Toward Interpretable and Actionable Data Analysis with Explanations and Causality

Sudeepa Roy
Duke University
Durham, NC, USA
sudeepa@cs.duke.edu

ABSTRACT

We live in a world dominated by data, where users from different fields routinely collect, study, and make decisions supported by data. To aid these users, the current trend in data analysis is to design tools that allow large-scale analytics, sophisticated predictive models, and beautiful visualizations. At this exciting time when both data and analytics tools are widely accessible to users, treating analyses as magical black boxes can painfully mislead users and make troubleshooting frustratingly time-consuming. For instance, although the perils of interpreting correlations inferred by predictive models as causation are well-documented, making such a distinction can be tricky for many users who do not have formal training in computer science or statistics. In this paper, we give an overview of our research toward bridging this gap along two main thrusts of *explanations* and *causality*. Explanations support a primary goal of data analysis – empowering users to be able to interpret the results in data analysis and troubleshoot the process. Causality complements explanations by supporting prescriptive or actionable analytics with counterfactuals and interventions, thereby helping sound decision making. In these thrusts, we explore the symbiotic relationship between core database techniques and complementary techniques from machine learning and statistics via interdisciplinary collaborations, and employ them to applications in domains like computer science education, law, and health.

PVLDB Reference Format:
Sudeepa Roy. Toward Interpretable and Actionable Data Analysis with Explanations and Causality. PVLDB, 15(12): 3812 - 3820, 2022.
doi:10.14778/3554821.3554902

1 INTRODUCTION

In today's world awash with data, users from different fields and with different backgrounds routinely collect, study, and make decisions supported by data. To aid these users, the current trend in data analysis is to design tools that allow large-scale analytics, sophisticated predictive models, and beautiful visualizations. However, as data and analytics tools become widely accessible to a broad range of users, the limitations of treating data analysis as magical black boxes can painfully mislead users and make troubleshooting frustratingly time-consuming. As an obvious example, the perils of interpreting correlations inferred by predictive models as causation are widely documented. However, making such a distinction can be tricky for many data analysts and domain experts who do not have formal training in computer science (CS) or statistics. On the other hand, in all stages of a *data analysis pipeline* from data preparation, data processing, to the final analysis of results, users may seek some insights and *explanations* in the form of 'why' and 'how' questions - 'why is my program wrong?', 'how do I fix it?', 'why is a value in the result higher/lower than another value or a value that I was expecting?', 'how do I get a desired result?' and so on. While dealing with real data we need further care in answering such questions, e.g., the data may be large and may have a complex structure with dependencies, the answers we provide should be interpretable and the process should be interactive, and in some scenarios, the data may have private information that must not be violated while answering such 'why'/'how' questions. In this paper, we give an overview of research we have been conducting to address these issues along two main thrusts of *explanations* and *causality*. Explanations support a primary goal of data analysis – empowering users to be able to interpret trends and anomalies in their analysis, debug their code, and get better insights about the process in general. Causality complements explanations by supporting prescriptive or actionable analytics with counterfactuals and interventions, thereby helping credible decision making.

Our research in explanations and causality aims to provide users sound, interpretable, and easy-to-use tools via a three-pronged approach: (a) develop theoretical and methodological foundations, (b) design algorithms and build efficient easy-to-use systems, and (c) explore the symbiotic relationship between data management techniques and complementary tools such as machine learning and statistics via interdisciplinary collaborations. The combination of these three approaches produces tools and techniques that have impact on real-world problems in two different ways: *First*, deeper insights into data, query results, and applications by explanations and causality help analysts make informed decisions and programmers debug/refine data-driven processes in industry, academia, and other domains. As an example, our tool RATest for *explaining wrong relational queries* [48, 49] that we discuss later is built on a formal foundation using provenance semirings [28], and has so far been used by more than 1000 students in undergraduate and graduate database courses at Duke since 2018 with enthusiastic feedback from the students. It provides automated and interactive help to both students and teaching assistants, which is more important than ever given the recent surge in enrollment in CS courses everywhere. *Second*, this research has an academic impact on other areas outside data management within and beyond CS by interdisciplinary collaborations. Our *Almost Matching Exactly (AME) lab*

Proceedings of the VLDB Endowment, Vol. 15, No. 12 ISSN 2150-8097.
doi:10.14778/3554821.3554902

[2] for causal inference research brings together researchers in CS (data management and machine learning), statistics, economics, political science, and law, where we develop new efficient algorithms for causal inference, and study their utility on practical problems from various application domains. For instance, our causal analysis research on various real datasets has led to new insights regarding the effect of a pre-trial community supervision program on new criminal charges, hospital stays and health insurance, and the effect of prenatal smoking on the health of newborns [60, 61, 68].

Related Work. There is a vast related work in database research on various notions of explanations for different contexts and applications, whereas causal inference (estimating the effect of a *treatment* on an *outcome*) has been studied mostly in the statistics and artificial intelligence (AI) communities (e.g., [55, 56, 59]) along with applications in clinical trials, public health, social studies, etc., with a recent interest in database research. For the sake of brevity, we omit a review of related work in this paper. We (with Boris Glavic and Alexandra Meliou) recently wrote a Foundations and Trends in Databases article on explanations for data-driven systems [26], and refer the reader to this article for a detailed discussion on different concepts of explanations from the literature.

2 EXPLANATIONS FOR DATABASE QUERIES

Understanding data, query answers, and plots is a key step in data analysis to enforce accountability in decision making or refine a process. Given one or more facts as observation made by a user in data analysis, an *explanation* can be considered as a *set of statements* that aims to clarify the causes, context, and/or consequences of those facts [71, 72]. Our research in *explanations in data analysis* can be divided into two main parts. (1) The *FIREFly (Formal Interactive Rich Explanations on-the-Fly)* project focuses on the final step in data analysis when some aggregate answers have been obtained and/or plots have been generated, and the users want to understand the trends and anomalies in these answers (e.g., '*why one value is higher/lower than expected*' or '*why a value is an outlier*'). We have been developing a toolkit for explanations for aggregate query answers that generates different types of high-level semantic explanations from the data in response to questions from the users. (2) The *HNRQ (Helping Novices learn Relational Queries)* project focuses on the query writing phase, and aims to help new programmers or students understand why their query is wrong and how it can be fixed. Below, we will discuss our work in these two directions followed by other work related to explanations. One common theme in all our approaches to explanations has been modeling what a human expert who is familiar with the application might think as a natural explanation, and then computing such explanations automatically in our explanation frameworks.

2.1 Explanations for Aggregate Query Answers

In the context of explaining query answers in database research, several frameworks for explanations both for non-aggregate and aggregate query answers have been proposed [26]. The most obvious notion of explanations use *data provenance* [8–10, 28], which explains why and how a query answer has been generated. In particular, the mathematical notion of *provenance semirings* has been developed for both non-aggregate and aggregate queries [3, 28]. However, for aggregate queries, when users want to understand why an output value is higher/lower than their expectation, or why a value is higher/lower than another value, simply showing how they have been generated does not provide much insights to them. Moreover, typically several input tuples contribute to a single aggregate output value. Hence fine-grained provenance might be overwhelming to the user, motivating the need for high-level explanations. The explanations we consider for explaining or comparing aggregate values are in the form of *explanation predicates* on attributes in the data summarizing a set of input tuples (also used in other work in the literature on explanations such as [20, 73]), however, in our different explanation frameworks these predicates assume different meaning.

2.1.1 Explanations motivated by causality, counterfactuals, and intervention. Different notions of *scientific explanations* have received considerable attention in philosophical discourse from pre-Socratic times through the modern period. Among them, a pioneering concept is that of *causal explanations* (we will discuss causality in Section 3), which primarily uses the concept of *counterfactuals* (if A had not occurred then B would not have occurred, therefore A explains B) and *interventions* (if we change A, B would change too). Our first work in this area [58] proposed a formal framework of explanations for one or more query answers adapting the notion of causality and interventions. It finds synopses of properties on input tuples by predicates on attributes as explanations, such that by restricting the database to tuples that entail a different value of these synopses, the query answers and the observation of the user (captured by a function on multiple query answers) change, thereby explaining the observation. Here is an example, the exact set up can be found in [58].

Example 2.1. From the DBLP data [12], we observed that SIGMOD publications from industrial research labs had a peak around year 2000 and a decreasing trend later. Some of the top explanations we found pointed at prominent industrial research labs (and their senior researchers) that were highly active around those years in database research, and later they either shut down or possibly shifted their focus. If publications by those labs are removed from the database or if these labs hypothetically did not exist, then the peak will be flattened to some extent.

This notion of explanations also allowed adding *'causal dependencies'* among tuples that can lead to recursive tuple deletion (e.g., if an author is removed from a database, their publications should be removed as well but not necessarily vice versa). We showed that a non-trivial recursive query can find the intervention of a given explanation predicate in polynomial time in *data complexity* [65], and gave an efficient solution for practical purposes using the OLAP data cube operator [27] to evaluate the scores of all explanation predicates simultaneously.

An interesting observation from our above work [58] is that the *intervention* of a given candidate explanation predicate in a dataset, i.e., the set of tuples that depend on this explanation, is independent of the queries or user questions, and only depends on this dataset. This idea led to our next work called *explanation-ready databases* [57], where interventions by complex explanation predicates involving aggregates and other tables in the dataset are

stored by pre-processing. When a user runs a query and asks a question, it simultaneously evaluates and ranks the explanations by an *incremental view maintenance* approach without sequentially iterating over all the explanations.

Example 2.2. In [57], one of the datasets we explored was the NSF grants dataset [15], where we observed a difference in funding obtained by two top graduate schools in CS in the USA. The top explanations computed by our approach included the investigators with more than USD 100 million awards in total (there were none in one of the schools), investigators with large number of awards, two particular investigators from one of the schools who got huge funding, and awards with multiple investigators, suggesting that a few investigators in one of the schools got a few big awards which contributed the most toward the difference between the funding in these two schools. If these factors were not present in the dataset, then the difference between total funding between these two schools will reduce by USD 400 million to 850 million. These interesting explanations needed information from tables that were unused in the original query that simply retrieved the total funding from different schools, hence could not be found by our previous work [58].

2.1.2 Explanations beyond provenance by counterbalance. Explanations motivated by causality and interventions described above [57, 58] give useful insights from the *provenance* of the query answers involved in the user question, by summarizing the properties of the input tuples that contribute significantly to the query answers. Therefore, a large fraction of the input database remains unused in those explanations, leading to our next idea for explanations: whether there can be interesting explanations from the *'rest of the database' that has not been used at all in producing the query answers of interest.* Our next approach called *'explanations by counterbalance'* [50] finds explanations *going beyond provenance*, and explains a high (low) outlier with another low (high) outlier in the opposite direction with respect to some *regression patterns* learned from the query answers. These regression patterns include a partition attribute X, a group-by attribute Y, and an aggregated attribute $Z = agg(W)$, and describe how Z varies with Y for sufficiently many values of X, e.g., for more than p% of authors (X), the number of papers (Z) is about constant over the years (Y). We measured the score of these explanations by combining how *'surprising'* they are in terms of deviations from the expected patterns and a distance measure of the explanation pattern from the output tuple that the user wants to understand.

Example 2.3. In [50], we investigated the Chicago crime data [11], to understand why the number of *battery crimes* in a given area in Chicago in a certain year is lower than the other years. We found explanations like the area had more battery crimes in the following and the previous year, and adjacent areas had more battery crimes in the same year, possibly suggesting that the observation is an outlier. As another example with the DBLP publication dataset [12], less than usual publications by a researcher in one conference in a year was explained by higher publications in that venue in adjacent years, and in other venues in the same year.

2.1.3 Explanations by augmented provenance. Our quest of finding interesting explanations from the unused part of the data in the query output of interest went further than the idea of counterbalancing. In the above work [50], we extracted explanations from the *same input tables* that were used in the query, using the *input tuples* in those tables that did not contribute to the query outputs of interest. In [41], we explored further how the *other tables* that were not used in the original query can be used to find interesting explanations apart from our work on explanation-ready databases [57] discussed earlier. In [41], our idea was to go through multiple related tuples from other tables to find interesting explanations for aggregate query answers. For instance, an increased number of publications in a machine learning conference by a database researcher can be explained by her new collaboration with a machine learning researcher. Assuming a dataset like DBLP [12] where different tables contain information about authors, authorship, and publications, such explanations can only be found by joining and aggregating multiple tables including self-joins on the authorship table for the information on collaboration. To formalize this idea, we used *join graphs* to capture how different tables relate to each other. Then using a join graph as input, we augmented the provenance of the query outputs of interest to input tuples belonging to other tables, followed by outputting top explanation predicates summarizing those input tuples as accurately as possible by adapting the ideas of precision, recall, and F-score.

Example 2.4. In [41], we studied the NBA dataset (extracted from [14, 16]) and observed that the team *Golden State Warriors* improved their number of wins significantly in the 2015-16 season compared to the 2012-13 season. By augmenting the provenance of the output tuples for the number of wins to unused tables containing information about lineups and players' scores, we found interesting explanations like *' Player S. Curry scored more than 23 points in 58 out of 73 games in 2015-16 compared to 21 out of 47 games in 2012-13'*, and, *'Players D. Green and K. Thompson's on-court minutes together were more than 19 minutes in 70 out of 73 games in the 2015-16 season compared to only 2 out of 47 games in the 2012-13 season'*. These two explanations were based on two different join graphs [41], and the second one includes a self-join on the lineup table to find out which players played together. [41] has other examples using the MIMIC-III intensive-care and hospital-stay dataset [35].

2.1.4 Explanations with differential privacy. Data analysis with real datasets might require dealing with data with private or sensitive information. Providing explanations for such datasets using any of the above approaches might affect data privacy. Hence, we wanted to understand how we can find interesting explanations while giving a privacy guarantee. *Differential privacy (DP)* [19] is the gold standard for protecting privacy in query processing and is critically important for sensitive data analysis. The core idea behind DP is that the answer to a query on the original database cannot be distinguished from the answer to the same query on a slightly different database, which is usually achieved by adding random noise to the query answer to create a small distortion. The problem is that, when a user wants to understand why a value is high or low, it may be simply due to adding noise to the query answers. However simply revealing that, or even revealing any deterministic ranking or scores of explanations, will again violate privacy. In our recent work [63], we developed a framework called *DPXPlain* to support DP in explanations by adapting the notion of interventions

from [58, 73]. DPXPlain provides explanations in multiple phases, while guaranteeing DP in all steps with a given privacy budget. It is able to give the user insights on (with high confidence) whether the higher/lower value of interest is due to the noise added by a DP mechanism, and if not, it further provides a confidence interval of relative influence and ranking of explanation predicates summarizing the input tuples that contribute the most to that output. The confidence intervals also provide insights on whether to trust an explanation, for instance, out of 100 possible explanations if we get an interval of [1, 5] for the rank of an explanation predicate with 95% confidence, we know that it is likely to be a good explanation, but if the interval is [1, 95], we can discard the explanation.

2.1.5 Tools with user interface for explanations. Explanations are integrally meant for helping users understand, debug, or improve a data-driven process, which makes developing efficient and easy-to-use systems for exploring these explanations important. For our intervention-based approach to explanations [58], we developed an end-to-end system called *LensXPlain* [47], for our counterbalance-based approach [50], we developed the CAPE (Counterbalancing with Aggregate Patterns) system [51], and for our augmented-provenance approach [41], we developed the CAJADE (Context-Aware Join-Augmented Deep Explanations) system [40]. These tools provide additional functionality through their graphical user interfaces, such as suggesting important attributes to include in explanations, helping users ask questions, illustrating how the explanations explain the user question, and refining explanations based on user preference. Note that these different approaches to explanations are incomparable to each other - they provide different types of insights in response to 'why' questions on aggregate query answers asked by a user. An important future work is to combine all these explanation approaches under one unified tool that can display different types of explanations to the users interactively.

2.2 Explanations for Debugging Queries

Writing correct queries is another important step in data analysis with data management systems. In a project called *HNRQ: Helping Novices Learn and Debug Relational Queries* [32], we are working on how we can help new programmers and students debug and learn database queries. We illustrate our approaches with an example.

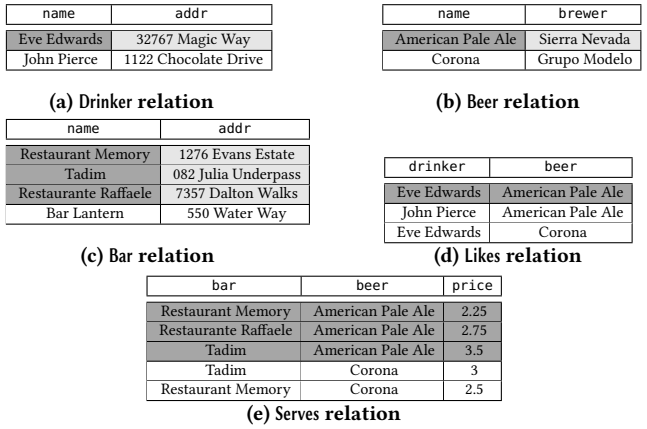
Example 2.5. Consider the Beers database shown in Figure 1 containing information about drinkers (Drinker table), beers (Beer table), bars (Bar table), which drinker likes which beer (Likes table), and which bar serves which beer (Serves table). Suppose a student has been asked to write the following query in some relational language (like SQL, relational algebra, or relational calculus):

Q_A*: Find beers liked by any drinker whose* **first name is 'Eve'** *along with bars that serve them* **at the highest price**.

Suppose instead, they write the following very similar (and a syntactically correct) query:

Q_B*: Find beers liked by any drinker whose* **name starts with 'Eve'** *along with bars that serve them* **not at the lowest price**.

We skip writing the actual queries here (see [25]), but the reader can check that Q_A is a non-monotone and more complex query, whereas Q_B is a monotone and simpler query requiring a self-join on the Serves table. For students or analysts who are learning SQL for the first time, or are trying to write more efficient equivalent queries for better performance, it is easy to make this type of mistakes.



name	addr
Eve Edwards	32767 Magic Way
John Pierce	1122 Chocolate Drive

(a) Drinker relation

name	brewer
American Pale Ale	Sierra Nevada
Corona	Grupo Modelo

(b) Beer relation

name	addr
Restaurant Memory	1276 Evans Estate
Tadim	082 Julia Underpass
Restaurante Raffaele	7357 Dalton Walks
Bar Lantern	550 Water Way

(c) Bar relation

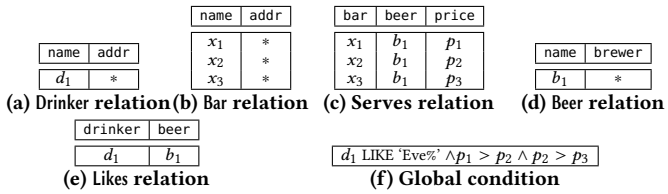
drinker	beer
Eve Edwards	American Pale Ale
John Pierce	American Pale Ale
Eve Edwards	Corona

(d) Likes relation

bar	beer	price
Restaurant Memory	American Pale Ale	2.25
Restaurante Raffaele	American Pale Ale	2.75
Tadim	American Pale Ale	3.5
Tadim	Corona	3
Restaurant Memory	Corona	2.5

(e) Serves relation

Figure 1: A database instance D_0 of the Beers dataset with five relations. The highlighted rows are sufficient to distinguish the wrong query from the correct query, and the light highlighted cells contain redundant information



name	addr
d_1	*

(a) Drinker relation

name	addr
x_1	*
x_2	*
x_3	*

(b) Bar relation

bar	beer	price
x_1	b_1	p_1
x_2	b_1	p_2
x_3	b_1	p_3

(c) Serves relation

name	brewer
b_1	*

(d) Beer relation

drinker	beer
d_1	b_1

(e) Likes relation

d_1 LIKE 'Eve%' $\wedge p_1 > p_2 \wedge p_2 > p_3$

(f) Global condition

Figure 2: A conditional instance $\mathcal{I}_0$ that distinguishes the correct and wrong queries in Example 2.5 and generalizes the highlighted cells in Figure 1.

The HNRQ project aims to help users understand such mistakes and debug their queries. In this project we collaborate with colleagues with expertise in CS Education and evaluate our solutions by user studies with the help of students with a varied level of experience in writing relational queries.

2.2.1 Explaining wrong queries with counterexamples. Suppose we have a hidden test database D available, which detects that Q_B in Example 2.5 is wrong since $Q_B(D) \neq Q_A(D)$ (e.g., if an auto-grader with a carefully constructed test instance is used). These test databases can be big to be able to detect different types of errors, and simply showing the database to users may not be enough to help them understand how their query was wrong. Consider the instance D_0 in Figure 1. The reader can verify that $Q_B(D_0) \neq Q_A(D_0)$. In particular, Q_B returns the tuples (Restaurante Raffaele, American Pale Ale), (Tadim, American Pale Ale), and (Tadim, Corona) while Q_A only returns the latter two tuples. In [48, 49], we developed a tool called *RATest* for explaining wrong relational algebra (RA) queries by providing a small database instance as a *counterexample* that differentiates the wrong query from the correct query. For instance, in Figure 1, the entire database is not needed to convince the user that their query is wrong, only showing the highlighted rows is sufficient for this purpose. We formulated this as an optimization problem: *given D such that $Q_B(D) \neq Q_A(D)$, can we compute the smallest subinstance $D' \subseteq D$ such that $Q_B(D') \neq Q_A(D)'$ holds?* We studied the complexity of this problem for different query classes.

Although the problem was NP-hard in general for non-monotone queries like Q_A, and even conceptually more difficult for queries with aggregates and group-by, we obtained practical solutions using data provenance and SAT/SMT (Satisfiability Modulo Theory) solvers. We conducted a detailed user study in an undergraduate database course with about 170 students and analyzed its usefulness in helping students debug their queries. In addition, the feedback we received from students was overwhelmingly positive, like, *"It was incredibly useful debugging edge cases in the larger dataset not provided in our sample dataset with behavior not explicitly described in the problem set.", "Overall, very helpful and would like to see similar testers for future assignments.", "I liked how it gave us a concise example showing what we did wrong."*, etc. Since then, RATest has been used by more than 1000 students in undergraduate and graduate database courses at Duke. We are working on making other tools developed in the HNRQ project available in our database classes.

2.2.2 Abstract conditional instances for explaining wrong queries. Small concrete database instances as counterexamples help students understand how their queries are wrong to some extent, but we observed multiple prospects for improvement. Again consider Figure 1: just showing the highlighted tuples does not explain *how* the query Q_B went wrong. Further, the cells that are lightly highlighted are completely redundant in explaining how Q_B is wrong. In addition, [48] starts with a given database instance differentiating the queries that may not be available (it does not check query equivalence). Finally, there may be multiple mistakes in a wrong query (see the two boldfaced parts in Q_B), and a given counterexample may not highlight all of them.

Therefore, in our follow-up work [25], we explored how we can output a set of *conditional instances* (c-instances) explicitly showing the differences between the queries Q_A and Q_B. Figure 2 shows a c-instance $\mathcal{I}_0$ that again differentiates Q_A and Q_B. However, $\mathcal{I}_0$ shows abstract tuples with variables instead of constants ($*$ are 'don't care' variables) and a condition that the variables must satisfy (there should be a drinker whose name starts with 'Eve' and the order of the prices in Serves table should be $p_1 > p_2 > p_3$). Thus, $\mathcal{I}_0$ not only generalizes the counterexample in Figure 1 (i.e., there exists an assignment to the variables that results in the instance in Figure 1 and satisfies the global condition), but, it also specifies the 'minimal' condition for which Q_B differs from Q_A (the global condition). This is one of the c-instances in our *universal solution* [25]; each c-instance captures some aspect of the difference between Q_B and Q_A– another c-instance $\mathcal{I}_1$ will contain only the first two tuples in Serves, and the global condition d_1 LIKE 'Eve%' $\wedge\neg$(d_1 LIKE 'Eve␣%') $\wedge p_1 > p_2$, suggesting that two queries will be different on an instance where the name of the drinker starts with 'Eve' rather than the first name being 'Eve'. Our user study with undergraduate and graduate students shows that although both [25, 48] help users detect errors, conditional instances help users detect multiple errors in wrong queries unlike concrete instances provided by [48].

2.2.3 Tracing the output of SQL queries. Our RATest system [48, 49] only supports RA (with some aggregates, having, and group-by) and can show the intermediate results from each operator explaining how an output is generated. When we extended the smallest counterexample computation to SQL, there were new challenges in showing how the output of a SQL query is computed, especially for queries with correlated/nested subqueries and outerjoins. Unlike *procedural* RA that specifies *how* the query is evaluated, SQL is *declarative* (only *what* needs to be computed is specified). Therefore, it is not obvious how to trace the output from a declarative SQL query, because its execution plan often differs from how it was originally written. This required addressing new challenges in generating counterexamples in the backend as well as developing a novel user interface in the frontend for tracing outputs. We developed the iRex system (demonstrated in [46], research paper in preparation) to help users trace SQL query evaluation and debug SQL queries interactively. This system will be deployed soon in our classes at Duke and a thorough evaluation will be performed.

2.3 Other Work on Explanations

While working on the two main directions in explanations discussed above, several other related problems surfaced, often they are of independent interest and has other applications. In [36, 37], we explored explanations for *contentions in shared clusters* when components coming from different queries like jobs and parallel tasks therein compete for different resources like CPU, memory, and network, by assignment of *blame* at different granularity at the levels of tasks, jobs, and queries. For explanations for aggregate queries, we observed that the search space of possible explanations is huge, and directly showing the top-k explanations by some scoring function may have repetitive information missing other interesting insights. Inspired by this problem, we studied optimizations for *interactive exploration and summarization of aggregate query answers*, where the user can see clusters of predicates on input attributes satisfying certain parameters for *diversity* (in terms of distance among the clusters), *relevance* (alignment with the original scoring function), and *coverage* (for top tuples by the original ranking), and visualize how changes in parameters affect clustering [69, 70]. In our first work on explanations [58], we used interventions, which can be defined as the set of input tuples that are *'affected'* by an explanation predicate, and these input tuples in turn may affect other tuples in the database recursively. Inspired by this idea, we explored *delta rules* [23, 24] in the context of data repair, when repairing tuples in one relation leads to cascaded updates in other relations, e.g., by foreign keys or by SQL triggers. We developed a generalized framework for such cascaded updates by a set of user-defined update rules and studied their possible semantics with algorithms and complexity analysis. In the other direction, to understand how to achieve a certain degree of intervention in aggregate query answers with minimal changes in the input, we studied the problem of *aggregated deletion propagation*, which may have applications in measuring robustness of query answers [34]. One key aspect in the work on explanations is scalability and interactivity, for which we often had to relax the requirement of getting the optimal solution in favor of more efficient solutions by heuristics [58]. In such interactive data analysis, users are likely to be interested in examining a subset of query answers or explanations satisfying some properties, This led to our work [66] on query optimization for complex queries with *having clauses* and other filtering predicates that filter out certain output tuples (called *iceberg queries* [21]). Such problems inspired our other work on query optimizations and data repair that we do not discuss in this paper [33, 43, 44, 64, 67].

3 CAUSAL ANALYSIS ↔ DATA MANAGEMENT: SCALABLE SOUND CAUSAL ANALYSIS FOR COMPLEX OBSERVATIONAL DATA

Our first work [58] on explaining aggregate queries was motivated by causality and intervention, but none of the explanations discussed in Section 2 are *'causal'*. All the approaches described in Section 2 mine interesting patterns that only intend to *'explain'* the questions asked by the user on the query answers in different ways, and give useful insights to understand and improve a data-driven process. '*Causal explanations*' might be considered as the gold standard for explanations in many applications, for instance, in our examples in Section 2, it would be nice to find 'what *caused* the number of database papers from industry to drop after year 2000', or 'what *caused* the number of battery crimes to decrease in a year in a Chicago area'. However, understanding causality and to be able to do a formal and sound causal analysis go way beyond causal explanations. Our research in explanations and pursuit for causal explanations introduced us to the rich research area of *causal analysis* in AI and also in statistics, and revealed a plethora of research directions on how causal analysis and data management research can make each other stronger.

Causal analysis – i.e., estimating the effect of a *treatment* on an *outcome* – lays the foundation of sound and robust policy making by providing a means to estimate the impact of a certain intervention to the world. It forms the stepping stone for **actionable data analysis** that correlation, association, or model-based prediction analysis cannot provide, and is practically indispensable in health, medicine, social sciences, and other domains (e.g., whether a new drug is effective in curing a disease, whether a job-training program helps improve employment prospects, or, whether giving incentives for not smoking in terms of reduction in insurance premium helps people quit smoking). For instance, if non-smokers pay reduced insurance premium anyway, and introducing the plan of reduced premium does not help smokers quit smoking, then a simple correlation analysis between people who pay less premium and who do not smoke may not be sufficient to convince policy makers in the government or in insurance companies that the new policy should be introduced – as cited widely in statistical studies, *correlation does not imply causation*.

Two popular models used in causal inference in statistical studies, AI, social science, or network analysis are *Rubin's potential outcome framework* [59] and *Pearl's graphical causal model* [55]. There are some *'units'* (e.g., patients) in either the *'treatment'* group (e.g., who receives a drug) or the *'control'* group (e.g., who receives a placebo). If the units were assigned to treatment or control group at random, as done in *randomized controlled experiments*, then the difference in the average outcome of these two groups gives a measure of causal effect of the treatment on the outcome. This does not hold when we cannot do randomized experiments (e.g., due to ethical reasons to answer questions like effect of smoking on health) and only can do an *'observational study'* based on the data available to us. Although the term *causality* has been overloaded and used in different applications for data management, *true causal inference* (e.g., inferring whether smoking causes lung diseases) was largely unexplored by the data management community until very recently. Similarly, the causal analysis community did not take much advantage of the convenient and robust data management techniques to make observational causal inference scalable and applicable to more complex data. Our research brings together techniques in databases, statistics, AI, and machine learning to develop causal analysis techniques for large complex datasets.

3.1 Interpretable Matching Algorithms and Scalable Data Management Techniques

In observational studies, since there is no guarantee on how the treatment was assigned, one popular and interpretable approach to handling observational data is *'matching'*. Matching in causal inference accounts for possible confounding covariates (that can affect treatment assignment) by matching treatment and control units on the same or similar values of the covariates and then studying their difference in the outcome. Therefore, the data analyst is able to see the matches of a treated unit that led to the estimation of its treatment effect, which leads to easy debugging and decision making (unlike dimensionality reduction techniques like *propensity scores* [56] and regression techniques that are uninterpretable and require the model to be specified correctly). However, in high dimensional data, exact matches on all covariates leaves most of the units unmatched, and produces confusing results by matching on irrelevant covariates. In our work on observational causal analysis, we have developed a number of scalable and interpretable *almost exact matching* methods that we describe below.

3.1.1 Almost exact matching algorithms FLAME and DAME. In our interdisciplinary *Almost Matching Exactly (AME)* lab [2], in collaboration with colleagues in machine learning and statistics, we developed two data-adaptive and interpretable approaches to matching for causal inference called *FLAME* (Fast Large-scale Almost Matching Exactly) [68] and *DAME* (Dynamic Almost Matching Exactly) [18]. DAME solves an optimization problem that matches units on as many covariates as possible by learning a distance metric, prioritizing matches on important covariates and utilizing pruning of the search space using an approach similar to the seminal *apriori algorithm* for frequent itemset mining [1]. FLAME approximates the solution found by DAME via a much faster backward feature selection procedure, and leverages techniques that are natural for query processing in the area of database management; the first implementation uses SQL queries that have been optimized through several decades (suitable for larger datasets), and the second one uses bit-vector techniques (suitable for smaller datasets). We showed with extensive evaluation that not only DAME and FLAME qualitatively improve over other state-of-the-art matching and other techniques for causal analysis, FLAME also scales to huge datasets with more than a million units where existing state-of-the-art methods fail. One option to use the better matching quality of DAME and better efficiency of FLAME is to run hybrid FLAME-DAME, which first removes obviously unimportant variables by running FLAME, then runs DAME to select variables to remove more carefully in the later iterations. We illustrated how FLAME can be used to obtain interpretable estimates of treatment effects for an important societal question: the effect of prenatal smoking on the health of the newborn child in terms of birth weight and NICU admissions using the Natality dataset [13], and presented another case study for DAME

on the causal effect of participation in the *Breaking the Cycle* social program [29] on reducing non-drug future arrest rates.

We maintain several Python and R packages of our algorithms with detailed documentation (available from our AME lab website [2]) that are being used in different applications by consumers of causal inference research.

3.1.2 Extensions and applications of almost exact matching methods. The algorithms FLAME and DAME had several applications for related problems in causal inference. We studied extensions of FLAME/DAME to data from *randomized experiments on network structures* [5] (treatment applied to a unit can affect the outcomes of its neighbors) by matching units almost exactly on counts of unique subgraphs within their neighborhood graphs. We also did a case study about factors affecting election participation and self-help-group participation using a dataset collected from several villages in India [7]. Moreover, we studied extensions of FLAME/DAME for *instrumental variables* [4] with an application to political canvassing. Instrumental variables correlate with the treatment assignment, affect the outcome only through their effect on the treatment, and are commonly used to reduce the effects of unmeasured confounding variables that affect treatment assignments. For datasets with continuous covariates, AME methods may not work well due to large number of distinct values. Hence we developed an *adaptive hyper-box matching* method [52] that matches units with others in unit-specific, hyper-box-shaped regions of the covariate space, with a study on the LaLonde dataset [39] on the effect of work training programs on future earnings. These regions are large enough such that many matches are created for each unit, and small enough such that the treatment effect is roughly constant throughout.

3.1.3 An application of interpretable matching algorithms for causal inference on evaluation of pre-trial programs. As a concrete application of causal inference and our matching algorithms, in a recent work [61], with colleagues from the Law department, we studied the effectiveness of participation in a pre-trial community supervision program on new criminal charges. We used multiple methods of observational causal inference to conduct this evaluation, including popular propensity score matching [56] and our FLAME/DAME matching algorithms. All methods of observational causal inference provided an estimated average treatment effect that is approximately zero, indicating that we could not conclude from this dataset that the program is effective at reducing new criminal charges in this community. In addition, the interpretable FLAME/DAME methods automatically performed important covariate selection and provided insights into how the matching and final results have been obtained. If this finding replicates across other datasets, after rigorously evaluating other pre-trial programs, policymakers may consider alternative strategies in managing low-risk populations or might consider focusing their resources on higher-risk populations instead. A rigorous causal inference analysis assists in actionable data analysis, which explanations, correlations, or predictions may not be able to provide.

3.2 Causal Analysis with Relational Data

Bringing data management and causality research together does not stop at scalable causal analysis using database queries. The traditional causal inference research primarily assumes the units to be homogeneous (all having the same covariates, treatment, and outcome variables) apart from some recent work on causal inference when homogeneous units are connected in a network (e.g., [53]). Therefore, new methods are needed to be developed to allow causal analysis on complex *'relational data'* where multiple tables contain different types of entities and relationships among them.

3.2.1 Causal relational learning. In many applications, data analysis is needed to be performed on large datasets with a number of inter-related tables, often with primary keys-foreign keys and other dependencies (e.g., hospitals - patients - providers - treatment - insurance data, product/business - customers - review data, authors - submissions - authorship - review data, or, students - enrollment - courses - taught-by - instructors data). We have been working on extending Rubin's and Pearl's models for causal inference [55, 59] for such relational data. The challenge with relational datasets is that some crucial assumptions needed for most of the existing causal inferences approaches (e.g., both treatment and outcome apply to the same unit, only one treatment level, and no interference among units) may no longer hold on them. As an example on students, courses, and instructors, a treatment of a special training can be applied to the instructors whereas the outcomes may be observed on students in terms of their grades, and there is a many-to-many relationship between the entities students and instructors through enrollment and course offerings. To make the causal inference process simpler and sound on relational data, in our work called *CARL* (*Causal Relational Learning*) [60], we developed a declarative framework to answer a number of possible causal questions on such data. We showed the effectiveness of CARL on real datasets and interesting causal questions. Here is an example.

Example 3.1. In [60], we studied the effect of authors' prestige on the scores they receive in single-blind and double-blind review processes using a paper review dataset [54]. We observed that there is a strong correlation between authors' prestige (from the ranking of the schools they belong to) and the review scores of the papers co-authored by them in both single-blind and double-blind reviews, which is probably as expected as papers from higher-ranked institutions are likely to be of good quality in general. However, when we do causal analysis, there is a significant impact of their prestige on the review scores for single-blind reviews and *not* for double-blind reviews, which illustrates the difference between correlation and causation, and may encourage conference organizers to adopt double-blind review processes for the sake of fairness. We also had other case studies in [60]: in the hospital-stay critical-care MIMIC-III dataset [35], we studied the effect of not having insurance on patient's mortality and length of hospital stay, and in another hospital-stay dataset (called NIS [30]), the effect of the size of a hospital on affordability of healthcare.

3.2.2 Answering what-if and how-to queries with causal dependencies of attributes. Hypothetical reasoning by answering *what-if* and *how-to* queries is a well-studied problem in core database research (e.g., [6, 17, 31, 38, 45]) with applications in decision-making and risk assessment in businesses. These queries aim to understand how (hypothetical) modifications of the input tuples affect the output tuples and vice versa. For instance, suppose we have a database

containing information about different products and reviews about products. An analyst may ask a what-if query: *'what would be the effect of increasing the price of Asus laptops by 10% on their average ratings?'*, or, a how-to query: *'how to maximize the average rating of laptops by updating their price such that the new price is at most 100 away from its original value?'*. In a recent work [22], as an application of causal analysis on relational data described above [60], we developed a novel probabilistic framework called *HYPER* for hypothetical reasoning in relational databases, which accounts for *collateral effects* of hypothetical updates in terms of *causal dependencies among attributes and tuples*. For instance, changing states in addresses can change tax rates, changing price of a product can affect the ratings of several related products, and making changes to one's social network may affect one's political inclinations. Such causal dependencies can be modeled by a *causal DAG* [55] that captures the effects of *exogenous* variables by a probability distribution on the *endogenous* variables. A 'what-if' question can be considered as an intervention on this *probabilistic causal model* that changes the distribution on the outcome variables, and in turn changes the distribution on the query answers, whereas a 'how-to' question can be modeled as an optimization problem over several satisfying what-if queries. We used techniques from causal inference, probabilistic databases, and solvers for integer programming to obtain efficient solutions for what-if and how-to queries.

4 FUTURE WORK AND CONCLUSIONS

With the recent focus on responsible data science in data management and ML/AI communities, explanations and causality are broad and timely topics with many exciting future directions both within data management research and of interdisciplinary nature. We are trying to bridge the gap between our work on explanations and causal inference, by computing *causal explanations* for aggregate and non-aggregate queries. We are also exploring explanations for other phases in data analysis like data cleaning, data labeling, and private synthetic data generation, and explanations using *Shapley values* [42, 62] for network data. For explaining wrong queries, we are working on generating useful *hints* that would allow a user fix a query. For causal analysis, we are working on *treatment effect modifiers* to understand the effect of a treatment on different subpopulations, and on the *sensitivity* of observational causal analysis methods in terms of the availability of confounding covariates that can affect the treatment assignment. In the long run, we envision a system that can combine different notions of explanations and causality with confidence measures, incorporate 'common knowledge' from knowledge bases, and output explanations in a form using natural language that is understandable and useful for decision making to a broad range of users. Combining explanations and causality with classical database systems and theory research, we will continue our pursuit toward effective, provable, interpretable, and actionable data analysis approaches in the future.

ACKNOWLEDGEMENTS

This work was primarily done by a group of outstanding students and postdocs: Amir Gilad, Zhengjie Miao, Harsh Parikh, Prajakta Kalmegh, Yuchao Tao, Marco Morucci, Vittorio Orlandi, Tianyu Wang, Chenjie Li, Qitian Zeng, Sainyam Galhotra, Xiao Hu, Brett Walenz, Yuhao Wen, Xiaodan Zhu, Travis Seale-Carlisle, Saksham Jain, Moe Kayali, Ester Livshits, Yameng Liu, M. Usaid Awan, Yihao Hu, Tiangang Chen, Andrew Lee, Shouzhuo Sun, Awa Dieng, Shweta Patwa, Laurel Orr, and many other graduate and undergraduate students. This work is the result of collaboration with a number of amazing collaborators: Jun Yang, Dan Suciu, Cynthia Rudin, Alexander Volfovsky, Ashwin Machanavajjhala, Boris Glavic, Babak Salimi, Lise Getoor, Shivnath Babu, Debmalya Panigrahi, Kristin Stephens-Martinez, Daniel Deutch, Benny Kimelfeld, and many others. I am grateful to Sayan Mukherjee for introducing me to the observational causal inference literature in statistics. This research was supported in part by NSF awards IIS-1552538, IIS-1703431, IIS-2008107, IIS-2147061, and NIH award R01-EB025021.

REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 487–499. Morgan Kaufmann, 1994.

[2] Almost Matching Exactly Lab, Duke University. https://almost-matching-exactly.github.io/.

[3] Y. Amsterdamer, D. Deutch, and V. Tannen. Provenance for aggregate queries. In *PODS*, pages 153–164, 2011.

[4] M. U. Awan, Y. Liu, M. Morucci, S. Roy, C. Rudin, and A. Volfovsky. Interpretable almost matching exactly with instrumental variables. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*, page 410, 2019.

[5] M. U. Awan, M. Morucci, V. Orlandi, S. Roy, C. Rudin, and A. Volfovsky. Almost-matching-exactly for treatment effect estimation under network interference. In S. Chiappa and R. Calandra, editors, *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 3252–3262. PMLR, 2020.

[6] A. Balmin, T. Papadimitriou, and Y. Papakonstantinou. Hypothetical queries in an OLAP environment. In *VLDB*, pages 220–231, 2000.

[7] A. Banerjee, A. G. Chandrasekhar, E. Duflo, and M. O. Jackson. The diffusion of microfinance. *Science*, 341(6144):1236498, 2013.

[8] P. Buneman, S. Khanna, and W. C. Tan. Why and where: A characterization of data provenance. In *Proc. 8th International Conference on Database Theory*, ICDT '01, pages 316–330, 2001.

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

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

[11] Dataset: Chicago Crime Data. https://data.cityofchicago.org/. *Chicago Data Portal.*

[12] Dataset: DBLP database. https://dblp.uni-trier.de/xml/. *Michael Ley and Schloss Dagstuhl.*

[13] Dataset: Natality. http://www.cdc.gov/nchs/data_access/ftp_data.htm. *Birth Records in the United States in 2010 from Centers for Disease Control and Prevention (CDC) and National Center for Health Statistics (NCHS).*

[14] Dataset: NBA Data. https://www.nba.com/.

[15] Dataset: NSF Awards. http://www.nsf.gov/awardsearch/download.jsp. *National Science Foundation.*

[16] Dataset: Play by Play Data. https://www.pbpstats.com/.

[17] D. Deutch, Z. G. Ives, T. Milo, and V. Tannen. Caravan: Provisioning for what-if analysis. In *CIDR*, 2013.

[18] A. Dieng, Y. Liu, S. Roy, C. Rudin, and A. Volfovsky. Interpretable almost-exact matching for causal inference. In *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, pages 2445–2453, 2019.

[19] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[20] K. El Gebaly, P. Agrawal, L. Golab, F. Korn, and D. Srivastava. Interpretable and informative explanations of outcomes. *Proc. VLDB Endow.*, 8(1):61–72, sep 2014.

[21] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing iceberg queries efficiently. In A. Gupta, O. Shmueli, and J. Widom, editors, *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA*, pages 299–310. Morgan Kaufmann, 1998.

[22] S. Galhotra, A. Gilad, S. Roy, and B. Salimi. Hyper: Hypothetical reasoning with what-if and how-to queries using a probabilistic causal approach. In Z. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 1598–1611. ACM, 2022.

[23] A. Gilad, D. Deutch, and S. Roy. On multiple semantics for declarative database repairs. In D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 817–831. ACM, 2020.

[24] A. Gilad, Y. Hu, D. Deutch, and S. Roy. Muse: Multiple deletion semantics for data repair. *Proc. VLDB Endow.*, 13(12):2921–2924, 2020.

[25] A. Gilad, Z. Miao, S. Roy, and J. Yang. Understanding queries by conditional instances. In Z. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 355–368. ACM, 2022.

[26] B. Glavic, A. Meliou, and S. Roy. Trends in explanations: Understanding and debugging data-driven systems. *Found. Trends Databases*, 11(3):226–318, 2021.

[27] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Min. Knowl. Discov.*, 1(1):29–53, Jan. 1997.

[28] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *Proceedings of the Twenty-sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '07, pages 31–40, 2007.

[29] A. V. Harrell, D. Marlowe, and J. Merrill. Breaking the cycle of drugs and crime in Birmingham, Alabama, Jacksonville, Florida, and Tacoma, Washington, 1997-2001. 2006.

[30] Healthcare Cost and Utilization Project (HCUP). HCUP Nationwide Inpatient Sample (NIS), 2006.

[31] H. Herodotou and S. Babu. Profiling, what-if analysis, and cost-based optimization of mapreduce programs. *Proc. VLDB Endow.*, 4(11):1111–1122, 2011.

[32] HNRQ (Helping Novices Learn and Debug Relational Queries) Project Webpage. https://dukedb-hnrq.github.io/.

[33] X. Hu, Y. Liu, H. Xiu, P. K. Agarwal, D. Panigrahi, S. Roy, and J. Yang. Selectivity functions of range queries are learnable. In Z. Ives, A. Bonifati, and A. E. Abbadi, editors, *SIGMOD '22: International Conference on Management of Data, Philadelphia, PA, USA, June 12 - 17, 2022*, pages 959–972. ACM, 2022.

[34] X. Hu, S. Sun, S. Patwa, D. Panigrahi, and S. Roy. Aggregated deletion propagation for counting conjunctive query answers. *Proc. VLDB Endow.*, 14(2):228–240, 2020.

[35] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[36] P. Kalmegh, S. Babu, and S. Roy. 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*, pages 918–935, 2019.

[37] P. Kalmegh, H. Lundberg, F. Xu, S. Babu, and S. Roy. iqcar: A demonstration of an inter-query contention analyzer for cluster computing frameworks. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1721–1724, 2018.

[38] L. V. S. Lakshmanan, A. Russakovsky, and V. Sashikanth. What-if OLAP queries with changing dimensions. In *ICDE*, pages 1334–1336, 2008.

[39] R. J. LaLonde. Evaluating Econometric Evaluations of Training Programs with Experimental Data. *The American Economic Review*, pages 604–620, 1986.

[40] C. Li, J. Lee, Z. Miao, B. Glavic, and S. Roy. Cajade: Explaining query results by augmenting provenance with context. In *VLDB*, 2022.

[41] C. Li, Z. Miao, Q. Zeng, B. Glavic, and S. Roy. Putting things into context: Rich explanations for query answers using join graphs. In *Proceedings of the 2021 International Conference on Management of Data, SIGMOD Conference 2021*, 2021.

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

[43] E. Livshits, B. Kimelfeld, and S. Roy. Computing optimal repairs for functional dependencies. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Houston, TX, USA, June 10-15, 2018*, pages 225–237. ACM, 2018.

[44] E. Livshits, R. Kochirgan, S. Tsur, I. F. Ilyas, B. Kimelfeld, and S. Roy. Properties of inconsistency measures for databases. 2021.

[45] A. Meliou and D. Suciu. Tiresias: the database oracle for how-to queries. In *SIGMOD*, pages 337–348, 2012.

[46] Z. Miao, T. Chen, A. Bendeck, K. Day, S. Roy, and J. Yang. I-rex: An interactive relational query explainer for SQL. *Proc. VLDB Endow.*, 13(12):2997–3000, 2020.

[47] Z. Miao, A. Lee, and S. Roy. Lensxplain: Visualizing and explaining contributing subsets for aggregate query answers. *Proc. VLDB Endow. (demonstration)*, 12(12):1898–1901, 2019.

[48] Z. Miao, S. Roy, and J. Yang. 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*, pages 503–520, 2019.

[49] Z. Miao, S. Roy, and J. Yang. 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 (demonstration)*, pages 1961–1964, 2019.

[50] Z. Miao, Q. Zeng, B. Glavic, and S. Roy. 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*, pages 485–502, 2019.

[51] Z. Miao, Q. Zeng, C. Li, B. Glavic, O. Kennedy, and S. Roy. CAPE: explaining outliers by counterbalancing. *Proc. VLDB Endow. (demonstration)*, 12(12):1806–1809, 2019.

[52] M. Morucci, V. Orlandi, S. Roy, C. Rudin, and A. Volfovsky. Adaptive hyperbox matching for interpretable individualized treatment effect estimation. In R. P. Adams and V. Gogate, editors, *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2020, virtual online, August 3-6, 2020*, volume 124 of *Proceedings of Machine Learning Research*, pages 1089–1098. AUAI Press, 2020.

[53] E. L. Ogburn, O. Sofrygin, I. Diaz, and M. J. van der Laan. Causal inference for social network data, 2017.

[54] OpenReview. https://openreview.net.

[55] J. Pearl. *Causality: models, reasoning, and inference*. Cambridge University Press, 2000.

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

[57] S. Roy, L. Orr, and D. Suciu. Explaining query answers with explanation-ready databases. *PVLDB*, 9(4):348–359, 2015.

[58] S. Roy and D. Suciu. A formal approach to finding explanations for database queries. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 1579–1590, 2014.

[59] D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331, 2005.

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

[61] T. M. Seale-Carlisle*, S. Jain*, C. Lee, C. Levenson, S. Ramprasad, B. Garrett, S. Roy, C. Rudin, and A. Volfovsky. Interpretable matching algorithms for causal inference for evaluating community supervision programs. *Manuscript*, 2022.

[62] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):310–317, 1953.

[63] Y. Tao, A. Gilad, A. Machanavajjhala, and S. Roy. Dpxplain: Privately explaining aggregate query answers. *Manuscript*, 2022.

[64] Y. Tao, X. He, A. Machanavajjhala, and S. Roy. Computing local sensitivities of counting queries with joins. In D. Maier, R. Pottinger, A. Doan, W. Tan, A. Alawini, and H. Q. Ngo, editors, *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*, pages 479–494. ACM, 2020.

[65] M. Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing, May 5-7, 1982, San Francisco, California, USA*, pages 137–146, 1982.

[66] B. Walenz, S. Roy, and J. Yang. Optimizing iceberg queries with complex joins. In *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*, pages 1243–1258, 2017.

[67] B. Walenz, S. Sintos, S. Roy, and J. Yang. Learning to sample: Counting with complex queries. *Proc. VLDB Endow.*, 13(3):390–402, 2019.

[68] T. Wang, M. Morucci, M. U. Awan, Y. Liu, S. Roy, C. Rudin, and A. Volfovsky. Flame: A fast large-scale almost matching exactly approach to causal inference. *Journal of Machine Learning Research*, 22(31):1–41, 2021.

[69] Y. Wen, X. Zhu, S. Roy, and J. Yang. Interactive summarization and exploration of top aggregate query answers. *Proc. VLDB Endow.*, 11(13):2196–2208, 2018.

[70] Y. Wen, X. Zhu, S. Roy, and J. Yang. Qagview: Interactively summarizing high-valued aggregate query answers. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1709–1712, 2018.

[71] Wikipedia article on Explanations. https://en.wikipedia.org/wiki/Explanation.

[72] J. Woodward and L. Ross. Scientific Explanation. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Summer 2021 edition, 2021.

[73] E. Wu and S. Madden. Scorpion: Explaining away outliers in aggregate queries. *Proc. VLDB Endow. (PVLDB)*, 6(8):553–564, June 2013.