Extended Provenance Management for Data Science Applications

Tanja Auge
Supervised by Prof. Andreas Heuer
University of Rostock, Germany
tanja.auge@uni-rostock.de

ABSTRACT

Research data management deals with tracking and archiving of data collected during scientific projects, experiments or observations. The path from data collection to publication should thus be kept comprehensible, reconstructable and plausible. The continuous growth of data, frequent schema changes as well as the varied evaluation of the data makes the storage of every possible database state a very complicated and lengthy task.

With the help of data provenance, however, we can determine which part of the primary research data must be stored long-term in order to ensure the reproducibility of the evaluations. It should also be possible to recalculate changes to data and schemata so that old data records do not have to be archived completely. In addition, the stored data must not conflict with existing privacy guidelines.

1. INTRODUCTION

The presentation and publication of research results increasingly requires the publication of the corresponding research data, which ensures the findability, accessibility, interoperability, and reusability in the sense of FAIR Data Principles[1]. In our research, we concentrate on the structured data, e.g. resulting from measurement series, experiments, or always-on sensors, stored in a relational database. The FAIR principle does not necessarily mean, that the entire database of a project has to be stored and published, but only that part of the database which is necessary for the traceability and/or reproducibility of the respective publication. The difficulty now is to generate exactly this minimal part of the original research database, called *sub-database*. The need for the data reduction can be due to high costs in collecting or evaluating the data (expensive or elaborately produced), to privacy aspects when evaluating personal data, or to intellectual property preservation. In our case, the evaluation of the research database is restricted to a relational query language, starting from conjunctive queries, and adding arithmetic or aggregation functions lateron.

[1]https://www.go-fair.org/

Proceedings of the VLDB 2020 PhD Workshop, August 31st, 2020. Tokyo, Japan. Copyright (C) 2020 for this paper by its authors. Copying permitted for private and academic purposes.

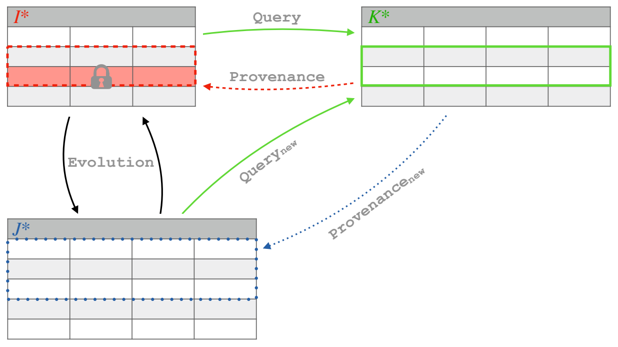

Figure 1: Query Evaluation, Provenance and Schema Evolution

Thus, our goal is not only to store the evaluation query and the query result itself, but also the relevant source data. However, if data and/or schema change frequently, the original database must be "frozen" and saved after each evaluation carried out on the dataset. To avoid this, we use *provenance management techniques* [7, 6] to calculate the sub-database before (red highlighted) or after evolution (blue highlighted) required to reproduce the query result (green highlighted in Figure 1).

After the new *general data protection regulation* (GDPR) becoming valid, it was apparent that the storage of research data, even without containing personal data, may fall under the aspect of privacy. Reasons for the additional privacy requirements are high costs, a lot of time as well as the great effort required to collect the research data. This implies a natural conflict of interest between publishing original data (*provenance*) and protecting these data (*privacy*) for reasons of competition.

All in all, this results in three central research questions, which are summarized in Figure 1:

(I.) How to calculate the minimal part of the original research database that has to be stored permanently to achieve replicable research? (red highlighted)

(II.) How to unify the theories behind data provenance and schema evolution? (blue highlighted)

(III.) How to combine Data Provenance and Privacy aspects in the case of query inversion? (locked tuple)

2. PROBLEM DESCRIPTION

Let us take a more detailed look at the different research questions. A detailed description of the problems (I.) and (II.) can be found in [1]. In the course of the PhD project the third question (III.) arose, which illustrates the practical relevance of the conflict between provenance and privacy.

Calculation of a minimal Sub-database (Figure 2). Provided that the query result (green highlighted) and the *evaluation query* is archived, one specific problem is, to determine the minimal (additional) information that is required for the reconstruction of the sub-database (red highlighted). Occasionally, we have to save entire tuples or parts of the database directly. Using *provenance management*, we can specify this necessary information. The so calculated *minimal sub-database* is able to reconstruct the results of the evaluation query under the following boundary conditions: (1) The number of tuples of the original is retained, (2) the sub-database can be homomorphically mapped to the original database, and (3) the sub-database is an intensional description of the original database.

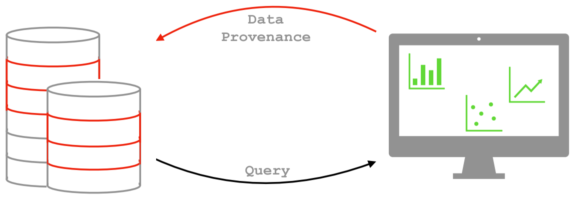

Figure 2: Calculation of a minimal sub-database

Unification of Provenance and Evolution (Figure 3). Previous provenance queries have usually been processed on a given fixed database and an evaluation query. The combination of data provenance with schema and data evolution should enable the evaluation of provenance queries with changing schemata. Under evolution the new query evaluation (green dotted) can be directly calculated as a composition of the original query evaluation and the inverse evolution (black). It is therefore sufficient to memorize one of the two minimal sub-databases I^* (red highlighted) or J^* (blue highlighted), the other sub-database can be calculated with the help of the inverse.

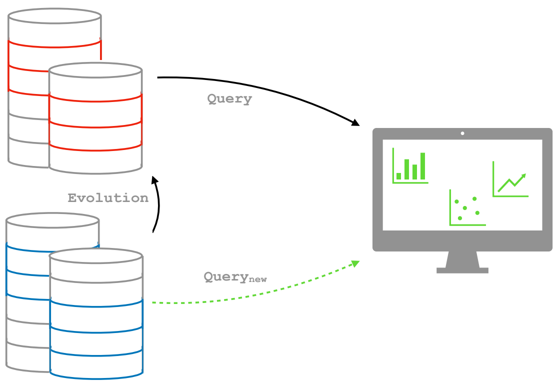

Figure 3: Unification of Provenance and Evolution

Privacy in the case of Query Inversion (Figure 4). The determination of a sub-database (red highlighted) based on the query result (green highlighted) is not always possible or permitted. For example, aggregated data cannot be inverted without storing additional information. Personal data, on the other hand, may not be published without a certain anonymization (see locked tuple). It is therefore necessary to generate a partial or generalized database that satisfies the provenance criteria on the one hand and does not contradict the privacy aspect on the other. This implies a natural conflict of interest between publishing original data (*provenance*) and protecting these data (*privacy*).

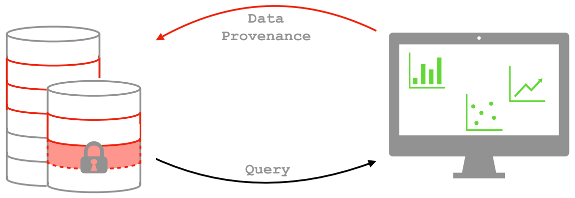

Figure 4: Privacy in the case of query inversion

3. STATE OF THE ART

Dependencies. The best known conditions are *key dependencies*, *functional dependencies* (FD) or *join dependencies* (JD). These can be extended to much more general dependencies called *(source-to-target) tuple generating dependencies* ((s-t) tgd) and *equality generating dependencies* (egd). While s-t tgds are used as a kind of inter-database dependencies, tgds – an s-t tgd on only one database schema – as well as egds can be seen as intra-database dependencies representing integrity constraints within a database [9, 10].

CHASE. The CHASE is a procedure that modifies a given object $\bigcirc$ by incorporating a parameter $\star$. We represent this by: $\text{chase}_\star(\bigcirc) = \circledast$. While the object $\bigcirc$ can represent both queries and instances, we understand the parameter $\star$ as set of dependencies like (s-t) tgds and/or egds. There are already first approaches to generalize the CHASE to (arbitrary) objects and parameters [3].

In our use case s-t tgds create new tuples and tgds/egds clean the database by replacing null values until the CHASEd database satisfies all given dependencies [13, 5]. The CHASE on instances can be used for data exchange, data integration, query answering on incomplete databases, or data cleaning, among others.

CHASE-inverse. A CHASE-inverse is an inverse function calculated via the CHASE algorithm. Weaker variants like the *relaxed CHASE-inverse* [10], *tuple-preserving relaxed* and *result equivalent CHASE-inverse* [2] do not guarantee an exact and unique inverse.

Data Provenance. Given a database instance I and an evaluation query Q, *data provenance* describes (1) where a result tuple r does come from (***where***-provenance), (2) why and (3) how r exists in the result $Q(I)$. ***Why***-provenance [6] specifies a witness base that identifies the tuples involved in the calculation of r. The question of how a result tuple r

is calculated is answered by ***how***-provenance using provenance polynomials. These polynomials give a concrete calculation of r. They are defined by a commutative semi-ring $(\mathbb{N}[X], +, \cdot, 0, 1)$ with $+$ for union and projection as well as $\cdot$ for natural join [14].

Why and ***where*** can be derived from the result of the ***how***-provenance. For this we can define a reduction based on the information content: ***where*** $\preceq$ ***why*** $\preceq$ ***how***. Therefore, we often only concentrate on ***how***-provenance. When including privacy aspects, however, the ***why***- and ***where***-provenance should not be neglected.

Provenance under Schema Evolution. The description of schema development using *schema modification operators* (SMO) such as `CREATE table`, `ADD` or `DROP column` enables schema and corresponding data changes [8]. First approaches to combining schema evolution and provenance are given in [12] and [11], the latter supporting a total of three types of provenance queries: (1) data provenance queries, (2) schema provenance queries and (3) statistics queries.

Privacy. Privacy refers the protection of (personal) data against unauthorized collection, storage and publication. Important criteria in this context are for example *k-anonymity* and *l-diversity* [15]. These are necessary, since a tuple can often be uniquely identified by apparently harmless attributes, so-called *quasi-identifiers*. The goal of our research is to reconstruct the original database as accurately as possible. This implies a natural conflict of interest between publishing original data (provenance) and protecting these data (privacy) [4].

4. PREVIOUS RESULTS

In order to unify the different theories, we represent evaluation queries, provenance queries and evolution functions as s-t tgds, so that the CHASE algorithm can be applied as a technique. The CHASE is thus a formalization of the evaluation as well as evolution of the research database. In a second step, called *BACKCHASE process*, we use the CHASE again to generate a provenance query based on the result of the evaluation query. Our theories of inverse functions can be developed from the already existing work of Fagin [10].

Calculation of a minimal Sub-database. Let I be a database instance, $\mathcal{M}$ a schema mapping defined by a s-t tgd and $I^* = \text{chase}_{\mathcal{M}^*}(\text{chase}_{\mathcal{M}}(I))$ the minimal sub-database calculated by applying CHASE twice, red highlighted in Figure 2. While an *exact CHASE-inverse* always reconstructs the original database itself, weaker variants only require *data exchange equivalence* and the existence of a homomorphism between I^* and I. To preserve the number of tuples we define the *tuple preserving relaxed CHASE-inverse* (tp-relaxed). To specify a CHASE-inverse for aggregated functions as well, we define the *result equivalent CHASE-inverse* [2]. Both definitions are based on the theory of Fagin [10].

The result equivalent CHASE-inverse is therefore the weakest CHASE-inverse. Overall, this results in the reduction

$$\text{result equivalent } \preceq \text{relaxed} \preceq \text{tp-relaxed} \preceq \text{exact},$$

which forms the sufficient conditions for the existence of a CHASE-inverse. The necessary conditions, on the other hand, refer to the existence of homomorphisms, an equal number of tuples as well as result equivalence [2].

An exact $(=)$, (tp-)relaxed $(\preceq_{\text{tp}})$ or result equivalent $(\leftrightarrow)$ CHASE-inverse can be specified for each basic operation like π, $\bowtie$, σ or `AVG`. Adding provenance information such as provenance polynomials [14] and (minimal) witness bases [6] allows the specification of stronger CHASE-inverse schema mappings then without. Thus, in the case of a projection, formalized as $R(a, b, c) \rightarrow S(a, c)$, the inverse function $S(a, c) \rightarrow \exists d : R(a, d, c)$ is tp-relaxed instead of relaxed. For other operations such as selection on the other hand, the inverse type cannot be improved despite additional information [2].

Unification of Provenance and Evolution. Given a database instance I and its evolution J, we can differentiate between 15 schema modifications like `CREATE table`, `ADD` or `DROP column`. Theses modifications can be formalized using the *schema modification operators* defined in [11]. We examined the most common schema modification operators with reference to their CHASE-inverses and extend them with ***why***- and ***how***-provenance as well as additional annotations. The most common operators are `DECOMPOSE`, `JOIN` and `MERGE Table` as well as `MERGE Column`. Currently we are evaluating the remaining 11 operators for their CHASE-inverse functions without and with using data provenance.

Among other things, we found that in some research institutions the SMOs defined by Zaniolo et al. are not sufficient [11]. In corporation with the Leibniz Institute for Baltic Sea Research Warnemünde we were able to determine that their schema modifications contain a lot of merging and splitting operations. We therefore define two operators `MERGE Column` and `SPLIT Column` as sequence of `ADD` and `DROP` operations. These merge function can be formalized as $R(a, b, c) \rightarrow S(b, f(a, c))$ with an inverse function $S(b, f(a, c)) \rightarrow \exists d, e : R(d, b, e)$. Depending on the use or not use of provenance information we can generate a tp-relaxed or exact CHASE-inverse resulting in a better reconstructed sub-database.

Privacy in the case of Query Inversion. For us, the term privacy goes beyond the term of (usually personal) data protection. Rather, we refer to the protection of research data in general. Reasons for protecting research data include economic (company protection), personal (personal data) or financial aspects. The creation of such data is often time-consuming and expensive. The identification of personal or internal company information should also be strictly prevented.

When reconstructing the minimal sub-database only those tuples may be reconstructed which do not contradict privacy aspects. Depending on the selected provenance, different data protection problems have to be considered [4]: (1) Using relation names as ***where***-provenance, there is generally not enough data worth protecting and reproducibility of the data is not guaranteed. Data protection aspects are therefore negligible. (2) In the case of ***why***, we may encounter privacy problems, if the variance of the distribution of attribute values is equal to zero. However, this only applies for special cases not known to the user interpreting the results of the provenance queries. (3) ***How***-provenance often calculates too much recoverable information, so that privacy aspects are likely to be a major problem with this approach.

(4) If we interpret ***where*** as tuple names and we save not only the scheme but the tuple itself, this can lead to major privacy problems. However, this second ***where*** approach is subject of our current work.

For solving problems generated by the different provenance queries, different approaches such as generalization and suppression, permutation of attribute values, differential privacy, and intensional (instead of extensional) answers have been developed. The next step is now to examine them for their compatibility with ***where***-, ***why***- and ***how***-provenance.

Generalization of the CHASE. We are currently working on adapting the CHASE variant presented in [5] to (arbitrary) objects $\bigcirc$ and parameters $\star$. Initial approaches to this are described in [3]. So the parameter $\star$ is always represented as a intra- or inter-database dependency and the predefined hierarchy of dependencies JD $\preceq$ tgd $\preceq$ s-t tgd and FD $\preceq$ egd allows to display all dependencies as s-t tgds respectively egds. Views can also be displayed as such dependencies. This allows the usage of a general parameter for all today's relevant CHASE applications like semantic optimization, answering queries using views, data exchange and data cleaning, query rewriting and many more.

The CHASE object $\bigcirc$ is either a query Q or a database instance I. In both cases variables/null values can be replaced by other variables/null values or constants. The variable substitution depends on certain conditions shown in [3]. Our goal here is to develop a tool that execute multiple CHASE applications. For the best of our knowledge, all currently existing tools are always designed for only one use case.

5. FUTURE WORK

Both in the calculation of a minimal sub-database and in the unification of provenance and evolution, there are still open questions to be answered. First, a concrete BACKCHASE process must be defined using provenance polynomials and witness bases. Secondly, we have to evaluate the remaining 11 SMOs for their CHASE-inverse functions without and with the use of data provenance. Further, we need to examine the different privacy approaches for their compatibility with ***where***-, ***why***-, and ***how***-provenance.

In addition to these specific questions, we also want to know whether our results are applicable for other use cases than research data management. And of course we hope to further develop the theory of generalized CHASE a little bit.

6. CONCLUSION

We presented the current status and plans to extend our work in the field of provenance management considering evolution and privacy aspects. We defined the tp-relaxed and result equivalent CHASE-inverse, examined the different evaluation and evolution operators for their inverse types without and with using data provenance and started to critically study the correlation between provenance and privacy.

7. ACKNOWLEDGMENTS

Special thanks go to my PhD supervisor Andreas Heuer and my mentor Goetz Graefe as well as to my colleagues from the database chair of the University of Rostock for their support during my PhD studies so far. Thanks to the Leibniz Institute for Baltic Sea Research Warnemünde for providing their research data and thanks also to my students for the many interesting discussions about privacy, provenance, evolution and the CHASE algorithm.

8. REFERENCES

[1] T. Auge and A. Heuer. Combining Provenance Management and Schema Evolution. In *IPAW*, volume 11017 of *Lecture Notes in Computer Science*, pages 222–225. Springer, 2018.

[2] T. Auge and A. Heuer. The Theory behind Minimizing Research Data — Result equivalent CHASE-inverse Mappings. In *LWDA*, volume 2191 of *CEUR Workshop Proceedings*, pages 1–12. CEUR-WS.org, 2018.

[3] T. Auge and A. Heuer. ProSA — Using the CHASE for Provenance Management. In *ADBIS*, volume 11695 of *Lecture Notes in Computer Science*, pages 357–372. Springer, 2019.

[4] T. Auge, N. Scharlau, and A. Heuer. Privacy Aspects of Provenance Queries. Accepted for ProvenanceWeek, 2020.

[5] M. Benedikt, G. Konstantinidis, G. Mecca, B. Motik, P. Papotti, D. Santoro, and E. Tsamoura. Benchmarking the Chase. In *PODS*, pages 37–52. ACM, 2017.

[6] P. Buneman, S. Khanna, and W. C. Tan. Why and Where: A Characterization of Data Provenance. In *ICDT*, volume 1973, pages 316–330. Springer, 2001.

[7] 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.

[8] C. Curino, H. J. Moon, A. Deutsch, and C. Zaniolo. Update rewriting and integrity constraint maintenance in a schema evolution support system: PRISM++. *Proc. VLDB Endow.*, 4(2):117–128, 2010.

[9] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data Exchange: Semantics and Query Answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[10] R. Fagin, P. G. Kolaitis, L. Popa, and W. C. Tan. Schema Mapping Evolution Through Composition and Inversion. In *Schema Matching and Mapping*, pages 191–222. Springer, 2011.

[11] S. Gao and C. Zaniolo. Provenance Management in Databases Under Schema Evolution. In *TaPP*. USENIX Association, 2012.

[12] B. Glavic, G. Alonso, R. J. Miller, and L. M. Haas. TRAMP: Understanding the Behavior of Schema Mappings through Provenance. *Proc. VLDB Endow.*, 3(1):1314–1325, 2010.

[13] S. Greco, C. Molinaro, and F. Spezzano. *Incomplete Data and Data Dependencies in Relational Databases*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.

[14] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40. ACM, 2007.

[15] P. Samarati. Protecting Respondents' Identities in Microdata Release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.