# Efficiency and Effectiveness of a Practical Provenance and Probabilistic DBMS

Silviu Maniu  Aryak Sen  *Pierre Senellart*


Logic and Algorithms in DB Theory and AI Reunion
Simons Institute, 23 January 2025

# In this talk

- What is a provenance-aware DBMS?
- What is a probabilistic DBMS?
- Is it possible to build such a DBMS, with:
  - Support for various forms of provenance
  - Support for a large, and practically useful, query language
  - Efficient computation of provenance and probabilities
- What to compare it with?
- Benchmarks

# Provenance in Databases

- A way to keep track of annotations through data processing tasks, that provide extra information about the results of these tasks
- Provenance framework:
  - A base data model (e.g., the relational model)
  - A query language (e.g., the positive relational algebra)
  - A data model for annotations (e.g., $\mathbb{K}$-annotated relations)
  - A semantics for queries over the annotation data model

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |
| Boolean [Imieliński and Lipski, 1984] | Arbitrary | Boolean function on tuples |

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |
| Boolean [Imieliński and Lipski, 1984] | Arbitrary | Boolean function on tuples |
| Semiring [Green et al., 2007] | Datalog | Semiring element on tuples |

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |
| Boolean [Imieliński and Lipski, 1984] | Arbitrary | Boolean function on tuples |
| Semiring [Green et al., 2007] | Datalog | Semiring element on tuples |
| M-Semiring [Geerts and Poggi, 2010] | Relational algebra | M-Semiring element on tuples |

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |
| Boolean [Imieliński and Lipski, 1984] | Arbitrary | Boolean function on tuples |
| Semiring [Green et al., 2007] | Datalog | Semiring element on tuples |
| M-Semiring [Geerts and Poggi, 2010] | Relational algebra | M-Semiring element on tuples |
| Where [Buneman et al., 2001] | Select, project, join, union | Bipartite graph on values |

# Example provenance frameworks on relations

| Provenance type | Query language | Annotations |
|---|---|---|
| Workflow [Davidson et al., 2007] | Opaque operations | Operation metadata on DAG nodes |
| Boolean [Imieliński and Lipski, 1984] | Arbitrary | Boolean function on tuples |
| Semiring [Green et al., 2007] | Datalog | Semiring element on tuples |
| M-Semiring [Geerts and Poggi, 2010] | Relational algebra | M-Semiring element on tuples |
| Where [Buneman et al., 2001] | Select, project, join, union | Bipartite graph on values |
| Semimodule [Amsterdamer et al., 2011] | Extended relational algebra | Semimodule annotation on values |

# Some important results

- Many application semirings, most of which can be extended into m-semirings, capturing useful meta-information: shortest path lengths, top-$k$ shortest path lengths and top-$k$ shortest paths, why-provenance, Trio's lineage, temporal intervals, counting...
- For the relational algebra, Boolean provenance semantics coincides with m-semiring semantics
- (M-)Semiring provenance semantics commutes with (m-)semiring homomorphism
- Universal semiring ($\mathbb{N}[X]$) and m-semiring (free m-semiring)
- Provenance tracking is generally PTIME
- Provenance circuits are generally more compact than provenance formulas

# Probabilistic database

- Compact representation of a probability distribution over possible databases
- Distributions can be finite, continuous [Abiteboul et al., 2011], discrete but infinite [Benedikt et al., 2010, Grohe and Lindner, 2020]
- Correlations across data items may be disallowed (TID) [Dalvi and Suciu, 2004], limited (BID) [Dalvi and Suciu, 2013], arbitrary (pc-tables) [Green and Tannen, 2006]
- Main problem: probabilistic query evaluation, i.e., computing the (marginal) probability of a data item in the output of a query; or sometimes computing the distribution of data values in the output of a query

# Main results

- Probabilistic query evaluation is #P-hard
- In the relational setting, becomes tractable when:
  - the Boolean provenance has some tractable representation (read-once, d-D circuit...)
  - the query is a safe UCQ [Dalvi and Suciu, 2013] over TIDs
  - the query is a safe CQ without self-joins [Dalvi et al., 2011] over BIDs
  - the data has bounded treewidth over BIDs [Amarilli et al., 2015], for any MSO query
  - the data and correlations have joint bounded treewidth [Amarilli, 2016], for any MSO query
- TIDs or BIDs are not a strong representation system for the relational algebra; pc-tables are

# ProvSQL: Provenance within PostgreSQL

[Senellart et al., 2018]

- Provenance annotations stored as Universally Unique Identifiers (UUIDs), in an extra attribute of each provenance-aware relation
- UUIDs of base tuples randomly generated; UUIDs of query results generated in a deterministic manner
- A provenance circuit relating UUIDs of elementary provenance annotations and arithmetic gates persistently stored on disk in memory-mapped files
- Query rewriting (after parsing, before planning) to automatically compute output provenance attributes in terms of the query and input provenance attributes

# Other databases with provenance/probability support

- Older probabilistic database systems can compute some forms of provenance (especially, Boolean provenance); but tied to specific version of PostgreSQL (8.3), hard to deploy

  Trio: `http://infolab.stanford.edu/trio/` [Benjelloun et al., 2006]
  MayBMS: `http://maybms.sourceforge.net/` [Huang et al., 2009]

- Perm `https://github.com/IITDBGroup/perm` [Glavic and Alonso, 2009] now obsolete system for provenance management; also tied to PostgreSQL 8.3
- ORCHESTRA `https://www.cis.upenn.edu/~zives/orchestra/` [Green et al., 2010] Java front end to DBMS with provenance support; not maintained
- GProM `http://www.cs.iit.edu/~dbgroup/projects/gprom.html` [Arab et al., 2018] similar to ProvSQL (though no probabilistic database capabilities), overlap of features; middleware
- ProbLog `https://dtai.cs.kuleuven.be/problog/` [Kimmig et al., 2011] probabilistic logic reasoning system in Python; has a (SQLite) DB backend

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:
  - linear-time processing of read-once Boolean circuits

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:
  - linear-time processing of read-once Boolean circuits
  - compilation to d-DNNFs for low-treewidth circuits

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:
  - linear-time processing of read-once Boolean circuits
  - compilation to d-DNNFs for low-treewidth circuits
  - use of external knowledge compilers (d4, c2d, dsharp...)

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:
  - linear-time processing of read-once Boolean circuits
  - compilation to d-DNNFs for low-treewidth circuits
  - use of external knowledge compilers (d4, c2d, dsharp...)
  - Monte-Carlo sampling

# ProvSQL features as of 2025

- PostgreSQL extension, works for all versions of PostgreSQL $\geqslant 10$, tested on Linux, Mac OS (x86/ARM), WSL
- Support of a large subset of SQL: multiset relational algebra + duplicate elimination + terminal aggregates (+ partial support for updates)
- Provenance computation in arbitrary semirings (through the universal semiring), in arbitrary m-semirings (through the free m-semiring), in arbitrary semimodules for aggregate provenance, where-provenance
- Specialization to various (m-)semirings: Boolean, counting, why-provenance, formulas, temporal intervals...
- Arbitrary probabilistic correlations representable
- Probabilistic query evaluation through Boolean provenance:
  - linear-time processing of read-once Boolean circuits
  - compilation to d-DNNFs for low-treewidth circuits
  - use of external knowledge compilers (d4, c2d, dsharp...)
  - Monte-Carlo sampling
- Shapley value and expected Shapley value computation

# Benchmarking ProvSQL

- We want to know:
  - What is the overhead of provenance computation?
  - Does it scale to large databases?
  - Is probabilistic query evaluation feasible in practice despite #P-hardness?
- Competitors:
  - GProM for why-provenance computation
  - MayBMS for probabilsitic query evaluation
  - ProbLog for probabilistic query evaluation
- Database: TPC-H, from 1 GB to 10 GB
- Queries: (some) TPC-H queries + ad-hoc queries on the same schema

# The case of ProbLog

Unfortunately does not scale at all, all data is brought into RAM even if using the DB backend. Simplest query of our benchmark:

```
:- use_module(library(db)).
:- sqlite_load('TPC-H.db').

1 :: result(O,P,S,LN,LS) :- lineitem(O,P,S,LN,Q,_,_,_,_,LS,_,_,_,_,SM,_), SM='AIR', Q>10 .
1 :: result(O,P,S,LN,LS) :- lineitem(O,P,S,LN,_,_,D,_,_,LS,_,_,_,_,_,_), D > .05 .

query(result(_,_,_,_,_)) .
```

- For TPC-H 100 MB, 5 GB of RAM used, timeout after 5 hours
- For TPC-H 1 GB, out-of-memory

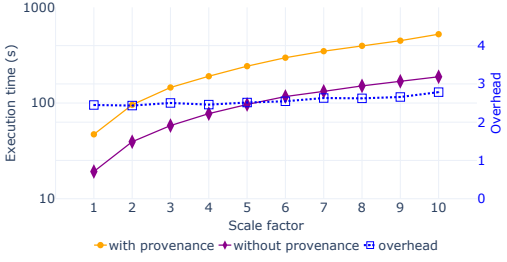
# Query support

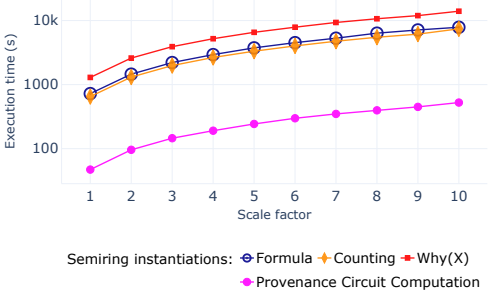
| | TPC-H | | | | | | TPC-H* | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Query** | 1 | 6 | 7 | 9 | 12 | 19 | 1 | 3 | 4 | 12 | 15 |
| ProvSQL (prov.) | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| ProvSQL (prob.) | O | Y | O | Y | Y | Y | Y | Y | Y | Y | Y |
| GProM | N | N | N | N | N | N | Y | Y | O | Y | Y |
| MayBMS | N | N | N | N | N | N | Y | Y | Y | Y | Y |

| | Custom | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Query** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| ProvSQL (prov.) | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| ProvSQL (prob.) | Y | Y | Y | Y | Y | Y | Y | Y | O | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| GProM | O | Y | Y | Y | O | O | Y | O | T | Y | Y | O | Y | Y | Y | N | Y | N |
| MayBMS | Y | Y | Y | T | O | Y | Y | Y | O | Y | Y | Y | Y | Y | Y | N | Y | N |

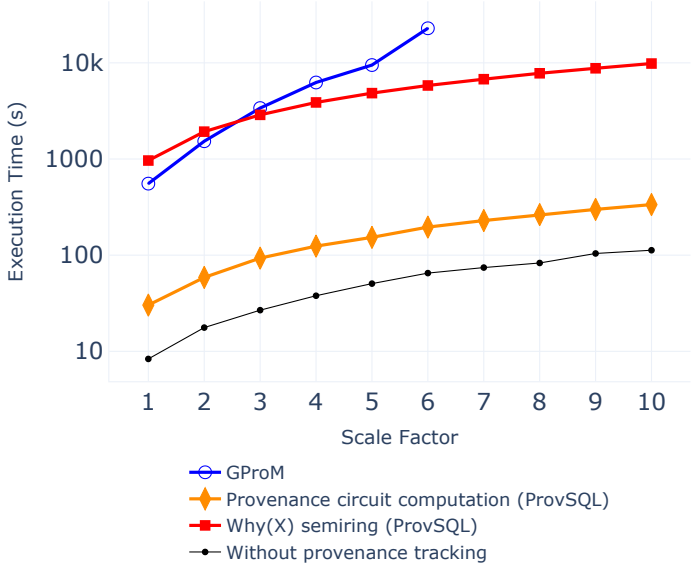
Not supported; Timeout after 3 000 s; Out-of-Memory Error

# Scalability of Provenance Computation in ProvSQL

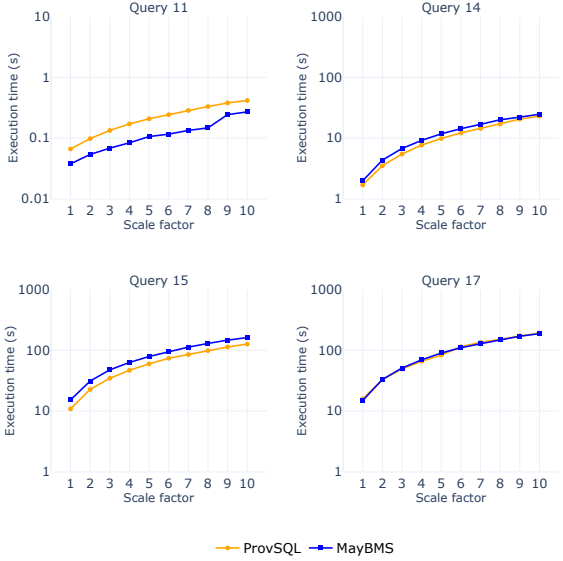


# Provenance Computation: ProvSQL vs GProM



# Probability Computation: ProvSQL vs MayBMS



# Missing and on-going features in ProvSQL

- Recursive query support: impossible because `WITH RECURSIVE` queries are very restricted in SQL (but see [Ramusat et al., 2021, Zhao et al., 2024] for how to implement provenance for Datalog within a Datalog evaluator)
- Updates: simple updates, following [Bourhis et al., 2020], in the process of being integrated; updates depending on multiple tables require more work
- Continuous distributions: prototype implementation available, in the spirit of [Abiteboul et al., 2011], hopefully can be integrated soon
- Extensional probabilistic query evaluation for safe queries: [Dalvi and Suciu, 2013] being considered, but requires architecture changes; ideally would be possible through compilation to a d-D circuit, but only known for particular cases [Monet, 2020]
- Plenty of optimizations still possible! (e.g., compiling isomorphic circuits only once for probabilistic query evaluation)

# Bibliography I


Serge Abiteboul, T-H. Hubert Chan, Evgeny Kharlamov, Werner Nutt, and Pierre Senellart. Capturing Continuous Data and Answering Aggregate Queries in Probabilistic XML. *ACM Transactions on Database Systems*, 36(4), 2011.

Antoine Amarilli. *Leveraging the Structure of Uncertain Data*. PhD thesis, Télécom ParisTech, 2016.

Antoine Amarilli, Pierre Bourhis, and Pierre Senellart. Provenance circuits for trees and treelike instances. In *Proc. ICALP*, pages 56–68, Kyoto, Japan, July 2015.

Yael Amsterdamer, Daniel Deutch, and Val Tannen. Provenance for aggregate queries. In *PODS*, 2011.

Bahareh Sadat Arab, Su Feng, Boris Glavic, Seokki Lee, Xing Niu, and Qitian Zeng. GProM - A swiss army knife for your provenance needs. *IEEE Data Eng. Bull.*, 41(1):51–62, 2018.

# Bibliography II


Michael Benedikt, Evgeny Kharlamov, Dan Olteanu, and Pierre Senellart. Probabilistic XML via Markov Chains. *Proceedings of the VLDB Endowment*, 3(1):770–781, September 2010. Presented at the VLDB 2010 conference, Singapore.

Omar Benjelloun, Anish Das Sarma, Alon Halevy, and Jennifer Widom. ULDBs: Databases with uncertainty and lineage. In *VLDB*, pages 953–964, 2006.

Pierre Bourhis, Daniel Deutch, and Yuval Moskovitch. Equivalence-invariant algebraic provenance for hyperplane update queries. In *SIGMOD*, pages 415–429. ACM, 2020. doi: 10.1145/3318464.3380578. URL `https://doi.org/10.1145/3318464.3380578`.

Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. Why and where: A characterization of data provenance. In *Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings.*, 2001.

# Bibliography III

Nilesh Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *Journal of the ACM (JACM)*, 59(6):1–87, 2013. URL https://homes.cs.washington.edu/~suciu/jacm-dichotomy.pdf.

Nilesh N. Dalvi and Dan Suciu. Efficient query evaluation on probabilistic databases. In Mario A. Nascimento, M. Tamer Özsu, Donald Kossmann, Renée J. Miller, José A. Blakeley, and K. Bernhard Schiefer, editors, *(e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004*, pages 864–875. Morgan Kaufmann, 2004. doi: 10.1016/B978-012088469-8.50076-0. URL http://www.vldb.org/conf/2004/RS22P1.PDF.

Nilesh N. Dalvi, Christopher Ré, and Dan Suciu. Queries and materialized views on probabilistic databases. *J. Comput. Syst. Sci.*, 77(3):473–490, 2011. doi: 10.1016/J.JCSS.2010.04.006. URL https://doi.org/10.1016/j.jcss.2010.04.006.

# Bibliography IV

Susan B. Davidson, Sarah Cohen Boulakia, Anat Eyal, Bertram Ludäscher, Timothy M. McPhillips, Shawn Bowers, Manish Kumar Anand, and Juliana Freire. Provenance in scientific workflow systems. *IEEE Data Eng. Bull.*, 30(4): 44–50, 2007. URL `http://sites.computer.org/debull/A07dec/susan.pdf`.

Floris Geerts and Antonella Poggi. On database query languages for k-relations. *J. Applied Logic*, 8(2), 2010.

Boris Glavic and Gustavo Alonso. Perm: Processing provenance and data on the same data model through query rewriting. In *ICDE*, pages 174–185, 2009.

Todd J. Green and Val Tannen. Models for incomplete and probabilistic information. *IEEE Data Eng. Bull.*, 29(1), 2006.

Todd J Green, Grigoris Karvounarakis, and Val Tannen. Provenance semirings. In *PODS*, 2007.

Todd J. Green, Grigoris Karvounarakis, Zachary G. Ives, and Val Tannen. Provenance in ORCHESTRA. *IEEE Data Eng. Bull.*, 33(3):9–16, 2010. URL `http://sites.computer.org/debull/A10sept/green.pdf`.

# Bibliography V

Martin Grohe and Peter Lindner. Infinite probabilistic databases. In Carsten Lutz and Jean Christoph Jung, editors, *23rd International Conference on Database Theory, ICDT 2020, March 30-April 2, 2020, Copenhagen, Denmark*, volume 155 of *LIPIcs*, pages 16:1–16:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi: 10.4230/LIPICS.ICDT.2020.16. URL https://doi.org/10.4230/LIPIcs.ICDT.2020.16.

Jiewen Huang, Lyublena Antova, Christoph Koch, and Dan Olteanu. MayBMS: a probabilistic database management system. In *SIGMOD*, pages 1071–1074, 2009.

Tomasz Imieliński and Jr. Lipski, Witold. Incomplete information in relational databases. *J. ACM*, 31(4), 1984.

Angelika Kimmig, Bart Demoen, Luc De Raedt, Vítor Santos Costa, and Ricardo Rocha. On the implementation of the probabilistic logic programming language problog. *Theory Pract. Log. Program.*, 11(2-3):235–262, 2011. doi: 10.1017/S1471068410000566. URL https://doi.org/10.1017/S1471068410000566.

# Bibliography VI

Mikaël Monet. Solving a special case of the intensional vs extensional conjecture in probabilistic databases. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 149–163. ACM, 2020. doi: 10.1145/3375395.3387642. URL https://doi.org/10.1145/3375395.3387642.

Yann Ramusat, Silviu Maniu, and Pierre Senellart. Provenance-Based Algorithms for Rich Queries over Graph Databases. In *Proc. EDBT*, pages 73–84, Nicosia, Cyprus, March 2021.

Pierre Senellart, Louis Jachiet, Silviu Maniu, and Yann Ramusat. ProvSQL: provenance and probability management in postgresql. In *VLDB*, 2018. Demonstration.

Hangdong Zhao, Shaleen Deep, Paraschos Koutris, Sudeepa Roy, and Val Tannen. Evaluating datalog over semirings: A grounding-based approach. *Proc. ACM Manag. Data*, 2(2):90, 2024. doi: 10.1145/3651591. URL https://doi.org/10.1145/3651591.