Journal of Computers Vol. 34 No. 2, April 2023, pp. 81-97
doi: 10.53106/199115992023043402007

Annolog: A Query Processing Framework for Modelling and Reasoning with Annotated Data

Haochen Zou*, Dejian Wang, Yang Xiao

Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada
{haochen.zou, dejian.wang, yang.xiao}@mail.concordia.ca

Received 27 May 2022; Revised 20 August 2022; Accepted 19 September 2022

Abstract. Data annotation is the categorization and labelling of data for applications, such as machine learning, artificial intelligence, and data integration. The categorization and labelling are done to achieve a specific use case in relation to solving problems. Existing data annotation systems and modules face imperfections such as knowledge and annotation not being formally integrated, narrow application range, and difficulty to apply on existing database management applications. To analyze and process annotated data, obtain the relationship between different annotations, and capture metainformation in data provenance and probabilistic databases, in this paper, we design a back-end query processing framework as a supplementary interface for the database management system to extend operation to datasets and boost efficiency. The framework utilizes Java language and the MVC model for development to achieve lightweight, cross-platform, and high adaptability identities. The contribution of this paper is mainly reflected in two aspects. The first contribution is to implement query processing, provenance semiring, and semiring homomorphism over annotated data. The second contribution is to combine query processing and provenance with SQL statements in order to enable the database manager to invoke operations to annotation.

Keywords: knowledge base system, data management system, data provenance, annotated data, query processing framework

1 Introduction

With the evolvement of applications, the scale of data keeps expanding. After obtaining the results of the query data, it is necessary to obtain an explanation of the query answers such as the description of data origin and the process of data acquisition. The extra explanation forms various types of information within the data itself. Metadata refers to the contextual information that establishes relationships between the data and the real-world aspects it applies to [1]. In other words, metadata is the data that describes information about a piece of data [2], thereby creating a relationship in terms of the content and functionality of that data [3]. Annotation is a type of metadata that keeps extra information about data [4]. In the datasets of a realistic project, the annotations can either represent multiplicities or probabilities. Each atomic formula in a rule or fact is associated with an annotated value. Annotated data contain descriptions and explanations of the data element. Data in various formats such as video, images, or text are labelled in the annotated data so that machines can understand it. Therefore, in the field of industry and academia, annotated data are crucial for understanding the input patterns in order to further process and produce accurate results. Based on the above-discussed requirements to analyze and process the annotated data and obtain the relationship between different types of annotations such as text annotation, sentiment annotation, intent annotation, semantic annotation, and named entity annotation [5-6], we have developed a framework called Annolog as a conservative extension of Datalog to model and reason with annotated data. The rules and facts in Annolog are the same forms as in the standard Datalog. Datalog is a database query language that syntactically is a subset of the Prolog programming language [7]. Each formula in Datalog is a function-free Horn clause [8]. Evaluation of Datalog queries on a finite database terminates in finite time-polynomial in the number of constants. Datalog has been the subject of numerous studies in the research domain of database management. The declarative advantages of Datalog together with its powerful query processing and optimization techniques have made it attractive to database research and development [9-10]. Consequently, the Datalog

* Corresponding Author

structure and paradigm are selected as the underlying infrastructure of the Annolog framework developed in this paper.

The key research problem in this paper is to satisfy the requirement of stakeholders and database managers to process the annotated data for further analysis. Specifically, the demands include the complete semantics, methodologies, and formulas on the annotated data in the relational databases, the operations to model the Select, Project, Join, and Union (SPJU) functions with relation features to the annotated data, and the capability to process the annotated data in various types of databases without affecting the functionality of the database management system itself. To address the pivotal research problem, Annolog is designed in this paper for usage either as a standalone reasoner or to work synchronously with the database management system engine. When evaluating a query on annotated data, it is advantageous to capture meta-information about the result of the query along with the result itself [11]. The annotated meta-information indicates the multiplicity and probability of the results, and why as well as how the outputs are generated. Annolog facilitates to capture of annotated meta-information in data provenance and probabilistic databases. As a standalone reasoner, Annolog can be part of the big data analysis module in applications such as decision support and result classification. Annolog performs as a supplement to the general rule-based database system utilized in big data analysis. By processing the annotation information of the datasets, we integrate data in an orderly manner, provide detailed information behind data, and enhance reason efficiency which is essential for making strategic decisions in applications in the field of education and healthcare. In order to meet the requirements of lightweight and expansibility, the Annolog framework module is implemented by adopting the MVC (Model-View-Controller) design pattern based on the Java platform to separate the inner side logic and outer display. Hence, the framework has the characteristics of cross-platform and high adaptability.

The main contributions, novelties, and achievements of this paper can be summarized as follows.

1. This paper defines semantics, query processing methodologies, provenance semirings, and semiring homomorphisms on the annotated data in the relational databases. These approaches can be considered complementary methods for analyzing annotated data in the databases from the database management systems.
2. A query processing framework is designed for modelling the annotation data in the database. The framework is able to realize the Select, Project, Join, and Union (SPJU) functions with bag, probability, why-provenance, and how-provenance features to the data processing operations. Compared with the existing query processing modules and architecture provided by the industry or researchers, the framework developed in this paper is able to handle the user-defined query formulas instead of processing the annotation data based on the predefined query formulas.
3. The developed query processing framework is consolidated into a lightweight API as an extension to the database management systems. Instead of managing the data elements in the specified types of databases designed in other studies, this framework is able to handle data records in various kinds of databases as long as the database is operated on by the SQL statements from the front-end system.
4. The proposed framework processes the data information parallelly and continuously without affecting the original functions of the database systems and is further evaluated on a simulated medical database and a public database of the number of new confirmed COVID-19 cases per hundred thousand citizens among different administrative areas of a city. The results from the two experiments based on the databases indicate the efficiency, validity, and accuracy characteristics of the framework for practical usage.

The rest of the paper is organized as follows. Section 2 describes the related works. Section 3 presents the methodology of the designed framework. In Section 4, the implementation of the framework is proposed, and the results of the developed experiments are displayed. Section 5 discusses the relative study and future works. Finally, Section 6 concludes the paper.

2 Related Work

Annotation is utilized implicitly or explicitly in people's daily life. The notion of annotated data has been defined and used in different and diverse disciplines. While there is no consensus on a precise definition of the annotation, there is an agreement on its purpose being a type of meta-information that provides meanings to an expression of a data element by describing, supporting, or restricting it. In order to automate such annotated data, it is required to formally represent, manage, process, and analyze the annotated data explicitly and develop desired procedures.

Data provenance is the whole process of data generation and evolution over time and operation accumulation. Traditional data integration technology focuses on the format of data, and the object of management is still the data itself. At present, it is necessary to process data sharing in a huge heterogeneous environment, therefore making the traditional data technology in data integration, query, and index appear inadequate. Data provenance involves the evolution of data that can be applied to track data across and within different data sources. At the same time, uncertainty is inevitable in the process of data evolution, so data provenance can trace the source of data and uncertainty and the evolution process. Since provenance and annotation are strongly associated, this paper focuses on designing a framework to implement data provenance technologies to process the annotated data information. Wang et al. categorize the provenance into three dimensions and proposed data provenance, lineage provenance, and environment provenance for labelling the origin of the data elements [12]. Müller et al. discovered how, where, and why data provenance can help improve the database SQL queries. They demonstrate the provenance analysis of SQL queries with how-provenance which determines which query expressions have been relevant for the evaluation of selected pieces of output data and why-provenance which determines relevant pieces of input data records [13]. Senellart et al. discussed important applications of different forms of data provenance based on the provenance semirings such as the Boolean provenance for probabilistic databases [14]. In this paper, we construct the above-discussed data provenance techniques including bag semantics, how-provenance, why-provenance, Boolean expression, and provenance semiring into the proposed architecture for processing and analyzing the annotated data.

Numerous techniques have been conducted on processing and analyzing annotated data and enriched operations over the provenance of combined data. Benzaken et al. proposed the first provenance-aware extended relational algebra formalized in a proof assistant (Coq) for a non-trivial subset of database queries [15]. Issa et al. introduced a novel framework for encoding inconsistency into relational tuples and tackling query answering for the union of conjunctive queries with respect to a set of denial constraints [16]. Li et al. extended the concept of data provenance in the manufacturing domain to acquire information about the data origin and data changes. They proposed an architecture to manage the provenance of process data, in which the data provenance is considered as annotation of process data [17]. Although the above studies suggest comprehensive provenance representations and extend considerations to databases for usage, they mainly process datasets by the query formulas defined by themselves instead of users' input. In addition, the frameworks these studies developed can only run independently instead of calling on existing data processing programs and data management systems according to forms such as an interface, which makes them difficult to apply to the existing applications.

Several modules and platforms have been proposed as supplementary operations for existing database management systems. Senellart et al. developed an open-source module for the PostgreSQL database management system. The module developed by them adds support for the computation of provenance and probabilities of query results [11]. Garima et al. presented a framework named HUKA that uses provenance polynomials for tracking the derivation of query results over knowledge graphs by encoding the edges involved in generating the answer [18]. Arab et al. presented an overview of GProM, a generic provenance middleware for relational databases. The system supports diverse provenance and annotation management tasks through query instrumentation [19]. Although the above works suggest application-independent modules for supporting provenance formalisms to annotated data, they can only apply in particular fields and domains, such as programs with PostgreSQL databases, knowledge-centric applications for knowledge graphs, and specific types of databases.

Therefore, we design Annalog, a back-end open-source framework based on the Java platform, which can be utilized by any kind of java database management system with the interface. Due to the cross-platform and easily portable characters of Java [20], Annalog is available on multiple system platforms once compiled. The developed framework does not have a limitation on the choice of the database type and is applicable to various databases and database management system applications which can be operated by standard SQL statements. The above-mentioned features ensure that Annalog can expand the data processing functions of database management applications without affecting their original operations. The platform is not restricted to specific databases and platforms, which realize lightweight, cross-platform and adaptability identities.

3 Methodology

In this section, we discuss the semantics of annotated data. We introduce query processes in the relational databases over annotated data and semantics on annotations represented by the algebraic framework of a semiring.

3.1 Semantics on Annotated Data

Annotated data are relational data and present various meanings in different applications [21]. Diverse applications require individual semantics of annotations that ought to be modified and processed [22]. For instance, we need to comprehend when the domain of the annotation column is a natural number and what is the multiplicity of each tuple (Bag Semantic). We also need to understand what the probability has the result given a feasibility distribution on the input data (Probability Semantic), why is the specific tuple obtained (Why-Provenance), and how is the specific tuple obtained? (How-Provenance). As displayed in Fig. 1(a), the table represents annotations about the relational data. In the example table, *r*, *s*, and *t* are the tuples IDs. A condition table is a primitive kind of table where the tuples are annotated with the formulas named conditions [23-24]. A bag table and a probability table are the elementary types of the condition tables where the annotations are the distinct multiplicity variables and the probability distributions respectively, as shown in Fig. 1(b) and Fig. 1(c).

A	B	Ann
2	5	r
1	2	s
6	3	t

(a)

A	B	Bag
2	5	2
1	2	6
6	3	5

(b)

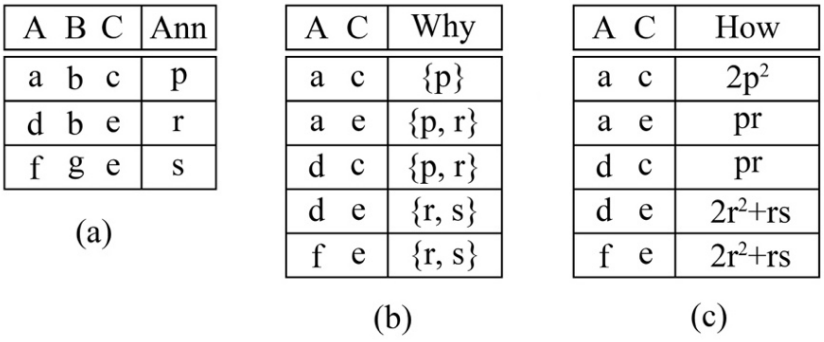
A	B	Prob
2	5	0.2
1	2	0.6
6	3	0.5

(c)

Fig. 1. The examples of an annotation table, a bag table, and a probability table respectively

It is possible that inside each database table, there are several copies of the same tuple. The extra bag annotation column aims to keep the number of copies in apiece tuple from the table in order to obtain the multiplicity information [25]. In the bag semantics, the annotations are natural numbers representing the multiplicity of the tuple in the multiset. A tuple not listed in the table has a multiplicity number of 0 [26]. The advantage of the above-discussed strategy is the reduction of the database complexity. Instead of storing the total number of tuples in the database, the table manages tuples by keeping their multiplicity information inside an extra bag annotation. Probability annotation is utilized in tables to enrich the probabilistic databases. In the table, tuples are associated with probabilistic values from 0 to 1 [27-28]. In the probability semantics, annotations are decimals or natural numbers. The probabilistic values represent the probability of the tuples presented in the database.

The example annotation table is illustrated in Fig. 2(a), where *p*, *r*, and *s* are the tuple IDs. For the source database *R* and the query *Q* of the table, there is a relation $Q(R) \underline{\underline{\text{def}}} \prod_{AC}(\prod_{AB} R \bowtie \prod_{BC} R \cup \prod_{AC} \prod_{BC} R)$. Except for the two semantics discussed above, where the numerical values are kept as annotations, there are other types of semantics in which annotations are represented by symbols. These annotations mostly establish a connection between the source tuples and the output tuples. Due to the nature of these annotations, we call them provenance [29]. Why-provenance explains which sets of the source tuples are involved in producing the output tuple, as the example displayed in Fig. 2(b). The domain relation of the annotation column is elucidated by *P(P(X))*, where *X* is the set of the tuple IDs and *P(X)* is the power set of *X*. How-provenance, also called the provenance polynomials, describes how the source tuples are combined based on the query to produce an output tuple [30]. The sample table of the How-provenance is illustrated in Fig. 2(c). The domain formula representing the How Provenances is *N[X]*, which indicates the set of polynomials with variables from *X* and coefficients from *N*.



A	B	C	Ann
a	b	c	p
d	b	e	r
f	g	e	s

(a)

A	C	Why
a	c	{p}
a	e	{p, r}
d	c	{p, r}
d	e	{r, s}
f	e	{r, s}

(b)

A	C	How
a	c	$2p^2$
a	e	pr
d	c	pr
d	e	$2r^2+rs$
f	e	$2r^2+rs$

(c)

Fig. 2. The examples of an annotation table, a why-provenance table, and a how-provenance table

3.2 Query Processing over Annotated Data

In the relation database, when evaluating queries over the annotated data, the primary question is how the annotation data should be managed when processing each query operation [31]. Specifically, for the Select, Project, Join, and Union (SPJU) queries, we need to acquire how to combine the annotations of the joined tuples and how to combine the annotations when there are different copies of the same tuple. A query in the Datalog is an aggregation of one or more rules [32]. If there is only one relationship appears in the rule header, then the value of that relationship is the answer to the query. If there are multiple relationships in the rule header, one of these relationships is the answer to the query, the other relationships play an auxiliary role in the definition of the answer [33].

The Union of the two relations is constructed utilizing two rules. Each rule has an atom corresponding to one of the relationships as its only sub-target. Both rules have the same IDB predicate at the head [34]. The parameters of a rule's header are the same as those of its sub-targets. The rule of $q = r \cup s$ is represented as follows:

$$Q(x_1. x_2, ..., x_n) \leftarrow R(x_1, x_2, ..., x_n), Q(x_1, x_2, ..., x_n) \leftarrow S(x_1, x_2, ..., x_n) . \quad (1)$$

The rule of Join $q = r \cap s$ is represented in the relation as follows:

$$Q(x_1, x_2, ..., x_n) \leftarrow R(x_1, x_2, ..., x_n) \; AND \; S(x_1, x_2, ..., x_n) . \quad (2)$$

In the rule of Select, if the selected conditions are all AND operations, they are represented by a single rule. Each condition is considered as an arithmetic sub-target concatenated with AND operations. For example, the rule $q = \sigma_{x > 10 \text{ and } y < 100}(r)$ is represented in the relation as follows:

$$Q(x_1, x_2, ..., x_n) \leftarrow R(x_1, x_2, ..., x_n) \; AND \; x > 10 \; AND \; y < 100 . \quad (3)$$

In the rule of Select, if the selection contains an OR operation involving p conditions, then it is required to represent the selection with p rules, where each of the rules defines the same header predicate [35]. The rule i makes a choice based on the ith of the p conditions and the rule $q = \sigma_{x > 10 \text{ or } y < 100}(r)$ is represented as follows:

$$Q(x_1, x_2, ..., x_n) \leftarrow R(x_1, x_2, ..., x_n) \; AND \; x > 10, Q(x_1, x_2, ..., x_n) \leftarrow R(x_1, x_2, ..., x_n) \; AND \; y < 0 . \quad (4)$$

A Projection of a relationship can be implemented utilizing a single rule with a single-sub object. The header parameters are variables that correspond to the projected property sheet in the desired order. For example, the rule of $q = \pi_{x1,x2,x3}(r)$ is represented in the relation as follows:

$$Q\,(x_1, x_2, x_3) \leftarrow R\,(x_1, x_2, ..., x_n)\,. \tag{5}$$

Datalog rules can simulate not only a single operation of relational algebra, but any algebraic expression [36]. The method is to examine the expression tree corresponding to the relational algebraic expression and establish an IDB predicate for each internal node of the tree. Each IDB predicate corresponds to one or more rules that are required to apply the operator to the corresponding node of the tree. Leaf nodes in the tree stand for the relationships in the database that are represented by the corresponding EDB predicate. Internal nodes in the tree stand for the relationships in the database that are represented by the corresponding IDB term [37]. For example, for the two relations: *Student (no, name, age, sex, dept)* and *SC (no, cno, grame)*, select name from the student who has taken course 2. One possible form of the relational algebraic expression completes this query is as follows:

$$\pi_{name}\,(\sigma_{student.sno=sc.sno}(student \times \sigma_{sc.cno=2}(SC)))\,. \tag{6}$$

In the bag semantics, the annotations of the joined tuples will be multiplied ($\times$), as displayed in Q_1 from Fig. 3(a). The annotations of different copies from the same tuple will be added ($+$), as shown in Q_2 from Fig. 3(b). Query answering on the database tables involves calculating not only the tuples in the output, but also the multiplicities. If there are two tables with the same content, when the query processor is asked to combine the annotation of the joined tuples, the annotation of the joined tuples will be multiplied [38]. On the other hand, the annotation of different copies from the same tuples will be added, which means that the engine will result with the addition of two annotations if process union operation and project operation [39].

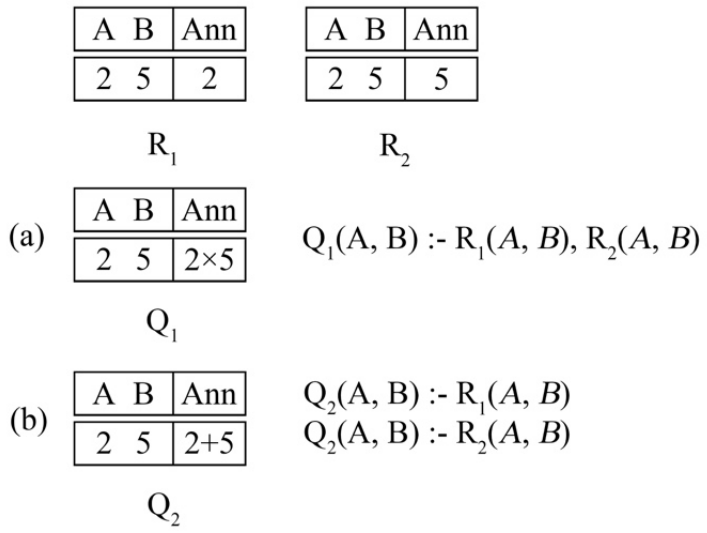


Fig. 3. The example of the bag semantics tables

Probability evaluation is an independence function $f(\alpha, \beta) = \alpha + \beta - \alpha\beta$ to combine multiple derivations of the same atom collected as a bag. For calculating the probability of annotations of the joined tuples, if the probability is independent, we use the multiplication of two annotations, for example, $p(r) \times p(s) = p(r \wedge s)$. For calculating the probability of combining the annotations in different copies of the same tuples, we use the independent formulas instead, for example, $p(r \vee s) = p(r) + p(s) - p(r) \times p(s)$. Our design aims at calculating the probability under the annotation column without creating a new column to store annotation and probability separately.

In the why-provenance semantics, for combining the annotations of the joined tuples, we use the pairwise union $\Cup$, where $A \Cup B = \{a \cup b\colon a \in A, b \in B\}$. For combining the annotations in different copies of the same tuple, we utilize the union $\cup$, as shown in Q_1 from Fig. 4(a) and Q_2 from Fig. 4(b) respectively.

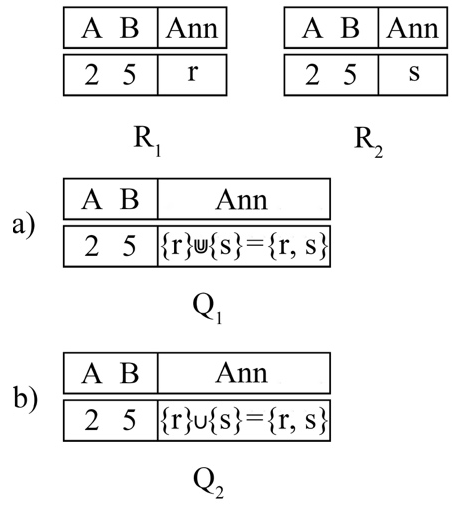


Fig. 4. The example of the why-provenance semantics tables

In the how-provenance semantics, for combining the annotations of the joined tuples, we use the multiplication as displayed in Q_1 from Fig. 5(a). For associating the annotations in different copies of the same tuple, we utilize the addition as shown in Q_2 from Fig. 5(b).

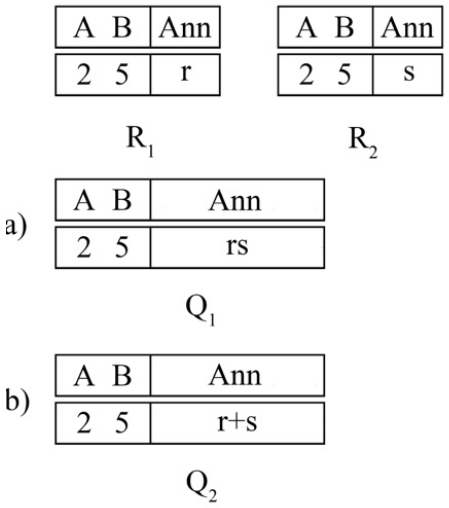


Fig. 5. The example of the how-provenance semantics tables

In the Boolean expression semantics, the domain of the annotation column is interpreted by *PosBool(X)* which is the set of all Boolean expressions over variables *X* which are positive [40]. The variables in Boolean expressions involve only disjunction, conjunction, and constants for true and false. For combining the annotations of the joined tuples, we utilize AND ($\wedge$) operation, as shown in Q_1 from Fig. 6(a). For combining the annotations in different copies of the same tuple, we use OR ($\vee$) operation, as shown in Q_2 from Fig. 6(b). One of the major applications of the Boolean provenance is the query evaluation in probabilistic databases [41].

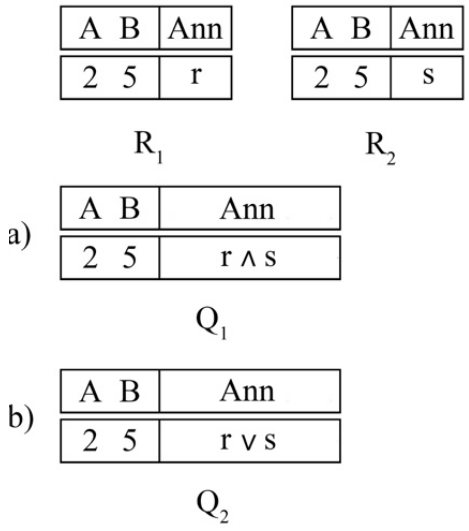


Fig. 6. The example of the Boolean expression semantics tables

3.3 Provenance Semiring and Semiring Homomorphism

The above-discussed semantics can be represented by the algebraic framework of provenance semiring. The relation of the framework is (K, $\oplus$, $\otimes$, 0, 1). In the relation, K illustrates the relations have an annotation column of the domain K. $\oplus$ and $\otimes$ indicates simulating two types of operations on the values of K. 0 and 1 are the identity elements of $\oplus$ and $\otimes$ respectively [42-43]. The natural number semiring is defined as (N, +, ∗, 0, 1). The polynomial semiring is ruled as (N, +, ∗, 0, 1). The why-provenance semiring is stated as ($P(P(X)), \cup, \Cup, \varnothing, \{\varnothing\}$). The positive Boolean expression semiring is regulated as: ($PosBool(X)$, $\wedge$, $\vee$, *false*, *true*), as shown in Fig. 7.

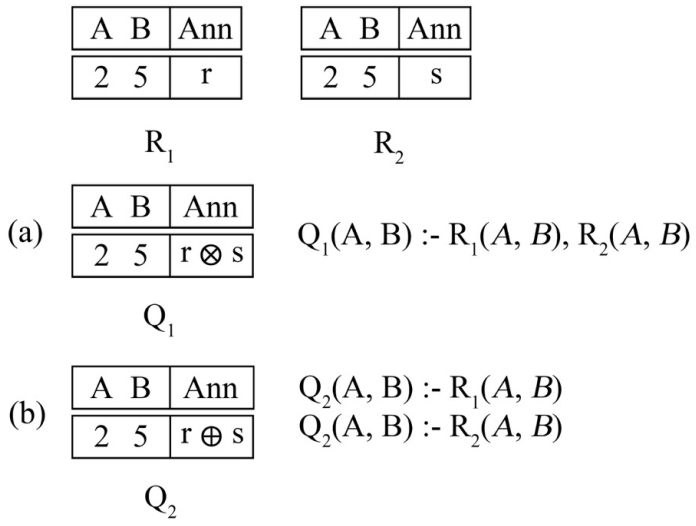


Fig. 7. The example of the semiring tables

In the provenance hierarchy, the relation $N[X]$ is the most general semiring and the relation B (the semiring of the standard algebra) is the least informative semiring [40]. Each path from a semiring K_1 downward to another

semiring K_2 indicates a semiring homomorphism h: $K_1 \rightarrow K_2$. For any semiring K, to evaluate RA^+ queries on K-relations, it is sufficient to know how to evaluate these queries over $N[X]$-relations [44]. Polynomial is a semiring of the symbolic expressions for recording the documentation as the tuple tags [45]. From the polynomials, it is clear to discover not only which the input tuples contribute, but also how an output tuple is produced. Let S be the set of the tuple IDs of a database instance I. The positive algebra provenance semiring for I is the semiring of polynomials with variables from S and coefficients from N. The operation of the relation is defined as: (N, +, ∗, 0, 1), where "+" stands for the union of two tuples and "*" stands for the join [46]. The provenance polynomials are the most informative semirings among semiring annotations by dint of their universality. In the relation, any function $v : X \rightarrow K$ can be extended uniquely to a semiring homomorphism $Eval_v : N[X] \rightarrow K$. Intuitively, $Eval_v$ operates by assigning the value $v(x)$ to each variable x in a polynomial expression, then evaluating the resulting expression in K [47]. By Combining the commutation with homomorphisms property [48], it allows the computations for any commutative semiring K to *factor* through the computations for the provenance polynomials. The polynomial semiring presents a universal semiring [49]. For instance, to detect the results of a query in why-provenance, people can find the results in polynomial provenance and then map the provenance in polynomials semantic. The source query is transferred at first, and then the query annotated in the why-provenance is executed. Since the why-provenance and executing queries over polynomials contain all other semirings, which have homomorphisms with polynomial semantics, the processes aim at utilizing the polynomials semiring as a universal memory [50]. The algebraic framework of provenance semiring tables is displayed in Fig. 8 as follows.

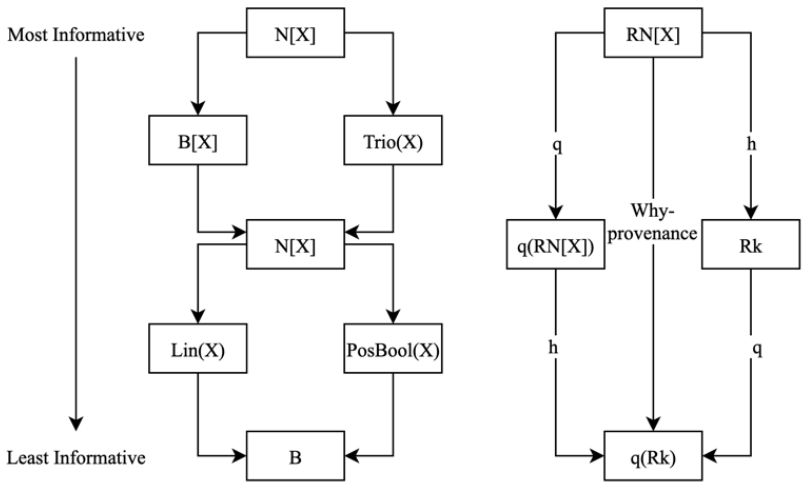


Fig. 8. The algebraic framework of the provenance semiring tables

4 Results and Implementation

We implement the top-down paradigm as the design solution. The goal of the proposed framework is specified together with Datalog. In the top-down method, proof trees are constructed from the top to the bottom [51]. Rules in top-down evaluation are considered the problem generators. Each goal is expected to be one problem that must be solved [52]. The initial goal is matched with the rules from the left-hand side and continuously generates problems corresponding to the right-hand side predicates of that rule. This process continues until no more new problems are generated [53].

4.1 Implementation of the Framework

The Annolog back-end annotation query processing framework is designed and developed as an additional tool

to the existing database systems. The framework is capable to realize the Union, Join, Project, and Select operations with bag, probability, why-provenance, and how-provenance features to data processing operations. The SQL statements information operated by users in the database management system will be transformed into the framework at first. Then the framework will analyze the SQL statements and turn the instructions into a relational calculus way as the annotated data is generally stored in relational databases. Afterwards, the framework will request the front-end to operate the datasets in the bag, probability, why-provenance, and how-provenance. Finally, after the tables are selected and the operations are guaranteed, the framework will provide results tables to the front-end systems as extra information. The query processing and modelling processes on the annotated data by the designed framework are not influencing the original operation results in database management systems, as shown in Fig. 9.

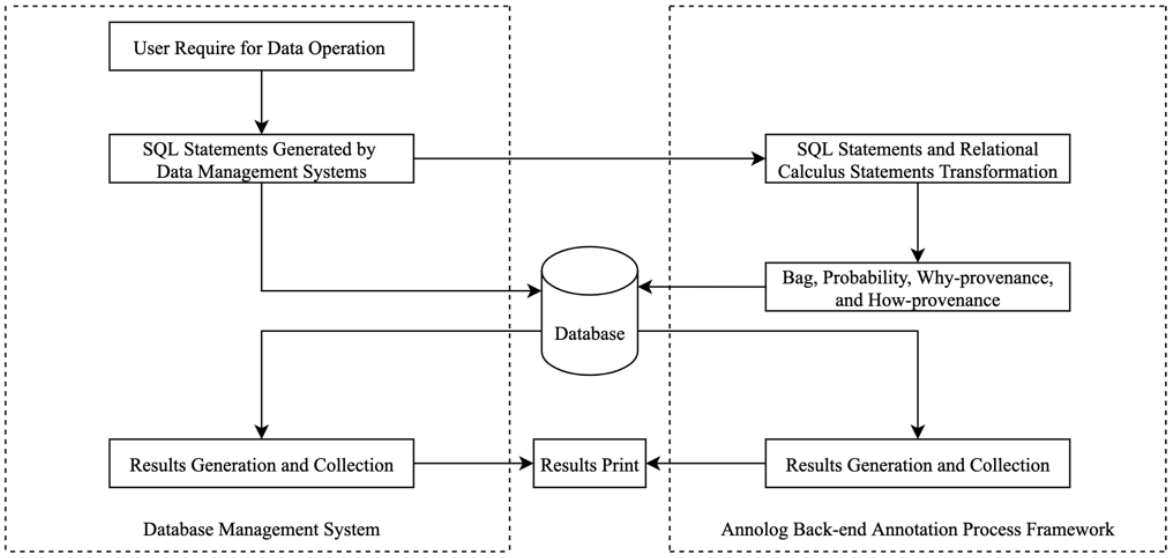


Fig. 9. The architecture of the proposed framework

4.2 Results of the Framework

The designed database for the system is displayed in Table 1, Table 2, and Table 3. Consider relational instances consisting of three relations Table 1, Table 2, and Table 3 with a corresponding number of denial constraints **IC** and a query **Qex**. In the relation to instances from Table 1, the first column is the patient identifier **PID**, the second column is the disease reference **RefID**, the third column is the date of a given event **Date**, and the last column is the annotation information of the record **Annotation**. The schema of the three tables is the same solely for illustration purposes and to maximize the number of joins across the tables.

Table 1. A hospital database with a set of denial constraints and a query Qex

PID	RefID	Date	Annotation
P01	R1	D1	2
P02	R5	D2	2
P03	R5	D2	3
P04	R7	D3	3
P05	R7	D3	5
P02	R5	D2	5
P05	R7	D3	3
P08	R9	D5	3
P10	R9	D9	7
P10	R9	D9	7

In the relation to instances from Table 2, the first column is the first course of treatment **CT1**, the second column is the second course of treatment **CT2**, the third column is the third course of treatment **CT3**, and the last column is the effective rate **Annotation**. Contents in the first three columns are the medicine names, for example, **MDa**. Contents in the last column of the table are the rate numbers, for example, **0.5**.

Table 2. A hospital database with a set of denial constraints and a query Qex

CT1	CT2	CT3	Annotation
MDa	MDb	MDc	0.5
MDd	MDb	MDe	0.5
MDf	MDg	MDe	0.3
MDa	MDf	MDb	0.4
MDb	MDc	MDd	0.2
MDc	MDd	MDe	0.9

In the relation to instances from Table 3, the first column is the first course of treatment **CT1**, the second column is the second course of treatment **CT2**, the third column is the third course of treatment **CT3**, and the last column is the effective rate **Annotation**. Contents in the first three columns are the medicine names, for example, **MDa**. Contents in the last column of the table are the rate variables, for example, **r**.

Table 3. A hospital database with a set of denial constraints and a query Qex

CT1	CT2	CT3	Annotation
MDa	MDb	MDc	o
MDd	MDb	MDe	p
MDf	MDg	MDe	q
MDa	MDf	MDb	r
MDb	MDc	MDd	s
MDc	MDd	MDe	t

When the user operates the Table 1 database in the database management system with SQL statements, for example, *SELECT PID, RefID, Date FROM TABLE 1* and chooses the bag annotated data operation, the Annolog back-end framework will first transform the SQL statements into the relational circulus expressions *PROJECT <PID, RefID, Date> (Table 1)*. Then the table after the bag operation is displayed or sent back directly to the front-end system for reuse. The detailed operating procedure and result are shown in Fig. 10.

```
SELECT PID, RefID, Date FROM TABLE 1
PROJECT <PID, RefID, Date> (Table 1)

PID    RefID      Date      Annotation
P01    R1         D1      2
P02    R5         D2      7
P03    R5         D2      3
P04    R7         D3      3
P05    R7         D3      8
P08    R9         D5      3
P10    R9         D9      14

Process finished with exit code 0
```

Fig. 10. The detailed bag operating procedure and result

When the user operates the Table 2 database in the database management system with SQL statements, for example, *SELECT CT1, CT2 FROM TABLE 2 JOIN SELECT CT2, CT3 FROM TABLE 2* and choose the probability annotated data operation, the Annolog back-end framework will first transform the SQL statements into relational circulus expressions *PROJECT <CT1, CT3> ((PROJECT <CT1, CT2>) (Table 2) JOIN (PROJECT <CT2, CT3>) (Table 2))*. Then the result of the table after the probability modification is displayed or sent back directly to the front-end system for reuse. The detailed operating procedure and result are shown in Fig. 11.

```
SELECT CT1, CT2 FROM TABLE 2 JOIN SELECT CT2, CT3 FROM TABLE 2
PROJECT <CT1, CT3> ((PROJECT <CT1, CT2>) (Table 2) JOIN (PROJECT <CT2, CT3>) (Table 2))

CT1    CT3    Annotation
MDa    MDc    0.4375
MDd    MDe    0.7370312
MDf    MDe    0.48616397
MDc    MDe    0.9855058
MDa    MDb    0.2944
MDb    MDd    0.07840001
MDd    MDc    0.25
MDa    MDe    0.25
```

Fig. 11. The detailed probability operating procedure and result

When the user operates the Table 3 database in the database management system with SQL statements, for example, *SELECT CT1, CT2 FROM TABLE 3 JOIN SELECT CT2, CT3 FROM TABLE 3* and choose the polynomial annotated data operation, the Annolog back-end framework will first transform the SQL statements into relational circulus expressions *PROJECT <CT1, CT3> ((PROJECT <CT1, CT2>) (Table 3) JOIN (PROJECT <CT2, CT3>) (Table 3))*. Then the result of the table after the probability operation is displayed or sent back directly to the front-end system for reuse. The detailed operating procedure and result are shown in Fig. 12.

```
SELECT CT1, CT2 FROM TABLE 3 JOIN SELECT CT2, CT3 FROM TABLE 3
PROJECT <CT1, CT3> ((PROJECT <CT1, CT2>) (Table 3) JOIN (PROJECT <CT2, CT3>) (Table 3))

CT1    CT3    Annotation
MDa    MDc    (o*o+o*o)
MDd    MDe    (((p*p+p*p)+q*p)+t*p)
MDf    MDe    ((p*q+(q*q+q*q))+t*q)
MDc    MDe    ((p*t+q*t)+(t*t+t*t))
MDa    MDb    (r*r+r*r)
MDb    MDd    (s*s+s*s)
MDd    MDc    o*p
MDa    MDe    p*o
```

Fig. 12. The detailed probability operating procedure and result

4.3 Accuracy of the Framework

To verify the accuracy of the results generated by the methodology from the query processing framework, we develop a dataset from the public health data source with the number of new confirmed COVID-19 cases per hundred thousand citizens among different administrative areas of Montreal from April 2020 to June 2020, as one-day data example illustrated in Fig. 13. The annotation information in this dataset is the confirmed COVID-19 cases per hundred thousand people. In the dataset, the number of new confirmed COVID-19 cases per hundred thousand citizens in Montreal is also recorded. The accuracy examination aims to implement the query processing framework to the area's data, generate the data of the city, and compare the generated results with the data from the dataset by checking the correlation coefficient of the data.

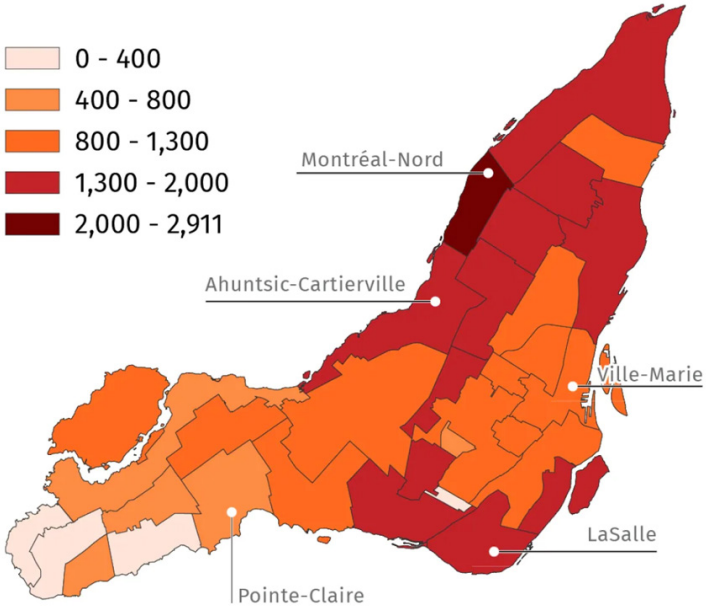


Fig. 13. The dataset with the number of new confirmed COVID-19 cases per hundred thousand citizens

The correlation coefficient is a statistical index that reflects the degree of linear correlation between variables. Due to the different research objects, there are many ways to define the correlation coefficient. The Pearson correlation coefficient is selected in this paper. The correlation coefficient is calculated according to the product difference method, which is also based on the deviation between two variables and their average value. The correlation degree between the two variables is reflected by multiplying the two deviations. The number of linear single-phase relations is emphasized. The definition of the Pearson correlation coefficient is as follows.

$$r = \frac{\sum (X - \bar{X})(Y - \bar{\bar{Y}})}{\sqrt{\sum (X - \bar{X})^2 \sum (Y - \bar{Y})^2}} \,. \tag{7}$$

The absolute value of r is between 0 and 1. Generally speaking, the closer r is to 1, the stronger the correlation between x and y. Conversely, the closer r is to 0, the weaker the correlation between x and y. The relation between the absolute value of r value and correlation degree is shown in Table 4 below.

Table 4. The relation between the absolute value of r value and correlation degree

Value range of $\lvert r \rvert$	Meaning of $\lvert r \rvert$
0.00 – 0.19	Very low correlation
0.20 – 0.39	Low correlation
0.40 – 0.69	Medium correlation
0.70 – 0.89	High correlation
0.90 – 1.00	Very high correlation

We define statements and relational calculus to the dataset of regional data, implement bag as well as probability operations, and generate the dataset of city data. Then we calculate the Pearson correlation coefficient value of the generated data and the data stored in the dataset. There are 91 comparison value results of the Pearson correlation coefficients, as displayed in Fig. 14 as follows.

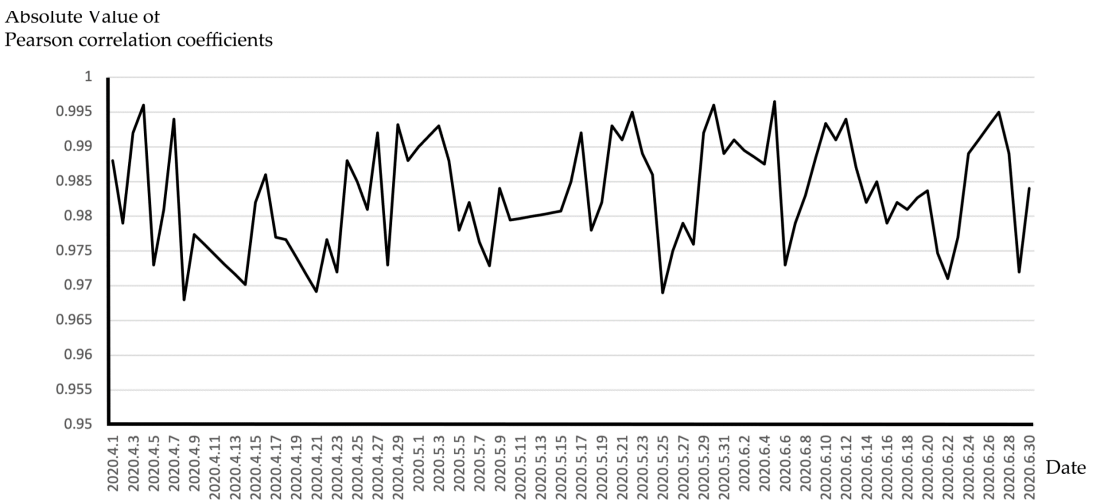


Fig. 14. Comparison results of the Pearson correlation coefficients

It can be illustrated from the result that all the Pearson correlation coefficients are in the range of 0.96 to 1.00, which concludes a significantly high correlation result between the value computed by the framework and the actual value. Therefore, the query processing framework developed in this paper is proved to be accurate. The Box and Whisker data analysis methodology is implemented to screen for the outliers. The outliers' data elements will be further examined for the detection of the causes. After analysis of the outlier data elements, it was found that if the number of the new confirmed COVID-19 cases among several administrative areas in the data element are significantly different, then the Pearson correlation coefficient value of the final urban data results generated by the query processing framework and the actual overall urban data results will be relatively low. Since the general rational query statement may not be able to handle the large differences in attributes within a data element. Therefore, subtle manual adjustments should be appended to the query, such as the addition of weights.

5 Discussion

The topic of this paper is to attach the importance to the role of the annotated data and improve the analytical methodology of processing the annotation data in database systems. Based on this research goal, we first discuss the semantics of the annotated data, introduce the query processes in the relational databases over annotated data, and interpret the semantics of annotations represented by the algebraic framework of semirings. Then we implement the top-down approach as the design solution to develop a query processing framework for modelling the annotations in database systems. The framework is a powerful expansion of the functionality of the existing database systems. In the framework, the SQL statements generated from the operation of users by the database management system will be transmitted into the platform at first. Then the statements will be transformed into a relational calculus way for further analyzing the annotated data in the relational databases. The framework will acquire the semantics operation to the annotated data and finally send back the results to the front-end database systems.

Existing studies focused on the various operations of the annotated data. Although these works processed and analyzed the annotated data, suggested comprehensive provenance representations, and enriched the operations and considerations over the provenance of combined data, they mainly developed and compiled on fixed and well-defined formulas or regulations. Therefore, to improve the limitation, we allow our platform to analyze diverse formulas by adding a port to receive the user's operations to the database in SQL statements and convert the statement to relational calculus parallelly and continuously. This process aims to support the framework deal with the annotated data flexibly without setting restrictions and limitations on the formulas. Several application-independent modules and platforms have been proposed as supplementary operations to annotate data for existing database management systems such as ProvSQL and HUKA etc. However, these frameworks have regulations on the formats of the database in database systems. For example, the ProvSQL framework compiles

on the PostgreSQL databases and the HUKA module is developed based on knowledge-centric applications for knowledge graphs. Therefore, to eliminate limitations to databases of the database systems that the framework is running on, instead of implementing operations to databases, we focus on the manipulations of the database management systems and design a supplementary interface for processing the data transforms inside the database management system. Hence, the framework can operate various kinds of annotated data in multiple types of databases.

Analysis of the annotated data can help stakeholders understand the information behind data in the database. A majority of the database management systems are developed based on the Java programming language. Hence, the framework we proposed in this paper is implemented in the Java programming language as well. In the future, we will focus on improving our framework for the cross-language platform design, so that it can be applied to more database management systems based on other programming languages such as Python and R programming language. We will continue focusing on expanding the types of statements that the framework can recognize and transform into the relational calculus. The extension will further improve the framework from processing the single SQL statement databases to multiple databases such as graph databases.

6 Conclusions

Annotated data play a significant role not only in the field of medical and education domains but also in the academic sectors such as artificial intelligence and machine learning. This paper connects annotation with knowledge and introduces a back-end query processing framework for modelling and reasoning with the annotated data in the database. Practice indicates that it can meet the requirements of datasets processing for annotated data management. As an external supplement module to the existing database management systems, the designed framework can realize the Select, Project, Join, and Union (SPJU) functions with bag, probability, why-provenance, and how-provenance features to the data processing operations parallelly and continuously without affecting the original functionality. This feature proves the characteristics of lightweight and expansibility of the developed model. In general, our study provides a data provenance technique and a portable solution for stakeholders to analyze and process the annotated data in the databases from the existing database management systems.

References

[1] M. Nagarajan, Semantic annotations in web services, in: J. Cardoso, A.P. Sheth (Eds.) Semantic Web services, processes and applications, Springer, Boston, 2006 (pp. 35-61).
[2] D. Haynes, Metadata for Information Management and Retrieval: Understanding metadata and its use, Facet Publishing, London, 2018.
[3] K. Blask, A. Förster, Designing an information architecture for data management technologies: Introducing the DIAMANT model, Journal of Librarianship and Information Science 52(2)(2020) 592-600.
[4] F.Z. Smaili, X. Gao, R. Hoehndorf, Opa2vec: combining formal and informal content of biomedical ontologies to improve similarity-based prediction, Bioinformatics 35(12)(2019) 2133-2140.
[5] H. Kilicoglu, A.B. Abacha, Y. Mrabet, S.E. Shooshan, L. Rodriguez, K. Masterton, D. Demner-Fushman, Semantic annotation of consumer health questions, BMC bioinformatics 19(1)(2018) 1-28.
[6] G. Abrami, A. Mehler, A. Lücking, E. Rieb, P. Helfrich, TextAnnotator: A flexible framework for semantic annotations, in: Proc. 15th joint ACL-ISO workshop on interoperable semantic annotation (ISA-15), 2019.
[7] M.N. Mansur, M. Christakis, V. Wüstholz, Metamorphic testing of Datalog engines, in: Proc. 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2021.
[8] C. Borraz-Sánchez, D. Klabjan, E. Pasalic, M. Aref, SolverBlox: Algebraic modeling in datalog, Declarative Logic Programming: Theory, Systems, and Applications (2018) 331-354.
[9] J. Wang, J. Wu, M. Li, J. Gu, A. Das, C. Zaniolo, Formal semantics and high performance in declarative machine learning using Datalog, The VLDB Journal 30(5)(2021) 859-881.
[10] O. Corcho, F. Priyatna, D. Chaves-Fraga, Towards a new generation of ontology based data access, Semantic Web 11(1) (2020) 153-160.
[11] P. Senellart, L. Jachiet, S. Maniu, Y. Ramusat, ProvSQL: Provenance and probability management in PostgreSQL, in: Proc. VLDB Endowment (PVLDB), 2018.
[12] J. Wang, D. Crawl, S. Purawat, M. Nguyen, I. Altintas, Big data provenance: Challenges, state of the art and opportunities, in: Proc. 2015 IEEE international conference on big data (Big Data), 2015.
[13] T. Müller, P. Engel, How, Where, and Why Data Provenance Improves Query Debugging: A Visual Demonstration of

Fine–Grained Provenance Analysis for SQL, in: Proc. 2022 IEEE 38th International Conference on Data Engineering (ICDE), 2022.
[14] P. Senellart, Provenance in databases: principles and applications, in: M. Krötzsch, D. Stepanova (Eds.) Reasoning Web. Explainable Artificial Intelligence, Springer, Berlin, 2019 (pp. 104-109).
[15] V. Benzaken, S. Cohen-Boulakia, É. Contejean, C. Keller, R. Zucchini, A Coq formalization of data provenance, in: Proc. 10th ACM SIGPLAN International Conference on Certified Programs and Proofs, 2021.
[16] O. Issa, A. Bonifati, F. Toumani, A relational framework for inconsistency-aware query answering, in: Proc. BDA'19, 2019.
[17] P. Li, O. Niggemann, A data provenance based architecture to enhance the reliability of data analysis for Industry 4.0, in: Proc. 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), 2018.
[18] G. Gaur, A. Bhattacharya, S. Bedathur, How and Why is An Answer (Still) Correct? Maintaining Provenance in Dynamic Knowledge Graphs, in: Proc. 29th ACM International Conference on Information & Knowledge Management, 2020.
[19] B.S. Arab, S. Feng, B. Glavic, S. Lee, X. Niu, Q. Zeng, GProM-a swiss army knife for your provenance needs, A Quarterly bulletin of the Computer Society of the IEEE Technical Committee on Data Engineering 41(1)(2018) 51-62.
[20] C. Wang, R. Li, W. Li, C. Qiu, X. Wang, SimEdgeIntel: A open-source simulation platform for resource management in edge intelligence, Journal of Systems Architecture 115(2021) 102016.
[21] J.R. López-Robles, J.R. Otegi-Olaso, I.P. Gómez, M.J. Cobo, 30 years of intelligence models in management and business: A bibliometric review, International journal of information management 48(2019) 22-38.
[22] M. Hassaballah, A.I. Awad, Deep learning in computer vision: principles and applications, CRC Press, Boca Raton, 2020.
[23] I.E. Oana, C.Q. Schneider, E. Thomann, Qualitative Comparative Analysis Using R: A Beginner's Guide, Cambridge University Press, Cambridge, 2021.
[24] A.K. Montoya, A.F. Hayes, Two-condition within-participant statistical mediation analysis: A path-analytic framework, Psychological Methods 22(1)(2020) 6-27.
[25] B. Sundarmurthy, P. Koutris, W. Lang, J. Naughton, V. Tannen, m-tables: Representing missing data, in: Proc. 20th International Conference on Database Theory (ICDT 2017), 2017.
[26] G. Cima, C. Nikolaou, E.V. Kostylev, M. Kaminski, B.C. Grau, I. Horrocks, Bag semantics of dl-lite with functionality axioms, in: Proc. International Semantic Web Conference, 2019.
[27] S. Feng, A. Huber, B. Glavic, O. Kennedy, Uncertainty annotated databases: A lightweight approach for approximating certain answers, in: Proc. 2019 International Conference on Management of Data, 2019.
[28] G. Gaur, A. Dang, A. Bhattacharya, S. Bedathur, Computing and Maintaining Provenance of Query Result Probabilities in Uncertain Knowledge Graphs, in: Proc. 30th ACM International Conference on Information & Knowledge Management, 2021.
[29] P. Buneman, W.C. Tan, Data provenance: What next? ACM SIGMOD Record 47(3)(2018) 5-16.
[30] D. Dosso, S.B. Davidson, G. Silvello, Data Provenance for Attributes: Attribute Lineage, in: Proc. 12th International Workshop on Theory and Practice of Provenance (TaPP 2020), 2020.
[31] T. Auge, A. Heuer, ProSA: Using the CHASE for Provenance Management, in: Proc. European Conference on Advances in Databases and Information Systems, 2019.
[32] C. Zaniolo, M. Yang, A. Das, A. Shkapsky, T. Condie, M. Interlandi, Fixpoint semantics and optimization of recursive datalog programs with aggregates, Theory and Practice of Logic Programming 17(5-6)(2017) 1048-1065.
[33] S. Zhang, K. Balog, Web table extraction, retrieval, and augmentation: A survey, ACM Transactions on Intelligent Systems and Technology (TIST) 11(2)(2020) 1-35.
[34] T.L. Pham, M.I. Ali, A. Mileo, Enhancing the scalability of expressive stream reasoning via input-driven parallelization, Semantic Web 10(3)(2019) 457-474.
[35] T. Deng, Y. Li, X. Liu, P. Wang, K. Lu, A Data-driven Parameter Planning Method for Structural Parts NC Machining, Robotics and Computer-Integrated Manufacturing 68(2021) 102080.
[36] L. Peterfreund, D.D. Freydenberger, B. Kimelfeld, M. Kröll, Complexity bounds for relational algebra over document spanners, in: Proc. 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2019.
[37] S. Lee, B. Ludäscher, B. Glavic, PUG: a framework and practical implementation for why and why-not provenance, The VLDB Journal 28(1)(2019) 47-71.
[38] I. Trummer, J. Wang, D. Maram, S. Moseley, S. Jo, J. Antonakakis, SkinnerDB: Regret-bounded query evaluation via reinforcement learning, in: Proc. 2019 International Conference on Management of Data, 2019.
[39] P. Pinoli, S. Ceri, D. Martinenghi, L. Nanni, Metadata management for scientific databases, Information Systems 81(2019) 1-20.
[40] K.M. Dannert, E. Grädel, Provenance Analysis: A Perspective for Description Logics? in: C. Lutz, U. Sattler, C. Tinelli, A.Y. Turhan, F. Wolter (Eds.) Description Logic, Theory Combination, and All That, Springer, Berlin, 2019 (pp. 266-285).
[41] P. Senellart, Provenance and probabilities in relational databases, ACM SIGMOD Record 46(4)(2017) 5-15.
[42] T.J. Green, V. Tannen, The semiring framework for database provenance, in: Proc. 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2017.
[43] P. Bourhis, D. Deutch, Y. Moskovitch, Equivalence-Invariant Algebraic Provenance for Hyperplane Update Queries, in:

Proc. 2020 ACM SIGMOD International Conference on Management of Data, 2020.
[44] Y. Zhang, V. Zhong, D. Chen, G. Angeli, C.D. Manning, Position-aware attention and supervised data improve slot filling, in: Proc. 2017 Conference on Empirical Methods in Natural Language Processing, 2017.
[45] W. Wu, L. Flokas, E. Wu, J. Wang, Complaint-driven training data debugging for query 2.0, in: Proc. 2020 ACM SIGMOD International Conference on Management of Data, 2020.
[46] E. Grädel, V. Tannen, Provenance analysis for logic and games, Moscow Journal of Combinatorics and Number Theory, 9(3)(2020) 203-228.
[47] S.E. Amiri, L. Chen, B.A. Prakash, Efficiently summarizing attributed diffusion networks, Data Mining and Knowledge Discovery 32(5)(2018) 1251-1274.
[48] J. Shimada, BE Is Not the Unique Homomorphism That Makes the Partee Triangle Commute, in Proc. 15th Meeting on the Mathematics of Language, 2017.
[49] K.M. Dannert, E. Grädel, Semiring provenance for guarded logics, in: J. Madarász, G. Székely (Eds.) Hajnal Andréka and István Németi on Unity of Science, Springer, Berlin, 2021 (pp. 53-79).
[50] G. Smyrnis, P. Maragos, Multiclass neural network minimization via tropical newton polytope approximation, in: Proc. International Conference on Machine Learning, 2020.
[51] Z. Chen, C.C. Su, A. Chen, Top-down or bottom-up? A network agenda-setting study of Chinese nationalism on social media, Journal of Broadcasting & Electronic Media 63(3)(2019) 512-533.
[52] J.J. Heckman, T. Rudelius, Top-down approach to 6D SCFTs, Journal of Physics A: Mathematical and Theoretical 52(9) (2019) 093001.
[53] T.F. Gordon, H. Friedrich, D. Walton, Representing argumentation schemes with constraint handling rules (CHR), Argument & Computation 9(2)(2018) 91-119.