SQL Query evaluation on Incomplete Databases with correctness guarantees

Etienne Toussaint
supervised by Leonid Libkin

ENS Paris-Saclay, The University of Edinburgh

September 6, 2016

University of Edinburgh, School of Informatics

Figure: Informatics Forum

- Research only building
- 200 members
- 6 Institutes

University of Edinburgh, School of Informatics

- Research only building
- 200 members
- 6 Institutes

Figure: Informatics Forum

Laboratory for Foundations of Computer Science

- 80 members
- 6 groups

Database Group

Database Group

- 6 professors
- Weekly Seminar

Database Group

Database Group

- 6 professors
- Weekly Seminar

Leonid Libkin's Team

Members 4-7 members

Theme Missing Information

Project VADA: Value Added Data Systems

SQL

- Implemented in all major (free and commercial) RDBMSs
- First standardized in 1986 (ANSI) and 1987 (ISO); several revisions afterwards
- $25B/year business
- Most common tool used by data scientists

SQL

- Implemented in all major (free and commercial) RDBMSs
- First standardized in 1986 (ANSI) and 1987 (ISO); several revisions afterwards
- $25B/year business
- Most common tool used by data scientists

[C. Date, Database in Depth, 2005] "You can never trust the answers you get from a database with nulls".

Certain Answers

Certain Answers

Answers independent of the interpretation of missing information

Certain Answers

Certain Answers

Answers independent of the interpretation of missing information

Formally defined as :

$$cert(Q, D) = \bigcap Q(v(D))$$

over all possible valuation v.

Certain Answers

Certain Answers

Answers independent of the interpretation of missing information

Formally defined as :

$$cert(Q, D) = \bigcap Q(v(D))$$

over all possible valuation v.

PAYMENTS

pay_id	ord_id	pay_date
pay1	ord1	2015-06-14
pay2	NULL	2015-07-25

Certain Answers

Certain Answers

Answers independent of the interpretation of missing information

Formally defined as :

$$cert(Q, D) = \bigcap Q(v(D))$$

over all possible valuation v.

PAYMENTS

pay_id	ord_id	pay_date
pay1	ord1	2015-06-14
pay2	NULL	2015-07-25

Certain Answers with nulls on bags

$$x \in^n cert_\perp(Q, D) \iff \forall v, v(x) \in^n Q(v(D))$$

Answers in SQL

ORDERS

ord_id	ord_date
ord1	2015-06-12
ord2	2015-07-11
ord3	2015-07-20

PAYMENTS

pay_id	ord_id	pay_date
pay1	ord1	2015-06-14
pay2	NULL	2015-07-25

A typical query to write: "Unpaid orders"

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
  (SELECT * FROM Payments P
          WHERE P.ord_id = O.ord_id)
```

Answers in SQL

ORDERS

ord_id	ord_date
ord1	2015-06-12
ord2	2015-07-11
ord3	2015-07-20

PAYMENTS

pay_id	ord_id	pay_date
pay1	ord1	2015-06-14
pay2	NULL	2015-07-25

A typical query to write: "Unpaid orders"

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
  (SELECT * FROM Payments P
          WHERE P.ord_id = O.ord_id)
```

Answer : ⟦ord2,ord3⟧

SQL and correctness

Correctness Guarantee

$$Eval(Q, D) \subseteq cert_{\perp}(Q, D)$$

SQL and correctness

Correctness Guarantee

$$Eval(Q, D) \subseteq cert_{\perp}(Q, D)$$

SQL evaluation has NOT correctness guarantee

- $UCQ^{\neq}$ fragment has correctness guarantee
- Problems arise with negative sub-queries

SQL and correctness

Correctness Guarantee

$$Eval(Q, D) \subseteq cert_{\perp}(Q, D)$$

SQL evaluation has NOT correctness guarantee

- $UCQ^{\neq}$ fragment has correctness guarantee
- Problems arise with negative sub-queries

Can we compute certain answers ?

- Finding certain answers is co-NP hard
- SQL query answering is AC^0

SQL and correctness

Correctness Guarantee

$$Eval(Q, D) \subseteq cert_{\perp}(Q, D)$$

SQL evaluation has NOT correctness guarantee

- $UCQ^{\neq}$ fragment has correctness guarantee
- Problems arise with negative sub-queries

Can we compute certain answers ?

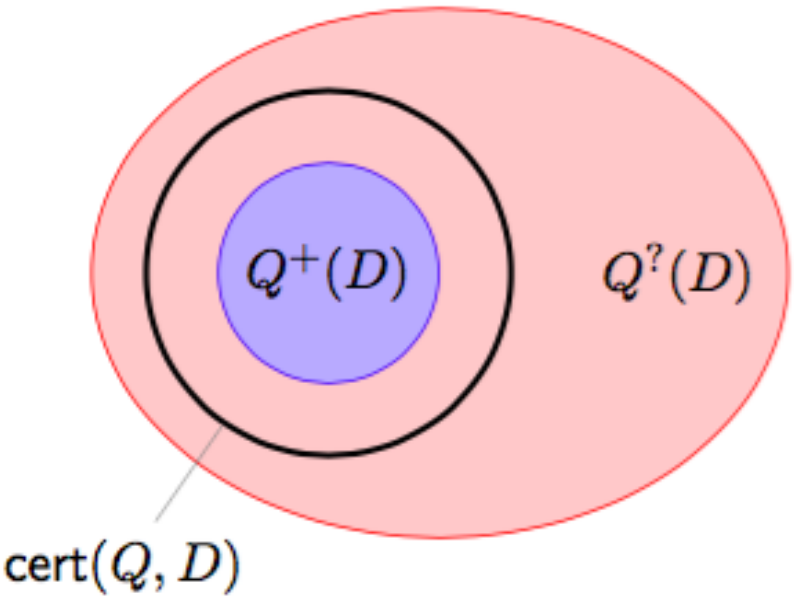
- Finding certain answers is co-NP hard
- SQL query answering is AC^0

Solution: Under approximation.

The Idea

Translate Q into $(Q^+, Q^?)$ where:

- Q^+ under-approximates certain answers
- $Q^?$ over-approximates certain answers




```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
  (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id)
```

Figure: Q

The translation

$$
\begin{aligned}
(H_1 \wedge H_2)^* &\rightarrow H_1^* \wedge H_2^* \\
(H_1 \vee H_2)^* &\rightarrow H_1^* \vee H_2^* \\
(r_i.a_i = c_i)^* &\rightarrow r_i.a_i = c_i \\
(r_i.a_i \neq c_i)^* &\rightarrow r_i.a_i \neq c_i \\
(r_i.a_i = r_j.a_j)^* &\rightarrow r_i.a_i = r_j.a_j \\
(r_i.a_i \neq r_j.a_j)^* &\rightarrow r_i.a_i \neq r_j.a_j \\
\mathit{exists}(Q)^* &\rightarrow \mathit{exists}(Q^+) \\
\mathit{notexists}(Q)^* &\rightarrow \mathit{notexists}(Q^?)
\end{aligned}
$$

Figure: Q^+

$$
\begin{aligned}
(H_1 \wedge H_2)^{**} &\rightarrow H_1^{**} \wedge H_2^{**} \\
(H_1 \vee H_2)^{**} &\rightarrow H_1^{**} \vee H_2^{**} \\
(r_i.a_i = c_i)^{**} &\rightarrow r_i.a_i = c_i \vee \mathit{null}(r_i.a_i) \\
(r_i.a_i \neq c_i)^{**} &\rightarrow r_i.a_i \neq c_i \vee \mathit{null}(r_i.a_i) \\
(r_i.a_i = r_j.a_j)^{**} &\rightarrow r_i.a_i = r_j.a_j \vee \mathit{null}(r_i.a_i) \vee \mathit{null}(r_j.a_j) \\
(r_i.a_i \neq r_j.a_j)^{**} &\rightarrow r_i.a_i \neq r_j.a_j \vee \mathit{null}(r_i.a_i) \vee \mathit{null}(r_j.a_j) \\
\mathit{exists}(Q)^{**} &\rightarrow \mathit{exists}(Q^?) \\
\mathit{notexists}(Q)^{**} &\rightarrow \mathit{notexists}(Q^+)
\end{aligned}
$$

Figure: $Q^?$

Theorem

Theorem

$$Q^+(D) \subseteq cert_\perp(Q, D)$$

Theorem

Theorem

$$Q^{+}(D) \subseteq cert_{\perp}(Q, D)$$

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id)
```

Figure: Q

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id
        OR P.ord_id IS NULL
        OR O.ord_id IS NULL)
```

Figure: Q^{+}

Theorem

Theorem

$$Q^+(D) \subseteq cert_\perp(Q, D)$$

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id)
```

Figure: Q

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id
        OR P.ord_id IS NULL
        OR O.ord_id IS NULL)
```

Figure: Q^+

Evaluation of Q^+ is slow !

- Quadratic vs Linear.
- Out of memory on big DB instances.

Planner

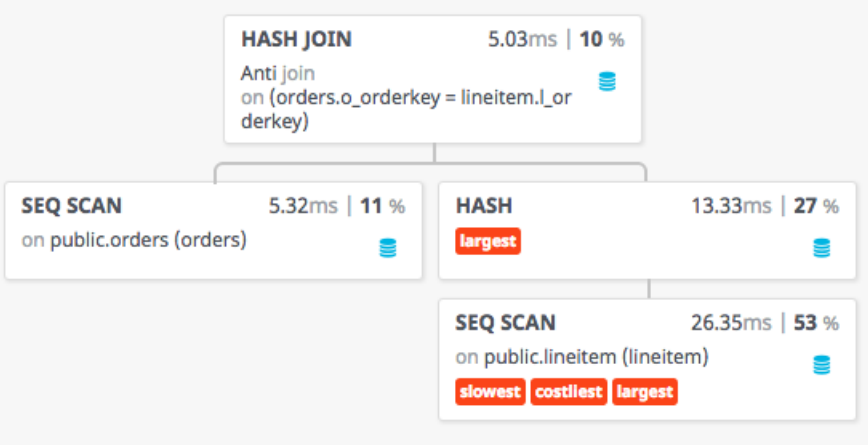


Figure: Q plan

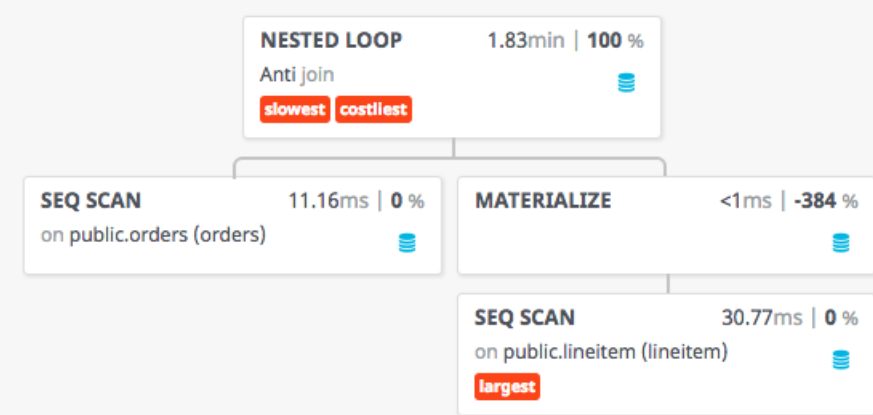


Figure: Q^+ plan

Remove redundant Null check

```
SELECT ord_id FROM Orders O
 WHERE NOT EXISTS
          (SELECT * FROM Payments P
           WHERE P.ord_id = O.ord_id
           OR P.ord_id IS NULL
           OR O.ord_id IS NULL
AND O.ord_id = 1
OR (P.ord_id = 2 AND O.ord_id = 1);
```

Split the query

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id
        OR P.ord_id IS NULL
        OR O.ord_id IS NULL)
```

Figure: Original Q

```
SELECT ord_id FROM Orders O
WHERE NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id = O.ord_id)
AND NOT EXISTS
 (SELECT * FROM Payments P
        WHERE P.ord_id IS NULL)
AND NOT EXISTS
 (SELECT * FROM Payments P
        WHERE O.ord_id IS NULL)
```

Figure: Split Q

Planner

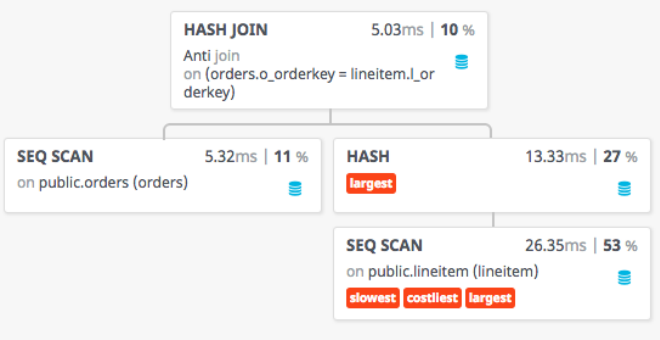


Figure: Q plan

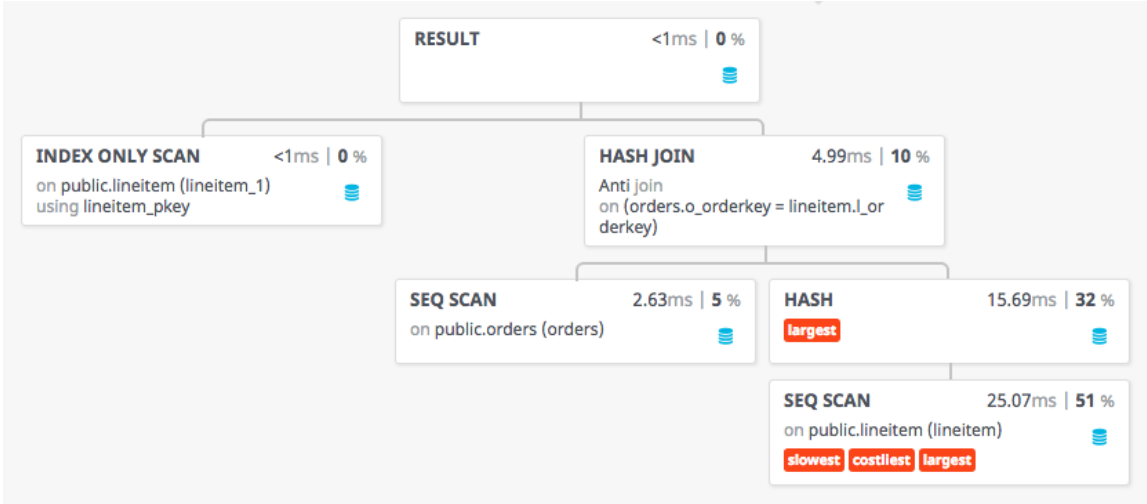


Figure: Q^+ optimize plan

Implementation

- PSQL extension (Scheme constraints)
- Language C
- Translation time insignificant (on most instances)
- Only a proof of concept
- Fair computation time

Implementation

- PSQL extension (Scheme constraints)
- Language C
- Translation time insignificant (on most instances)
- Only a proof of concept
- Fair computation time
- Marked Nulls version

Known issues

- SQL Semantic

Known issues

- SQL Semantic
- SQL Null

Known issues

- SQL Semantic
- SQL Null
  - Isomorphism

Known issues

- SQL Semantic
- SQL Null
  - Isomorphism

Figure: Orders

ord_id	ord_date
1	$NULL_a$
2	$NULL_b$

Known issues

- SQL Semantic
- SQL Null
  - Isomorphism

Figure: Orders

ord_id	ord_date
1	$NULL_a$
2	$NULL_b$

Figure: Orders $\times$ Orders

ord_id	ord_date	ord_id	ord_date
1	$NULL_a$	1	$NULL_a$
2	$NULL_b$	1	$NULL_a$
1	$NULL_a$	2	$NULL_b$
2	$NULL_b$	2	$NULL_b$

Known issues

- SQL Semantic
- SQL Null
  - Isomorphism
  - Null multiplicity

Figure: Orders

ord_id	ord_date
1	$NULL_a$
2	$NULL_b$

Certain answers definition

Certain Answers with nulls on bags

$$x \in^n cert_\perp(Q, D) \iff \forall v, v(x) \in^n Q(v(D))$$

Figure: Oders

ord_id	ord_date
1	2015
2	2016
$NULL_a$	2016

Certain answers definition

Certain Answers with nulls on bags

$$x \in^n cert_\perp(Q, D) \iff \forall v, v(x) \in^n Q(v(D))$$

Figure: Oders

ord_id	ord_date
1	2015
2	2016
$NULL_a$	2016

Wanted : $[\![1,2,NULL_a]\!]$

Certain answers definition

Certain Answers with nulls on bags

$$x \in^n cert_\perp(Q, D) \iff \forall v, v(x) \in^n Q(v(D))$$

Figure: Oders

ord_id	ord_date
1	2015
2	2016
$NULL_a$	2016

Wanted : $[\![1,2,NULL_a]\!]$
Definition : $[\![1,2,NULL_a,NULL_a]\!]$

Conclusion

SQL Evaluation with Correctness Guarantee

- Theory:
  - SQL semantics
  - Certain answers with nulls definition
  - Better approximation ?

Conclusion

SQL Evaluation with Correctness Guarantee

- Theory:
  - SQL semantics
  - Certain answers with nulls definition
  - Better approximation ?
- Practice:
  - Generalized Optimisation
  - Implemented directly in DBMS
  - Accelerate query computation ?

Conclusion

SQL Evaluation with Correctness Guarantee

- Theory:
  - SQL semantics
  - Certain answers with nulls definition
  - Better approximation ?
- Practice:
  - Generalized Optimisation
  - Implemented directly in DBMS
  - Accelerate query computation ?

SQL is only the beginning !