The version of record is available at: http://dx.doi.org/10.1561/1900000074

Trends in Explanations: Understanding and Debugging Data-driven Systems

Boris Glavic[1], Alexandra Meliou[2] and Sudeepa Roy[3]

[1] *Illinois Institute of Technology, Chicago; bglavic@iit.edu*
[2] *University of Massachusetts Amherst; ameli@cs.umass.edu*
[3] *Duke University; sudeepa@cs.duke.edu*

ABSTRACT

Humans reason about the world around them by seeking to understand why and how something occurs. The same principle extends to the technology that so many of human activities increasingly rely on. Issues of trust, transparency, and understandability are critical in promoting adoption and proper use of systems. However, with increasing complexity of the systems and technologies we use, it is hard or even impossible to comprehend their function and behavior, and justify surprising observations through manual investigation alone. Explanation support can ease humans' interactions with technology: explanations can help users understand a system's function, justify system results, and increase their trust in automated decisions.

Our goal in this article is to provide an overview of existing work in explanation support for data-driven processes, through a lens that identifies commonalities across varied problem settings and solutions. We suggest a classification of explainability requirements across three dimensions: the target of the explanation ("What"), the audience of the

Boris Glavic, Alexandra Meliou and Sudeepa Roy (2021), "Trends in Explanations: Understanding and Debugging Data-driven Systems", Foundations and Trends® in Databases: Vol. 11, No. 3, pp 226–318. DOI: 10.1561/1900000074.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

explanation ("Who"), and the purpose of the explanation ("Why"). We identify dominant themes across these dimensions and the high-level desiderata each implies, accompanied by several examples to motivate various problem settings. We discuss explainability solutions through the lens of the "How" dimension: How something is explained (the form of the explanation) and how explanations are derived (methodology). We conclude with a roadmap of possible research directions for the data management community within the field of explainability in data systems.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

1

Introduction

All the way from infancy to advanced scientific inquiry, humans pose and seek answers to "why" questions; simply, explanations are the means through which humans perceive and reason about the world. It is thus not surprising that as technology increasingly permeates all aspects of human activity, the need for explanations arises in algorithmic and data-driven systems.

Data-driven technologies are at the core of many everyday interactions, such as social network connections, news personalization, and product recommendations, but also drive critical decision-making, such as autonomous vehicle actions, diagnosis and treatment of diseases, and even criminal sentencing. Explanation support can ease our interactions with these systems: explanations help users justify and consequently trust the function of these systems, they help developers diagnose problems and improve the systems, and they increase stakeholders' confidence in system decisions. Several research domains have recognized these emerging requirements, and work on supporting explanations in computing systems has flourished.

In this article, we present a research roadmap for the data management community, examining existing efforts in explanation support,

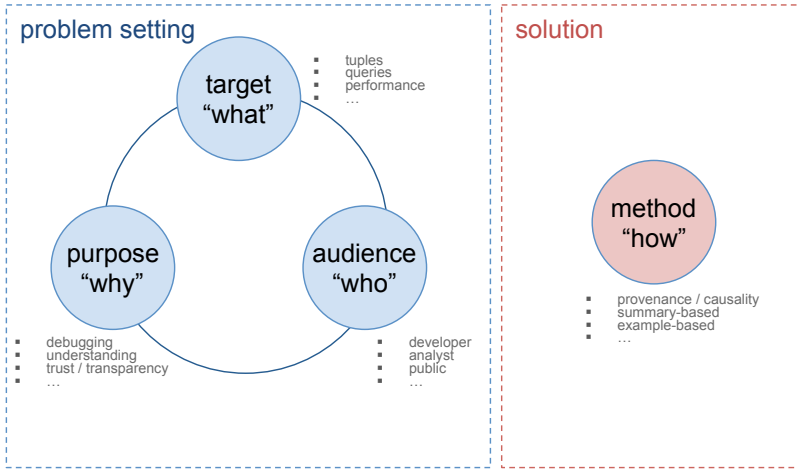


Figure 1.1: High-level classification of the explanation research space. We examine different problem settings pertaining to explanations on three axes: target ("what"), purpose ("why"), and audience ("who"). We review existing solutions ("how") that appeared in the literature, including different objectives they targeted.

and open questions and meaningful directions in the evolving map of understandability requirements in computing and data-driven systems. We examine this research space from three perspectives that specify the particular requirements of an explanation problem setting: the explanation's target ("*what* to explain"), the explanation's purpose ("*why* do we seek to explain"), and the explanation's audience ("*who* receives the explanations"). We then proceed to discuss, at a high level, methods proposed in prior work for deriving explanations within particular problem settings ("*how* to explain"). We briefly describe below these four aspects (what, why, who, and how) of explanation research, with an overview in Figure 1.1:

"What." The first perspective under which we examine explanation support is the explanation's target: what it is we are trying to explain. Prior work has focused on a variety of aspects, including surprising results, outliers, queries, differences in results, repairs, and performance. Our goal is to categorize existing work within these targets, note gaps interesting to explore, and identify possible explanation-worthy targets.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

"Why." The purpose of the explanation often determines important desiderata for the derived explanations. For example, explanations aimed at debugging problems need to be comprehensive and detailed, while explanations aimed at illustrating functionality should be simpler and high level. Explanations may also be used as a tool to promote trust in a system's operation, in which case, they may target aspects of a system's functionality, rather than particular data or results.

"Who." The audience of an explanation is often tied to the explanation's purpose (i.e., the "why" and "who" are often linked). In addition to diverse purposes, explanations can serve diverse audiences. For example, explanation support may be an important tool for technical users of data systems, who typically aim to improve the systems (e.g., developers), or gain deep understanding of data processing steps (e.g., analysts). But explanations can also target a broader audience, such as data enthusiasts or the general public, who generally expect illustrative justifications for their observations.

"How." The research community has followed a variety of methods in deriving explanations, and the corresponding explanation products can be of different types. A significant portion of explanation-related work in the data management community is provenance-based, and seeks explanations in the lineage of query results. In other veins, researchers have sought query-based explanations, or explanations based on summaries or examples. The explanation objectives adopted in prior work can vary as well. Different explanation frameworks seek to optimize different metrics, including measures of understandability, flexibility, or generalizability. Our goal is to provide a categorization of the existing methodologies, note possible new approaches and combinations, group metrics used in the literature under high-level objectives, and suggest additional useful ones for the community to consider.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

1.1 Scope of this article

The need for explanations is universal. Thus, it is not surprising that explanations have been studied in many fields of Computer Science. Our intention is not to cover the entire body of this work, as this would be impossible to do in sufficient depth within this article. Rather, we primarily focus on relevant work from the database community and discuss work from other fields when this work is relevant to data management or introduces techniques that we believe could prove useful in a data management context in the future.

In particular, explainable AI (commonly known as XAI) is an important emerging field with research contributions from the database community, and we will provide a brief review of this work for context. However, this is a vast and active research area. Providing a detailed overview of explanations in the context of AI is out of scope for this article. Guidotti *et al.* (2019) provide a good overview of methods for explaining the behavior of blackbox models, e.g., by locally approximating them through simpler, but more interpretable methods. At the technical level, many of the approaches discussed in this article use methods for summarizing information and presenting it in a form that is fit for human consumption. As such, they use or are closely related to unsupervised methods developed by the data mining community such as association rule mining (Agrawal and Srikant, 1994) and clustering (Han and Kamber, 2001). On the other hand, there is an extensive literature in the computer systems community on monitoring, finding, and explaining big data systems' performance and errors, such as root cause analysis (e.g., Yoon *et al.* (2016), Ousterhout *et al.* (2015), and Roy *et al.* (2015b)). While these methods are interesting in their own right, we will only discuss them to the degree necessary for understanding their use for generating explanations with a brief overview, and refer to the relevant literature if the reader is interested in these topics.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

2

Explanation Needs: Who, Why, and What

The contributions of data management research to explanation support are broad and varied, ranging from provenance-based methods for tracing changes in data, to summarization algorithms, and debugging systems. In this article, we use the general term *explanation framework* to refer to all kinds of explanation support, regardless of the nature of their solution or the problem setting. Given the diverse nature of existing approaches, and the broad applicability of explanations, we posit that a unified explanation framework that can handle all understandability requirements in a data-driven system is unlikely. While enhancing understandability is a common overall objective, explanation frameworks adopt more targeted goals (e.g., with respect to explanation form or size), driven by requirements of particular problem settings. In this chapter, we explore aspects of problem settings that drive these goals, and which, in turn, have impact on possible solutions.

We identify three general axes of problem specifications that impact the goals of explanation frameworks: *Who*, *Why*, and *What*. "Who" specifies the audience of an explanation: who is seeking an explanation. "Why" specifies the purpose of an explanation: why is the explanation needed (e.g., to debug a problem, or to illustrate a functionality). "What"

The version of record is available at: http://dx.doi.org/10.1561/1900000074

specifies the target of an explanation: what is it that we are trying to explain (e.g., a surprising observation, or a system malfunction).

In this chapter, we introduce the Who-Why-What classification of explanation problems. Notably, these axes can help identify general desiderata for explanation settings, without being tied to particular applications, domains, or datasets. For example, we note that explanations targeted at technical users should be more detailed than those targeted at a general audience. We structure this chapter from the perspective of the problem setting, anchored across the axes of *Who*, *Why*, and *What*, and the requirements or desiderata that each entails. The desiderata of a problem setting are generally decoupled from particular solutions, though they can often make the problem more amenable to certain methodologies for deriving explanations. We will keep the discussion at a high-level of abstraction in this chapter using examples, and will not discuss methodologies of explanation frameworks: (i) what objects constitute an explanation (e.g., query input tuples); (ii) how are these objects presented to the user (e.g., using predicates to compactly describe part of the input data that is responsible for an erroneous query result); (iii) what is the search space for objects take make up an explanation (e.g., do we search for explanations in the provenance or also in other related data); (iv) how to measure the contribution of a candidate explanation for justifying the observed effect the users wants us to explain (e.g., to what degree does a query result change if the remove the input data encoded by a candidate explanation). These aspects of the "How" of explanation frameworks will be discussed in detail in the following chapter.

We note that the who-why-what categorization is not fully discriminative. For example, a particular methodology can facilitate explanations for multiple purposes (e.g., result interpretation and debugging). It is also possible that the same problem setting can be addressed by more than one methodological approach (e.g., a provenance-based and a summarization-based solution). The primary goals of this categorization are to provide guidelines on appropriate explanation objectives, and to highlight relationships across different approaches.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

2.1 "Who" needs explanations

In this section, we discuss the audience of an explanation: who is the seeker and recipient of an explanation. In most cases, since explanations are typically tied to understandability expectations, the audience is a human user. However, conceivably, explanations can be provided to a system component in the automation of a task like error tracing or debugging. Below, we list three general categories of explanation audiences, which capture most problem settings in the existing literature, though others are possible.

- **Developers.** Data-driven systems, like all systems, can exhibit bugs and suboptimal behavior. As data is a big component of these systems' function, problems with their operation may become apparent through the use and processing of data, and certain aspects of the data may be the triggers of such incorrect or suboptimal behaviors. The involvement of data as a core system component further complicates debugging and optimizations. Explanation systems that automatically derive causes of evident buggy behavior can assist developers in the task of implementing corrections and improvements. System developers have technical expertise and access to system components, but they may not have good understanding of the data domain. Suitable explanations can involve system components and operations that are deemed responsible for observed behaviors, but also data elements and patterns that can identify cases not well-handled in the code. Such explanations will tend to be low-level, detailed, and comprehensive.

- **Data analysts.** The task of data analysis involves the extraction and interpretation of interesting observations from data. Therefore, explanations are an inherent expectation for a data analyst, who often resorts to deriving them manually, typically through further and deeper analysis. Analysts are typically experts in the data domain and have breadth of knowledge in the use of analytics tools, but they may lack technical expertise of the data handling systems and the underlying analytics technologies. As a result, the outcomes of data analysis can be hard to interpret, and mistakes in

The version of record is available at: http://dx.doi.org/10.1561/1900000074

the use and requirements of the technologies can lead to erroneous observations and incorrect insights. To support data analysis, data systems need to provide tools dedicated to deriving explanations in analytics tasks, with analysts as the target audience. Relevant explanations should be data-focused, high-level, and illustrative of the analytics technologies, thus targeting the expertise of the target audience. Low-level, more fine-grained explanations that rely on internal components and processes of the analytics pipeline may not be appropriate, as the analysts have little knowledge of and cannot intervene in these elements.

- **Data enthusiasts and the general public.** The democratization of computational resources and analytics tools, coupled with the continuous increase in the access and availability of interesting datasets has lead to the involvement of a more general, non-expert group of users in the perusal and basic analysis of data. These users may have task-specific requirements (e.g., computational journalists), or merely general interest and curiosity. Such *data enthusiasts* lack deep expertise in the data domain, and typically do not have technical knowledge of the tools and underlying systems. Explanations can help them reason about the observations they make, better assess their validity, and judge their insightfulness. Since these users are non-technical, explanations need to be high level and illustrative of the systems' function. As data-driven systems permeate so many aspects of human activity, the general public has invested interest in understanding these systems' operation. Explanations can help enhance the public's trust by providing high-level justifications of data-driven decisions.

2.2 "Why" we want to explain

Explanations can serve a variety of functions, ranging from understandability requirements to guiding debugging efforts. The purpose of the explanation, often correlates with the explanation's audience, and in itself also drives desiderata with respect to the explanation format and success metrics. Here, we review several common themes in explana-

The version of record is available at: http://dx.doi.org/10.1561/1900000074

N(ewsFeeds)

nid	*story*	*tag*
1	... for the 2023 AFC Asian Cup Xi'an ...	Sports
2	... economic downturn affected sensitive ...	Business
3	... with sequences shot in Xi'an ...	Movies
4	... when President Biden meets former ally ...	Biden
5	... Xi'an slow down hiring ...	Business
6	... Oscars 2021: Academy's 'best' choice ...	Movies
7	... launches cloud lab in Xi'an ...	Technology
8	... struggles to corral votes for health bill ...	Health
9	... SIGMOD conference this year in Xi'an ...	DB_conf
10	... Copenhagen host to VLDB 2021 ...	DB_conf
11	... SIGMOD in Xi'an promises to be ...	DB_conf
12	... contact sports can resume Monday ...	Sports

R(outing)

tag
Biden
DB_conf
Sports
Technology

Query answer:
P(ersonalized alerts)

cities
Paris
Xi'an
Athens

Figure 2.1: Example of a personalized alert-feed (P) as a result of a query filtering all news (N) based on a carefully constructed routing table (R).

tion settings from the perspective of the purpose of the explanation: "why" we want to explain. These themes include understanding results (Section 2.2.1), debugging data and systems (Section 2.2.2), debugging performance (Section 2.2.3), and enabling responsible data analysis through fairness, trust, and reproducibility of the process (Section 2.2.4). We discuss each theme through examples, often drawn and adapted from the existing literature.

2.2.1 Understanding and interpreting results

Enhancing understandability is a core goal of explanation research. Prior work has focused in particular on surprising observations in data processing results. Explanations can provide justification and evidence to establish the validity of an observation, or assist with the interpretation of a finding. We discuss a few examples from published work that deal with a variety of understandability and interpretation requirements.

Example 2.1 (Meliou *et al.*, 2010)**.** A major travel agency monitors a large number of news feeds in order to identify trends, opportunities, or alerts about various cities. Central to this activity is a carefully personalized routing table and query, which filters what information to forward to each specialized travel agent by carefully chosen keywords. Figure 2.1 shows the routing table for one user R, as well as a sample news feed. The query issuing alerts to this user is:

The version of record is available at: http://dx.doi.org/10.1561/1900000074


```
SELECT C.name
FROM NewsFeeds N, Routing R, City C
WHERE C.name substring N.story and N.tag = R.tag
GROUP BY C.name
HAVING count(*) > 20
```

The result is a list of cities that are drawn to the attention of this particular agent, shown in Figure 2.1. As popular destinations, Paris and Athens are predictable answers. But this agent is surprised to see Xi'an in the results, and would like to understand what news or keywords lead it to appear on her watch list, so that she can better direct her promotion efforts.

In this example, the user seeks to understand the presence of particular data items in the result of a query. Better understanding of the results would lead to more informed actions on the part of the agent. The results do not need to be investigated because they are assumed to be incorrect; rather, decision making may rely on underlying parameters that drive this result and should be more closely investigated. Given these expectations, it would be likely that explanations should involve parameters of the query or the data that cause the relevant data items to appear in the result.

The same understandability requirements extend to the absence of expected results as the following example shows.

Example 2.2 (Lee *et al.*, 2020)**.** Figure 2.2 shows a sample of a real-world dataset recording Airbnb (bed & breakfast) listings and their availability. Each listing has an id, name, property type (`Ptype`), room type (`Rtype`), neighborhood (`Neighbor`), and neighborhood group (`NGroup`). Neighborhood groups are larger areas that include multiple neighborhoods. `Availability` stores the ids of listings with available dates and a price for each date. Bob, an analyst at Airbnb, investigates a customer complaint about the lack of availability of shared rooms on 2016-11-09 in Queen Anne (`NGroup` = `queen anne`). He uses the query shown below to determine which listings (names and room types) are available on that date in Queen Anne.

```
SELECT Name, RType
FROM Listing L, Availability A
```

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Listing

Id	Name	Ptype	Rtype	NGroup	Neighbor
8403	central place	apt	shared	queen anne	east
9211	plum	apt	entire	ballard	adams
2445	cozy homebase	house	private	queen anne	west
8575	near SpaceNeedle	apt	shared	queen anne	lower
4947	seattle couch	condo	shared	downtown	first hill
2332	modern view	house	entire	queen anne	west

Availability

Id	Date	Price
9211	2016-11-09	130
2445	2016-11-09	45
2332	2016-11-09	350
4947	2016-11-10	40

Query Result

Name	Rtype
cozy homebase	private
modern view	entire

Figure 2.2: Explaining missing answers: why are there no shared rooms available for rent on Airbnb in the Queen Anne neighborhood on November 9th in 2016?

```
WHERE   NGroup = 'queen anne'
        AND A.Date = '2016-11-09'
        AND L.Id = A.Id
```

The query result confirms the customer's complaint, since none of the available listings are shared rooms. Bob now needs to investigate what led to this missing result. An explanation framework for missing answers can provide Bob with important information that helps him to understand the shortage of shared rooms.

The two previous examples differ somewhat in the explanation target ("What"), as the former seeks to explain the presence and the latter the absence of particular results. However, the explanation purpose ("Why") is for both to provide a better understanding of the observed outcome of a data operation (in this case a query). Both these examples focused on queries without an aggregate output. In data analysis, however, aggregate outputs (involving count, sum, average, max, etc.) appear frequently, aiming to analyze *trends* over a period of time (e.g., to compare total sales over years), or communicate summary statistics

The version of record is available at: http://dx.doi.org/10.1561/1900000074

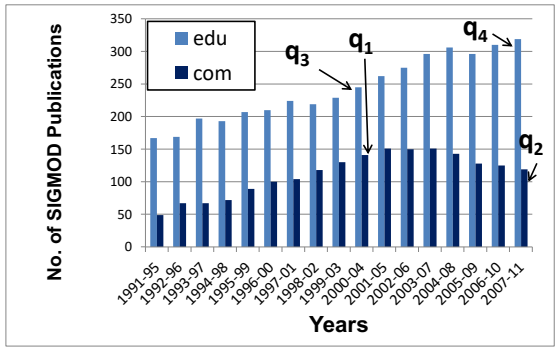


Figure 2.3: Number of SIGMOD publications in a five years windows, broken down into papers from industry ('com') and academia ('edu'). While both increase until 2000-2007, afterward the number of papers from academia continue to increase while that from industry decreases.

(e.g., to compare total sales in different locations). We discuss an example on explanations for aggregated outputs next.

Example 2.3 (Roy and Suciu, 2014)**.** Figure 2.3 shows the number of publications in SIGMOD during a moving five years window (up to 2011) co-authored by researchers from industry and academia. The graph was generated by running a SQL query over the DBLP publication dataset integrated with an affiliation table (affiliation information was available only for a subset of authors, so the graph does not include all papers). SIGMOD accepts more papers over the years, therefore the increasing trend from academia may be justifiable. However, somewhat surprisingly, the papers from industry have a peak in early 2000. With the recent discourse in many conferences about their impact, relevance in the era of big data and deep neural networks, and the path to move forward, this might be an interesting observation that an analyst may wish to investigate further. Explanation frameworks could assist in interpreting these results, saving in efforts of manual investigation.

The explanation target ("What") differs again in this example, but the purpose remains close to the previous two: understand and interpret an observation in the query output. Here, the observation is a trend in the result of an aggregate query, and the reason it provokes the need for interpretation is its difference from another observed trend.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

In a similar vein, the following example also identifies an explanation-worthy target in differences, but this time this is a difference in results of distinct queries over separate datasets. When two datasets represent the same elements in the physical world, it is expected that analyzing or querying either should produce the same result. When that is not the case, the differences need to be understood and interpreted before an analyst is able to derive conclusions. The understandability and interpretation goals are again the same, despite the different setting, and explanations may need to highlight aspects of the data or elements of the query that contribute to the observed deviations.

Example 2.4 (Wang and Meliou, 2019)**.** Two publicly-available academic datasets, the UMass-Amherst dataset on undergraduate programs[1], and the National Center for Education Statistics (NCES) dataset[2], are both from reputable sources and contain high-quality information. Nevertheless, querying both datasets for the number of undergraduate degree programs at UMass Amherst yields vastly different answers.

	UMass-Amherst data	**NCES data**
Schema:	Major(Major, Degree, School)	School(ID, Univ_name, City, Url) Stats(ID, Program, bach_degr)
Query:	Q_1 : `SELECT COUNT(Major) FROM Major;`	Q_2 :`SELECT SUM(bach_degr) FROM School, Stats WHERE Name = 'UMass-Amherst' AND School.ID=Stats.ID;`
Answer:	113	90

Existing explanation solutions can only be applied with respect to one of these datasets at a time, by asking questions such as "Why is the result of Q_1 (respectively, Q_2) high (respectively, low)?" But these would not provide meaningful explanations in this case, as each tuple contributes the same to the aggregate of Q_1, and prioritizing tuples with low `bach_degr` in the provenance of Q_2 would be arbitrary, not grounded on the actual differences with Q_1.

We will discuss approaches from the literature that explain such questions on query results in Chapter 3.

[1] https://www.umass.edu/gateway/academics/undergraduate

[2] https://nces.ed.gov: An open dataset presented. We simplified its schema for this example.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Student S

name	major	
Mary	CS	t_1
John	ECON	t_2
Jesse	CS	t_3

Registration R

name	course	dept	grade	
Mary	216	CS	100	t_4
Mary	230	CS	75	t_5
Mary	208D	ECON	95	t_6
John	316	CS	90	t_7
John	208D	ECON	88	t_8
Jesse	216	CS	95	t_9
Jesse	316	CS	90	t_{10}
Jesse	330	CS	85	t_{11}

Result of Q_1

name	major	
John	ECON	r_1

Result of Q_2

name	major	
Mary	CS	r_2
John	ECON	r_3
Jesse	CS	r_4

Figure 2.4: Toy database instance for Example 2.5, and results of Q_1 and Q_2.

2.2.2 Debugging data and systems

The results of an analysis or a data transformation often expose errors in the data, the operations, the system that executed the transformation, or the environment the task was executed in. Similar to using explanations to establish the validity of and justify surprising results, explanations can also serve as a tool to aid debugging efforts or other improvements in a system's function or an analytical process. In contrast to the examples in the previous section, where explanations were used to enhance understanding and interpretation of results, the distinction here is the explicit assumption that particular data items, values, queries, or other system components are incorrect, or otherwise indicate errors and problems in other parts of the system. We provide a set of examples from existing literature that demonstrate such issues in relational and non-relational systems.

Example 2.5 (Miao *et al.*, 2019a)**.** Consider two relations storing information about students `Student`(`name`, `major`) and course registrations `Registration`(`name`, `course`, `dept`, `grade`). A toy test instance is given in Figure 2.4. Suppose an instructor in a database course asked the students in that course to write a SQL query: *'find students who registered for* exactly one *CS course'*. Below we show the correct query Q_1, and an incorrect query Q_2 that a student submitted, which actually finds students who registered for *one or more* CS courses.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Q_1:

```
SELECT s.name,s.major
FROM Student s, Registration r
WHERE s.name = r.name AND r.dept = 'CS'
EXCEPT
SELECT s.name,s.major
FROM Student s, Registration r1, Registration r2
WHERE s.name = r1.name
      AND s.name = r2.name
      AND r1.course <> r2.course
      AND r1.dept = 'CS'
      AND r2.dept = 'CS'
```

Q_2:

```
SELECT s.name,s.major
FROM Student s, Registration r
WHERE s.name = r.name AND r.dept = 'CS'
```

Figure 2.4 shows the results of Q_1 and Q_2. Here the goal of the instructor or the teaching assistants is to help the student understand the mistake in their query using a meaningful and easy-to-follow explanation.

In this example, there is explicit knowledge that a query is erroneous. The error is known, and the point is not to detect the error, but rather to explain it. An explanation in this setting should illustrate to the user (in this case the student) why the query they submitted is erroneous. This problem can be approached from different angles —we will discuss one in the "How" chapter—but the key purpose of the explanation is to help the user to understand the error.

The following example shares this theme (i.e., the explanation provides a high level illustration of the error), but here we are tracing the error to errors in a data pipeline that produced the erroneous data.

Example 2.6 (Wang *et al.*, 2015a)**.** Large-scale information extraction pipelines process unstructured or semi-structured data, such as the web tables of Figure 2.5, to extract structured information typically in the form of knowledge triples (bottom of Figure 2.5). These pipelines are often imperfect, and can introduce errors in the extracted data. In this example, the extractors are assigning a default date value whenever

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Musicians – Table 1

Name	Date of Birth	Date of Death
P. Fontaine	c.1380	c.1450
J. Vide	unknown	1433

Composers – Table 2

Name	Date of Birth	Date of Death
G. Legrant	fl.1405	N/A
H. Lantins	fl.c.1420	unknown

Extracted triples | Triple properties

ID	knowledge triple	*source* URL	tableID	*subject* type	instance	*predicate* type	instance	*object* type	instance
t_1	{P. Fontaine, Profession, Musician}	wiki	tbl #1	People	P. Fontaine	Bio	Profession	Profession	Musician
t_2	{P. Fontaine, DoB, c.1380}	wiki	tbl #1	People	P. Fontaine	Bio	DoB	Date	c.1380
t_3	{P. Fontaine, DoD,c.1450}	wiki	tbl #1	People	P. Fontaine	Bio	DoD	Date	c.1450
t_4	{J. Vide, Profession, Musician}	wiki	tbl #1	People	J. Vide	Bio	Profession	Profession	Musician
t_5	**{J. Vide, DoB, 01/01/1900}**	**wiki**	**tbl #1**	**People**	**J. Vide**	**Bio**	**DoB**	**Date**	**01/01/1900**
t_6	{J. Vide, DoD, 1433}	wiki	tbl #1	People	J. Vide	Bio	DoD	Date	1433
t_7	{G. Legrant, Profession, Composer}	wiki	tbl #2	People	G. Legrant	Bio	Profession	Profession	Composer
t_8	{G. Legrant, DoB, fl.1405}	wiki	tbl #2	People	G. Legrant	Bio	DoB	Date	fl.1405
t_9	**{G. Legrant, DoD, 01/01/1900}**	**wiki**	**tbl #2**	**People**	**G. Legrant**	**Bio**	**DoD**	**Date**	**01/01/1900**
t_{10}	{H. Lantins, Profession, Composer}	wiki	tbl #2	People	H. Lantins	Bio	Profession	Profession	Composer
t_{11}	{H. Lantins, DoB, fl.c.1420}	wiki	tbl #2	People	H. Lantins	Bio	DoB	Date	fl.c.1420
t_{12}	**{H. Lantins, DoD, 01/01/1900}**	**wiki**	**tbl #2**	**People**	**H. Lantins**	**Bio**	**DoD**	**Date**	**01/01/1900**

Figure 2.5: An information extraction pipeline processes the web tables (top) and derives 12 knowledge triples (bottom). Each triple has four property dimensions with different granularity levels. The extractors assign a default value to dates that are unknown ("01/01/1900"), leading to three incorrect triples (t_5, t_9, and t_{12}).

date information is missing. Manual investigation of these problems is impractical, as the errors can be large scale; e.g., in the case of Knowledge Vault (Dong *et al.*, 2014) errors can span billions of triples. Automated explanation frameworks are necessary to diagnose systemic problems in large scale pipelines, which can be extremely complex and may include obscure and black box components.

The errors in this example are identified in the output of the extraction pipeline, as noted in Figure 2.5. The purpose of explaining the errors in this case is to better understand how they occurred in the first place. The errors in this example are systemic, inherent to the process that generates the result triples and thus will keep occurring unless a repair is applied to the part of the pipeline that causes the problem. Simply, purging the errors from the output will not solve the issue, as the pipeline will keep generating flawed data. Therefore, an explanation in this setting should offer clarity about the origin of the problem, potentially serving as a debugging mechanism.

The following example continues the debugging theme: errors are noted in the output of several processing steps, and an explanation can offer clarity to how they were introduced. A distinction here is that the

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Taxes: D_0

ID	income	owed	pay
t_1	$9500	$950	$8550
t_2	$90000	$22500	$67500
t_3	$86000	$21500	$64500
t_4	$86500	$21625	$64875

Query log: $\mathcal{Q}$

q_1:
```
UPDATE Taxes SET owed=income*0.3
  WHERE income>=85700
```
q_2:
```
INSERT INTO Taxes
  VALUES (85800, 21450, 64350)
```
q_3:
```
UPDATE Taxes SET pay=income-owed
```

Taxes: D_4

ID	income	owed	pay
t_1	$9500	$950	$8550
t_2	$90000	$27000	$63000
t_3	$86000	$25800	$60200
t_4	$86500	$25950	$60550
t_5	$85800	$21450	$64350

Figure 2.6: A recent change in tax rate brackets calls for a tax rate of 30% for those with income above $87500. The accounting department issues query q_1 to implement the new policy, but the predicate of the `WHERE` clause condition transposed two digits of the income value. As a result, the owed amount of t_3 and t_4 were calculated incorrectly. This mistake is obscured by q_2, which inserted a tuple with correct income and owed amount, and was later further propagated to additional fields by query q_3, which calculates the pay check amount based on the corresponding income and (incorrect) owed values.

problem setting is relational. In contrast to the previous example that involved complex steps in the pipeline and, for all practical purposes, black box components, the relational domain allows for more flexibility in examining and reasoning about the inner workings of the data processing steps. As such, the methodologies and solutions ("How") will frequently differ, even though the explanation's purpose is the same.

Example 2.7 (Wang *et al.*, 2017)**.** An accounting firm implements an adjustment to tax brackets on their customer dataset (Figure 2.6). The adjustment sets the tax rate to 30% for income levels above $87,500, and is implemented by query q_1. A digit transposition mistake in the query, results in an incorrect owed amount for tuples t_3 and t_4. Query q_2, which inserts a tuple with slightly higher income than t_3 and t_4 and the correct information, obscures this mistake. This mistake is further propagated by query q_3, which calculates the paycheck amount based on the corresponding *income* and amount *owed*.

Some of the mistakes in the database may be individually reported to the firm by the customers. But, fixing these errors on an individual reporting basis further obscures the problem, and leaves several erroneous values unaddressed. Instead, the accounting firm should first understand how the errors were introduced, and such an explanation would help determine and implement an appropriate repair strategy.

In our final example in this section, errors are again noted in the output of data processing, but in this case they do not occur consistently,

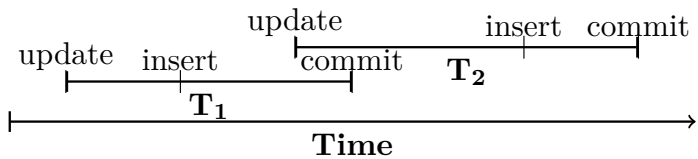
The version of record is available at: http://dx.doi.org/10.1561/1900000074

Alice's Withdrawal Transaction

```
UPDATE account SET bal = bal - :amount
WHERE cust = :name AND typ = :type;

INSERT INTO overdraft (
  SELECT cust, a1.bal + a2.bal
  FROM account a1, account a2
  WHERE a1.cust = :name AND a1.cust = a2.cust
        AND a1.typ != a2.typ AND a1.bal + a2.bal < 0);
```

Execution Order of Transactions T_1 and T_2



Bind Parameters for Transactions T_1 and T_2

Transaction	:name	:amount	:type
T_1	Bob	70	Checking
T_2	Bob	40	Savings

(a) Database before execution of T_1 and T_2

account

cust	typ	bal
Bob	Checking	50
Bob	Savings	30

overdraft

cust	bal

(b) Database after execution of T_1

account

cust	typ	bal
Bob	Checking	-20
Bob	Savings	30

overdraft

cust	bal

(c) Database after execution of T_2

account

cust	typ	bal
Bob	Checking	-20
Bob	Savings	-10

overdraft

cust	bal

Figure 2.7: Erroneous transaction execution. Explanations for how transactions interacted in the history are needed to understand the cause of the error (interleaving of transaction execution under non-serializable transaction isolation levels).

The version of record is available at: http://dx.doi.org/10.1561/1900000074

as they are linked to concurrency issues. While the purpose of explanations remain the same—understanding the origin of the problem and assisting in debugging—the particular problem setting would require a different approach compared the the previous examples.

Example 2.8. Alice is a developer at a bank that runs a database (e.g., Oracle) using the *snapshot isolation (SI)* concurrency control protocol (Berenson *et al.*, 1995) which does not guarantee serializable schedules. She is tasked with writing a transaction for withdrawing money from a customer's checking or savings account (a table `account(cust,typ,bal)`). If after the withdrawal the total balance of the checking and savings account for the customer are below 0, then an overdraft record should be inserted into a table `overdraft(cust,bal)`. Alice implements the transaction shown in Figure 2.7 that runs an update followed by an insert using a query that detects overdrafts. After some tests that are uneventful, Alice's solution is deployed. However, it turns out that Alice's transaction does not always report overdrafts correctly. Assume that transactions T_1 and T_2 as shown in Figure 2.7 have been executed concurrently with T_2 committing last. Fig. 2.7 shows the database state before and after execution of T_1 and T_2.

As shown in Figure 2.7 (c), these transactions cause an overdraft for Bob that is evident in the database state after T_2's commit (since $-20 + (-10) < 0$). However, neither T_1 nor T_2 have reported this overdraft. The cause of this problem is that SI does not guarantee serializability. In fact, it can lead to a concurrency anomaly called *write-skew* (Berenson *et al.*, 1995). Under SI, a transaction T runs over a private snapshot of the database that contains changes made by transactions that committed before T started. Thus, T_1 and T_2 do not see each other's changes. Both transactions compute the total balance using an outdated balance for the other account. For instance, T_2 sees the previous balance of \$50 for Bob's checking account and the condition of the overdraft check evaluates to $50 + (-10) = 40 \not< 0$.

Such errors are hard to debug, because they only occur if the execution of transactions is interleaved in a certain way. Alice's application may run for days without causing any issues. An explanation framework that can identify the root cause for the absence of the expected

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Team	City	Country	Year	League	Place
F.C Barcelona	Barcelona	Spain	2019	La Liga	1
Atletico Madrid	Madrid	Spain	2019	La Liga	2
Real Madrid	Madrid	Spain	2019	La Liga	2
F.C Barcelona	Barcelona	Catalonia	2018	La Liga	1
Atletico Madrid	Capital	España	2018	La Liga	2
Real Madrid	Madrid	Spain	2018	La Liga	3

Team	City	Country	Year	League	Place
F.C Barcelona	Barcelona	Spain	2019	La Liga	1
Atletico Madrid	Madrid	Spain	2019	La Liga	2
Real Madrid	Madrid	Spain	2019	La Liga	3
F.C Barcelona	Barcelona	Spain	2018	La Liga	1
Atletico Madrid	Madrid	Spain	2018	La Liga	2
Real Madrid	Madrid	Spain	2018	La Liga	3

Figure 2.8: Dirty table (left) and resulting clean table (right), after the application of a black box repair system.

result based on a transactional history can aid Alice in debugging her implementation.

The use of explanations in debugging data and systems can extend to different data transformation processes. Data routinely undergoes various stages of processing, such as extraction, cleaning, integration, and sampling, before a dataset is ready for use. These processes may be separate from the main data analysis task, but due to the changes they incur to the data, analysts often need to investigate the validity of these changes. We present an example from prior literature that focuses on constraint-based data repair, but the motivation for explanations generalizes to other processes and settings.

Example 2.9 (Deutch *et al.* (2020))**.** Figure 2.8 offers an example of a data repair system that takes dirty data as input, and produces a transformed, clean table as a result. In this repair transformation, the system decides which constraints to apply and determines which repairs are preferable. Systems like this cannot guarantee the correctness of the repair, so their decisions often need to be investigated, justified, and, sometimes, overruled. Explanation frameworks can help support analytics in this preprocessing stage, helping enhance the trust in the data used.

2.2.3 Performance analysis

Data analysis frequently involves the use of complex pipelines, often deployed in parallel fashion over on-demand computing resources. Analysts can often struggle with the complexities of the setup, and may face challenges with various performance aspects of these systems. Explanations can elucidate the reasons for unexpected performance behaviors,

The version of record is available at: http://dx.doi.org/10.1561/1900000074

potentially indicating ways to improve the analytics pipelines and their deployment. This can still be seen as a debugging setting, where the purpose of the explanation is to improve an issue in the system; here the issue is poor performance instead of erroneous results. However, we see this category as distinct, in that understanding the reasons for performance deviations may impact decisions related to the analysis setup, calibrating tradeoffs between cost and resources, etc., rather than repairing an inherent bug in the process.

We present two examples where system performance is perceived to deviate compared to another "normal" execution. In both cases, the purpose of the explanation is to identify the causes for the degradation. Typically, explanations for system performance need to involve configuration, interference, shared resources, and other general systems' issues. Depending on the audience of the explanation, e.g., and administrator who can modify the system, or an analyst who may be actively monitoring the progress of an analysis, the explanation requirements and goals may ultimately differ.

Example 2.10 (Kalmegh *et al.*, 2019)**.** Consider the dataflow-DAG of a data analytical TPCDS Query-3 shown in Figure 2.9. Suppose an admin notices a slowdown for Query-3 in an execution, compared to a previous execution, and wants to analyze the reasons of its slowdown. As a first step of troubleshooting, she wants to identify whether Query-3 was a victim of concurrency-caused contention or not (*i.e.*, whether the reasons were systemic or due to configuration). If she finds that the slowdown may be caused by concurrent execution with other queries in the shared cluster, she may have to find which of the concurrent queries are more responsible for the slowdown of Query-3 so that she can configure the scheduler accordingly.

Example 2.11 (Zhang *et al.*, 2017)**.** Figure 2.10 shows the data queuing size of a monitored Hadoop job, against the time elapsed since the beginning of the job. On the left, the job progress is normal: the intermediate results output by the mappers start to queue at the beginning and reach a peak after a short period of time. This is because a number of mappers have completed in this period while the reducers

The version of record is available at: http://dx.doi.org/10.1561/1900000074


```
SELECT i.i_brand_id, sum(ss_ext_sales_price) sum_agg
FROM   date_dim dt, store_sales ss, item i
WHERE  dt.d_date_sk = ss.ss_sold_date_sk
AND ss.ss_item_sk = i.i_item_sk AND i.i_manufact_id = 128
GROUP  BY i.i_brand_id;
```

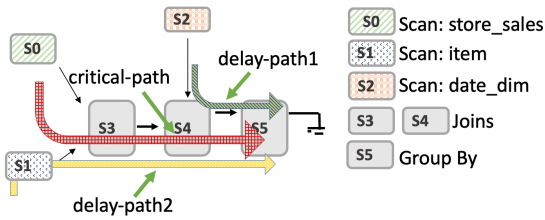


Figure 2.9: Execution DAG of TPCDS Query-3 showing the computation that each stage performs. Stages S0, S1 and S2 are IO-intensive as they scan input data; S3, S4 are both network and IO-intensive owing to a shuffle operation required for a Join; S5 is more CPU bound due to the aggregate operation. A dataflow DAG may consist of many chains of stages running in parallel. A query can slowdown due to delay in one or more of its component stages. Some delays propagate to the end while others get mitigated by faster later stages. Here two such paths, *delay-path1* and *delay-path2*, get mitigated later, but the *critical-path*, the sequence of stages with maximum overall runtime, contributes to the final slowdown.

have not been scheduled to consume the map output. Afterwards, the queued data size decreases and then stabilizes for a long period of time, meaning that the mappers and reducers are producing and consuming data at constant rates, until the queued data reduces to zero at the end of the job. During a different execution of the same job, a Hadoop user observes the plot on the right: there is a long initial period where the data queuing size increases gradually but continually, and this phase causes the job completion time to be delayed by more than 500 seconds. The user needs this performance difference explained, to potentially take steps to improve the job's execution time. In this case, the performance degradation is due to high memory usage of other programs in the Hadoop cluster. However, this fact is not obvious from the visualization of the user's monitoring query, Q1. It requires analyzing additional data beyond what is used to compute Q1 (which used data relevant only to the user's Hadoop job, but not all the jobs in the system).

The version of record is available at: http://dx.doi.org/10.1561/1900000074

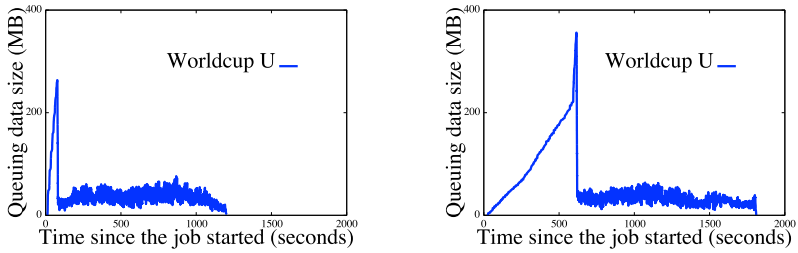


Figure 2.10: Hadoop cluster monitoring: Data queuing size of a normal Hadoop job (left), and data queuing size of an abnormal Hadoop job (right).

2.2.4 Responsible data analysis

Data analysis aims to extract interesting and useful insights from data. Mishandling, misinterpretation, and poor understanding of tools, can be detrimental in analytics, leading to flawed results. Explanation frameworks can support analytics through establishing the validity of findings, or highlighting misconceptions and misuse. We discuss several ways in which explanations can assist responsible analytics.

Fairness and bias

Data-driven systems often include learned components, trained over datasets that may themselves be imperfect and even biased. As a result, systems are likely to amplify these imperfections and biases. When systems exhibit bias and discriminatory behavior, explanations can serve as a tool to trace the reasons for the biased behavior, and point to ways to repair it. The goal of altering broader aspects of a system's behavior, such as result bias, is distinct from debugging, which has more precise targets. In general debugging, we typically observe concrete errors within data or system components. In this case, however, the issue is not confined to particular data or queries; rather it is manifested over broader operations. We present an example where initial biases in the data can lead a learned system to demonstrate discriminatory behavior. The purpose of the explanation is to map the problematic behavior to the underlying causes, so they can be addressed, if needed.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

id	name	gender	age	race	zip code	phone	credit score
t_1	Shanice Wilson	F	31	AA	60617	555-391-7654	low
t_2	Demetrius Smith	M	52	AA	60617	524-764-0032	low
t_3	Andre Holland	M	60	AA	60649	321-716-0187	low
t_4	Connor Wilson	M	51	W	01060	908-375-1073	high
t_5	Emily Strike	F	47	W	01009		high
t_6	Hannah Plath	F	28	W		918-938-8172	low
t_7	Jacob Alston	M	36	W	60636		high
t_8	Garrett Johnson	M	31	W	27780	510-276-9182	high
t_9	Logan Drake	M	32	W	01002		high
t_{10}	Brett Smith	M	28	W		413-726-1082	high

Figure 2.11: A sample dataset with 10 entities. A logistic regression classifier trained over this dataset discriminates against African Americans (`race` = 'AA') and women (`gender` = 'F').

Example 2.12. A classifier trained over the data in Figure 2.11 is bound to demonstrate bias, as the data itself is highly imbalanced. Specifically, the trained classifier, is likely to associate African Americans (`race` = 'AA') and women (`gender` = 'F') with low credit scores. Even if sensitive attributes, such as gender and race, are witheld during the training process, other, seemingly innocuous attributes can serve as a proxy, resulting in the same effect (e.g., zip code is correlated with race). System developers can use explanations to understand the reasons of the resulting bias, which can help them seek and train on a more balanced dataset.

Algorithmic fairness and the study of discrimination in computing systems has been a very active area of research in recent years, and has produced a large body of work across several disciplines. While a majority of the existing work is rooted in the machine learning community, data management research has shown growing interest in these topics. Our goal is not to cover this area in depth, but, rather, to highlight the connections to explainability, as explanations can help guide fairness repairs as illustrated in the above example.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Auditing

Many organizations are subject to strict auditing requirements enforced by law. To comply with the law, these organizations are required to keep a record of their data handling processes. Such records can also be used for identifying security breaches and for forensic analysis in the event of a security breach.

Example 2.13. Consider a health care provider that maintains an audit log (Snodgrass *et al.*, 2004) to keep track of accesses to their database and uses temporal database technology to be able to access past versions of their data. If a user account is compromised and this security breach has been detected, then the records kept in the audit log enable the health care provider to investigate the breach, e.g., to determine what data has been modified by the compromised account. However, the large number of transactions run every day and the large amounts of data the provider maintains make it hard for an analyst to understand the impact of the breach, since this may require browsing through thousands of SQL statements executed by the compromised account and millions of tuple versions accessed and created by these statements. Thus, there is a need for techniques to extract high-level explanations from these low level details.

Reproducibility

Reproducibility of computational scientific experiments is one of the grand challenges of our time. Computational experiments may fail to be reproducible for a variety of reasons, such as dependencies on libraries, environment settings, lack of documentation for how to run the experiments, and non-determinism in the experimental code.

Example 2.14. Alice, a biologist, published a paper about the new gene analysis algorithm she has developed. The algorithm takes as input multiple genomes, prepares the data using a range of publicly available tools, then combines the prepared data and runs a statistical analysis over the results. Another scientist, Bob, wants to reproduce the results of Alice's algorithm as published in her paper. Even if Alice is diligent in documenting her process, there are many possible pitfalls

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Bob may face during his attempt to reproduce Alice's results, such as incompatibilities between his and Alice's execution environments (e.g., different library versions), missing datasets, or non-determinism of steps in Alice's workflow.

2.3 "What" we want to explain

In this section, we discuss the *target* of the explanation: *What* we are trying to explain. The explanation target corresponds to the precise element that an explanation seeks to justify, e.g., a particular value, or a query, and it is distinct from the explanation's purpose. For example, Examples 2.5 and 2.6 both describe settings where the purpose of the explanation is debugging and understanding of errors; however, in the former, the explanation target is a query, whereas in the latter, the explanation target is erroneous data. The explanation target ("What") is also distinct from the explanation itself; the latter is tied to the solution approach, which we discuss in the following chapter ("How"), and can potentially vary, even for the same explanation problem setting. Here, we discuss explanation targets that have appeared in the literature, frequently referring to prior examples and occasionally introducing new ones.

2.3.1 Data

The most common explanation target in the data management literature is, perhaps not surprisingly, data. As data is a central component of data-driven systems, surprising observations and unexpected behaviors are often indicated within the data itself. Examples 2.1, 2.6, and 2.7 all identify data as the explanation target, where values or extraneous information indicates particular data as surprising or problematic. The solutions methods and derived explanations may ultimately differ in each case, but the common factor is *what* we seek to explain, which is specific data items or values. Existing literature that focuses on data as the explanation target, typically considers *output data* in particular, which we discuss next. However, any data, even outside the behavior of a system, can serve as the target, and existing explanation frameworks

The version of record is available at: http://dx.doi.org/10.1561/1900000074

have been used in this manner. For example, Data X-Ray (Wang *et al.*, 2015a) has been used to derive explanations for traffic incidents over an accident report dataset (Wang *et al.*, 2015b).

2.3.2 Data processing output

When we consider explanations in the context of data-driven systems, an explanation-worthy observation is typically tied to the system's operation and function. Therefore, much of the existing literature on explanations targets the *output* of data processing operations, such as queries (Example 2.3), updates (Example 2.7), or more complex, and potentially non-relational, data processing pipelines (Example 2.6). Within the processing output, explanations may target different aspects of the data:

- *Data items or values.* Example 2.1 shows an instance of explanations targeting the presence of particular data items or values in the result of a query: certain tuples or their values may be unexpected and, thus, and merit investigation. The absence of expected results, such as in Example 2.2, can also be the target of explanations. In the literature, this category is frequently referred to as *missing answers* or *why-not* explanations.

- *Outliers.* Sometimes, what makes a data value surprising is its relationship to other values. Results that deviate substantially from the mean of the output distribution may be indicative of a problem or a phenomenon that needs to be better understood. For example, the Scorpion (Wu and Madden, 2013) and the CAPE systems (Miao *et al.*, 2019c) target outliers in the result of aggregate queries.

- *Trends.* Output explanations may not always target particular data items, but groups of items that indicate a surprising trend. In Example 2.3, the target of the explanation is the decreasing trend of industry publications at SIGMOD, which is surprising given the increasing trend of papers from academia.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

2.3.3 Performance

Aside from unexpected data results, interesting issues and surprising observations may sometimes be evidenced through other aspects of a system's behavior. Common observable aspects of performance, such as overall runtime or use of resources, can indicate a problem with the system or interactions that were not previously understood and merit explanation. Examples 2.10 and 2.11 demonstrate two such cases in relational and non-relational cases. Understanding the reasons for unexpected performance can facilitate the improvement of the corresponding systems. Performance considerations can also include the overall quality of a system's function. For example, if a system produces generally poor results based on some metrics (e.g., in Example 2.12 the learned system is bound to produce discriminatory results), explanations can target these aspects of performance, with the goals of understanding the behavior and potentially changing it.

2.3.4 Queries

Any component of a system can be an explanation target. This hasn't been explored much in general data processing pipelines, as individual components can be complex. However, in the context of relational data, prior work has identified queries as explanation targets. Example 2.5 identifies the erroneous student query as the explanation target. Note that we shouldn't confound the explanation target (*what* we are explaining) with the derived explanation (*how* we explain it). In this example, "what" focuses on the query; the "how" is not another query—the correct query is already known—rather it is a dataset that succinctly highlights the problems in the submitted query (Miao *et al.*, 2019a). We discuss explanation approaches ("how") in the next chapter.

2.3.5 Differences

When we think of *what* merits explanation, we typically think of something surprising that deviates from some norm or expectation. A lot of the work on explanation research tends to overlook the reason itself for seeking an explanation, merely focusing on deriving an explanation as-

The version of record is available at: http://dx.doi.org/10.1561/1900000074

suming that something is labeled by some oracle as explanation-worthy. But the precise indication of the deviation—what makes something surprising, and, thus, explanation-worthy—can provide important clues on what the proper explanation should be. Some work relies on rough indicators of the deviation, e.g., a value is surprisingly *low* or *high* (Roy and Suciu, 2014). But these indicators are often vague and don't specify the nature and extent of the deviation precisely. When possible, more accurate specification of the deviation can lead to more targeted explanations.

The difference between an expected and an unexpected value or behavior, or particular change in data or behavior can provide such a specification. Example 2.4 explicitly highlights such differences in query results as the explanation target: given two trusted datasets that are supposed to represent the same data, albeit under a different schema, one should expect queries that seek the same information to return the same results. When they do not, explanation frameworks can assist in interpreting these differences. In some of our other examples, differences are an implicit focus of explanations: Example 2.9 focuses on the modifications inflicted by a repair algorithm on a dataset, while Example 2.11 specifies a performance abnormality against another execution that progresses normally. In both these cases, the change is what captures the deviation that is deemed interesting, making it the explanation's target.

2.3.6 Machine learning models and predictions

In recent years, machine learning has become prevalent in a wide range of application domains. Systems using learned models are used to make or influence critical decisions with practical human and societal impact, ranging from product recommendations, to medical diagnosis and treatment, and even criminal sentencing. As such systems incur a profound impact on people's daily lives, there is a pressing need to understand, justify, and enhance trust in automated decision making by ML models. Explanations have targeted both the decisions, as well as the models themselves, e.g., why does a model have certain characteristics, and explainability is an active field in the machine learning community. We discuss an example that highlights this problem setting.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

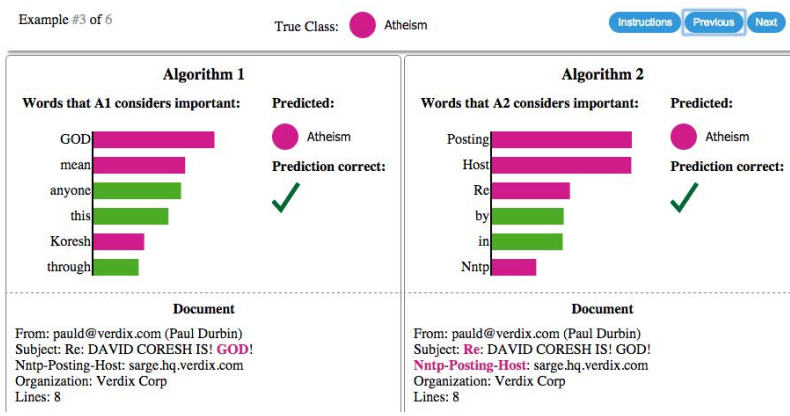


Figure 2.12: Example from Ribeiro *et al.*, 2016, explaining individual predictions of competing classifiers that try to determine if a document is about "Christianity" or "Atheism". The bar chart represents the importance given to the most relevant words, also highlighted in the text. Color indicates which class the word contributes to (green for "Christianity", magenta for "Atheism").

Example 2.15 (Ribeiro *et al.*, 2016)**.** Figure 2.12 shows explanations for the predictions of two classifiers over the same data. The right side indicates the words that most contribute to the prediction of a support vector machine with an RBF kernel, trained on unigrams to differentiate "Christianity" from "Atheism." Although this classifier achieves 94% held-out accuracy, and one would be tempted to trust it based on this, the explanation for this instance shows that predictions are made for quite arbitrary reasons (words "Posting", "Host", and "Re" have no connection to either Christianity or Atheism). After getting such insights from explanations, it is clear that this dataset has serious issues (which are not evident just by studying the raw data or predictions), and that this classifier, or held-out evaluation, cannot be trusted.

We will briefly review some relevant approaches to explainability for ML models and explainable AI (referred to as XAI) in the next chapter, but as mentioned earlier, a detailed discussion on XAI is beyond the scope of this article.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

3

Explanations and Methodologies: How

In the previous chapter, we discussed the dimensions of "Who", "Why", and "What" that specify the explanation needs of a particular problem setting. A problem setting is not explicitly tied to particular approaches to explanations and solutions, and there is often more than one way to tackle the same problem. However, it is possible that particular problem settings are more amenable to certain approaches. Our intent has been to decouple the dimensions relevant to the problem specification from the particular methodologies and frameworks proposed in the literature. We discuss the latter in this chapter. Specifically, we examine how the existing literature has tackled some of these problems settings, and identify several high-level desiderata, methodologies, and commonalities across techniques.

We characterize the solution space of explanation settings as the "How" dimension. We first examine some general principles of explanation optimality, i.e., how to measure the quality of an explanation, and how to compare two explanations to determine which is preferable. We present several high-level objectives that have been proposed in the literature, and suggest some additional ones to consider (Section 3.1). Then, we identify common explanation types (what a derived

The version of record is available at: http://dx.doi.org/10.1561/1900000074

explanation looks like), and high-level aspects of methodology (e.g., intervention-based, summary-based, etc.) in Section 3.2. Finally, we discuss particular approaches from the existing literature for a selection of problem settings (Sections 3.3 to 3.9).

3.1 Explanation objectives: measuring explanation quality

As a tool primarily meant for human consumption, explanations remain highly-subjective notions. Their effectiveness is not always quantitatively measurable, and, ideally, explanation quality should be evaluated through targeted qualitative analysis and user studies collaborating with domain experts and cognitive scientists. The varied desiderata implied by the different problem settings further complicate this issue. As we discussed, explanations meant for debugging purposes likely need to be more specific and more detailed than those meant for illustrative purposes. Similarly, explanations targeting a general audience would look different that those targeting domain or technical experts.

The existing literature has taken varied approaches to defining explanations, specifying their desirable properties, and proposing metrics for evaluating them. However, ultimately, there is no universally accepted set of objectives that explanations should satisfy. Our goal here is to distill some high-level desiderata for explanations that are frequently associated with explanation quality.

- **Succinctness.** The core purpose of explanations is to aid our understanding of an observation. However, understandability is a subjective and not directly measurable notion. The explanation literature has frequently used *succinctness* as a substitute for understandability. Intuitively, explanations should be small enough to be easily understood. Large explanations, while accurate, are unlikely to aid the goal of understandability. For example, consider the case where the goal is to explain a trend in aggregate results (Example 2.3). Providing the *why provenance* that consists of all relevant input tuples (Cheney *et al.*, 2009) as an explanation is unlikely to be effective, because aggregation results typically depend on a large number of input tuples. Thus, the set of relevant input

The version of record is available at: http://dx.doi.org/10.1561/1900000074

tuples, or even some algebraic form like provenance semirings for aggregates (Amsterdamer *et al.*, 2011), would likely be impractical for human consumption. Therefore, succinctness is intuitively a good proxy for understandability, and explanation frameworks commonly use it as an explicit objective.

- **Interpretability.** An important dimension of explainability is the target audience ("Who"). Explanations need to be in a form that is easily interpretable by their intended audience. For example, explanations in the form of raw data may often be unsuitable for non-experts, whereas higher-level meta-data that summarizes aspects of the relevant data is likely to be easier to interpret. Intuitively, interpretability as a goal drives design choices in the explanation form, rather than specifying an explicit optimization objective.

- **Actionability.** A driving motivation for producing explanations within a data system is the particular purpose ("Why"), which may include goals like debugging, debiasing, or performance enhancements. Explanations need to point to actionable suggestions for satisfying the purpose they were designed for. Therefore, they need to consider which components of the data or system can be subject to intervention, and the practicality of such interventions.

- **Measurability and Comparability.** The search space for possible explanations is typically very large. The goal of optimization objectives is to efficiently search through this space and narrow down the selection of relevant explanations as much as possible. Metrics for explanation quality should enable pruning parts of the search space when ranking explanations.

- **Computability.** Related to the above point, explanations are typically intended for interactive data analysis, and therefore, should be efficiently computable even if the search space is large.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

3.2 General methodologies

Existing literature has employed a variety of approaches in the design of explanation frameworks. Before discussing particular solutions, we first present the high-level principles of common methodologies in this research space.

- **Provenance-based** methods rely on maintained provenance information to derive explanations. Such provenance-based explanations trace the target output to particular inputs or to components in the pipeline that participated in the derivation of this output. Provenance-based explanation frameworks require maintenance of potentially large provenance data, or relatively simple computations where provenance information can be generated on the fly or reverse-engineered.

- **Intervention-based** methods rely on implementing changes to the data (e.g., removal of tuples) or the system (e.g., forcing a program state), and observing the effect that such interventions have on the system output or behavior. When the effect (e.g., output or behavior) that provoked the need for explanation is reversed by the intervention, this indicates a connection between the explanation target an the inflicted change. Interventional methodologies are often used when reasoning about causality, as causality often cannot be inferred from observational data alone. However, interventions can be computationally intensive, as systems need to be rerun, and computations need to be repeated, often over a large number of interventions. A common approach for addressing this combinatorial problem is to use properties of the system under study to exploit commonalities between computations for sets of interventions or to prune interventions that are guaranteed to be of lower quality than previously explored explanations.

- **Statistical or observation-based** methods reason over previously gathered data, rather than deciding on and executing interventions. They are more practical in cases where there is

The version of record is available at: http://dx.doi.org/10.1561/1900000074

already enough data to use for such analysis, but they are often limited with respect to distinguishing correlation from causation.

- **Summarization-based** methods reduce the size of explanations through summarization. Summarization techniques that have been applied involve feature extraction, taxonomies, mappings, or general meta-data that encompass larger sets of lower-level explanations. For example, if a framework derives sets of tuples as explanations, but the sets are large, summarization methods may increase succinctness by clustering the inputs, and replacing the cluster by a higher-level description, e.g., by identifying commonalities among the elements of a cluster (e.g., all customers are from the US). As such, summarization-based approaches can unearth trends in data or computations.

- **Example- or Counterexample-based** methods rely on the human inclination to perceive things through examples. These frameworks typically derive illustrative explanations, where an effect is explained through an example or counterexample. For example, a particular output can be explained by demonstrating a particular part of the data flow that derives it; or, a mistake in a query can be elucidated by a small data sample that highlights the deviation from the expected result.

- **Model-based** methodologies reduce the problem of generating explanations into instances of another type of problem which can be solved using existing tools. Common targets of such transformations are learning problems, satisfiability problems, or optimization problems.

Note that explanation frameworks may combine several of these methodologies. For example, summarization techniques are often used to extract higher-level explanations from provenance information (e.g., Cate *et al.*, 2015; Lee *et al.*, 2020). Now that we have described these general methodologies, we will highlight particular approaches in the literature, organizing them with respect to different problems settings.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

3.3 Explaining query answers through data provenance

Perhaps the most prevalent approach to explaining, debugging, and tracing query answers is through tracking *provenance* or *lineage*. Provenance has been studied in the database research community for decades, and has been used broadly in various contexts and granularities including in scientific workflows and storage systems (e.g., Muniswamy-Reddy *et al.*, 2006; Davidson *et al.*, 2007). Most relevant to this article is the concept of *fine-grained data provenance* that records the origin of the query answers in terms of the input data (e.g., Cui and Widom, 2001; Buneman *et al.*, 2001; Green *et al.*, 2007; Cheney *et al.*, 2009). Provenance can be recorded at different granularity levels, e.g., simply by showing the set of input tuples that contributed to a query answer (*why-provenance*), or by showing the process that generated the query answer using a Boolean formula (*how-provenance*), which records conjunctive uses of input tuples through joins ($\wedge$) and alternative uses of tuples through projection or union ($\vee$). Consider a toy example of a database D with two relations: $R(A, B)$ and $S(B, C)$, with tuples $r_1 = R(a_1, b)$, $r_2 = R(a_2, b)$, and $s = S(b, c)$, and the query $Q{=}\pi_C(R \bowtie S)$, which joins R and S on B and projects the result to C. There is a single output in $Q(D)$, i.e., c, with lineage that can be expressed as the Boolean formula $(r_1 \vee r_2) \wedge s$. This expression indicates that the output c will be generated if and only if tuple s is present, *and*, *at least one of* r_1, r_2 is present. This Boolean lineage formula can serve as an explanation, providing a succinct and precise representation of the generation of the output c.

Provenance semirings (Green *et al.*, 2007) also provide a general framework for expressing provenance information. Such expressions explain in detail *how* a query result is derived from the input data and *which* input tuples contributed to which results. Building upon such concepts, the notion of provenance has also been extended to aggregate queries (Amsterdamer *et al.*, 2011; Glavic *et al.*, 2013). Other examples of provenance models that record such information include provenance traces (Cheney *et al.*, 2014) and provenance models for recursive Datalog queries that track rule derivations (Köhler *et al.*, 2012; Deutch *et al.*, 2014; Lee *et al.*, 2018).

The version of record is available at: http://dx.doi.org/10.1561/1900000074

There is a vast and rich literature on provenance in database research (e.g., see the article by Glavic (2021), Cheney *et al.* (2009), or the tutorial by Tannen (2010)). Further, as discussed in Section 3.1, why- or how-provenance may not satisfsy our desiderata for good explanations in many of our targeted applications. In this article, we mainly focus on other methods, some of which still use provenance, though in different forms.

3.3.1 Refining provenance information

So far, we discussed provenance as a prevalent tool for supporting explanations for query answers, such as the setting of Example 2.1. Such explanations may simply include all input tuples that contributed to the output (why-provenance) or a more fine-grained description of the precise derivation using a Boolean expression (how-provenance). In the toy example we presented earlier in the section, the Boolean expression of the provenance was small enough to provide a succinct and accurate explanation of the answer. However, in practice, provenance expressions can grow large and become hardly interpretable by a human user, thus, diminishing their value as explanations. Prior work has explored ways to refine provenance information by prioritizing input tuples based on a metric of their contribution towards a particular output. We discuss two types of metrics here.

Responsibility

In our toy example, the Boolean provenance expression $(r_1 \vee r_2) \wedge s$ contains only three tuples. But the way these tuples contribute to the output is different. Tuple s is necessary for producing the output, whereas only one of r_1 and r_2 is needed. Thus, intuitively, the contribution of s to the particular query answer should be ranked higher than the contributions of each of r_1 and r_2. Meliou *et al.* (2010) proposed the notion of *responsibility* as a measure for tuple contributions to a query answer. This metric was adapted from prior work in the causality literature (Pearl, 2000; Chockler and Halpern, 2004; Halpern and Pearl, 2001) and is based on the notion of intervention and counterfactual causality. In simple terms, an input tuple $t \in D$ is a *counterfactual*

The version of record is available at: http://dx.doi.org/10.1561/1900000074

cause of an answer $a \in Q(D)$ if removing t from the input also removes a from $Q(D)$; hence, in our toy example, s is a counterfactual cause for output c. Extending counterfactuals, an input tuple $t \in D$ is an *actual cause* of an answer $a \in Q(D)$ with *contingency* Γ, where Γ is a set of input tuples, if removing Γ from D makes t counterfactual. Hence, in our toy example, r_1 is an actual cause with contingency $\Gamma = \{r_2\}$. Finally, the responsibility of a tuple t is defined as $\rho = \frac{1}{1+\min_\Gamma |\Gamma|}$. Thus, in our toy example, $\rho_s = 1$ and $\rho_{r_1} = 0.5$. Meliou *et al.* (2011) extended the notion of responsibility to multiple outputs and views. Freire *et al.* (2015) later revisited the notion of responsibility and analyzed it for cases where functional dependencies are present in the data.

Shapley values

Livshits *et al.* (2020) proposed an alternative method for measuring tuple contributions, adapting the notion of *Shapley Values* from the Game Theory literature (Shapley, 1953). The concept comes from a cooperative game that is played by a set $\mathcal{A}$ of *players*, and there is a *wealth function* v that assigns wealth $v(\mathcal{S})$ to each coalition $\mathcal{S} \subseteq \mathcal{A}$ of players. For example, in a publication/citation database, the players can be researchers, and $v(\mathcal{S})$ can be the total number of citations to papers by an author in $\mathcal{S}$. The Shapley value aims to distribute the wealth $v(\mathcal{A})$ among the players $a \in \mathcal{A}$ by quantifying the contribution of each player to the overall wealth (in this example, contribution of an author for citations).

Formally, in a *cooperative game* with players $\mathcal{A}$, there is a function v (called the *characteristic function* or the *wealth function*) $v : 2^{\mathcal{A}} \to \mathbb{R}$ with $v(\emptyset) = 0$. For any subset $\mathcal{S} \subseteq \mathcal{A}$ of players, $v(\mathcal{S})$ represents the value of the outcome produced when the players of $\mathcal{S}$ cooperate. As mentioned above, the Shapley value measure the contribution of a player a to the outcome as the expectation of the difference between the value $v(\mathcal{S})$ and $v(\mathcal{S} - \{a\})$ over all coalitions $\mathcal{S}$ not including a, i.e.,

$$\text{Shapley}(\mathcal{A}, v, a) = \sum_{\mathcal{S} \subseteq \mathcal{A} - \{a\}} \frac{|\mathcal{S}|\,! \cdot (|\mathcal{A}| - |\mathcal{S}| - 1)\,!}{|\mathcal{A}|\,!} \Big(v(\mathcal{S} \cup \{a\}) - v(\mathcal{S}) \Big)$$

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Note that computation of Shapley values requires computing the expected new contribution of a player with respect to a random permutation of the other players. since the number of possible permutations is exponential, the computation of the Shapley value is hard in the general case. Livshits *et al.* (2020) gives complexity results for both conjunctive and aggregate queries, and proposes approximation algorithms for the hard cases; in a later work, (Reshef *et al.*, 2020) studied the complexity of computation of Shapley values to queries with negation.

Responsibility and Shapley values both relate to intervention-based methods, as they try to quantify the contribution of a single input tuple to the query answer in addition to other tuples present in the database. The Shapley value considers more fine-grained incremental contributions to the output than responsibility (by considering all possible permutations of all possible subsets before adding the input tuple), and therefore is computationally more expensive (Livshits *et al.*, 2020).

3.4 Explaining aggregate query outputs and outliers

Queries in data analysis are likely to have *aggregates* (sum, count, min, max, etc.), often plotted as scatter plots, line graphs, or bar charts for ease of analysis. We provide a simple scenario: there is an aggregate SQL query Q possibly with a group-by operator, a database D, and the user is trying to understand the result tuples in the query answer denoted by $Q(D)$. We use $\texttt{agg}(t)$ to denote the aggregate value of an answer tuple $t \in Q(D)$, and $\texttt{gb}(t)$ to denote the non-aggregate attributes (if any) for t; for simplicity, we assume a single aggregate value in Q unless specified otherwise.

Example 3.1.

Consider a simplified schema for the publication data of Example 2.3, with three tables: `Author(aid, name, inst, domain)`, `Pub(pid, title, year, venue, area)`, and `Authored(aid, pid)`. Unique keys are underlined. Suppose `domain` can have values 'edu' (academia in the United States) or 'com' (industry), which can be found for researchers from the domain of their webpages whenever available; `inst` denotes the institution where the researcher works. The output

The version of record is available at: http://dx.doi.org/10.1561/1900000074

bars can be obtained by a SQL query Q_{sigmod} joining these three tables, selecting for `venue = 'SIGMOD'` and ranges of years with a 5-year moving window, and for each window outputting the number of *distinct* `pids` as each paper can have multiple authors. Here, the group by attributes $\texttt{gb}(t)$ contain (`year-range, domain`) and $\texttt{agg}(t)$ denotes the distinct publication count as shown in the bars in Figure 2.3. For example, for the tuple corresponding to the bar q_1 shown in Figure 2.3, $\texttt{gb}(q_1)$ is (`2000-04, 'com'`) and $\texttt{agg}(q_1)$ is the height of the bar q_1 denoting publication counts in SIGMOD from industry in years 2000–2004.

When the user studies the output of $Q(D)$, she might find some interesting, unexpected, or counter-intuitive values, sometimes in comparison with the other values. Such questions might include (but not limited to):

1. **(Outliers)** Why is the value of $\texttt{agg}(t)$ high/low for a $t \in Q(D)$? In other words, why is t an *outlier* in $Q(D)$? (e.g., why is $\texttt{agg}(q_4)$ high in Figure 2.3?).

2. **(Comparisons)** Why is the value of $\texttt{agg}(t_1)$ higher/lower than $\texttt{agg}(t_2)$, for $t_1, t_2 \in Q(D)$? (e.g., why is $\texttt{agg}(q_1)$ higher than $\texttt{agg}(q_2)$ in Figure 2.3?).

3. **(Complex comparisons)** Why is $\texttt{agg}(t_1)$ higher (lower) than $\texttt{agg}(t_2)$, but $\texttt{agg}(t_3)$ is lower (higher) than $\texttt{agg}(t_1)$, for $t_1, t_2, t_3, t_4 \in Q(D)$? (e.g., why is $\texttt{agg}(q_1)$ higher than $\texttt{agg}(q_2)$ while $\texttt{agg}(q_3)$ is lower than $\texttt{agg}(q_4)$ in Figure 2.3?).

4. **(Trends)** Why do I see a trend (e.g., increasing/decreasing) in the `agg` values for tuples $t_1, t_2 \cdots, t_\ell$? (e.g., why do the light blue bars for SIGMOD publications from academia have an increasing trend in Figure 2.3?)

Such questions are likely to come from users who are data analysts, researchers, decision makers of relevant application domains, or sometimes data enthusiasts and the general public (*"Who"*). Some of these researchers might have a technical background in computer science, statistical analysis, or data science, while some users might

The version of record is available at: http://dx.doi.org/10.1561/1900000074

be more familiar with the domain instead. Therefore, the goal is to provide meaningful explanations for this range of users as outlined in Section 3.1. Next, we discuss a few approaches to explanations from the literature for such aggregate query answers.

3.4.1 Intervention-based approaches

We start with an overview of intervention-based methods to explaining aggregate query answers and outliers, proposed in Wu and Madden, 2013, Roy and Suciu, 2014, and Roy *et al.*, 2015a. Here, we discuss some high-level ideas from these approaches and refer to the papers for technical details.

Motivation from causality. The basic idea of *intervention-based* approaches is to (1) assume a compact representation of the tuples identified in the explanation question along with the direction of the deviation (*high* or *low*) as perceived by the user, and then (2) make changes to the database D that would push the representation to the *opposite direction*. For example, if the user asks *'why is* $\texttt{agg}(t)$ *high'*, a good intervention will change the data instance D to a new data instance D' (typically, as close as possible to D) such that $\texttt{agg}(t)$ will be much lower in $Q(D')$. The concept of interventions stems from the literature on *causal analysis* (e.g., Pearl, 2000, see Chapter 4 for more details on causal analysis):

- **(causality by intervention)** A variable Y is a *cause* for another variable Z if changing Y changes Z, i.e., $\Delta Y \Rightarrow \Delta Z$. In other words, *an intervention* on Y causes a change in Z.

Motivated by the causal notion of interventions, we can derive a similar concept of explanations for database query answers:

- **(explanations by intervention)** A set of input tuples $\Delta D \subseteq D$ is an explanation for (one or more designated query answers in) $Q(D)$ if *'intervening on'* that subset also changes $Q(D)$ to some extent, i.e., $\Delta D \Rightarrow \Delta Q(D)$, where ΔD denotes the subset of input tuples changed by the intervention. Interventions causing higher changes in the output are intuitively stronger explanations.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Interventions can include the *deletion* of tuples, the *addition* of tuples, or the *modification* of tuples. Deletion is often a preferred choice for intervention (Wu and Madden, 2013; Roy and Suciu, 2014; Roy *et al.*, 2015a), since adding or modifying tuples raises concerns regarding the validity of such updates in a dataset in practice. In this section, we will focus on deletions as the intervention.

Allowing an explanation to be defined as an arbitrary subset of the input tuples has disadvantages with respect to common explainability objectives (Section 3.1): (1) an arbitrary subset of tuples can be large, thus, failing the expectation for *succinctness*; (2) tuples in an arbitrary set may not share common properties, thus leading to poor *interpretability*; (3) the search space over all possible subsets of a dataset is exponential in the size of the input data, leading to poor *computability*. As a result, it is common to opt for explanations in more restricted, compact forms.

Predicates as explanations. A natural way to think collectively about groups of tuples is through predicates. Predicates provide a natural way to summarize commonalities among input tuples, and they can be combined through conjunctions to create more restricted groups. Thus, producing predicates as explanations has been a popular approach in the literature (Wu and Madden, 2013; Roy and Suciu, 2014; Gebaly *et al.*, 2014; Roy *et al.*, 2015a; Lee *et al.*, 2020). Predicate-based explanations are succinct and easily understandable, effectively summarizing the properties of the tuples relevant to the explanation.

Since explanations are modeled as predicates, interventions can operate directly on the predicates: Given an explanation predicate ϕ, the intervention on ϕ, denoted by Δ_ϕ, corresponds to the subset of tuples in D *defined* by the predicate ϕ. This type of *deterministic and unique* interventions on predicates leads to a significant reduction in the search space, which is only exponential with respect to schema complexity, but polynomial with respect to the most common concern of *data complexity* (Vardi, 1982). However, more complex (Meliou *et al.*, 2011; Meliou and Suciu, 2012) as well as stochastic interventions are possible, though little work has been done in this direction (we discuss further in Chapter 4).

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Ranking explanations. We now recast the explanation questions listed after Example 3.1 using a simple formalism: we denote the answer tuples that are the explanation target with $S \subseteq Q(D)$, and we use $f(S, D, Q)$ to represent the target relationship (e.g., comparison, trend, etc.); then the explanation seeks to answer why $f(S, D, Q)$ is high or low:

- **(Outliers)** $S = \{t\}$ and $f(S, D, Q) = \texttt{agg}(t)$.
- **(Comparisons)** $S = \{t_1, t_2\}$, and $f(S, D, Q)$ can be $\frac{\texttt{agg}(t_1)}{\texttt{agg}(t_2)}$, $(\texttt{agg}(t_1) - \texttt{agg}(t_2))$, etc.
- **(Complex comparisons)** $S = \{t_1, t_2, t_3, t_4\}$, and $f(S, D, Q)$ can be $\frac{\texttt{agg}(t_1)/\texttt{agg}(t_2)}{\texttt{agg}(t_3)/\texttt{agg}(t_4)}$, $(\frac{\texttt{agg}(t_1)}{\texttt{agg}(t_2)} - \frac{\texttt{agg}(t_3)}{\texttt{agg}(t_4)})$, etc.
- **(Trends)** $S = \{t_1, t_2 \cdots, t_\ell\}$, and $f(S, D, Q)$ can be the slope of the best fitted line by linear regression.

If the question involves why f is *'high'*, a good explanation predicate ϕ will *decrease* the value of f after intervention. Therefore, a possible choice of ranking function is ordering the explanation predicates ϕ in increasing order of changes: $\rho_\phi = |f(S, D, Q) - f(S, (D - \Delta_\phi), Q)|$ (Roy and Suciu, 2014; Roy *et al.*, 2015a). However, without other safeguards, the highest-ranked explanation may be the one that removes all tuples, which is not meaningful or useful. One way to avoid this is to penalize the scoring function based on the number of deleted tuples, e.g., Wu and Madden, 2013 uses $\rho_\phi = \frac{|f(S,D,Q) - f(S,(D-\Delta_\phi),Q|}{|\Delta_\phi|}$. Similarly, if the question involves why f is *'low'*, a good explanation predicate ϕ will *increase* the value of f after intervention; however, we should note that a monotone f cannot increase in value after tuple deletion (alternative approaches are discussed in Section 3.4.2).

Other ranking considerations may account for producing diverse explanations (i.e., avoid having the top-ranked explanations be too similar), which may be helpful for interactive exploration. Generally, having a good balance of *diversity*, *coverage*, and *utility*, as done in general top-k query answers (Joglekar *et al.*, 2016; Wen *et al.*, 2018), can also be considered while ranking explanations.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Overview of algorithms. With a well-defined objective function ρ_ϕ, the goal is to devise algorithms that can find top-k explanation predicates ϕ as efficiently as possible. The Scorpion system (Wu and Madden, 2013) considers a database with a single table (the materialized universal table with the join output for multiple tables), and finds top explanations in two ways: (1) with a *top-down decision tree partitioner*, or (2) with a *bottom-up partitioner* that starts with single-attribute predicates and then intersects them to construct multi-attribute predicates. Scorpion also uses several optimizations like sampling and parallel partitioning.

On the other hand, the key idea in Roy and Suciu (2014) is to treat tables in a multi-relational database separately and take into account mutual dependencies for the existence of tuples. A simple kind of dependency is referential integrity constraint with cascade delete semantics (if a tuple with the primary key is deleted, all tuples with a foreign key referring to this primary key are deleted). However, other types of user-defined constraints might also exist, e.g., if an author is removed, then eventually all of their publications should be removed from the database, which requires an extension to foreign keys. Roy and Suciu (2014) give a recursive program that can compute interventions for a given explanation predicate ϕ in the presence of mutual dependencies among tuples, which gives the exact intervention Δ_ϕ and scoring function ρ_ϕ, but is not efficient. They also give an efficient optimization heuristic based on the SQL *data cube* operator (Gray *et al.*, 1997) for evaluating ρ_ϕ for all possible explanation predicates ϕ. For Examples 2.3 and 3.1, this returned interesting explanations like leading industrial labs (and their senior database researchers) that were highly active in database research in early 2000 but later had a shutdown or possibly shift in research focus, explaining the decline in industry papers in SIGMOD, as well as relatively new but highly productive academic institutions as they contributed more to the increasing trend in the academia papers compared to more established academic research groups. Subsequently, Roy *et al.* (2015a) proposed the notion of *explanation-ready databases* that pre-compute and store interventions of possible explanations as they are independent of queries and user questions, then evaluate all explanations simultaneously using concepts from *incremental view*

The version of record is available at: http://dx.doi.org/10.1561/1900000074

maintenance (Ceri and Widom, 1991; Ahmad *et al.*, 2012) to find the top ones when a user question arrives.

3.4.2 Counterbalance-based approaches

Insertions or updates are difficult to implement as modes of intervention, while preserving the semantics or properties of the data. Therefore, intervention-base methods usually rely on *deletion* of input tuples; this restricts the explanations to the *provenance* (contributing input tuples) of the answer tuples in the user's questions. This, in turn, leads to two limitations of the intervention-based approaches:

1. **(Context is ignored)** Explanations are limited in scope as interventions are restricted to tuples that directly contribute to the output tuples identified in the user question. Other parts of the input are ignored in the space of possible explanations, possibly losing useful contextual information from the rest of the data, which can be the bulk of the available data.

2. **(Cannot explain 'why low')** Intervention-based approaches intend to reverse as much as possible the trend perceived as deviating by the user (e.g., if the user asks why a value is high, good explanations would lower that value as much as possible through hypothetical tuple deletion). However, if the function f on query results that we want to change is monotone, then the output cannot be made higher by tuple deletion. This makes intervention-based approaches unsuitable for explaining some 'why low' questions.

Miao *et al.* (2019c) proposed an alternative to intervention-based explanations, called *explanations by counterbalance*. Suppose one notices a drop in crime rate in one area of a city in a particular year compared to the other adjacent years. To explain this drop, one can find higher crime rate in this area in the year before, which might have made the authorities redirect resources to this area to reduce the crime rate. Similarly, a lower than usual number of publications in a venue by a researcher can be explained by higher than usual number of publications in other venues in the same year by the same researcher.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

To obtain such explanations, Miao *et al.* (2019c) mine common *patterns* based on the group-by attributes and aggregate values with a large enough support and confidence from the data (e.g., the number of annual publications by many researchers is about constant, although the constant may vary from researcher to researcher). Then, with respect to these patterns, a low (respectively, high) outlier can be explained by a high (respectively, low) outlier, that together 'counterbalance' each other. The effectiveness of the explanations for the purpose of ranking is decided by (1) the *distance* of the explanations from the user question (e.g., publications in years further away have less weight), and (2) the *surprisingness in the aggregate value* (e.g., if the researcher publishes about 2-3 papers in venue X typically in a year, publishing 10 papers in that venue a year is more surprising than publishing 4 papers). We refer to Miao *et al.* (2019c) for technical details and concrete examples.

3.4.3 Summarization-based approaches

While the notion of provenance (Section 3.3) has often powered explanations of aggregate query answers (Amsterdamer *et al.*, 2011; Glavic *et al.*, 2013), fine-grained provenance is often too large for human consumption. Examples of use cases where fine-grained provenance is too detailed are queries with aggregation which derive a small number of outputs from a large number of input rows, and *why-not provenance* (Lee *et al.*, 2020) that may consist of a very large number of derivations even for simple queries. Summarization methods aim to refine explanations in ways that improve their succinctness, and, by consequence, their understandability.

Summarization by patterns or predicates

Selection patterns or predicates (discussed in Section 3.4.1) can serve as a summarization mechanism. For example, given a relation augmented with a binary outcome attribute, *explanation tables* (Gebaly *et al.*, 2014; Gebaly *et al.*, 2018) aim to find ets of patterns (predicates) that affect the outcome attribute the most. Gebaly *et al.* (2014) presented an athlete's exercise log as an example for this approach. Each row in the log records the `day` of the week and `time` of the exercise, the `meal` eaten

The version of record is available at: http://dx.doi.org/10.1561/1900000074

before the exercise, and a binary outcome attribute indicating whether the exercise target `goal` was met. An explanation table consists of a set of patterns. Each pattern is a tuple of values and wildcards (similar to other work on predicate-based explanations, the wildcard symbol "*" represents all possible values of an attribute) along with the number of tuples it matches (count) and a fraction indicating how many of these tuples have a positive outcome (the outcome attribute is one). For instance, an explanation table for the exercise log may contain an entry $(Sat, *, *)$ with a count of 20 and a fraction of 0.2. This pattern states that the exercise goal was only met 20% of the time out of the 20 Saturdays recorded in the log, independent of the time and what the athlete has eaten.

Lee *et al.* (2020) offer another example of pattern-based explanations: they generate approximate summaries for why-not provenance to solve both the scalability as well as usability challenges stemming from the large size of why-not provenance. This approach highlighted the need of approximate techniques for application domains like why-not provenance where even enumerating all fine-grained provenance as input to summarization is not feasible.

Abuzaid *et al.* (2021) propose to integrate pattern-based summarization of the differences between two datasets into relational query processing in the form of a new logical operator called DIFF and demonstrated that several existing pattern-based explanations approaches can be expressed using the DIFF operator. Furthermore, the authors discussed several logical and physical optimizations for the DIFF operator. Earlier, Sarawagi and Sathe (Sarawagi, 2000; Sarawagi and Sathe, 2000; Sathe and Sarawagi, 2001) proposed operators (RELAX, DIFF, SURPRISE) for interactive exploration of OLAP data cubes that continuously adapt to the knowledge of the user on the data and guide her to the most informative parts of the cube from the viewpoint of the user.

Summarization with taxonomies

Another class of summarization approaches that has been used in this context is summarization based on *taxonomies* (Glavic *et al.*, 2015),

The version of record is available at: http://dx.doi.org/10.1561/1900000074

or more generally *ontologies* to summarize data for the purpose of explanations. Cate *et al.* (2015) used ontologies which are mapped to concepts in a database in the spirit of ontology-based database access (Calvanese *et al.*, 2007) to generalize a missing answer to a set of missing answers explained through a higher-level concept in the ontology. As an example, a user may wonder why there are no train connections from New York to Paris. The approach would explain this through the generalization of this missing answer: there are no train connections from any city in the US (the concept `UScity` subsumes New York) to any city in Europe (the concept `EuropeanCity` subsumes Paris). Interestingly, this work showed that in lieu of an user-provided taxonomy, a taxonomy can be build over queries (such as the selection patterns mentioned above) based on query subsumption: if a query Q_1 is contained in a query Q_2 then the ontological concept corresponding to the set of values described by Q_1 is a specialization of the ontological concept corresponding to the values described by Q_2. In this example, the query $\sigma_{county=US}$ (US cities) subsumes the query $\sigma_{state=NY}$ (NYstateCities).

Other approaches that use taxonomies are Data X-Ray (Wang *et al.*, 2015a), which creates summaries to describe errors in data based on hierarchical meta-data, and Glavic *et al.* (2015), which applies ideas from Cate *et al.* (2015) to explain the provenance of answers and non-answers.

Summarization with natural language

Another summarization mechanism is to factorize provenance information. Deutch *et al.* (2017) leveraged the structure of the user's explanation query in natural language to factorize the provenance by replacing subexpressions with counts. For example, for a query returning authors that from a particular university who have published at least one paper, an explanation for the provenance of a result tuple may replace the set of papers published by an author in the provenance with the number of such papers.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

3.5 Explaining queries

In many cases, explanations focus on understanding how a query, rather than the input data, affected a result. For example, when constructing a complex query, a user may want to understand which parts of the query are responsible for producing (or, failing to produce) an unexpected (or, expected) result. That is, explaining an answer or missing answer through properties of a query can aide users in debugging the query. Following terminology used for distinguishing between these two types of explanations for missing answers, we refer to such explanations as *query-based* as opposed to *instance-based* like the intervention and summarization-based approaches described in Section 3.4.

We can classify query-based explanation problems based on what should be explained and what properties of queries are explored for the explanation. A common class of query-based explanation problems explains a particular result, i.e., *why the query returned the particular answer*. Examples of this class of problems include *debugging an erroneous query result* by identifying which parts of the query caused the erroneous output to be returned (Glavic *et al.*, 2010; Alexe *et al.*, 2006; Fehrenbach and Cheney, 2019), *explaining a missing answer* by pointing out parts of the query that should be changed to make the missing answer appear in the result (Chapman and Jagadish, 2009; Tran and Chan, 2010; Bidoit *et al.*, 2014), and *program slicing* that identifies which parts of an input program are sufficient for producing an output or intermediate result of interest (Cheney, 2007; Xu *et al.*, 2005; Tip, 1994; Weiser, 1981). Note that, in this problem setting, we assume that the query (or program) can be broken into parts, and these individual parts are considered to be potential causes for the observed outcome (existing or missing answers). For example, we may define the parts of a query expressed in relational algebra to be the operators of the query. Closely related to this type of explanations are approaches that explain why a query result is empty (this is often called the empty answer problem) or why a query returns too many results (Mottin *et al.*, 2013). While identifying the parts of a query that are responsible for producing a result (or failing to produce a result), this may not be sufficient information to repair the query and resolve the error. An

The version of record is available at: http://dx.doi.org/10.1561/1900000074

alternative type of explanation may directly derive a *repaired version of the query*. Finally, counterexample-based mechanisms can also be used in query-based explanations (Chu *et al.*, 2017b; Miao *et al.*, 2019a). We proceed to discuss some of these approaches at a high-level, and refer the reader to the corresponding papers for more details.

3.5.1 Query-based explanations for missing answers

The problem of query-based explanations for missing answers was first studied by Chapman and Jagadish (2009). Their approach assumes that the input data is sufficient for producing the missing answer, and, thus, a query repair exists that can derive it. Given a query Q, database D and tuple t with the same schema as Q such that $t \notin Q(D)$, the approach identifies a set of operators that are "frontier picky". Intuitively, these are the operators that are responsible for removing intermediate data items, which were derived from input data items that could have contributed to the missing answers. In particular, the approach identifies the input tuples u that have attribute values compatible with the missing answer t (i.e., u and t agree on their common attributes). An operator is called frontier picky if there are successors of compatible tuples (tuples that have at least one compatible tuple in their provenance) in the input of the operator, but no successors of such tuples exist in the output from the operator. In other words, all tuples that could have produced the missing answer were removed by such an operator. The rationale is that only such tuples can be in the provenance of the missing answer if the picky operators are fixed so that they no longer filter out the missing answer. Bidoit *et al.* (2015) introduced why-not polynomials that represent query-based explanations as polynomials which encode all alternative (addition) of sets of conditions (multiplication) that caused a compatible input to be filtered out. The main advantage of this approach is that it enumerates all possible explanations and utilizes factorization to construct the set of explanations from explanations for simpler subqueries. Diestelkämper *et al.* (2021) presents an approach for nested data models and queries that scales to large data sizes and is based on a sound formalization rooted in query repairs. This approach is also the first to detect errors based on misuse of attributes (e.g., the

The version of record is available at: http://dx.doi.org/10.1561/1900000074

missing answer is caused by projecting on work address instead of home address).

Repairing queries

Several approaches moved beyond identifying the parts of the query responsible for a missing answer, to deriving repairs, i.e., modify the query, such that the modified version contains the (previously missing) answer in its result. Query refinement techniques such as Mishra and Koudas (2009) and Vélez *et al.* (1997) generate such repairs. Tran and Chan (2010) presented an early approach for this problem that tries to balance the similarity between the modified query and the original query as well as the side effects of the modification on the query's result. The approach enumerates possible changes, e.g., to selection conditions of queries, as potential repairs. Bidoit *et al.* (2016) refine queries using a syntax-independent representation of queries. Closely related to query refinement are techniques for query reverse engineering (QRE) (Kalashnikov *et al.*, 2018; Tran *et al.*, 2014) where the task is to given a query answer R and input database D to find a query such that $Q(D) = R$. The main difference to query refinement is that instead of refining an existing query, in QRE, no query is provided as reference.

3.5.2 Explaining wrong queries by counterexamples

In this section, we focus on the setting motivated by Example 2.5, where a user knows that a query is erroneous, and an explanation needs to elucidate the problems in the user's query. Typically, in a classroom setting, such errors in a wrong query Q_2 are detected (manually or by an autograder) based on a reference test database D and a correct query Q_1 such that $Q_1(D) \neq Q_2(D)$. However, as the reference database D is meant to capture many ways in which a query can go wrong, it is too large to shed light on the student's mistake. In addition, in a classroom setting, it is often undesirable to reveal the entire test database D. The *RATest* tool (Miao *et al.*, 2019a; Miao *et al.*, 2019b) derives an explanation for the wrong query Q_2 by aiming to find a smallest database sub-instance $D' \subseteq D$ such that $Q_1(D') \neq Q_2(D')$. In Example 2.5, to convince the student that their query is wrong,

The version of record is available at: http://dx.doi.org/10.1561/1900000074

the instructor can provide the full contents of the S and R tables as a counterexample comprising 11 tuples. However, a smaller and better counterexample can simply contain three tuples (e.g., t_1 from S and t_4, t_5 from R in Figure 2.4) to illustrate the non-equivalence of Q_1, Q_2. This problem is NP-hard in the general case (when the input queries are non-monotone) with respect to data complexity (Vardi, 1982). Moreover, the problem of minimizing counterexamples becomes much more challenging for queries with aggregates and group-by (e.g., with a "HAVING count(*) >= 10000" condition in the query). Miao *et al.* (2019a) give efficient solutions by tracking provenance semirings for non-aggregates and aggregates (Green *et al.*, 2007; Amsterdamer *et al.*, 2011), parameterization for queries with group-by and having conditions, and using SMT-solvers and other optimizations. They also discuss the effectiveness of this approach by deploying this tool in an undergraduate database course. The primary focus of Miao *et al.* (2019a) is simple relational algebra and SQL queries with at most one step of group-by and aggregates at the end. Miao *et al.* (2020) demonstrates a tool for complex SQL queries including set operators and nested subqueries where even the tracing problem becomes non-trivial.

Two related topics studied in the literature are deciding query equivalence and test data generation. In general, query equivalence is undecidable (Abiteboul *et al.*, 1995). A practical system, *Cosette* (Chu *et al.*, 2017b; Chu *et al.*, 2017a), aims to decide SQL equivalence without any test instances; it encodes SQL queries to constraints using symbolic execution, and uses a constraint solver to find counterexamples (with symbolic tuples and expressions) that differentiate two input queries. Another approach called *XData* (Chandra *et al.*, 2015) aims to capture as many erroneous queries as possible but does not consider a particular wrong query. Other methods for generating test data instances and helping users understand the behavior of their dataflow programs are discussed in Veanes *et al.* (2010) and Olston *et al.* (2009).

3.6 Explaining data differences and evolution

Explanations typically target something surprising that deviates from some norm or expectation. Certain problem settings may allow for

The version of record is available at: http://dx.doi.org/10.1561/1900000074

the specification of these deviations in a precise manner, usually by contrasting a normal and an abnormal pattern of data or behavior. For example, a value or pattern may appear deviant in comparison with another observed value or pattern. This can arise in many settings: different datasets or analyses may derive different results, time series data may show a significant pattern change, or data evolution over time may result in significant value changes that merit explanation. While the element we seek to explain may still be a tuple or a set of tuples, the fact that the source of the discrepancy is the difference or comparison with another tuple or pattern can help guide explanations more precisely.

3.6.1 Data differences

The explanation literature has often focused on some crude measures for labeling an element, e.g., a tuple, as explanation-worthy, such as indicating that its value is surprisingly low or high. This typically alludes to a tuple being an outlier, deviating from a general trend within a dataset; it does not usually refer to a direct comparison with a particular different value. But if this comparison is made more explicit and precise, it can steer explanations to target the particular discrepancy more explicitly.

A situation where this direct contrast of values or results is possible and effective is when the same analysis or query is performed over different datasets. Data management research on explanations has focused on the assumption that data resides in a single dataset, under one common schema. But the reality of today's data diverges from that ideal. More often than not, datasets evolve separately, under different schemas, and even datasets from trustworthy sources frequently end up diverging, both in format and content, causing headaches to downstream applications and users. While datasets may be related and overlapping, their separate production and evolution can lead to disagreements, even when datasets come from trustworthy sources. A case study of this problem is summarized in Example 2.4. In this example, a query over the data from the National Center for Education Statistics, a curated and trustworthy source, suggests a number of majors provided by the

The version of record is available at: http://dx.doi.org/10.1561/1900000074

University of Massachusetts Amherst that vastly disagrees with the UMass database.

When different datasets provide different answers to semantically similar questions, understanding the reasons for the discrepancies is challenging and cannot be handled by single-dataset solutions. The key insight within this setting is that the underlying assumption of similarity between two datasets can supply a lot of additional information on where the significant differences lie. There are two key considerations in modeling this setting: (1) how to encode this similarity, and (2) what a proper explanation looks like in this setting. These two considerations are naturally linked, as the model for encoding similarity can serve as the medium to highlight the differences.

A model-based approach

Explain3D (Wang and Meliou, 2019) models the similarity across datasets using traditional schema mapping techniques. Specifically, traditional mapping techniques can provide an initial probabilistic mapping $\mathcal{M}$ between the tuples of two datasets, D_1 and D_2. When two queries Q_1 and Q_2 over D_1 and D_2, respectively, are expected to produce the same result, but they do not, we can use their expected agreement to refine the initial mapping $\mathcal{M}$. Specifically, the goal is to identify the most likely deterministic mapping $\mathcal{M}^*$ given the queries' provenance and the initial mapping $\mathcal{M}$. Intuitively, a particular mapping pinpoints specific discrepancies; given the expectation of agreement between two results and the principle of Occam's razor, a mapping that indicates the fewest discrepancies is preferred.

Ultimately, the derived mapping can serve as explanation, or it can be further processed to derive more high-level explanations. But the main point here is that the encoding of the expected agreement (the tuple mapping in Explain3D) carries the information to highlight the observed disagreement. This intuition appears fundamental and we expect generalizes to many explanation settings: if the expected agreement or pattern can be encoded in sufficient detail, this encoding can serve as a pattern for modeling and deriving appropriate explanations.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Summarizing differences

The *DIFF* operator (Abuzaid *et al.*, 2021) used in *Macrobase* (Abuzaid *et al.*, 2018) that we already discussed in Section 3.4.3 summarizes the differences between two datasets using patterns. While technically, this approach does not require the two datasets to be compared have the same schema, it is still not suited very well for generating explanations that explain the difference between the two schema of the input instead of just the data. *DataDiff* (Yilmaz *et al.*, 2018) computes a summary of the difference between two datasets that includes transformations such as deleting all rows matching a predicate (pattern) and or adding or removing attributes.

3.6.2 Changing patterns in time series

A natural setting where *change* takes a central role is in time series and streaming data. Such datasets typically represent evolving phenomena—environmental monitoring, resource use, supply chains—and monitoring applications seek to find, and potentially explain, interesting patterns. What makes a pattern interesting, is a deviation from an expected norm, e.g., a sudden spike in resource use, or high temperature measurements during the winter. Instead of isolating the deviation and describing it crudely (e.g., why is the resource use / temperature high), we gain more information by specifying it more precisely, through direct comparison with some reference data. For example, an interval of a stream can be labeled as anomalous with respect to another *reference* interval. Providing an explicit comparison point can target the explanations more precisely as the causes of deviation between the anomaly and the reference.

A solution to this setting again needs to consider the encoding of the differences to specify an explanation format. EXStream (Zhang *et al.*, 2017) models stream intervals in a feature space; for example, in a resource monitoring application, relevant features can include the mean free memory during a time interval, the swap space, etc. Then, the explanation is also feature-based: a conjunction of features that best represents the differences between two intervals. In EXStream, the optimal explanation is defined through a submodular optimization problem

The version of record is available at: http://dx.doi.org/10.1561/1900000074

relying on an entropy-based distance function, but other definitions of optimality are possible.

3.7 Explaining query performance

A different type of explanations that has appeared in the data management literature targets query performance, as illustrated by the examples in Section 2.2.3. There is a large body of work in the literature using very different techniques for cluster monitoring, performance debugging, root cause diagnosis, configuration recommendation, etc.; we mention a few tools here and refer the reader to the corresponding papers and the references therein for a detailed study.

The tool iQCAR (inter-Query Contention Analyzer), proposed in Kalmegh *et al.* (2019) (see Example 2.10) attributes blame for the slowdown of a query to concurrent queries using *blocked times* (Ousterhout *et al.*, 2015), time a task is blocked for resources like CPU, network, memory, or IO used concurrently by other contentious queries, and helps an admin understand why a query is slow in an execution which might take hours of effort to do manually. Another explanation tool for debugging performance in cluster computing is *PerfXplain* (Khoussainova *et al.*, 2012) that uses a decision-tree approach to provide explanations for slowdown of MapReduce jobs comparing multiple executions. Here the users can specify the expected and observed performance of pairs of MapReduce jobs as well as a *despite* clause which captures how similar the jobs are, and the system outputs top explanations based on metrics like relevance, precision, and generality. EXStream (Zhang *et al.*, 2017) similarly contrasts performance metrics across executions, in the setting of event stream monitoring. The methodology in this framework is based on solving a submodular optimization problem to identify combinations of high-level features, such as mean free memory during a time interval, the swap space, etc., that best explain the divergence in performance. There are several root-cause diagnosis tools (Dias *et al.*, 2005; Borisov *et al.*, 2009; Yoon *et al.*, 2016; Ousterhout *et al.*, 2015; Roy *et al.*, 2015b), as well as tools for *cluster monitoring*, such as Ganglia (*Ganglia Monitoring System* 2019), Spark UI (*Spark Monitoring and Instrumentation* 2019), and Ambari (*Apache Ambari* 2019), that provide query metrics.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Finally, *configuration monitoring* tools, such as Starfish (Herodotou *et al.*, 2011), Dr. Elephant (*Dr. Elephant* 2019), and OtterTune (Van Aken *et al.*, 2017), analyze performance and suggest changes in configuration.

3.8 Explainable AI and Machine Learning

Machine learning techniques are increasingly being used to make decisions that have significant real-world impact, such as determining mortgage rates and even to recommend criminal sentences. Therefore, providing human-understandable explanations of such automated decisions and processes is now a critical need to ensure transparency, troubleshooting, and fairness. Explainable AI (XAI) is a new, active field that aims to address explainability challenges in these systems (e.g., Lundberg and Lee, 2017; Ribeiro *et al.*, 2018; Rudin, 2019; Barredo Arrieta *et al.*, 2020). This is a vast and fast moving research area, and it is beyond the scope of this article to provide an exhaustive overview. Instead, we will focus the discussion on a selected set of techniques rooted in data management or approaches that we deem interesting to data management researchers.

One of the most prevalent machine learning tasks is classification. Complex models such as deep neural networks have been demonstrated to excel in classification tasks. A commonly cited disadvantage of using such complex models is that these models are quite opaque and, thus, it is hard to understand why a test data point is assigned to a particular class. A frequently studied XAI problem is: a classifier f was used to classify a test data point $\mathbf{x}^*$ and the user wants to understand the classification outcome $f(\mathbf{x}^*)$; the same question can arise when an ML model is used for estimation instead of classification.

3.8.1 Explaining Model Predictions through Feature Attribution

One possible type of explanations for an unexpected prediction outcome is to identify which features of the test data point $\mathbf{x}^*$ are responsible for the prediction. This class of techniques is often referred to as *feature attribution* methods, because they attribute responsibility for a prediction outcome to features of the data point $\mathbf{x}^*$ (possibly also

The version of record is available at: http://dx.doi.org/10.1561/1900000074

considering related data points). Methods in this category use a variety of techniques to determine the responsibility of a feature. Note that the two approaches we discuss here (Shapley values and causality) were already discussed in Section 3.3.1, but for a different purpose: refining provenance by computing the degree of responsibility each data item in the provenance has for an observed query result.

One common approach is to model the prediction as a co-operative game where the players are the input features and the goal is to determine a fair attribution of the payoff of the game (the prediction outcome) to the players. Shapley values can be used to determine the contribution of each feature (Shapley, 1953; Lundberg *et al.*, 2018; Merrick and Taly, 2019; Frye *et al.*, 2020; Aas *et al.*, 2019). Shapley values have attractive theoretical properties. However, there are several possible ways of how the co-operative game modeling the prediction task can be designed that can significantly affect the generated explanations (Merrick and Taly, 2019). Consider a prediction task with M input features $\mathcal{X} = \{X_j\}$ and an input point $\mathbf{x}^*$. The Shapley value for a feature X_j with respect to $\mathbf{x}^*$, denoted as ϕ_j attributes to X_j part of the difference between $f(\mathbf{x}^*)$ and the expected prediction of the model over all data points: $\phi_0 = E[f(\mathbf{x})]$ such that:

$$f(\mathbf{x}^*) = \phi_0 + \sum_{j=1}^{M} \phi_j$$

The Shapely value ϕ_j for feature X_j is computed by calculating the mean over all subsets $\mathcal{S}$ of $\mathcal{X} - \{X_j\}$ of the difference between the expected prediction of the classifier for all points that agree with $\mathbf{x}^*$ on $\mathcal{S}$ and all points that agree with $\mathbf{x}^*$ on $\mathcal{S} \cup \{X_j\}$:

$$\phi_j = \sum_{\mathcal{S} \subseteq \mathcal{X} - \{X_j\}} \frac{|\mathcal{S}|!\,(M - |\mathcal{S}| - 1)!}{M!} \Big(v(\mathcal{S} \cup \{X_j\}) - v(\mathcal{S}) \Big)$$
$$v(\mathcal{S}) = E[f(\mathbf{x}) | \mathbf{x}_{\mathcal{S}} = \mathbf{x}^*{}_{\mathcal{S}}]$$

Shapley values have also been applied to measure global importance of features across all data points (Owen and Prieur, 2017). Computing Shapley values is exponential in the number of features and, thus,

The version of record is available at: http://dx.doi.org/10.1561/1900000074

intractable for more than a few features. Several approximate versions have been proposed, e.g., most of the approaches cited above rely on some approximation scheme. Note that the use of Shapely values is not just restricted to explaining the impact a feature has on the outcome of a prediction; In fact, it can be applied to explain the impact an input feature has on the result of any type of blackbox function. For instance, Shapley values have been used to explain data repairs with respect to a set of denial constraints for a large class of data repair algorithms (Deutch *et al.*, 2020).

An alternative class of algorithms produces *contrastive and causal explanations* that identify *interventions*, i.e., perturbations of the test data point or of subset of the test data, that change the prediction for a data point of interest. *Contrastive and causal* XAI methods explain ML model predictions in terms of minimal *interventions* on input features that change the prediction (Bertossi *et al.*, 2020; Datta *et al.*, 2016; Verma *et al.*, 2020; Wachter *et al.*, 2017; Laugel *et al.*, 2018; Karimi *et al.*, 2020; Ustun *et al.*, 2019; Mahajan *et al.*, 2019; Mothilal *et al.*, 2020; Ying *et al.*, 2019; Galhotra *et al.*, 2021). For instance, Bertossi *et al.* (2020) introduce an explanation score for features based on counterfactual causality and responsibility. Consider a binary classifier f and a data point $\mathbf{x}^*$ for which $f(\mathbf{x}^*) = 1$. An intervention that changes $\mathbf{x} = \mathbf{x}^*[X_i = v]$, i.e., changing the value of feature X_i to v in data point $\mathbf{x}^*$ is called a *counterfactual cause* for the prediction outcome $f(\mathbf{x}^*)$ if $f(\mathbf{x}) = 0$. That is, counterfactual causes change the prediction outcome. Based on this notion, Bertossi *et al.* (2020) introduce the COUNTER-score for each feature X_i of $\mathbf{x}^*$ that is defined as the difference between the prediction at $\mathbf{x}^*$ and the expectation of the prediction of $f(\mathbf{x})$ over points $\mathbf{x} = \mathbf{x}^*[X_i = v]$ for some v. This is the expectation of the change in prediction over all interventions on X_i:

$$\text{COUNTER}(\mathbf{x}^*, X_i) = f(\mathbf{x}^*) - E[f(\mathbf{x}) \mid \mathbf{x}_{\mathcal{X}-\{X_i\}} = \mathbf{x}^*{}_{\mathcal{X}-\{X_i\}}]$$

As discussed in Section 3.4.1, approaches based on intervention and causality have been successfully applied for explaining outcomes of aggregate queries and to refine provenance information by associating data with a degree of responsibility instead of using a binary decision

The version of record is available at: http://dx.doi.org/10.1561/1900000074

(in the provenance or not). In summary, both Shapley values and counterfactual causality are principled techniques for explaining the result of a blackbox function by assigning responsibility or blame to input features.

Yet another line of work (Sagadeeva and Boehm, 2021; Chung *et al.*, 2019) utilizes pattern-based summaries as described in Section 3.4.3 to compactly describe a subset of the test data on which the classifier performs poorly.

3.8.2 Explaining Black-box Predictions Using Interpretable Models

An alternative to explaining a prediction based on input features, is to compute a surrogate interpretable model g (e.g., a decision tree or linear classifier) that *approximates* the blackbox model f (e.g., a deep neural network) in the vicinity of the data point $\mathbf{x}^*$. The intuition is that by using a simpler, but more interpretable model, the opaque behavior of f with respect to $\mathbf{x}^*$ and points similar to $\mathbf{x}^*$ can be explained to a user.

LIME (Ribeiro *et al.*, 2016) is one of these approaches that explains a blackbox classifier f's prediction for a data point by locally approximating it though an interpretable model. In this work, the search for such an interpretable model is cast as an optimization problem: find the model g from a set of possible models G that minimizes the sum of the error of the prediction for the points in the local neighborhood (determined by a distance function π_x) of the data point $\mathbf{x}^*$ of interest and the complexity of the model $\Omega(g)$ (e.g, depth of a decision tree):

$$\underset{g \in G}{\operatorname{argmin}} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

To improve the efficiency of this process Ribeiro *et al.* (2016) computes the loss over a random sample of the training data that is biased towards points that are close to $\mathbf{x}^*$ with respect to π_x. Other representative instances of the idea of explaining outcomes through surrogate models are Ribeiro *et al.* (2018) Lundberg and Lee (2017). To the best of our knowledge, the idea of explaining a complex function through a simpler function that locally approximates the complex function has

The version of record is available at: http://dx.doi.org/10.1561/1900000074

not been applied for explaining query results. It would be interesting to investigate whether this idea translates to explanations for query answers.

An alternative to finding interpretable models is to directly train models that are more interpretable (Rudin, 2019). For instance, Letham *et al.* (2015) and *LIBRE* (Mita *et al.*, 2020) learn classifiers that are built from simple conditional rules.

3.8.3 Explanations based on Training Data

The two techniques discussed so far have in common that they explain the model's behavior, but do not provide any insight into why the model is exhibiting this behavior. To answer this question, we need to dig deeper and analyze how the training data affects the model trained over this data and, thus, indirectly affects the outcome of prediction made by the model. Methods following this paradigm, explain a prediction for a data point or other property of the model (e.g., the prediction performance of the classifier or whether it is biased against a protected group) by attributing responsibility for the prediction to training data points.

The general techniques discussed in Section 3.8.1 are also applicable to generating explanations based on training data. For instance, an intervention based on training data may be modeled as deleting a subset D_{inter} of the training data D. The effect this intervention has on the prediction outcome then can be determined by retraining the classifier over $D - D_{inter}$. Let us use f_{inter} to denote this classifier. The effect D_{inter} has on the prediction of data point $\mathbf{x}^*$ is then the difference between $f_{inter}(\mathbf{x}^*)$ and $f(\mathbf{x}^*)$.

One line of work has applied Shapley values for this purpose. Data Shapley (Ghorbani and Zou, 2019) assigns value to (sets of) training data points based on the their contribution to the performance of a classifier over a test dataset. Distributional Shapley (Kwon *et al.*, 2021; Ghorbani *et al.*, 2020) extends Data Shapley to take into account that the training data is only a sample from an unknown underlying distribution.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Recall that computing measures like Shapley values for feature attribution is already computationally infeasible because of their combinatorial nature. When computing such measures based on interventions to the training data, this additionally requires retraining the classifier over subsets of the training data. To address this issue, several techniques have been developed to approximate the effect of removing or updating a training data point. A popular technique is based on *influence functions* which have a long history in statistics (Cook and Weisberg, 1982). Koh and Liang (2017) propose the use of influence functions for explaining how training data affects the predictions made by a model. While the quality of the approximation produced by influence functions is typically good for predicting the influence of single data points, the error increases for larger subsets of the training data. Basu *et al.* (2019) propose using second order approximations instead. This reduces the approximation error for sets. Influence functions have also been explored in the context of explanations for queries. For instance, Wu *et al.* (2020) use influence functions to find training data points that are responsible for an error in a query result where the query has access to a model trained over that data. Making further connections between database techniques/explanations in databases and XAI will be an interesting direction to explore.

3.9 Other related topics

Beyond the work on explaining query answers and outliers, query formulation, data differences and transformation, and query performances, several other approaches exist to get useful insights from data and processes. Explaining interactions with a data-driven system is inherently related to visualizations, as most of the explanation systems are likely to be interactive with suitable user interfaces. There is a line of work on *summarizing relational data or query answers* often with interactive exploration in mind, which focuses on *diversity, relevance, and/or coverage*, to give the maximum interesting information to users while inspecting a much shorter summary (Vieira *et al.*, 2011; Qin *et al.*, 2012; Drosou and Pitoura, 2012; Joglekar *et al.*, 2017; Wen *et al.*, 2018; Kim *et al.*, 2020). On the other hand, systems like ZenVisage (Siddiqui *et al.*, 2016) or

The version of record is available at: http://dx.doi.org/10.1561/1900000074

DeepEye (Luo *et al.*, 2018) help users explore large datasets by automatically *generating and recommending interesting visualizations*. A related topic is *explaining recommendations* that tries to address 'why' questions to users or system designers regarding why certain items are recommended by the algorithm or the models used in recommendation systems (e.g, see the article by Zhang and Chen, 2020 and references therein).

In addition to general explanations, application-specific explanations have also been studied in the literature. Das *et al.* (2011) and Thirumuruganathan *et al.* (2012) studied the problem of *Meaningful Ratings Interpretation (MRI)* based on the idea of data cube to help a user easily interpret ratings of items on Yelp or IMDb; e.g., instead of simply giving the average rating of a movie, MRI attempts to give explanations like "male reviewers under 30 from NYC love this movie". Barman *et al.* (2007) and Agarwal *et al.* (2007) studied explaining changes in hierarchical summaries in data warehouses identifying optimally parsimonious explanations. Fabbri and LeFevre (2011) and Bender *et al.* (2014) studied explanation-based auditing of access logs. Ré and Suciu (2008) and Kanagal *et al.* (2011) studied the problem of computing the top-k influential variables (the input variables that can significantly modify the output probabilities) and top-k explanations (to answer questions like "why a tuple is in the output" and "why the probability of an output tuple is greater than another one") for conjunctive queries on probabilistic databases; Abiteboul *et al.* (2014) studied a similar problem for recursive datalog queries.

The need for explanations also arises in error detection, data repair, and data integration, where the users needs to understand the result produced by a semi-automated or automated data curation algorithms or pipeline. Chalamalla *et al.* (2014) generate explanations for violations to a set of integrity constraints defined over a query result as patterns that describe subsets of the input data which are responsible for the violation. The QFix system (Wang *et al.*, 2017) explains errors in a dataset generate through a sequence of updates by identifying which updates are responsible for the error. Most data cleaning and integration techniques use heuristics to select one solution for repair or integration task, because typically there is insufficient information available for uncovering the unknown ground truth solution. For instance, when

The version of record is available at: http://dx.doi.org/10.1561/1900000074

imputing missing values (Stekhoven and Bühlmann, 2012), the correct value to be substituted for a missing value is unknown. One possibility to help a user to understand how to choices made by the cleaning algorithm affect their result is to treat the alternative choices available to the algorithm as a probabilistic or incomplete database and using uncertain query processing to propagate this information through queries (Beskales *et al.*, 2009; Yang *et al.*, 2015). This idea was first popularized in consistent query answering (Bertossi, 2011). The Vizier system (Feng *et al.*, 2021; Brachmann *et al.*, 2019) visualizes such uncertainty in its provenance and explains which analysis results are trustworthy and why.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

4

A Research Roadmap

Explanations in data systems is a relatively young field that is enjoying continued interest and a steady presence in data management venues. Our goal has been to identify and summarize general principles in the specification of explanation problems ("Who", "Why", and "What"), and high-level objectives in designing explanations and methodologies for deriving them ("How"). To conclude this article, we identify gaps in the existing work, and potential ripe ground for next directions to pursue, some being more open-ended compared to the others.

Causal explanations

Many of the examples we discussed in the previous chapters implicitly assume that the ideal explanation would be one or more *causes* of the observation; e.g., in Example 2.3, we would ideally like to find the *cause* that led to the slowdown of publications from industry in SIGMOD in recent years. *Causality* and the related concept of *intervention* have been used in various contexts in database research for explanations, borrowing concepts from the study of causal analysis in AI (Pearl, 2000), e.g., in Meliou *et al.*, 2010, Roy and Suciu, 2014, and Fariha *et al.*, 2020. However, the study of *true cause* as explanations, e.g., 'how

The version of record is available at: http://dx.doi.org/10.1561/1900000074

much smoking can be attributed to the cause of lung cancer' or 'how much a new vaccine can be attributed to a slowdown of the spread of a contagious disease', has not been explored much in the database community (recently, Salimi *et al.* (2020) proposed a framework for causal analysis in relational databases).

As is commonly noted, *'correlation is not causation'*; thus, answering such causal questions, which goes beyond inferring correlation or predictions, is critical for making principled and informed decisions in clinical research, epidemiology, economics, and social sciences. The notion of causal inference has been studied not only in AI (by Pearl's Graphical Causal Model (Pearl, 2000)), but also for several decades in Statistical Science (e.g., by Rubin-Neyman's potential Outcome Model (Rosenbaum and Rubin, 1983; Rubin, 2005)). In general, *randomized controlled trials* (such as clinical trials for medical research), where the participating *units* are randomly divided into *treatment and control* groups, and the average difference of outcome in these two groups is computed as the *'average treatment effect'*, is considered the gold standard for causal analysis. However, due to cost or ethics concerns, one often needs to do *observational causal studies* using observed data under various (untestable) assumptions as studied in the above models. This connects observational causal studies with explanations for data and queries in database research, since many explanations settings start with an observed/collected dataset.

However, there are several challenges that need to be addressed to adapt standard causal analysis techniques to explanation questions related to data processing. A significant difference is that typical causal studies consider a single and flat table with information regarding units (treatment, outcome, other attributes or confounding covariates that should be conditioned on, etc.), while in relational databases and data lakes there can be several tables / datasets (e.g., Author-Pub-Authored tables in Example 3.1). Further, a critical assumption is that there is 'no interference' among units stored in this table (treatment assigned to one unit cannot affect the outcome of another unit); in contrast, in relational databases and data processing in general, multiple tables are related to each other with common attributes, foreign keys, many-to-many join patterns, and intricate integrity constraints. In addition, there can

The version of record is available at: http://dx.doi.org/10.1561/1900000074

be derived data or 'views' generated by complex queries or analysis pipelines in the causal question, and the 'treatment' and 'outcome' attribute can appear in different tables.

A further complication is that datasets in typical statistical studies are much smaller than the 'big data' commonly encountered in data analytics, making sound yet *scalable* causal analysis a practical challenge (it has been observed that SQL queries can make standard causal analysis approaches much more scalable (Wang *et al.*, 2021)). Finally, there might not be an obvious translation of many explanation questions in database research to the causal analysis framework in terms of 'treatment' and 'outcome'. Strengthening the connections between causal analysis and explanations for data-driven systems, borrowing concepts from Statistics, AI, and ML, is a relatively new but important research direction.

Confidence in explanations

The various approaches discussed in Chapter 3 propose ways to navigate the search space to derive explanations that optimize some desired properties (summarized in Section 3.1). In Section 3.4 in particular, we discussed scoring functions that rank possible explanations in the search space. Most of these methods aim to return best possible explanations based on the available data. However, existing approaches do not delve deep into the question of whether the input dataset even has enough information to provide meaningful explanations (e.g., in the example of Section 3.4.2, fewer than usual publications by an author in a venue in a year may be explained by a sabbatical, involvement with a startup, or more engagement in administrative duties, but this information is unavailable in the DBLP publication database), or how likely it is for the returned explanations to be 'true' explanations for the questions posed by the user. With this in mind, it is important to consider whether we can provide a *confidence measure* for the returned explanations. The lack of a training dataset or gold standard for explanations adds to the challenges of this problem. Collaboration with domain experts, integration with other datasets, and integration and cross-reference with 'common knowledge' known to humans (e.g., using knowledge bases, such as YAGO (Suchanek *et al.*, 2007) or NELL (Mitchell, 2012), or by web crawling) can be beneficial for this purpose.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Explaining data evolution

Data is often dynamic and subject to change. In fact, constants are not generally interesting in data analysis; what analysis tends to focus on is understanding how and why something in the data changes. As we discussed in Section 3.6, to explain a change more precisely, it is best to specify the change succinctly (e.g., in reference to other data). Existing frameworks however, do not explicitly account for the cases where diverging datasets may have a common original source, or simply a dataset may be a later, evolved copy of another. Leveraging evolution aspects of the data such as schema restructuring, and broad data updates can lead to better targeted explanations.

Syntactic vs semantic explanations

Existing work on explaining data differences has come up with different explanation models, but these models often focus on *syntactic* differences, i.e., low-level, fine-grained differences between two reference elements. For example, to explain the differences between two diverging query results, Explain3D (Wang and Meliou, 2019) attempts to find minimal modifications to an initial data mapping from one dataset to the other. The resulting mapping is an explanation, but it is too low-level, specifying divergence tuple-by-tuple. The ultimate explanation to be served to the user should often be something more high-level; e.g., the number of majors derived from two educational datasets differs because one dataset does not include associate degrees. These high-level explanations are *semantic* and provide a more holistic view of the difference. While such semantic explanations can often be derived from syntactic explanations through summarization techniques, a key observation is that explanation desiderata are optimized over the low-level syntactic explanations, rather than the final semantic explanation. This gap potentially indicates an opportunity for improving explanation products, by targeting the optimization solutions directly on them. While our example focused on the explanation of differences, this observation generalizes to all explanation settings.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Flexible explanations

In our classification, we discussed explanation desiderata as driven by the dimensions of "Who", "Why", and "What" of the problem specification. For example, we noted that explanations for the purposes of debugging need to be more low-level and detailed than those meant to enhance understandability for a non-technical audience. However, some settings need to accommodate a variety of users and purposes, requiring explanation frameworks to flexibly adapt to differing needs. Such adaptive frameworks could support navigating explanations at different "zoom" levels, and potentially switch among different explanation types for the same problem.

Explainable explanations

Explanations are an interpretive tool, helping data users better understand and explore their data. As an element meant primarily for human consumption, explanations bring forth interactivity challenges that have not been deeply explored. Most existing works focus on the basic action of seeking explanations, but more complex interactions may be necessary for users to achieve their analysis goals. Users may wish to refine, explore, and analyze explanation results, essentially imposing explainability requirements on the explanations themselves. Just as one places more trust on observations that can be explained, *explainable explanations* are likely to enhance the confidence that one places on the explanations themselves, which, in turn, will lead to more trust on the data or observations. For example, if a framework returns a feature-based explanation such as `[(zipcode = 01003) AND (age<22)]` for the question *"why was there a spike in COVID cases in Hampshire county in MA, during February 2021?"*, this can give rise to new questions on the explanation itself. For example, a user may wish to understand *"why is this the best explanation?"* or *"why is the explanation `[(zipcode = 01003)]` not sufficient?"* To answer such questions, the framework would need to produce evidence to illustrate how the explanation was derived, and why other options are suboptimal. Ultimately, explanation support may need to extend to variable lengths of recursive investigation, which in turn would likely augment

The version of record is available at: http://dx.doi.org/10.1561/1900000074

or alter the explanation objectives and potentially point to particular methodologies as more amenable to explainability.

Other directions

Aside from these broader research directions, there are many possibilities for improvements over existing approaches, both in terms of functionality and efficiency. For example, intervention-based approaches (Section 3.4.1) have primarily focused on tuple deletions as the mode of explanation, and deterministic and well-defined interventions. Modeling meaningful additions or updates, while preserving the semantics of the data, will require creative solutions. Another possibility is studying stochastic interventions to model interdependencies among tuples when changes are made to the database; e.g., in the context of Example 3.1 if an author is removed, a possible intervention model is that with some probability each paper written by this author should be removed from the database, and this probability might be a function of properties (like contributions or seniority) of all authors of that paper. As another example, showing a small counterexample or illustrating how a "wrong" query can be repaired (Section 3.5) offers useful intuition to students or new programmers, but focusing on a particular counterexample or repair method may also be misleading as they can miss the high-level properties of the desired queries or draw attention to only one type of errors. Based on the context ("What" and "Why") and the intended users ("Who"), it is important to study the best possible representation of explanations as an integrated research agenda for all explanation questions.

Finally, the big question of '*what is an explanation*' is closely related to philosophy (e.g., Aristotle's *'Four Causes'*, or Hempel's *'Deductive-nomological model'*) and has been studied for centuries. Exploring such notions of explanations, establishing close collaborations with domain experts, cognitive scientists, experts in human-computer interaction, and education specialists, exploring interesting practical applications and designing meaningful user studies, should be consistent pillars in our research efforts to establish the efficacy of different notions of explanations for data-driven systems.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Acknowledgements

This work was partially supported by the National Science Foundation under grants IIS-1453543, IIS-1552538, OAC-1640864, IIS-1703431, CCF-1763423, IIS-1943971, IIS-1956123, and IIS-2008107, and by the National Institutes of Health under grant R01EB025021.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

References

Aas, K., M. Jullum, and A. Løland. (2019). "Explaining Individual Predictions When Features Are Dependent: More Accurate Approximations To Shapley Values". *CoRR*. abs/1903.10464. arXiv: 1903.10464. URL: http://arxiv.org/abs/1903.10464.

Abiteboul, S., D. Deutch, and V. Vianu. (2014). "Deduction with Contradictions in Datalog". In: *Proc. 17th International Conference on Database Theory (ICDT), Athens, Greece, March 24-28, 2014*. Ed. by N. Schweikardt, V. Christophides, and V. Leroy. OpenProceedings.org. 143–154. DOI: 10.5441/002/icdt.2014.17.

Abiteboul, S., R. Hull, and V. Vianu. (1995). *Foundations of Databases: The Logical Level*. 1st. USA: Addison-Wesley Longman Publishing Co., Inc.

Abuzaid, F., P. Bailis, J. Ding, E. Gan, S. Madden, D. Narayanan, K. Rong, and S. Suri. (2018). "Macrobase: Prioritizing Attention in Fast Data". *ACM Trans. Database Syst.* 43(4): 15:1–15:45. DOI: 10.1145/3276463.

Abuzaid, F., P. Kraft, S. Suri, E. Gan, E. Xu, A. Shenoy, A. Ananthanarayan, J. Sheu, E. Meijer, X. Wu, J. F. Naughton, P. Bailis, and M. Zaharia. (2021). "DIFF: a Relational Interface for Large-Scale Data Explanation". *VLDB J.* 30(1): 45–70. DOI: 10.1007/s00778-020-00633-6.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Agarwal, D., D. Barman, D. Gunopulos, N. E. Young, F. Korn, and D. Srivastava. (2007). "Efficient and effective explanation of change in hierarchical summaries". In: *KDD*. San Jose, California, USA: ACM. 6–15. DOI: 10.1145/1281192.1281197.

Agrawal, R. and R. Srikant. (1994). "Fast Algorithms for Mining Association Rules in Large Databases". In: *Proceedings of the 20th International Conference on Very Large Data Bases. VLDB '94*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 487–499.

Ahmad, Y., O. Kennedy, C. Koch, and M. Nikolic. (2012). "DBToaster: Higher-order Delta Processing for Dynamic, Frequently Fresh Views". *Proc. VLDB Endow.* 5(10): 968–979. DOI: 10.14778/2336664.2336670.

Alexe, B., L. Chiticariu, and W. Tan. (2006). "SPIDER: a schema mapPIng DEbuggeR". In: *Proceedings of the 32nd international conference on Very large data bases*. VLDB Endowment. 1179–1182.

Amsterdamer, Y., D. Deutch, and V. Tannen. (2011). "Provenance for Aggregate Queries". In: *PODS '11*. Athens, Greece: Association for Computing Machinery. 153–164. URL: https://doi.org/10.1145/1989284.1989302.

"Apache Ambari". (2019). http://ambari.apache.org.

Barman, D., F. Korn, D. Srivastava, D. Gunopulos, N. E. Young, and D. Agarwal. (2007). "Parsimonious Explanations of Change in Hierarchical Data". In: *ICDE*. 1273–1275. DOI: 10.1109/ICDE.2007.368991.

Barredo Arrieta, A., N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. (2020). "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". *Information Fusion*. 58: 82–115. DOI: https://doi.org/10.1016/j.inffus.2019.12.012.

Basu, S., X. You, and S. Feizi. (2019). "Second-Order Group Influence Functions for Black-Box Predictions". *CoRR*. abs/1911.00418. arXiv: 1911.00418. URL: http://arxiv.org/abs/1911.00418.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Bender, G., L. Kot, and J. Gehrke. (2014). "Explainable Security for Relational Databases". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. SIGMOD '14*. Snowbird, Utah, USA: Association for Computing Machinery. 1411–1422. DOI: 10.1145/2588555.2593663.

Berenson, H., P. Bernstein, J. Gray, J. Melton, E. O'Neil, and P. O'Neil. (1995). "A critique of ANSI SQL isolation levels". *SIGMOD Record*. 24(2): 1–10. DOI: 10.1145/223784.223785.

Bertossi, L. (2011). "Database Repairing and Consistent Query Answering".

Bertossi, L. E., J. Li, M. Schleich, D. Suciu, and Z. Vagena. (2020). "Causality-based Explanation of Classification Outcomes". In: *Proceedings of the Fourth Workshop on Data Management for End-To-End Machine Learning, In conjunction with the 2020 ACM SIGMOD/PODS Conference, DEEM@SIGMOD 2020, Portland, OR, USA, June 14, 2020*. Ed. by S. Schelter, S. Whang, and J. Stoyanovich. New York, NY, USA: ACM. 6:1–6:10. DOI: 10.1145/3399579.3399865.

Beskales, G., M. Soliman, I. Ilyas, and S. Ben-David. (2009). "Modeling and querying possible repairs in duplicate detection". *Proceedings of the VLDB Endowment*. 2(1): 598–609. DOI: 10.14778/1687627.1687695.

Bidoit, N., M. Herschel, and A. Tzompanaki. (2015). "Efficient Computation of Polynomial Explanations of Why-Not Questions". In: *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*. Ed. by J. Bailey, A. Moffat, C. C. Aggarwal, M. de Rijke, R. Kumar, V. Murdock, T. K. Sellis, and J. X. Yu. ACM. 713–722. DOI: 10.1145/2806416.2806426.

Bidoit, N., M. Herschel, K. Tzompanaki, *et al.* (2014). "Query-Based Why-Not Provenance with NedExplain". In: *Extending Database Technology (EDBT)*. DOI: 10.5441/002/edbt.2014.14.

Bidoit, N., M. Herschel, and K. Tzompanaki. (2016). "Refining SQL Queries based on Why-Not Polynomials". In: *8th USENIX Workshop on the Theory and Practice of Provenance (TaPP 16)*. Washington, D.C.: USENIX Association.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Borisov, N., S. Babu, S. Uttamchandani, R. Routray, and A. Singh. (2009). "Why Did My Query Slow Down?" *arXiv preprint arXiv:0907.3183*.

Brachmann, M., C. Bautista, S. Castelo, S. Feng, J. Freire, B. Glavic, O. Kennedy, H. Müeller, R. Rampin, W. Spoth, and Y. Yang. (2019). "Data Debugging and Exploration with Vizier". In: *Proceedings of the 2019 International Conference on Management of Data. SIGMOD '19*. Amsterdam, Netherlands: Association for Computing Machinery. 1877–1880. DOI: 10.1145/3299869.3320246.

Buneman, P., S. Khanna, and W. C. Tan. (2001). "Why and Where: A Characterization of Data Provenance". In: *Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings*. Ed. by J. V. den Bussche and V. Vianu. Vol. 1973. *Lecture Notes in Computer Science*. Springer. 316–330. DOI: 10.1007/3-540-44503-X_20.

Calvanese, D., G. D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati. (2007). "Ontology-based Database Access". In: *Proceedings of the Fifteenth Italian Symposium on Advanced Database Systems, SEBD 2007, 17-20 June 2007, Torre Canne, Fasano, BR, Italy*. Ed. by M. Ceci, D. Malerba, and L. Tanca. 324–331.

Cate, B. ten, C. Civili, E. Sherkhonov, and W.-C. Tan. (2015). "High-level why-not explanations using ontologies". In: *Proceedings of the 34th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. ACM. New York, NY, USA: Association for Computing Machinery. 31–43. DOI: 10.1145/2745754.2745765.

Ceri, S. and J. Widom. (1991). "Deriving Production Rules for Incremental View Maintenance". In: *Proceedings of the 17th International Conference on Very Large Data Bases. VLDB '91*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 577–589.

Chalamalla, A., I. F. Ilyas, M. Ouzzani, and P. Papotti. (2014). "Descriptive and Prescriptive Data Cleaning". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. SIGMOD '14*. Snowbird, Utah, USA: Association for Computing Machinery. 445–456. DOI: 10.1145/2588555.2610520.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Chandra, B., B. Chawda, B. Kar, K. V. M. Reddy, S. Shah, and S. Sudarshan. (2015). "Data generation for testing and grading SQL queries". *VLDB J.* 24(6): 731–755.

Chapman, A. and H. V. Jagadish. (2009). "Why Not?" In: *SIGMOD '09: Proceedings of the 35th SIGMOD International Conference on Management of Data*. 523–534.

Cheney, J. (2007). "Program Slicing and Data Provenance". *IEEE Data Engineering Bulletin*. 30(4): 22–28.

Cheney, J., A. Ahmed, and U. A. Acar. (2014). "Database Queries that Explain their Work". In: *Proceedings of the 16th International Symposium on Principles and Practice of Declarative Programming, Kent, Canterbury, United Kingdom, September 8-10, 2014*. 271–282.

Cheney, J., L. Chiticariu, and W.-C. Tan. (2009). "Provenance in Databases: Why, How, and Where". *Foundations and Trends® in Databases*. 1(4): 379–474.

Chockler, H. and J. Y. Halpern. (2004). "Responsibility and Blame: A Structural-Model Approach". *J. Artif. Int. Res.* 22(1): 93–115.

Chu, S., D. Li, C. Wang, A. Cheung, and D. Suciu. (2017a). "Demonstration of the Cosette Automated SQL Prover". In: *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. Ed. by S. Salihoglu, W. Zhou, R. Chirkova, J. Yang, and D. Suciu. ACM. 1591–1594.

Chu, S., C. Wang, K. Weitz, and A. Cheung. (2017b). "Cosette: An Automated Prover for SQL". In: *CIDR 2017, 8th Biennial Conference on Innovative Data Systems Research, Chaminade, CA, USA, January 8-11, 2017, Online Proceedings*. www.cidrdb.org.

Chung, Y., T. Kraska, N. Polyzotis, K. H. Tae, and S. E. Whang. (2019). "Slice finder: Automated data slicing for model validation". In: *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE. 1550–1553.

Cook, R. D. and S. Weisberg. (1982). *Residuals and influence in regression*. New York: Chapman and Hall.

Cui, Y. and J. Widom. (2001). "Lineage Tracing for General Data Warehouse Transformations". In: *Proceedings of 27th International Conference on Very Large Data Bases*. 471–480.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Das, M., S. Amer-Yahia, G. Das, and C. Yu. (2011). "MRI: Meaningful Interpretations of Collaborative Ratings". *PVLDB*. 4(11): 1063–1074.

Datta, A., S. Sen, and Y. Zick. (2016). "Algorithmic Transparency via Quantitative Input Influence: Theory and Experiments with Learning Systems". In: *IEEE Symposium on Security and Privacy, SP 2016, San Jose, CA, USA, May 22-26, 2016*. IEEE Computer Society. 598–617. DOI: 10.1109/SP.2016.42.

Davidson, S. B., S. C. Boulakia, A. Eyal, B. Ludäscher, T. M. McPhillips, S. Bowers, M. K. Anand, and J. Freire. (2007). "Provenance in Scientific Workflow Systems". *IEEE Data Eng. Bull.* 30(4): 44–50.

Deutch, D., N. Frost, and A. Gilad. (2017). "Provenance for Natural Language Queries". *Proceedings of the VLDB Endowment*. 10(5).

Deutch, D., N. Frost, A. Gilad, and O. Sheffer. (2020). "T-REx: Table Repair Explanations". In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. SIGMOD '20*. Portland, OR, USA: Association for Computing Machinery. 2765–2768. DOI: 10.1145/3318464.3384700.

Deutch, D., T. Milo, S. Roy, and V. Tannen. (2014). "Circuits for Datalog Provenance". In: *Proc. 17th International Conference on Database Theory (ICDT)*. 201–212.

Dias, K., M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood. (2005). "Automatic Performance Diagnosis and Tuning in Oracle." In: *CIDR*. 84–94.

Diestelkämper, R., S. Lee, M. Herschel, and B. Glavic. (2021). "To not miss the forest for the trees - A holistic approach for explaining missing answers over nested data". In: *Proceedings of the 46th International Conference on Management of Data*. 405–417.

Dong, X., E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. (2014). "Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '14*. New York, New York, USA: Association for Computing Machinery. 601–610. DOI: 10.1145/2623330.2623623.

"Dr. Elephant". (2019). http://www.teradata.com.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Drosou, M. and E. Pitoura. (2012). "DisC Diversity: Result Diversification Based on Dissimilarity and Coverage". *Proc. VLDB Endow.* 6(1): 13–24.

Fabbri, D. and K. LeFevre. (2011). "Explanation-based auditing". *Proc. VLDB Endow.* 5(1): 1–12. URL: http://dl.acm.org/citation.cfm?id=2047485.2047486.

Fariha, A., S. Nath, and A. Meliou. (2020). "Causality-Guided Adaptive Interventional Debugging". In: *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*. Ed. by D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo. ACM. 431–446.

Fehrenbach, S. and J. Cheney. (2019). In: *Proceedings of the 17th ACM SIGPLAN International Symposium on Database Programming Languages, DBPL 2019, Phoenix, AZ, USA, June 23, 2019*. 74–84.

Feng, S., A. Huber, B. Glavic, and O. Kennedy. (2021). "Efficient Uncertainty Tracking for Complex Queries with Attribute-level Bounds". In: *Proceedings of the 46th International Conference on Management of Data*. 528–540.

Freire, C., W. Gatterbauer, N. Immerman, and A. Meliou. (2015). "A Characterization of the Complexity of Resilience and Responsibility for Self-join-free Conjunctive Queries". *PVLDB*. 9(3): 180–191. DOI: 10.14778/2850583.2850592.

Frye, C., C. Rowat, and I. Feige. (2020). "Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. URL: https://proceedings.neurips.cc/paper/2020/hash/0d770c496aa3da6d2c3f2bd19e7b9d6b-Abstract.html.

Galhotra, S., R. Pradhan, and B. Salimi. (2021). "Explaining Black-Box Algorithms Using Probabilistic Contrastive Counterfactuals". *CoRR*. abs/2103.11972. arXiv: 2103.11972. URL: https://arxiv.org/abs/2103.11972.

"Ganglia Monitoring System". (2019). http://ganglia.info.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Gebaly, K. E., P. Agrawal, L. Golab, F. Korn, and D. Srivastava. (2014). "Interpretable and Informative Explanations of Outcomes". *Proc. VLDB Endow.* 8(1): 61–72.

Gebaly, K. E., G. Feng, L. Golab, F. Korn, and D. Srivastava. (2018). "Explanation Tables". *IEEE Data Eng. Bull.* 41(3): 43–51.

Ghorbani, A., M. P. Kim, and J. Zou. (2020). "A Distributional Framework For Data Valuation". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. *Proceedings of Machine Learning Research*. PMLR. 3535–3544. URL: http://proceedings.mlr.press/v119/ghorbani20a.html.

Ghorbani, A. and J. Y. Zou. (2019). "Data Shapley: Equitable Valuation of Data for Machine Learning". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. *Proceedings of Machine Learning Research*. PMLR. 2242–2251. URL: http://proceedings.mlr.press/v97/ghorbani19c.html.

Glavic, B. (2021). "Data Provenance: Origins, Applications, Algorithms, and Models". *Foundations and Trends® in Databases*: to appear.

Glavic, B., G. Alonso, R. J. Miller, and L. M. Haas. (2010). "TRAMP: Understanding the Behavior of Schema Mappings through Provenance". *Proceedings of the Very Large Data Bases Endowment.* 3(1): 1314–1325.

Glavic, B., S. Köhler, S. Riddle, and B. Ludäscher. (2015). "Towards Constraint-based Explanations for Answers and Non-Answers". In: *Proceedings of the 7th USENIX Workshop on the Theory and Practice of Provenance.*

Glavic, B., R. J. Miller, and G. Alonso. (2013). "Using SQL for Efficient Generation and Querying of Provenance Information". *In search of elegance in the theory and practice of computation: a Festschrift in honour of Peter Buneman*: 291–320.

Gray, J., S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. (1997). "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals". *Data Min. Knowl. Discov.* 1(1): 29–53.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Green, T. J., G. Karvounarakis, and V. Tannen. (2007). "Provenance semirings". In: *PODS*. 31–40.

Guidotti, R., A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. (2019). "A Survey of Methods for Explaining Black Box Models". *ACM Comput. Surv.* 51(5): 93:1–93:42. DOI: 10.1145/3236009.

Halpern, J. Y. and J. Pearl. (2001). "Causes and Explanations: A Structural-Model Approach: Part i: Causes". In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence. UAI'01*. Seattle, Washington: Morgan Kaufmann Publishers Inc. 194–202.

Han, J. and M. Kamber. (2001). *Data Mining: Concepts and Techniques*. Morgan Kaufmann.

Herodotou, H., H. Lim, G. Luo, N. Borisov, L. Dong, F. B. Cetin, and S. Babu. (2011). "Starfish: a self-tuning system for big data analytics." In: *Cidr*. Vol. 11. No. 2011. 261–272.

Joglekar, M., H. Garcia-Molina, and A. Parameswaran. (2017). "Interactive data exploration with smart drill-down". *IEEE Transactions on Knowledge and Data Engineering*. 31(1): 46–60.

Joglekar, M., H. Garcia-Molina, and A. G. Parameswaran. (2016). "Interactive data exploration with smart drill-down". In: *32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016*. IEEE Computer Society. 906–917. DOI: 10.1109/ICDE.2016.7498300.

Kalashnikov, D. V., L. V. Lakshmanan, and D. Srivastava. (2018). "FastQRE: Fast Query Reverse Engineering". In: *Proceedings of the 2018 International Conference on Management of Data*. ACM. 337–350.

Kalmegh, P., S. Babu, and S. Roy. (2019). "iQCAR: inter-Query Contention Analyzer for Data Analytics Frameworks". In: *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. 918–935.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Kanagal, B., J. Li, and A. Deshpande. (2011). "Sensitivity analysis and explanations for robust query evaluation in probabilistic databases". In: *SIGMOD*. Athens, Greece: ACM. 841–852. DOI: 10.1145/1989323.1989411.

Karimi, A., G. Barthe, B. Balle, and I. Valera. (2020). "Model-Agnostic Counterfactual Explanations for Consequential Decisions". In: *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*. Ed. by S. Chiappa and R. Calandra. Vol. 108. *Proceedings of Machine Learning Research*. PMLR. 895–905. URL: http://proceedings.mlr.press/v108/karimi20a.html.

Khoussainova, N., M. Balazinska, and D. Suciu. (2012). "Perfxplain: debugging mapreduce job performance". *PVLDB*. 5(7): 598–609.

Kim, A., L. V. Lakshmanan, and D. Srivastava. (2020). "Summarizing Hierarchical Multidimensional Data". In: *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE. 877–888.

Koh, P. W. and P. Liang. (2017). "Understanding Black-box Predictions via Influence Functions". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by D. Precup and Y. W. Teh. Vol. 70. *Proceedings of Machine Learning Research*. PMLR. 1885–1894. URL: http://proceedings.mlr.press/v70/koh17a.html.

Köhler, S., B. Ludäscher, and Y. Smaragdakis. (2012). "Declarative datalog debugging for mere mortals". *Datalog in Academia and Industry*: 111–122.

Kwon, Y., M. A. Rivas, and J. Zou. (2021). "Efficient Computation and Analysis of Distributional Shapley Values". In: *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*. Ed. by A. Banerjee and K. Fukumizu. Vol. 130. *Proceedings of Machine Learning Research*. PMLR. 793–801. URL: http://proceedings.mlr.press/v130/kwon21a.html.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Laugel, T., M. Lesot, C. Marsala, X. Renard, and M. Detyniecki. (2018). "Comparison-Based Inverse Classification for Interpretability in Machine Learning". In: *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations - 17th International Conference, IPMU 2018, Cádiz, Spain, June 11-15, 2018, Proceedings, Part I*. Ed. by J. Medina, M. Ojeda-Aciego, J. L. V. Galdeano, D. A. Pelta, I. P. Cabrera, B. Bouchon-Meunier, and R. R. Yager. Vol. 853. *Communications in Computer and Information Science*. Springer. 100–111. DOI: 10.1007/978-3-319-91473-2_9.

Lee, S., B. Ludäscher, and B. Glavic. (2018). "PUG: a framework and practical implementation for why and why-not provenance". *The VLDB Journal*. 28(1): 47–71. DOI: 10.1007/s00778-018-0518-5.

Lee, S., B. Ludäscher, and B. Glavic. (2020). "Approximate Summaries for Why and Why-not Provenance". *PVLDB*. 13(6): 912–924.

Letham, B., C. Rudin, T. H. McCormick, and D. Madigan. (2015). "Interpretable Classifiers Using Rules and Bayesian Analysis: Building a Better Stroke Prediction Model". *CoRR*. abs/1511.01644. arXiv: 1511.01644. URL: http://arxiv.org/abs/1511.01644.

Livshits, E., L. E. Bertossi, B. Kimelfeld, and M. Sebag. (2020). "The Shapley Value of Tuples in Query Answering". In: *23rd International Conference on Database Theory, ICDT 2020, March 30-April 2, 2020, Copenhagen, Denmark*. Ed. by C. Lutz and J. C. Jung. Vol. 155. *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 20:1–20:19.

Lundberg, S. M., G. G. Erion, and S. Lee. (2018). "Consistent Individualized Feature Attribution for Tree Ensembles". *CoRR*. abs/1802.03888. arXiv: 1802.03888. URL: http://arxiv.org/abs/1802.03888.

Lundberg, S. M. and S. Lee. (2017). "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett. 4765–4774.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Luo, Y., X. Qin, N. Tang, and G. Li. (2018). "DeepEye: Towards Automatic Data Visualization". In: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 101–112.

Mahajan, D., C. Tan, and A. Sharma. (2019). "Preserving Causal Constraints in Counterfactual Explanations for Machine Learning Classifiers". *CoRR*. abs/1912.03277. arXiv: 1912.03277. URL: http://arxiv.org/abs/1912.03277.

Meliou, A., W. Gatterbauer, K. F. Moore, and D. Suciu. (2010). "Why So? or Why No? Functional Causality for Explaining Query Answers". In: *Proceedings of the 4th International VLDB workshop on Management of Uncertain Data (MUD) in conjunction with VLDB* (MUD). Singapore. 3–17.

Meliou, A., W. Gatterbauer, and D. Suciu. (2011). "Reverse Data Management". *PVLDB*. 4(11): 1490–1493.

Meliou, A. and D. Suciu. (2012). "Tiresias: The Database Oracle for How-To Queries". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)* (SIGMOD). Scottsdale, AZ. 337–348. DOI: 10.1145/2213836.2213875.

Merrick, L. and A. Taly. (2019). "The Explanation Game: Explaining Machine Learning Models With Cooperative Game Theory". *CoRR*. abs/1909.08128. arXiv: 1909.08128. URL: http://arxiv.org/abs/1909.08128.

Miao, Z., T. Chen, A. Bendeck, K. Day, S. Roy, and J. Yang. (2020). "I-Rex: An Interactive Relational Query Explainer for SQL". *Proc. VLDB Endow.* 13(12): 2997–3000.

Miao, Z., S. Roy, and J. Yang. (2019a). "Explaining Wrong Queries Using Small Examples". In: *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. 503–520.

Miao, Z., S. Roy, and J. Yang. (2019b). "RATest: Explaining Wrong Relational Queries Using Small Examples". In: *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. Ed. by P. A. Boncz, S. Manegold, A. Ailamaki, A. Deshpande, and T. Kraska. ACM. 1961–1964.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Miao, Z., Q. Zeng, B. Glavic, and S. Roy. (2019c). "Going Beyond Provenance: Explaining Query Answers with Pattern-based Counterbalances". In: *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*. Ed. by P. A. Boncz, S. Manegold, A. Ailamaki, A. Deshpande, and T. Kraska. ACM. 485–502.

Mishra, C. and N. Koudas. (2009). In: *EDBT 2009, 12th International Conference on Extending Database Technology, Saint Petersburg, Russia, March 24-26, 2009, Proceedings*. 862–873.

Mita, G., P. Papotti, M. Filippone, and P. Michiardi. (2020). "LIBRE: Learning Interpretable Boolean Rule Ensembles". In: *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*. 245–255. URL: http://proceedings.mlr.press/v108/mita20a.html.

Mitchell, T. M. (2012). "Never Ending Learning". In: *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*. Ed. by L. D. Raedt, C. Bessiere, D. Dubois, P. Doherty, P. Frasconi, F. Heintz, and P. J. F. Lucas. Vol. 242. *Frontiers in Artificial Intelligence and Applications*. IOS Press. 5.

Mothilal, R. K., A. Sharma, and C. Tan. (2020). "Explaining machine learning classifiers through diverse counterfactual explanations". In: *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*. Ed. by M. Hildebrandt, C. Castillo, E. Celis, S. Ruggieri, L. Taylor, and G. Zanfir-Fortuna. ACM. 607–617. DOI: 10.1145/3351095.3372850.

Mottin, D., A. Marascu, S. B. Roy, G. Das, T. Palpanas, and Y. Velegrakis. (2013). "A probabilistic optimization framework for the empty-answer problem". *Proceedings of the VLDB Endowment*. 6(14): 1762–1773.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Muniswamy-Reddy, K., D. A. Holland, U. Braun, and M. I. Seltzer. (2006). "Provenance-Aware Storage Systems". In: *Proceedings of the 2006 USENIX Annual Technical Conference, Boston, MA, USA, May 30 - June 3, 2006*. Ed. by A. Adya and E. M. Nahum. USENIX. 43–56.

Olston, C., S. Chopra, and U. Srivastava. (2009). "Generating example data for dataflow programs". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*. Ed. by U. Çetintemel, S. B. Zdonik, D. Kossmann, and N. Tatbul. ACM. 245–256.

Ousterhout, K., R. Rasti, S. Ratnasamy, S. Shenker, and B.-G. Chun. (2015). "Making Sense of Performance in Data Analytics Frameworks". In: *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. 293–307.

Owen, A. B. and C. Prieur. (2017). "On Shapley Value for Measuring Importance of Dependent Inputs". *SIAM/ASA J. Uncertain. Quantification*. 5(1): 986–1002. DOI: 10.1137/16M1097717.

Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. USA: Cambridge University Press.

Qin, L., J. X. Yu, and L. Chang. (2012). "Diversifying Top-K Results". *Proc. VLDB Endow.* 5(11): 1124–1135. DOI: 10.14778/2350229.2350233.

Ré, C. and D. Suciu. (2008). "Approximate lineage for probabilistic databases". *Proc. VLDB Endow.* 1(1): 797–808. URL: http://dl.acm.org/citation.cfm?id=1453856.1453943.

Reshef, A., B. Kimelfeld, and E. Livshits. (2020). "The Impact of Negation on the Complexity of the Shapley Value in Conjunctive Queries". In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*. Ed. by D. Suciu, Y. Tao, and Z. Wei. ACM. 285–297. DOI: 10.1145/3375395.3387664.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Ribeiro, M. T., S. Singh, and C. Guestrin. (2016). ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. Ed. by B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi. ACM. 1135–1144. DOI: 10.1145/2939672.2939778.

Ribeiro, M. T., S. Singh, and C. Guestrin. (2018). "Anchors: High-Precision Model-Agnostic Explanations". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. Ed. by S. A. McIlraith and K. Q. Weinberger. AAAI Press. 1527–1535.

Rosenbaum, P. R. and D. B. Rubin. (1983). "The central role of the propensity score in observational studies for causal effects". *Biometrika*. 70(1): 41–55.

Roy, S., L. J. Orr, and D. Suciu. (2015a). "Explaining Query Answers with Explanation-Ready Databases". *Proc. VLDB Endow.* 9(4): 348–359. DOI: 10.14778/2856318.2856329.

Roy, S. and D. Suciu. (2014). "A Formal Approach to Finding Explanations for Database Queries". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data. SIGMOD '14*. Snowbird, Utah, USA: Association for Computing Machinery. 1579–1590. DOI: 10.1145/2588555.2588578.

Roy, S., A. C. König, I. Dvorkin, and M. Kumar. (2015b). "PerfAugur: Robust diagnostics for performance anomalies in cloud services". In: *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*. Ed. by J. Gehrke, W. Lehner, K. Shim, S. K. Cha, and G. M. Lohman. 1167–1178.

Rubin, D. B. (2005). "Causal Inference Using Potential Outcomes: Design, Modeling, Decisions". *Journal of the American Statistical Association*. 100(Mar.): 322–331.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Rudin, C. (2019). "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead". *Nature Machine Intelligence*. 1(5).

Sagadeeva, S. and M. Boehm. (2021). "SliceLine: Fast, Linear-Algebra-based Slice Finding for ML Model Debugging". In: *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*. Ed. by G. Li, Z. Li, S. Idreos, and D. Srivastava. ACM. 2290–2299. DOI: 10.1145/3448016.3457323.

Salimi, B., H. Parikh, M. Kayali, L. Getoor, S. Roy, and D. Suciu. (2020). "Causal Relational Learning". In: *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*. Ed. by D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo. ACM. 241–256.

Sarawagi, S. (2000). "User-Adaptive Exploration of Multidimensional Data". In: *VLDB 2000, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt*. Ed. by A. E. Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K. Whang. Morgan Kaufmann. 307–316.

Sarawagi, S. and G. Sathe. (2000). "i^3: Intelligent, Interactive Investigaton of OLAP data cubes". In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16-18, 2000, Dallas, Texas, USA*. Ed. by W. Chen, J. F. Naughton, and P. A. Bernstein. ACM. 589. DOI: 10.1145/342009.336564.

Sathe, G. and S. Sarawagi. (2001). "Intelligent Rollups in Multidimensional OLAP Data". In: *VLDB 2001, Proceedings of 27th International Conference on Very Large Data Bases, September 11-14, 2001, Roma, Italy*. Ed. by P. M. G. Apers, P. Atzeni, S. Ceri, S. Paraboschi, K. Ramamohanarao, and R. T. Snodgrass. Morgan Kaufmann. 531–540. URL: http://www.vldb.org/conf/2001/P531.pdf.

Shapley, L. S. (1953). "A Value for n-Person Games". *Contributions to the Theory of Games II*: 307–317.

Siddiqui, T., A. Kim, J. Lee, K. Karahalios, and A. G. Parameswaran. (2016). "Effortless Data Exploration with zenvisage: An Expressive and Interactive Visual Analytics System". *Proc. VLDB Endow.* 10(4): 457–468.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Snodgrass, R. T., S. S. Yao, and C. Collberg. (2004). "Tamper detection in audit logs". In: *VLDB*. 504–515.

"Spark Monitoring and Instrumentation". (2019). http://spark.apache.org/docs/latest/monitoring.html.

Stekhoven, D. J. and P. Bühlmann. (2012). "Missforest - Non-Parametric Missing Value Imputation for Mixed-Type Data". *Bioinform.* 28(1): 112–118. DOI: 10.1093/bioinformatics/btr597.

Suchanek, F. M., G. Kasneci, and G. Weikum. (2007). "Yago: a core of semantic knowledge". In: *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. Ed. by C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy. ACM. 697–706.

Tannen, V. (2010). "Provenance for database transformations". In: *EDBT 2010, 13th International Conference on Extending Database Technology, Lausanne, Switzerland, March 22-26, 2010, Proceedings*. Vol. 426. *ACM International Conference Proceeding Series*. ACM. 1.

Thirumuruganathan, S., M. Das, S. Desai, S. Amer-Yahia, G. Das, and C. Yu. (2012). "MapRat: meaningful explanation, interactive exploration and geo-visualization of collaborative ratings". *Proc. VLDB Endow.* 5(12): 1986–1989. URL: http://dl.acm.org/citation.cfm?id=2367502.2367554.

Tip, F. (1994). *A Survey of Program Slicing Techniques*. Centrum voor Wiskunde en Informatica.

Tran, Q. T., C. Y. Chan, and S. Parthasarathy. (2014). "Query Reverse Engineering". *VLDB J.* 23(5): 721–746. DOI: 10.1007/s00778-013-0349-3.

Tran, Q. T. and C.-Y. Chan. (2010). "How to ConQueR why-not questions". In: *SIGMOD '10: Proceedings of the 2010 international conference on Management of data*. Indianapolis, Indiana, USA: ACM. 15–26.

Ustun, B., A. Spangher, and Y. Liu. (2019). "Actionable Recourse in Linear Classification". In: *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, Atlanta, GA, USA, January 29-31, 2019*. 10–19. DOI: 10.1145/3287560.3287566.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Van Aken, D., A. Pavlo, G. J. Gordon, and B. Zhang. (2017). "Automatic database management system tuning through large-scale machine learning". In: *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM. 1009–1024.

Vardi, M. Y. (1982). "The Complexity of Relational Query Languages (Extended Abstract)". In: *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing. STOC '82*. San Francisco, California, USA: ACM. 137–146. DOI: 10.1145/800070.802186.

Veanes, M., N. Tillmann, and J. de Halleux. (2010). "Qex: Symbolic SQL Query Explorer". In: *Logic for Programming, Artificial Intelligence, and Reasoning - 16th International Conference, LPAR-16, Dakar, Senegal, April 25-May 1, 2010, Revised Selected Papers*. Ed. by E. M. Clarke and A. Voronkov. Vol. 6355. *Lecture Notes in Computer Science*. Springer. 425–446.

Vélez, B., R. Weiss, M. A. Sheldon, and D. K. Gifford. (1997). In: *SIGIR '97: Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 27-31, 1997, Philadelphia, PA, USA*. 6–15.

Verma, S., J. P. Dickerson, and K. Hines. (2020). "Counterfactual Explanations for Machine Learning: A Review". *CoRR*. abs/2010.10596. arXiv: 2010.10596. URL: https://arxiv.org/abs/2010.10596.

Vieira, M. R., H. L. Razente, M. C. N. Barioni, M. Hadjieleftheriou, D. Srivastava, C. Traina, and V. J. Tsotras. (2011). "On query result diversification". In: *2011 IEEE 27th International Conference on Data Engineering*. 1163–1174.

Wachter, S., B. D. Mittelstadt, and C. Russell. (2017). "Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR". *CoRR*. abs/1711.00399. arXiv: 1711.00399. URL: http://arxiv.org/abs/1711.00399.

Wang, T., M. Morucci, M. U. Awan, Y. Liu, S. Roy, C. Rudin, and A. Volfovsky. (2021). "FLAME: A Fast Large-scale Almost Matching Exactly Approach to Causal Inference". *Journal of Machine Learning Research*. 22(31): 1–41. URL: http://jmlr.org/papers/v22/19-853.html.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Wang, X., X. L. Dong, and A. Meliou. (2015a). "Data X-Ray: A Diagnostic Tool for Data Errors". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)* (SIGMOD). 1231–1245. DOI: 10.1145/2723372.2750549.

Wang, X., M. Feng, Y. Wang, L. Dong, and A. Meliou. (2015b). "Error Diagnosis and Data Profiling with Data X-Ray". *PVLDB*. 8(12): 1984–1987. DOI: 10.14778/2824032.2824117.

Wang, X. and A. Meliou. (2019). "Explain3D: Explaining Disagreements in Disjoint Datasets". *PVLDB*. 12(7): 779–792.

Wang, X., A. Meliou, and E. Wu. (2017). "QFix: Diagnosing errors through query histories". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)* (SIGMOD). 1369–1384. DOI: 10.1145/3035918.3035925.

Weiser, M. (1981). "Program slicing". *Proceedings of the 5th international conference on Software engineering*: 439–449.

Wen, Y., X. Zhu, S. Roy, and J. Yang. (2018). "Interactive Summarization and Exploration of Top Aggregate Query Answers". *Proc. VLDB Endow.* 11(13): 2196–2208.

Wu, E. and S. Madden. (2013). "Scorpion: Explaining Away Outliers in Aggregate Queries". *Proc. VLDB Endow.* 6(8): 553–564.

Wu, W., L. Flokas, E. Wu, and J. Wang. (2020). "Complaint-driven Training Data Debugging for Query 2.0". In: *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*. Ed. by D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo. ACM. 1317–1334. DOI: 10.1145/3318464.3389696.

Xu, B., J. Qian, X. Zhang, Z. Wu, and L. Chen. (2005). "A brief survey of program slicing". *ACM SIGSOFT Software Engineering Notes*. 30(2): 1–36.

Yang, Y., N. Meneghetti, R. Fehling, Z. H. Liu, and O. Kennedy. (2015). "Lenses: an on-demand approach to ETL". *Proceedings of the VLDB Endowment*. 8(12): 1578–1589.

The version of record is available at: http://dx.doi.org/10.1561/1900000074

Yilmaz, G. S., T. Wattanawaroon, L. Xu, A. Nigam, A. J. Elmore, and A. G. Parameswaran. (2018). "DataDiff: User-Interpretable Data Transformation Summaries for Collaborative Data Analysis". In: *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*. Ed. by G. Das, C. M. Jermaine, and P. A. Bernstein. ACM. 1769–1772. DOI: 10.1145/3183713.3193564.

Ying, Z., D. Bourgeois, J. You, M. Zitnik, and J. Leskovec. (2019). "GNNExplainer: Generating Explanations for Graph Neural Networks". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. 9240–9251.

Yoon, D. Y., N. Niu, and B. Mozafari. (2016). "DBSherlock: A Performance Diagnostic Tool for Transactional Databases". In: *Proceedings of the 2016 International Conference on Management of Data. SIGMOD '16*. San Francisco, California, USA: ACM. 1599–1614. DOI: 10.1145/2882903.2915218.

Zhang, H., Y. Diao, and A. Meliou. (2017). "EXStream: Explaining Anomalies in Event Stream Monitoring". In: *20th International Conference on Extending Database Technology (EDBT)* (EDBT). 156–167. DOI: 10.5441/002/edbt.2017.15.

Zhang, Y. and X. Chen. (2020). "Explainable Recommendation: A Survey and New Perspectives". *Found. Trends Inf. Retr.* 14(1): 1–101.