traditional citations to the proposed credit distribution strategies to see the difference in reward for data authors and curators. Using both real-world and synthetic queries, we distribute credit to the authors responsible for the data under the different strategies. Our results show that credit rewards authors of data that is cited fewer times, but that has a higher impact on the query results.

To do so, we need to identify a set of authors and queries that cite data curated by them. Considering GtoPdb, each target family page has a list of curators, representing the people who are co-creators and curators of the data comprising the page. This list can be obtained using the last query shown in Figure 3. Each time a target family page is cited, we assign one *citation* to each author associated with the page. The authors also receive *credit* in the amount assigned to the data used by the query to construct the webpage, equally divided between the authors of the webpage.

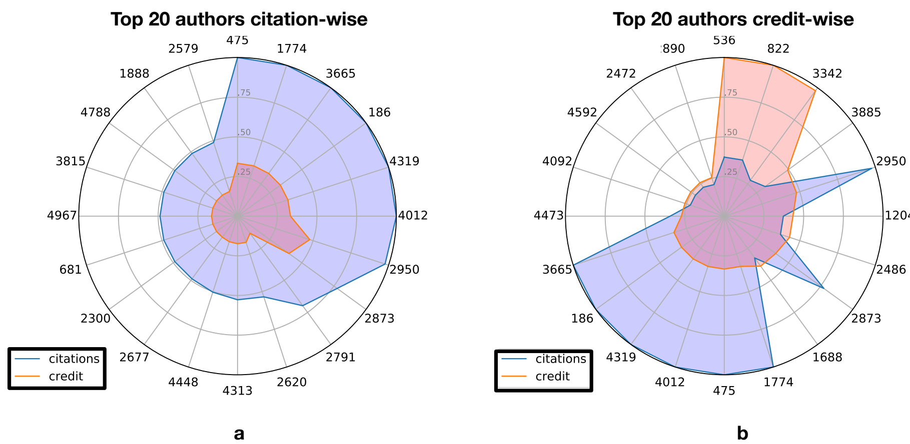

Figure 12: Radars presenting the top 20 authors citation-wise and credit wise, together with their (normalized between 0 and 1) values of citations and credit.

Results: Real-world queries. As described in Section 6.1, we consider real-world queries taken from papers published in the BJP which reference web-pages in GtoPdb. Since for these queries there is no difference in the distribution of credit between the DSs, only one value for credit is used.

The results are shown in the radar plots of Figure 12, in which each number on the outer circle (e.g. 475, 1774 and 3665) represents an author (id) and the blue (red) line represents the normalized value of credit generated by citations (credit), respectively. The first radar plot, Figure 12.a, shows the top 20 authors in terms of *citations*, ordered in a clockwise direction, whereas Figure 12.b orders the authors based on *credit*. Comparing the author ids used in the outer circles of these two plots, it can immediately be seen that the "top authors" are very different using these two metrics, although there is some overlap (for example, authors 1774, 475, and 4012).

Diving a bit deeper to focus on the red and blue areas in each of the plots reveals that there is a significance difference between citations and credit: The top 20 authors in terms of citations do not have the highest values of credit (Figure 12.a). Conversely, the authors with the highest values of credit do not necessarily have a large number of citations (Figure 12.b). For example, author 536 has the highest value of credit, but is not even in the top 20 authors in terms of citations. This means that authors like 536, 822, and 3342 in Figure 12.b receive much more credit from their relatively few citations than authors like 475, who receives the largest number of citations.

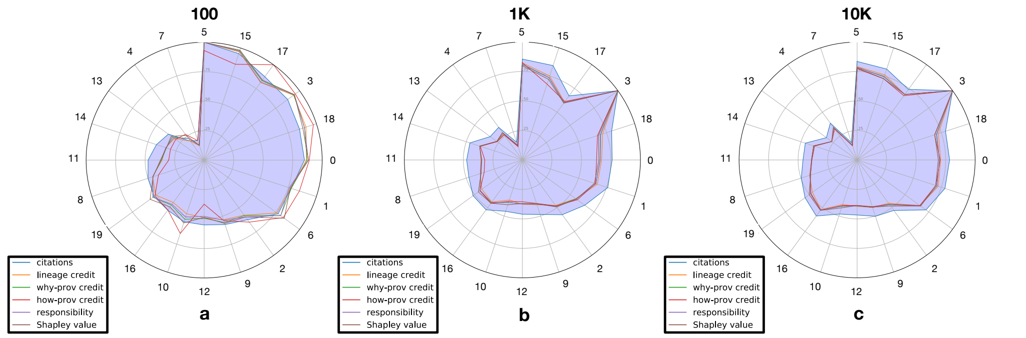

Figure 13: Radars presenting the 20 synthetic authors with corresponding citation and quantities of credit distributed through the 4 DSs (all values normalized between 0 and 1) through different numbers of polynomials (respectively, 100, 1K and 10K). The order is the one defined by figure a, i.e. descending order of citations obtained from 100 polynomials.

That is, the data underlying certain webpages is more "valuable" in terms of credit than a citation to the webpage.

The reason for the difference between citation and credit is partly due to the experimental setup: each output tuple carries a credit of 1, and there can be many tuples used to generate a webpage. Thus a webpage that is created from more tuples will have a higher credit value than one created from fewer tuples. Furthermore, authors who collaborated with fewer people will receive a biggest share of the equally divided credit. However, all authors will receive a citation of one.

Credit distribution therefore rewards authors differently than traditional citations: an author who has curated larger quantities of cited data and collaborated with fewer co-authors, will receive larger quantities of credit. Thus, credit rewards them for their larger contribution to the database.

Results: Synthetic queries. We used the same synthetic polynomials described in Section 6.2, and we distributed credit with the first 100, $1K$, and $10K$ of them. Since these polynomials are created by randomly selecting tuples from three tables, they usually correspond to a set of data curated by authors who, in reality, did not collaborate. To make the size of the author set more realistic, we therefore created 20 synthetic authors, and randomly assigned one author to blocks of consecutive tuples in the database, with the size of each block varying between 10 and 40, to simulate different quantities of work performed by an author. Every time an author appears as curator of one or more tuples used in a polynomial, we assign them one citation. They also receive four kinds of credit, each one using a different DS.

Figure 13 shows three radar plots, one for the results obtained with 100 polynomials, one with 1K polynomials, one with 10K polynomials. Each plot shows the top 20 authors in terms of citations (hence the authors and clockwise ordering is the same in each of the plots), and additionally shows the the normalized values of citation (blue line), lineage-based credit (yellow line), why-provenance-based credit (green line), how-provenance-based credit (red line), responsibility-based credit (violet line), and the Shapley value-based credit (brown line).

As can be seen, given the synthetic nature of the queries, the correlation between the number of citations and the quantity of credit assigned to the authors appears to be a much stronger than with the real-world queries of Figure 12. In fact, for Figure 13.a the linear correlation between the citation number and all four types of credit is always above 0.94 with p values in the order of 3e-8. The credit distributed via lineage is closest to the number of citations (a linear correlation of 0.99, p value of 2e-16 in Figure 13.a), while the other three types of credit behave slightly differently (a linear correlation of around 0.95 or above in all other four cases in Figure 13.a). Similar observations can be made for Figure 13.b and 13.c.

What these figures show is that, in certain cases, authors who do not have a large number of citations receive more credit than others, as for example authors 17, 18 and 10 in Figure 13.a, and especially when credit is distributed using how-provenance. This again shows how credit gives a different perspective on the role of data and authors by going beyond the limitations of traditional citations.

It is worth noting that, when scaling up to $1K$ and $10K$ polynomials, the credit distributions become almost identical (the linear correlation for the values of Figure 13.c is more than 0.99 with a p-value of 1.32e-32). This is consistent with what we observed in Figure 10.

7. Discussion

Before concluding, we discuss some design decisions: the focus on Credit Distribution (as opposed to Credit Generation), and the choice of Distribution Strategies.

7.1. Credit Generation

In this paper we focused on Credit Distribution, the problem of distributing credit generated by a citation to the parts of the database referenced by the query. A different problem is Credit Generation, the task of generating credit which is then distributed. Credit Generation presents a series of issues which are shared by traditional citation practices. For instance, defining the quantity of credit to be generated for a given citation is still an open problem. Different types of citations may generate different quantities of credit. Data cited as previous work or as useful for previous work may generate less credit than other data extensively used to produce the results presented in a paper. The computation of credit could be done manually (although we must consider the complexity of the task, human biases and the resources required to carry it out) or automatically, but it must be based on a shared definition of impact which is still not agreed upon for data or for traditional citation. For this reason, we used a uniform credit assignment.

There is also the problem of *transitive credit distribution*, i.e., how to transitively propagate credit from one cited unit to another unit that was used to produce the one being cited. For this, a graph of cited units that propagate credit between the units depending on influence could be used. How to propagate credit is an open and non-trivial problem that needs to consider the importance and impact of a citation in a work, be it a paper or data, and how to eventually compute the quantity of credit to be propagated.

Finally, in our experiments we assumed that the credit carried by an output tuple is one. Thus, each tuple in the output has equal importance. As described above, this assumption may be revised and different credit to different output tuples could be assigned. Note that from the distribution model viewpoint no change is required since the DCD is defined for a generic value k.

7.2. Choice of Distribution Strategies

In this paper we presented four different DSs, so the natural question is which one to use. This depends on the task at hand. When we want to highlight the tuples being used in the database by a workload, the lineage-based DS may be sufficient. When we also want to know the relative impact of tuples in the context of the query, the other DSs should be used since they give a better understanding of the importance of data.

In the real-world based experiments, the four DSs behaved the same, which was due to the specific nature of the data and the queries being used.

However, the why-provenance of a query will differ from the lineage of the same query whenever the output tuples can be computed in more than one way by the query, i.e., if there is more than one witness. This is usually true when join and projection operators are used in the query.

To address the question of what types of queries are likely to extract cited data, we turn to the results of published studies on the characteristics of query workloads and the complexity of their queries [39, 54, 59]. These studies show that operations such as inner-/outer-joins and projections occur in a significant number of queries. Therefore why- and how-provenances may become quite complex in certain cases and provide a distribution of credit that is significantly different from the one obtained with lineage.

* Is there more to say here? What are the general queries for which responsibility is hard to compute, and can the various provenances handle them at all? I know that provenance semi-rings has been extended to SPJU and aggregate queries, so imagine this means the others can be extended since it is a general framework. *

From a complexity standpoint, all four DS are similar since we focused on SPJ queries. However, responsibility is hard to compute for general queries. In terms of implementation, lineage is the simplest to compute since it only cares about a tuple being used, while the other provenances also need additional information to be taken into consideration.

8. Conclusions and Future Work

This paper defines four new distribution strategies based on why-provenance, how-provenance, responsibility, and the Shapley Value, and it compares them against the lineage-based distribution strategy defined in [27]. The first, why-provenance-based DS, uses the concept of a witness, and gives more credit to tuples that appear in more than one witness. In this way, tuples that are more important to the query and are used in different ways are rewarded more. The second, how-provenance-based DS, considers the frequency with which a tuple or combination of tuples is used in the query through the information contained in a provenance polynomial. In this case, the how-provenance-based DS is more sensitive than the why-provenance-based DS to the role and importance of tuples. The third DS exploits the notion of responsibility, a real value which ranks the lineage tuples based on their degree of causality in generating the output. The responsibility-based DS was shown to behave similarly to the why-provenance based DS. The fourth DS uses the Shapley value function, used to rank the facts of the database, seen as players, in producing the required result. To do so, the wealth function in the Shapley value's definition was adapted for general free-variable queries on the database.

To show the differences between the five DSs, we performed extensive experiments based on GtoPdb, a curated scientific relational database, using both real and synthetic queries. In the first set of experiments, we used select-project-join (SPJ) queries extracted from citations to webpages in GtoPdb found in papers published in the British Journal of Pharmacology. Using these "real" queries, we distributed credit to tuples in different tables of the database, highlighting tuples that were more frequently used. We showed that, with these queries, the four strategies produce the same distribution. This is because the SPJ queries were fairly simple, and did not use self-joins. Therefore the formulas underlying the different DSs had the same output.

In the second set of experiments, we synthetically produced more complex provenance polynomials, corresponding to more complex queries, that resulted in exponents and coefficients in the provenance polynomials that were greater than (or equal to) 1. These experiments highlighted the differences between the four DSs. While the DS based on lineage rewards all the tuples used by a query equally, the strategies based on why-provenance and responsibility give more credit to tuples that are more critical to the query. In particular, why-provenance considers the different ways in which a tuple is used in a query, while responsibility considers the relative importance of a tuple in the generation of the output. The DS based on the Shapley value similarly rewards the tuples based on their participation. The more impactful the role of a tuple, the higher its reward in credit. This distribution proved to be different from the previous two and to reward even more tuples that are used in more than one witness. How-provenance is even more sensitive to the tuple's role: it also considers the frequency with which a tuple or a set of tuples is used.

In the third set of experiments, we showed how the differences between the DS are compounded over time, i.e. when more and more queries are processed by the system.

In the fourth set of experiments we compared traditional citations to authors to the credit accrued to them via the DSs. We showed how, in both real-world and synthetic scenarios, credit rewards authors who contribute/curate data that has the highest impact, and therefore receives the biggest quantity of credit, and not necessarily the data with the highest citation count. In this sense, credit appears to be an useful new measure to discover data and their corresponding curators that have a high impact in the research world, even when they are cited few times or do not appear at all in the data that are cited (i.e. the case of data used to build the output of a query but that is not visualized in the output itself).

In future work, we plan to explore different strategies to generate and distribute credit. In this paper we assumed that each output tuple carries credit 1. In more sophisticated scenarios we can employ different strategies to compute credit, that reflect the importance of cited data. Other, more sophisticated, strategies could also be used to decide how credit is distributed between the authors, beyond the uniform distribution used here, in a way to reflect the work performed by them on the cited data. There are also a number of other intriguing applications for credit over relational databases. One such application is *data pricing*, which gives a price to a query submitted by a user who wants to buy the produced information. Currently, a common strategy used for data pricing is based on query rewriting: A database stores a set of views with their price. When a new query arrives, the system rewrites it using the stored views to obtain a query price, a process that can be computationally expensive. We plan to distribute credit through carefully planned and representative queries, and use credit information to define a new, faster, and potentially more flexible pricing function.

Another application is *data reduction* [48], which addresses the problem of reducing the vast – and rapidly expanding – amount of data that is being produced. Data credit can help address this problem by identifying "hotspots" and "coldspots" of data. A hotspot is data in a database (e.g. a tuple) with a high quantity of credit, which is therefore valuable for the set of queries that execute frequently over the data and distribute the credit. A coldspot is data with a low quantity of credit which can therefore be considered as less important, and could be deleted, summarized, or moved to cheaper and/or less efficient memory.

Acknowledgement

The work was partially supported by the ExaMode project, as part of the European Union H2020 program under Grant Agreement no. 825292.

References

[1] Abadi, D., Ailamaki, A., Andersen, D., Bailis, P., Balazinska, M., Bernstein, P., Boncz, P., Chaudhuri, S., Cheung, A., Doan, A., Dong, L., Franklin, M. J., Freire, J., Halevy, A., Hellerstein, J. M., Idreos, S., Kossmann, D., Kraska, T., Krishnamurthy, S., Markl, V., Melnik, S., Milo, T., Mohan, C., Neumann, T., Chin Ooi, B., Ozcan, F., Patel, J., Pavlo, A., Popa, R., Ramakrishnan, R., Ré, C., Stonebraker, M., and Suciu, D. (2020). The seattle report on database research. *SIGMOD Rec.*, 48(4):44–53.

[2] Alawini, A., Davidson, S. B., Hu, W., and Wu, Y. (2017). Automating data citation in citedb. *PVLDB*, 10(12):1881–1884.

[3] Alawini, A., Davidson, S. B., Silvello, G., Tannen, V., and Wu, Y. (2018). Data citation: A new provenance challenge. *IEEE Data Eng. Bull.*, 41(1):27–38.

[4] Altman, M., Borgman, C. L., Crosas, M., and Martone, M. (2015). An Introduction to the Joint Principles for Data Citation. *Bulletin of the Association for Information Science and Technology*, 41(3):43–45.

[5] Baggerly, K. (2010). Disclose all data in publications. *Nature*, 467(7314):401–401.

[6] Bechhofer, S., Buchan, I. E., De Roure, D., Missier, P., Ainsworth, J. D., Bhagat, J., Couch, P. A., Cruickshank, D., Delderfield, M., Dunlop, I., Gamble, M., Michaelides, D. T., Owen, S., Newman, D. R., Sufi, S., and Goble, C. A. (2013). Why linked data is not enough for scientists. *Future Gener. Comput. Syst.*, 29(2):599–611.

[7] Belter, C. W. (2014). Measuring the Value of Research Data: A Citation Analysis of Oceanographic Data Sets. *PLoS ONE*, 9(3):e92590.

[8] Bertin-Mahieux, T., Ellis, D., Whitman, B., and Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, pages 591–596.

[9] Borgman, C. L. (2016). Data Citation as a Bibliometric Oxymoron. In Sugimoto, C. R., editor, *Theories of Informetrics and Scholarly Communication*, pages 93–116. De Gruyter Mouton.

[10] Buneman, P. (2006). How to cite curated databases and how to make them citable. In *18th International Conference on Scientific and Statistical Database Management, SSDBM*, pages 195–203. IEEE Computer Society.

[11] Buneman, P., Christie, G., Davies, J. A., Dimitrellou, R., Harding, S. D., Pawson, A. J., Sharman, J. L., and Wu, Y. (2020). Why data citation isn't working, and what to do about it. *Database J. Biol. Databases Curation*, 2020.

[12] Buneman, P., Davidson, S. B., and Frew, J. (2016). Why data citation is a computational problem. *Commun. ACM*, 59(9):50–57.

[13] Buneman, P., Khanna, S., and Tan, W. C. (2001). Why and where: A characterization of data provenance. In *Database Theory - ICDT 2001, 8th International Conference*, pages 316–330.

[14] Buneman, P. and Silvello, G. (2010). A rule-based citation system for structured and evolving datasets. *IEEE Data Eng. Bull.*, 33(3):33–41.

[15] Callaghan, S., Donegan, S., Pepler, S., Thorley, M., Cunningham, N., Kirsch, P., Ault, L., Bell, P., Bowie, R., Leadbetter, A. M., Lowry, R. K., Moncoiffé, G., Harrison, K., Smith-Haddon, B., Weatherby, a., and Wright, D. (2012). Making Data a First Class Scientific Output: Data Citation and Publication by NERC's Environmental Data Centres. *International Journal of Digital Curation*, 7(1):107–113.

[16] Candela, L., Castelli, D., Manghi, P., and Tani, A. (2015). Data Journals: A Survey. *Journal of the Association for Information Science and Technology*, 66(9):1747–1762.

[17] Cheney, J., Chiticariu, L., and Tan, W. (2009). Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474.

[18] Chockler, H. and Halpern, J. Y. (2004). Responsibility and blame: A structural-model approach. *J. Artif. Intell. Res.*, 22:93–115.

[19] CODATA-ICSTI Task Group on Data Citation Standards and Practices (2013). *Out of Cite, Out of Mind: The Current State of Practice, Policy, and Technology for the Citation of Data*, volume 12.

[20] Cousijn, H., Feeney, P., Lowenberg, D., Presani, E., and Simons, N. (2019). Bringing citations and usage metrics together to make data count. *Data Science Journal*, 18(1).

[21] Cronin, B. (1984). *The Citation Process. The Role and Significance of Citations in Scientific Communication.* London: Taylor Graham.

[22] Cronin, B. (2001). Hyperauthorship: A postmodern perversion or evidence of a structural shift in scholarly communication practices? *JASIST*, 52(7):558–569.

[23] Cui, Y., Widom, J., and Wiener, J. L. (2000). Tracing the lineage of view data in a warehousing environment. *ACM Trans. Database Syst.*, 25(2):179–227.

[24] Davidson, S. B., Deutch, D., Milo, T., and Silvello, G. (2017). A model for fine-grained data citation. In *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research*. www.cidrdb.org.

[25] Deutch, D., Frost, N., Kimelfeld, B., and Monet, M. (2022a). Computing the Shapley Value of Facts in Query Answering. In Bonifati, A. and Abbadi, A. E., editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, June 12-17, 2022*. ACM.

[26] Deutch, D., Frost, N., Kimelfeld, B., and Monet, M. (2022b). Computing the shapley value of facts in query answering. *the 2022 Special Interest Group on Management of Data conference (SIGMOD).* in print.

[27] Dosso, D. and Silvello, G. (2020). Data credit distribution: A new method to estimate databases impact. *Journal of Informetrics*, 14(4):101080.

[28] Dubernet, M. L., Antony, B. K., Ba, Y. A., et al. (2016). The virtual atomic and molecular data centre (VAMDC) consortium. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 49(7):074003.

[29] Eiter, T. and Lukasiewicz, T. (2002). Complexity results for structure-based causality. *Artif. Intell.*, 142(1):53–89.

[30] Fang, H. (2018). A discussion of citations from the perspective of the contribution of the cited paper to the citing paper. *JASIST*, 69(12):1513–1520.

[31] Garfield, E. (1999). Journal impact factor: a brief review. *Can. Med. Assoc.*, 979-980.

[32] Gößwein, B., Miksa, T., Rauber, A., and Wagner, W. (2019). Data identification and process monitoring for reproducible earth observation research. In *2019 15th International Conference on eScience (eScience)*, pages 28–38. IEEE.

[33] Green, T. J., Karvounarakis, G., and Tannen, V. (2007). Provenance semirings. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 31–40. ACM.

[34] Halpern, J. Y. and Pearl, J. (2013). Causes and explanations: A structural-model approach — part 1: Causes. *CoRR*, abs/1301.2275.

[35] Harding, S. D., Sharman, J. L., Faccenda, E., Southan, C., Pawson, A. J., Ireland, S., Gray, A. J. G., Bruce, L., Alexander, S. P. H., Anderton, S., Bryant, C., Davenport, A. P., Doerig, C., Fabbro, D., Levi-Schaffer, F., Spedding, M., Davies, J. A., and Nc-Iuphar (2018). The IUPHAR/BPS guide to PHARMACOLOGY in 2018: updates and expansion to encompass the new guide to IMMUNOPHARMACOLOGY. *Nucleic Acids Research*, 46(Database-Issue):D1091–D1106.

[36] Hartley, J. (2017). Authors and their citations: a point of view. *Scientometrics*, 110(2):1081–1084.

[37] Hey, T., Tansley, S., and Tolle, K. M. (2009). Jim Gray on eScience: a transformed scientific method.

[38] Honor, L. B., Haselgrove, C., Frazier, J. A., and Kennedy, D. N. (2016). Data citation in neuroimaging: proposed best practices for data identification and attribution. *Frontiers in neuroinformatics*, 10:34.

[39] Jain, S., Moritz, D., Halperin, D., Howe, B., and Lazowska, E. (2016). Sqlshare: Results from a multi-year sql-as-a-service experiment. In *Proceedings of the 2016 International Conference on Management of Data*, pages 281–293.

[40] Joshi-Tope, G., Gillespie, M., Vastrik, I., D'Eustachio, P., Schmidt, E., de Bono, B., Jassal, B., Gopinath, G. R., Wu, G. R., Matthews, L., Lewis,

S., Birney, E., and Stein, L. (2005). Reactome: a knowledgebase of biological pathways. *Nucleic Acids Research*, 33(Database-Issue):428–432.

[41] Katz, D. (2014). Transitive credit as a means to address social and technological concerns stemming from citation and attribution of digital products. *Journal of Open Research Software*, 2(1).

[42] Kosten, J. (2016). A classification of the use of research indicators. *Scientometrics*, 108(1):457–464.

[43] Lawrence, B., Jones, C., Matthews, B., Pepler, S., and Callaghan, S. (2011). Citation and Peer Review of Data: Moving Towards Formal Data Publication. *International Journal of Digital Curation*, 6(2):4–37.

[44] Livshits, E., Bertossi, L. E., Kimelfeld, B., and Sebag, M. (2020). The shapley value of tuples in query answering. In Lutz, C. and Jung, J. C., editors, *23rd International Conference on Database Theory, ICDT 2020, March 30-April 2, 2020, Copenhagen, Denmark*, volume 155 of *LIPIcs*, pages 20:1–20:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[45] Martone, M. (2014). Joint declaration of data citation principles. *FORCE11. San Diego CA. Data Citation Synthesis Group.* `https://www.force11.org/datacitationprinciples`, online September 2020.

[46] Meho, L. I. and Yang, K. (2007). Impact of data sources on citation counts and rankings of LIS faculty: Web of science versus scopus and google scholar. *Journal of the american society for information science and technology*, 58(13):2105–2125.

[47] Meliou, A., Gatterbauer, W., Moore, K. F., and Suciu, D. (2010). The complexity of causality and responsibility for query answers and non-answers. *Proc. VLDB Endow.*, 4(1):34–45.

[48] Milo, T. (2019). Getting rid of data. *Journal of Data and Information Quality (JDIQ)*, 12(1):1–7.

[49] Nosek, B. A., Alter, G., Banks, G. C., Borsboom, D., Bowman, S. D., Breckler, S. J., Buck, S., Chambers, C. D., Chin, G., Christensen, G., Contestabile, M., Dafoe, A., Eich, E., Freese, J., Glennerster, R., Goroff, D., Green, D. P., Hesse, B., Humphreys, M., Ishiyama, J., Karlan, D., Kraut, A., Lupia, A., Mabry, P., Madon, T., Malhotra, N., Mayo-Wilson,

E., McNutt, M., Miguel, M., Paluck, E. L., Simonsohn, U., Soderberg, C., Spellman, B. A., Turitto, J., VandenBos, G., Vazire, S., Wagenmakers, E. J., Wilson, R., and Yarkoni, T. (2015). Promoting an open research culture. *Science*, 348(6242):1422–1425.

[50] Peters, I., Kraker, P., Lex, E., Gumpenberger, C., and Gorraiz, J. (2016). Research data explored: An extended analysis of citations and altmetrics. *Scientometrics*, 107(2):723–744.

[51] Pröll, S. and Rauber, A. (2013). Scalable data citation in dynamic, large databases: Model and reference implementation. In *Proceedings of the 2013 IEEE International Conference on Big Data, 6-9 October 2013, Santa Clara, CA, USA*, pages 307–312.

[52] Rauber, A., Ari, A., van Uytvanck, D., and Pröll, S. (2016). Identification of Reproducible Subsets for Data Citation, Sharing and Re-Use. *Bulletin of IEEE Technical Committee on Digital Libraries, Special Issue on Data Citation*, 12(1):6–15.

[53] Rauber, A., Asmi, A., van Uytvanck, D., and Proell, S. (2015). Data citation of evolving data: Recommendations of the working group on data citation (wgdc). *Result of the RDA Data Citation WG*, 20.

[54] Remil, Y., Bendimerad, A., Mathonat, R., Chaleat, P., and Kaytoue, M. (2021). " what makes my queries slow?": Subgroup discovery for sql workload analysis. *arXiv preprint arXiv:2108.03906*.

[55] Shapley, L. S. (1954). A value for n-person games. In Kuhn, H. W. and Tucker, A. W., editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton.

[56] Silvello, G. (2018). Theory and practice of data citation. *J. Assoc. Inf. Sci. Technol.*, 69(1):6–20.

[57] Simmhan, Y., Plale, B., and Gannon, D. (2005). A survey of data provenance in e-science. *SIGMOD Record*, 34(3):31–36.

[58] Spengler, S. (2012). Data Citation and Attribution: A Funder's Perspective. In of Sciences' Board on Research Data, N. A. and Information, editors, *Report from Developing Data Attribution and Citation Practices*

and Standards: An International Symposium and Workshop, pages 177–178. National Academies Press: Washington DC.

[59] Vogelsgesang, A., Haubenschild, M., Finis, J., Kemper, A., Leis, V., Mühlbauer, T., Neumann, T., and Then, M. (2018). Get real: How benchmarks fail to represent the real world. In *Proceedings of the Workshop on Testing Database Systems*, pages 1–6.

[60] Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J., da Silva Santos, L. B., Bourne, P. E., et al. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3.

[61] Wu, Y., Alawini, A., Davidson, S. B., and Silvello, G. (2018). Data citation: Giving credit where credit is due. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD*, pages 99–114.

[62] Zeng, T., Wu, L., Bratt, S., and Acuna, D. E. (2020). Assigning credit to scientific datasets using article citation networks. *Journal of Informetrics*, 14(2).

[63] Zou, C. and Peterson, J. B. (2016). Quantifying the scientific output of new researchers using the zp-index. *Scientometrics*, 106(3):901–916.

[64] Zwölf, C. M., Moreau, N., and Dubernet, M.-L. (2016). New Model for Datasets Citation and Extraction Reproducibility in VADMC. *Journal of Molecular Spectroscopy*, 327:122–137.