Modern Datalog Engines

Other titles in Foundations and Trends® in Databases

Natural Language Interfaces to Data
Abdul Quamar, Vasilis Efthymiou, Chuan Lei and Fatma Özcan
ISBN: 978-1-63828-028-6

Trends in Explanations: Understanding and Debugging Data-driven Systems
Boris Glavic, Alexandra Meliou and Sudeepa Roy
ISBN: 978-1-68083-880-0

Differential Privacy for Databases
Joseph P. Near and Xi He
ISBN: 978-1-68083-850-3

Database Systems on GPUs
Johns Paul, Shengliang Lu and Bingsheng He
ISBN: 978-1-68083-848-0

Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases
Gerhard Weikum, Xin Luna Dong, Simon Razniewski and Fabian Suchanek
ISBN: 978-1-68083-836-7

Cloud Data Services: Workloads, Architectures and Multi-Tenancy
Vivek Narasayya and Surajit Chaudhuri
ISBN: 978-1-68083-774-2

Modern Datalog Engines

Bas Ketsman
Vrije Universiteit Brussel
bas.ketsman@vub.be

Paraschos Koutris
University of Wisconsin-Madison
paris@cs.wisc.edu

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Databases

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

B. Ketsman and P. Koutris. *Modern Datalog Engines*. Foundations and Trends® in Databases, vol. 12, no. 1, pp. 1–68, 2022.

ISBN: 978-1-63828-043-9

Foundations and Trends® in Databases

Volume 12, Issue 1, 2022

Editorial Scope

Topics

Foundations and Trends® in Databases publishes survey and tutorial articles in the following topics:

- Data Models and Query Languages
- Query Processing and Optimization
- Storage, Access Methods, and Indexing
- Transaction Management, Concurrency Control and Recovery
- Deductive Databases
- Parallel and Distributed Database Systems
- Database Design and Tuning
- Metadata Management
- Object Management
- Trigger Processing and Active Databases
- Data Mining and OLAP
- Approximate and Interactive Query Processing
- Data Warehousing
- Adaptive Query Processing
- Data Stream Management
- Search and Query Integration
- XML and Semi-Structured Data
- Web Services and Middleware
- Data Integration and Exchange
- Private and Secure Data Management
- Peer-to-Peer, Sensornet and Mobile Data Management
- Scientific and Spatial Data Management
- Data Brokering and Publish/Subscribe
- Data Cleaning and Information Extraction
- Probabilistic Data Management

Information for Librarians

Foundations and Trends® in Databases, 2022, Volume 12, 4 issues. ISSN paper version 1931-7883. ISSN online version 1931-7891. Also available as a combined paper and online subscription.

Contents

Modern Datalog Engines

Bas Ketsman[1] and Paraschos Koutris[2]

[1] *Vrije Universiteit Brussel, Belgium; bas.ketsman@vub.be*
[2] *University of Wisconsin-Madison, USA; paris@cs.wisc.edu*

ABSTRACT

Recent years have seen a resurgence of interest from both the industry and research community in Datalog. Datalog is a declarative query language that extends relational algebra with recursion. It has been used to express a wide spectrum of modern data management tasks, such as data integration, declarative networking, graph analysis, business analytics, and program analysis. The result of this long line of research is a plethora of Datalog engines, which support different variants of Datalog, and have different technical specifications and capabilities. In this monograph, we provide an overview of the architecture and technical characteristics of these Datalog engines. We identify common architectural decisions and evaluation methods, as well as data structures and layouts used to speed up the query execution. We also discuss in what ways Datalog engines differ when they specialize to workloads with different characteristics (for example, data analytics vs program analysis vs graph analysis). One particular focus is how modern Datalog engines scale to massively parallel environments.

Bas Ketsman and Paraschos Koutris (2022), "Modern Datalog Engines", Foundations and Trends® in Databases: Vol. 12, No. 1, pp 1–68. DOI: 10.1561/1900000073.

1

Introduction

Recent years have seen a resurgence of interest from both the industry and research community in Datalog. Datalog is a declarative query language that extends Relational Algebra with recursion. It offers a simple declarative interface to the developer, while allowing for general optimizations that can speed up evaluation both in single-machine and parallel settings. During the past two decades, Datalog has been used to express a wide spectrum of tasks in different application domains, such as data integration (Fagin *et al.*, 2003), declarative networking (Loo *et al.*, 2006), graph analysis (Seo *et al.*, 2015; Shkapsky *et al.*, 2016), business analytics (Aref *et al.*, 2015), program analysis (Whaley and Lam, 2004; Smaragdakis and Balatsouras, 2015), and security (Marczak *et al.*, 2010).

Datalog received a lot of academic interest during the late 1980s and early 1990s. During this time, the theoretical background of Datalog was firmly established, including its syntax, semantics, and evaluation methods. Moreover, several mature Datalog systems were developed, among them Coral (Ramakrishnan *et al.*, 1993), LDL (Chimenti *et al.*, 1990), and Glue-Nail (Derr *et al.*, 1994). However, these systems were not widely adopted and their development ceased.

This trend was reversed during the last decade, with the development and deployment of several Datalog engines, both from academia and industry. The list of systems includes LogicBlox (Aref *et al.*, 2015), BigDatalog (Shkapsky *et al.*, 2016), SociaLite (Seo *et al.*, 2015), bddbddb (Whaley and Lam, 2004), Soufflé (Scholz *et al.*, 2016), RecStep (Fan *et al.*, 2019), and Graspan (Wang *et al.*, 2017). In order to deal with the huge volume of data they have to process, the design of these Datalog engines focused around efficiency and scalable performance. Apart from this, their architecture and applications show a lot of variation. Some Datalog systems were developed to target program analysis tasks, a fundamental area in the field of programming languages (e.g., bddbddb, Soufflé). Other Datalog engines specialized on data management problems, such as graph processing (e.g, SociaLite) or business analytics (e.g., LogicBlox). This variation has created a diverse ecosystem of systems that support different variants of Datalog, make different design decisions, and have different capabilities.

In this monograph, we dive deep into this ecosystem and provide an overview of the architecture and technical characteristics of the above Datalog systems. We identify common architectural decisions and evaluation methods, as well as data structures and layouts used to speed up the query execution. We also discuss in what ways Datalog engines differ when they specialize to workloads and inputs with different characteristics (for example, data analytics vs program analysis vs graph analysis). One particular focus of this monograph is how modern Datalog engines scale to massively parallel environments, which is necessary to support the processing of very large datasets. This is a particularly challenging task, since evaluating a Datalog program is generally not an embarrassingly parallel task.

Finally, we remark that Datalog engines have a different focus than SQL engines. While SQL engines are typically treated as one of many interconnected elements in a fully-blown, transactional database management system, Datalog engines are generally more stripped down with a strong focus on efficient recursive query processing. Another difference is that most SQL engines make use of bag-semantics with duplicate elimination as an afterthought, while Datalog engines rely on set-semantics and therefore require specialized data structures and techniques for efficiently dealing with sets throughout the entire computation.

1.1 An Overview of Datalog Systems

Next, we briefly introduce and describe the Datalog systems that will be the focus of this monograph.

LogicBlox (Aref *et al.*, 2015) is a proprietary Datalog system that provides a general platform for developing enterprise software. It has been used as the underlying engine for both points-to program analysis by Doop (Bravenboer and Smaragdakis, 2009), as well as for security applications by SecureBlox (Marczak *et al.*, 2010).

BigDatalog (Shkapsky *et al.*, 2016) is a parallel Datalog engine built on top of a modified version of Apache Spark. Its main applications are business and graph analytics.

SociaLite (Seo *et al.*, 2015) is a parallel Datalog engine used mainly for social network analysis. It is optimized for executing programs on graphs, and has both a single-threaded and parallel implementation.

bddbddb (Whaley and Lam, 2004) is a single-threaded implementation of Datalog designed specifically to support context-sensitive program analysis. Its novelty is the use of binary decision diagrams (BDDs) to compactly represent relations.

Soufflé (Scholz *et al.*, 2016) is a state-of-the-art open-source Datalog system that is used for different types of program analysis, with an emphasis on scalability. It compiles Datalog programs to fast parallel C++ code.

RecStep (Fan *et al.*, 2019) is a multi-threaded implementation of Datalog focused on main-memory execution. It compiles Datalog programs directly to a sequence of SQL queries, which is then executed on Quickstep, a parallel in-memory relational database engine (Patel *et al.*, 2018).

RapidNet (RapidNet, n.d.) is a distributed Datalog engine for the NDLog (Abadi and Loo, 2007) Datalog variant. It compiles Datalog programs directly into C++ programs which run over the NS-3 network simulator (ns-3, n.d.).

Myria and Naiad Finally, we will include in our study two efforts of executing Datalog programs on top of scalable parallel systems: Myria (Halperin *et al.*, 2014) and Naiad (Murray *et al.*, 2013). These do not constitute fully-fledged Datalog implementations, since they require that the user manually specifies an execution plan for the Datalog program they want to run.

In addition to the aforementioned systems, there exist several other Datalog engines: AbcDatalog (Bembenek *et al.*, n.d.), the μZ system built on top of Z3 (Hoder *et al.*, 2011), DES, Differential Datalog (Ryzhyk and Budiu, 2019), the DLV engine (Leone *et al.*, 2006), which supports Disjunctive Datalog, Dyna (Eisner and Filardo, 2011), a system that extends Datalog for AI applications, Bud (Alvaro *et al.*, 2011) and WebdamLog (Moffitt *et al.*, 2015). A few other efforts have focused on adding recursion or fixpoint computation on top of existing relational systems without achieving the expressivity of full Datalog. These include extensions of MapReduce (Afrati *et al.*, 2011; Shaw *et al.*, 2012), or more recent work by Gu *et al.* (2019) and Jachiet *et al.* (2020). Although the above systems will not be the focus of this monograph, we will present some of their techniques that are relevant to recursive computation.

1.2 Commercial Impact

In addition to the success of LogicBlox, several other commercial systems that use Datalog or dialects of Datalog have been developed over the last decade. We discuss next some of these developments.

Datomic is a transactional and distributed database that uses an extended form of Datalog as the basic query language. Semmle, a platform that supports code analysis over large code bases (acquired by Github), has developed its own query language, called .QL, which can be viewed as an object-oriented version of Datalog. Nicira (acquired by

VMware) also uses a restricted variant of the Datalog language, called nLog, for software defined networking. Finally, Datalog was used as the underlying language for declarative analytics in Netsil (acquired by Nutanix).

1.3 Relationship with Prior Work

This survey aims to present how modern large-scale systems evaluate Datalog. It is orthogonal to a rich set of papers and surveys on the Datalog language. What follows is a short but not complete list of relevant work that is complementary to this monograph.

- The most related survey to this monograph is by Green *et al.* (2013). It covers the core Datalog language and its extensions, semantics, query optimizations, incremental view maintenance, along with discussing modern applications of Datalog. Although there is some content overlap with this monograph, our focus is on system building and parallel evaluation. Furthermore, there has been tremendous progress in the adoption of Datalog techniques and many new use cases have been added after its publication.
- Several articles provide a general overview of Datalog and techniques for optimizing recursive computation (Ceri *et al.*, 1989; Ramakrishnan and Ullman, 1995; Bancilhon and Ramakrishnan, 1986). The basic principles of Datalog are also covered in several database textbooks (for example, see Abiteboul *et al.* (1995) and Ullman (1989)).
- A long line of work in the Programming Languages community has also studied the application of Datalog to different types of program analysis tasks (Reps, 1993; Lam *et al.*, 2005). For a comprehensive survey on this subject, we refer the reader to Smaragdakis and Balatsouras (2015).

1.4 Organization

The monograph is organized as follows.

The Datalog Language In Section 2, we introduce the core Datalog language and present its syntax and semantics. Then, we study how different Datalog engines extend the core language to increase its expressibility, with features that are necessary to make it usable in practical settings. These features include negation, (recursive) aggregation, arithmetic operations and functions. We discuss how some Datalog engines choose combinations of features that lead to imprecise language semantics.

Methods for Datalog Evaluation In Section 3, we discuss the fundamental methods used for Datalog evaluation. We focus on the most widely used technique, called *semi-naïve evaluation*, and we discuss the design choices of implementing it. Next, we study how modern Datalog engines use parallelism to further speed up evaluation and why this can be done effectively – both from a theoretical and practical viewpoint. Our focus is both on parallel and multi-threaded environments. We look at the design choices for parallel evaluation across different axes: data partitioning methods, synchronous vs asynchronous evaluation, and data shuffling strategies.

Data Layouts and Indices In Section 4, we study the data layouts used by Datalog engines. Even though the input of Datalog programs is relational data, many systems choose formats other than the traditional row-store: these include tries, binary decision diagrams, bit matrices, and tail-nested tables. These layouts are specialized to be performant for specific inputs and workloads (e.g., graph-oriented data, context-sensitive pointer analysis, dense relations). We also discuss indexing techniques that are specialized and fine-tuned to Datalog evaluation.

Optimization Techniques In Section 5, we present the different optimizations that are used by Datalog engines, both on the language level (e.g., program transformations and rewritings), as well as low-level optimizations on the operator level. Specifically, we study how we can rewrite a Datalog program to push selection and aggregation through recursion, or to reduce the number of iterations and strata.

Finally, we conclude the monograph in Section 6 where we discuss opportunities for future research directions and new possible applications for Datalog engines.

References

ns-3. "Network Simulator 3". URL: http://www.nsnam.org/.

Abadi, M. and B. T. Loo. (2007). "Towards a Declarative Language and System for Secure Networking". In: *NetDB*. USENIX Association.

Abiteboul, S., M. Bienvenu, A. Galland, and É. Antoine. (2011). "A rule-based language for web data management". In: *PODS*. ACM. 293–304.

Abiteboul, S., R. Hull, and V. Vianu. (1995). *Foundations of Databases*. Addison-Wesley.

Afrati, F. N., V. R. Borkar, M. J. Carey, N. Polyzotis, and J. D. Ullman. (2011). "Map-reduce extensions and recursive queries". In: *EDBT*. ACM. 1–8.

Afrati, F. N., S. S. Cosmadakis, and M. Yannakakis. (1995). "On Datalog vs. Polynomial Time". *J. Comput. Syst. Sci.* 51(2): 177–196.

Afrati, F. N. and C. H. Papadimitriou. (1987). "The Parallel Complexity of Simple Chain Queries". In: *PODS*. ACM. 210–213.

Afrati, F. N. and J. D. Ullman. (2010). "Optimizing joins in a map-reduce environment". In: *EDBT*. Vol. 426. ACM. 99–110.

Afrati, F. N. and J. D. Ullman. (2012). "Transitive closure and recursive Datalog implemented on clusters". In: *EDBT*. ACM. 132–143.

Agrawal, S., S. Chaudhuri, and V. R. Narasayya. (2000). "Automated Selection of Materialized Views and Indexes in SQL Databases". In: *VLDB*. Morgan Kaufmann. 496–505.

Albarghouthi, A., P. Koutris, M. Naik, and C. Smith. (2017). "Constraint-Based Synthesis of Datalog Programs". In: *CP*. Vol. 10416. *Lecture Notes in Computer Science*. Springer. 689–706.

Alvaro, P., N. Conway, J. M. Hellerstein, and W. R. Marczak. (2011). "Consistency Analysis in Bloom: a CALM and Collected Approach". In: *CIDR*. 249–260.

Alvaro, P., W. R. Marczak, N. Conway, J. M. Hellerstein, D. Maier, and R. Sears. (2010). "Dedalus: Datalog in Time and Space". In: *Datalog*. Vol. 6702. *Lecture Notes in Computer Science*. Springer. 262–281.

Ameloot, T. J., B. Ketsman, F. Neven, and D. Zinn. (2016). "Weaker Forms of Monotonicity for Declarative Networking: A More Fine-Grained Answer to the CALM-Conjecture". *ACM Trans. Database Syst.* 40(4): 21:1–21:45.

Ameloot, T. J., F. Neven, and J. V. den Bussche. (2013). "Relational transducers for declarative networking". *J. ACM*. 60(2): 15:1–15:38.

Aref, M., B. ten Cate, T. J. Green, B. Kimelfeld, D. Olteanu, E. Pasalic, T. L. Veldhuizen, and G. Washburn. (2015). "Design and Implementation of the LogicBlox System". In: *SIGMOD Conference*. ACM. 1371–1382.

Bancilhon, F. and R. Ramakrishnan. (1986). "An Amateur's Introduction to Recursive Query Processing Strategies". In: *SIGMOD Conference*. ACM Press. 16–52.

Beame, P., P. Koutris, and D. Suciu. (2017). "Communication Steps for Parallel Query Processing". *J. ACM*. 64(6): 40:1–40:58.

Beeri, C. and R. Ramakrishnan. (1987). "On the Power of Magic". In: *PODS*. ACM. 269–284.

Bellomarini, L., E. Sallinger, and G. Gottlob. (2018). "The Vadalog System: Datalog-based Reasoning for Knowledge Graphs". *Proc. VLDB Endow.* 11(9): 975–987.

Bembenek, A., S. Chong, and M. Gaboardi. "AbcDatalog". URL: http://abcdatalog.seas.harvard.edu/.

Bollig, B. and I. Wegener. (1996). "Improving the Variable Ordering of OBDDs Is NP-Complete". *IEEE Trans. Computers*. 45(9): 993–1002.

Bravenboer, M. and Y. Smaragdakis. (2009). "Strictly declarative specification of sophisticated points-to analyses". In: *OOPSLA*. ACM. 243–262.

Brin, S. and L. Page. (1998). "The Anatomy of a Large-Scale Hypertextual Web Search Engine". *Comput. Networks*. 30(1-7): 107–117.

Cabibbo, L. (1995). "On the Power of Stratified Logic Programs with Value Invention for Expressing Database Transformations". In: *ICDT*. Vol. 893. *Lecture Notes in Computer Science*. Springer. 208–221.

Calì, A., G. Gottlob, T. Lukasiewicz, B. Marnette, and A. Pieris. (2010). "Datalog+/-: A Family of Logical Knowledge Representation and Query Languages for New Applications". In: *LICS*. IEEE Computer Society. 228–242.

Ceri, S., G. Gottlob, and L. Tanca. (1989). "What you Always Wanted to Know About Datalog (And Never Dared to Ask)". *IEEE Trans. Knowl. Data Eng.* 1(1): 146–166.

Chimenti, D., R. Gamboa, R. Krishnamurthy, S. A. Naqvi, S. Tsur, and C. Zaniolo. (1990). "The LDL System Prototype". *IEEE Trans. Knowl. Data Eng.* 2(1): 76–90.

Conway, N., W. R. Marczak, P. Alvaro, J. M. Hellerstein, and D. Maier. (2012). "Logic and lattices for distributed programming". In: *SoCC*. ACM. 1.

Cosmadakis, S. S. and P. C. Kanellakis. (1986). "Parallel Evaluation of Recursive Rule Queries". In: *PODS*. ACM. 280–293.

Dean, J. and S. Ghemawat. (2004). "MapReduce: Simplified Data Processing on Large Clusters". In: *OSDI*. USENIX Association. 137–150.

Derr, M. A., S. Morishita, and G. Phipps. (1994). "The Glue-Nail Deductive Database System: Design, Implementation, and Evaluation". *VLDB J.* 3(2): 123–160.

DeWitt, D. J. and J. Gray. (1992). "Parallel Database Systems: The Future of High Performance Database Systems". *Commun. ACM*. 35(6): 85–98.

Eisner, J. and N. W. Filardo. (2011). "Dyna: Extending Datalog For Modern AI". In: *Datalog Reloaded*. Ed. by O. de Moor, G. Gottlob, T. Furche, and A. Sellers. Vol. 6702. *Lecture Notes in Computer Science*. Springer. 181–220.

Fagin, R., P. G. Kolaitis, R. J. Miller, and L. Popa. (2003). "Data Exchange: Semantics and Query Answering". In: *ICDT*. Vol. 2572. *Lecture Notes in Computer Science*. Springer. 207–224.

Fan, Z., J. Zhu, Z. Zhang, A. Albarghouthi, P. Koutris, and J. M. Patel. (2019). "Scaling-Up In-Memory Datalog Processing: Observations and Techniques". *Proc. VLDB Endow.* 12(6): 695–708.

Ganguly, S., A. Silberschatz, and S. Tsur. (1990). "A Framework for the Parallel Processing of Datalog Queries". In: *SIGMOD Conference*. ACM Press. 143–152.

Ganguly, S., A. Silberschatz, and S. Tsur. (1992). "Parallel Bottom-Up Processing of Datalog Queries". *J. Log. Program.* 14(1&2): 101–126.

Graefe, G. (2010). "A survey of B-tree locking techniques". *ACM Trans. Database Syst.* 35(3): 16:1–16:26.

Greco, S., C. Zaniolo, and S. Ganguly. (1992). "Greedy by Choice". In: *PODS*. ACM Press. 105–113.

Green, T. J., S. S. Huang, B. T. Loo, and W. Zhou. (2013). "Datalog and Recursive Query Processing". *Found. Trends Databases*. 5(2): 105–195.

Gu, J., Y. H. Watanabe, W. A. Mazza, A. Shkapsky, M. Yang, L. Ding, and C. Zaniolo. (2019). "RaSQL: Greater Power and Performance for Big Data Analytics with Recursive-aggregate-SQL on Spark". In: *SIGMOD Conference*. ACM. 467–484.

Halperin, D., V. T. de Almeida, L. L. Choo, S. Chu, P. Koutris, D. Moritz, J. Ortiz, V. Ruamviboonsuk, J. Wang, A. Whitaker, S. Xu, M. Balazinska, B. Howe, and D. Suciu. (2014). "Demonstration of the Myria big data management service". In: *SIGMOD Conference*. ACM. 881–884.

Hellerstein, J. M. (2010). "The declarative imperative: experiences and conjectures in distributed logic". *SIGMOD Rec.* 39(1): 5–19.

Hoder, K., N. Bjørner, and L. M. de Moura. (2011). "μZ- An Efficient Engine for Fixed Points with Constraints". In: *CAV*. Vol. 6806. *Lecture Notes in Computer Science*. Springer. 457–462.

Jachiet, L., P. Genevès, N. Gesbert, and N. Layaïda. (2020). "On the Optimization of Recursive Relational Queries: Application to Graph Queries". In: *SIGMOD Conference*. ACM. 681–697.

Jordan, H., P. Subotic, D. Zhao, and B. Scholz. (2019a). "A specialized B-tree for concurrent datalog evaluation". In: *PPoPP*. ACM. 327–339.

Jordan, H., P. Subotic, D. Zhao, and B. Scholz. (2019b). "Brie: A Specialized Trie for Concurrent Datalog". In: *PMAM@PPoPP*. ACM. 31–40.

Kanellakis, P. C. (1986). "Logic Programming and Parallel Complexity". In: *ICDT*. Vol. 243. *Lecture Notes in Computer Science*. Springer. 1–30.

Ketsman, B., A. Albarghouthi, and P. Koutris. (2020). "Distribution Policies for Datalog". *Theory Comput. Syst.* 64(5): 965–998.

Ketsman, B. and C. Koch. (2020). "Datalog with Negation and Monotonicity". In: *ICDT*. Vol. 155. *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 19:1–19:18.

Kifer, M. (1998). "On the Decidability and Axiomatization of Query Finiteness in Deductive Databases". *J. ACM*. 45(4): 588–633.

Kossmann, D. (2000). "The State of the art in distributed query processing". *ACM Comput. Surv.* 32(4): 422–469.

Koutris, P. and J. Wijsen. (2021). "Consistent Query Answering for Primary Keys in Datalog". *Theory Comput. Syst.* 65(1): 122–178.

Lam, M. S., J. Whaley, V. B. Livshits, M. C. Martin, D. Avots, M. Carbin, and C. Unkel. (2005). "Context-sensitive program analysis as database queries". In: *PODS*. ACM. 1–12.

Leone, N., G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. (2006). "The DLV system for knowledge representation and reasoning". *ACM Trans. Comput. Log.* 7(3): 499–562.

Loo, B. T., T. Condie, M. N. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica. (2006). "Declarative networking: language, execution and optimization". In: *SIGMOD Conference*. ACM. 97–108.

Loo, B. T., T. Condie, M. N. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica. (2009). "Declarative networking". *Commun. ACM*. 52(11): 87–95.

Madsen, M., M. Yee, and O. Lhoták. (2016). “From Datalog to flix: a declarative language for fixed points on lattices”. In: *PLDI*. ACM. 194–208.

Malewicz, G., M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. (2010). “Pregel: a system for large-scale graph processing”. In: *SIGMOD Conference*. ACM. 135–146.

Marczak, W. R., S. S. Huang, M. Bravenboer, M. Sherr, B. T. Loo, and M. Aref. (2010). “SecureBlox: customizable secure distributed data processing”. In: *SIGMOD Conference*. ACM. 723–734.

Martinez-Angeles, C. A., I. de Castro Dutra, V. S. Costa, and J. Buenabad-Chávez. (2013). “A Datalog Engine for GPUs”. In: *KDPD*. Vol. 8439. *Lecture Notes in Computer Science*. Springer. 152–168.

Mazuran, M., E. Serra, and C. Zaniolo. (2013). “Extending the power of datalog recursion”. *VLDB J.* 22(4): 471–493.

Meyer, U. and P. Sanders. (1998). “Delta-Stepping: A Parallel Single Source Shortest Path Algorithm”. In: *ESA*. Vol. 1461. *Lecture Notes in Computer Science*. Springer. 393–404.

Moffitt, V. Z., J. Stoyanovich, S. Abiteboul, and G. Miklau. (2015). “Collaborative Access Control in WebdamLog”. In: *SIGMOD Conference*. ACM. 197–211.

Murray, D. G., F. McSherry, R. Isaacs, M. Isard, P. Barham, and M. Abadi. (2013). “Naiad: a timely dataflow system”. In: *SOSP*. ACM. 439–455.

Patel, J. M., H. Deshmukh, J. Zhu, N. Potti, Z. Zhang, M. Spehlmann, H. Memisoglu, and S. Saurabh. (2018). “Quickstep: A Data Platform Based on the Scaling-Up Approach”. *Proc. VLDB Endow.* 11(6): 663–676.

Piatetsky-Shapiro, G. (1983). “The Optimal Selection of Secondary Indices is NP-Complete”. *SIGMOD Rec.* 13(2): 72–75.

Raghothaman, M., J. Mendelson, D. Zhao, M. Naik, and B. Scholz. (2020). “Provenance-guided synthesis of Datalog programs”. *Proc. ACM Program. Lang.* 4(POPL): 62:1–62:27.

Ramakrishnan, R., F. Bancilhon, and A. Silberschatz. (1987). “Safety of Recursive Horn Clauses With Infinite Relations”. In: *PODS*. ACM. 328–339.

Ramakrishnan, R., W. G. Roth, P. Seshadri, D. Srivastava, and S. Sudarshan. (1993). "The CORAL Deductive Database System". In: *SIGMOD Conference*. ACM Press. 544–545.

Ramakrishnan, R. and J. D. Ullman. (1995). "A survey of deductive database systems". *The Journal of Logic Programming*. 23(2): 125–149.

RapidNet. "RapidNet Declarative Networking Engine". URL: http://netdb.cis.upenn.edu/rapidnet/.

Reps, T. W. (1993). "Demand Interprocedural Program Analysis Using Logic Databases". In: *Workshop on Programming with Logic Databases (Book), ILPS. The Kluwer International Series in Engineering and Computer Science 296*. Kluwer. 163–196.

Ross, K. A. and Y. Sagiv. (1992). "Monotonic Aggregation in Deductive Databases". In: *PODS*. ACM Press. 114–126.

Rudolph, S. and M. Thomazo. (2016). "Expressivity of Datalog Variants - Completing the Picture". In: *IJCAI*. IJCAI/AAAI Press. 1230–1236.

Ryzhyk, L. and M. Budiu. (2019). "Differential Datalog". In: *Datalog*. Vol. 2368. *CEUR Workshop Proceedings*. CEUR-WS.org. 56–67.

Scholz, B., H. Jordan, P. Subotic, and T. Westmann. (2016). "On fast large-scale program analysis in Datalog". In: *CC*. ACM. 196–206.

Seib, J. and G. Lausen. (1991). "Parallelizing Datalog Programs by Generalized Pivoting". In: *PODS*. ACM Press. 241–251.

Seo, J., S. Guo, and M. S. Lam. (2013a). "SociaLite: Datalog extensions for efficient social network analysis". In: *ICDE*. IEEE Computer Society. 278–289.

Seo, J., S. Guo, and M. S. Lam. (2015). "SociaLite: An Efficient Graph Query Language Based on Datalog". *IEEE Trans. Knowl. Data Eng.* 27(7): 1824–1837.

Seo, J., J. Park, J. Shin, and M. S. Lam. (2013b). "Distributed SociaLite: A Datalog-Based Language for Large-Scale Graph Analysis". *Proc. VLDB Endow.* 6(14): 1906–1917.

Seshadri, P., H. Pirahesh, and T. Y. C. Leung. (1996). "Complex Query Decorrelation". In: *ICDE*. IEEE Computer Society. 450–458.

Sewall, J., J. Chhugani, C. Kim, N. Satish, and P. Dubey. (2011). "PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors". *Proc. VLDB Endow.* 4(11): 795–806.

Shaw, M., P. Koutris, B. Howe, and D. Suciu. (2012). "Optimizing Large-Scale Semi-Naive Datalog Evaluation in Hadoop". In: *Datalog.* Vol. 7494. *Lecture Notes in Computer Science.* Springer. 165–176.

Shkapsky, A., M. Yang, M. Interlandi, H. Chiu, T. Condie, and C. Zaniolo. (2016). "Big Data Analytics with Datalog Queries on Spark". In: *SIGMOD Conference.* ACM. 1135–1149.

Shkapsky, A., M. Yang, and C. Zaniolo. (2015). "Optimizing recursive queries with monotonic aggregates in DeALS". In: *ICDE.* IEEE Computer Society. 867–878.

Si, X., W. Lee, R. Zhang, A. Albarghouthi, P. Koutris, and M. Naik. (2018). "Syntax-guided synthesis of Datalog programs". In: *ESEC/SIGSOFT FSE.* ACM. 515–527.

Smaragdakis, Y. and G. Balatsouras. (2015). "Pointer Analysis". *Found. Trends Program. Lang.* 2(1): 1–69.

Stonebraker, M. (1985). "The Case for Shared Nothing". In: *HPTS.*

Subotić, P., H. Jordan, L. Chang, A. Fekete, and B. Scholz. (2018). "Automatic Index Selection for Large-Scale Datalog Computation". *Proc. VLDB Endow.* 12(2): 141–153.

Sudarshan, S. and R. Ramakrishnan. (1991). "Aggregation and Relevance in Deductive Databases". In: *VLDB.* Morgan Kaufmann. 501–511.

Ullman, J. D. (1989). *Principles of Database and Knowledge-Base Systems, Volume II.* Computer Science Press.

Ullman, J. D. and A. V. Gelder. (1988). "Parallel Complexity of Logical Query Programs". *Algorithmica.* 3: 5–42.

Veldhuizen, T. L. (2014). "Triejoin: A Simple, Worst-Case Optimal Join Algorithm". In: *ICDT.* OpenProceedings.org. 96–106.

Wang, J., M. Balazinska, and D. Halperin. (2015). "Asynchronous and Fault-Tolerant Recursive Datalog Evaluation in Shared-Nothing Engines". *Proc. VLDB Endow.* 8(12): 1542–1553.

Wang, K., A. Hussain, Z. Zuo, G. Xu, and A. Amiri Sani. (2017). "Graspan: A Single-Machine Disk-Based Graph System for Interprocedural Static Analyses of Large-Scale Systems Code". *SIGARCH Comput. Archit. News.* 45(1): 389–404.

Wang, Y. R., M. A. Khamis, H. Q. Ngo, R. Pichler, and D. Suciu. (2022). "Optimizing Recursive Queries with Program Synthesis". *CoRR*. abs/2202.10390.

Whaley, J. and M. S. Lam. (2004). "Cloning-based context-sensitive pointer alias analysis using binary decision diagrams". In: *PLDI*. ACM. 131–144.

Wolfson, O. (1989). "Sharing the Load of Logic-Program Evaluation". *IEEE Data Eng. Bull.* 12(1): 58–64.

Wolfson, O. and A. Ozeri. (1990). "A New Paradigm for Parallel and Distributed Rule-Processing". In: *SIGMOD Conference*. ACM Press. 133–142.

Wolfson, O. and A. Ozeri. (1993). "Parallel and Distributed Processing of Rules by Data Reduction". *IEEE Trans. Knowl. Data Eng.* 5(3): 523–530.

Wolfson, O. and A. Silberschatz. (1988). "Distributed Processing of Logic Programs". In: *SIGMOD Conference*. ACM Press. 329–336.

Yang, M., A. Shkapsky, and C. Zaniolo. (2017). "Scaling up the performance of more powerful Datalog systems on multicore machines". *VLDB J.* 26(2): 229–248.

Zaharia, M., R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica. (2016). "Apache Spark: a unified engine for big data processing". *Commun. ACM.* 59(11): 56–65.

Zaniolo, C., M. Yang, A. Das, and M. Interlandi. (2016). "The Magic of Pushing Extrema into Recursion: Simple, Powerful Datalog Programs". In: *AMW*. Vol. 1644. *CEUR Workshop Proceedings*. CEUR-WS.org.

Zaniolo, C., M. Yang, A. Das, A. Shkapsky, T. Condie, and M. Interlandi. (2017). "Fixpoint semantics and optimization of recursive Datalog programs with aggregates". *Theory Pract. Log. Program.* 17(5-6): 1048–1065.

Zhang, W., K. Wang, and S. Chau. (1995). "Data Partition and Parallel Evaluation of Datalog Programs". *IEEE Trans. Knowl. Data Eng.* 7(1): 163–176.