# LinCQA: Faster Consistent Query Answering with Linear Time Guarantees

Zhiwei Fan†, Paraschos Koutris†, Xiating Ouyang†, Jef Wijsen‡

†University of Wisconsin – Madison, ‡University of Mons

## Consistent Query Answering (CQA)

While data cleaning is widely adopted to repair the inconsistent/dirty data, finding the "right repair" remains a challenge. Alternatively, the idea of CQA is to compute the answers that are guaranteed to be returned in **all** repairs.

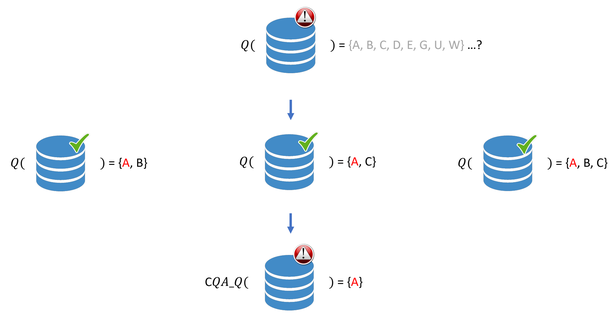



Fig. 1: An example of CQA. The middle layer shows the different sets of answers returned for each possible repair, and the bottom layer explains that the answer "A" is returned in **all** repairs.

$$\mathsf{CQA}_Q(\mathbf{db}^{!}) = \bigcap_{\mathbf{db}^{✓} \text{ is a repair of } \mathbf{db}^{!}} Q(\mathbf{db}^{✓}).$$

We study inconsistent databases that could violate the primary key constraint: every primary key could correspond to multiple distinct tuples in the database.

**Problem**: CQA($Q$), where $Q$ is a SPJ query

**Input**: a database $\mathbf{db}$ that violates the primary key constraint

**Output**: the answers **guaranteed** to be returned by $Q$ on all repairs of $\mathbf{db}$

Course

| course_id | faculty_id |
|---|---|
| **CS 703** | 2 |
| **CS 703** | 5 |
| **MATH 770** | 3 |
| **MATH 770** | 7 |
| **CS 787** | 8 |
| **CS 787** | 9 |

CS_Faculty

| faculty_id | name | area |
|---|---|---|
| **2** | Adam | DB |
| **2** | Alice | OS |
| **5** | Bob | PL |
| **9** | Cathy | ML |
| **9** | Carrol | ML |

Executing the following blue SQL query on the database returns the inconsistent answers **CS 703** and **CS 787**. The rewritten query by adding the red segments would find the consistent answers (i.e., **CS 703**) that are returned in every repair of the database.

```
SELECT DISTINCT Course.c_id
FROM Course, Faculty
WHERE Course.f_id = Faculty.f_id
AND (all f_id's for the same c_id
     appear in Faculty)
```

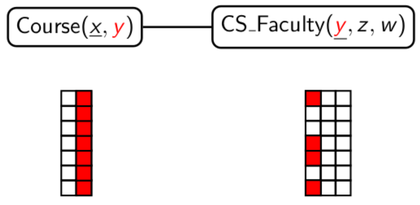
Fig. 2: An example of first-order (FO) rewriting.

Not all queries admit a first-order rewriting! And the classification on the rewritability remains an open problem.

## Acyclic Queries in Linear Time

Evaluating an acyclic query is a well-studied problem by using hash joins on the join tree. For example, the **Boolean** blue SQL query is acyclic:

$$q() \text{:-} \mathsf{Course}(\underline{x}, y), \mathsf{Faculty}(\underline{y}, z, \text{'DB'}).$$




Yannakakis [VLDB'81] Our result

consistent answer

The ~~answer~~ of every **Boolean** acyclic query can be evaluated in $O(|\text{db}|)$.

∧

with a pair-pruning join tree (PPJT)

Queries with projection can be reduced to **Boolean** queries

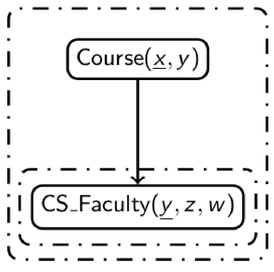
## Pair-pruning Join Tree (PPJT)

We consider acyclic self-join-free SPJ queries (each table name occurs once).

**Definition**: A join tree rooted at some atom is a PPJT if

the root of every subtree is unattacked in the subtree.

For example, $q$ has a PPJT:




- Queries on star/snowflake schema (e.g. TPC-H, TPC-DS)
- Two tables
- acyclic queries in $\mathcal{C}_{\text{forest}}$ [Fuxman and Miller, SIGMOD'05]

**Proposition**: If a query $q$ have a pair-pruning join tree (PPJT), then CQA($q$) has a first-order rewriting that runs in linear time.

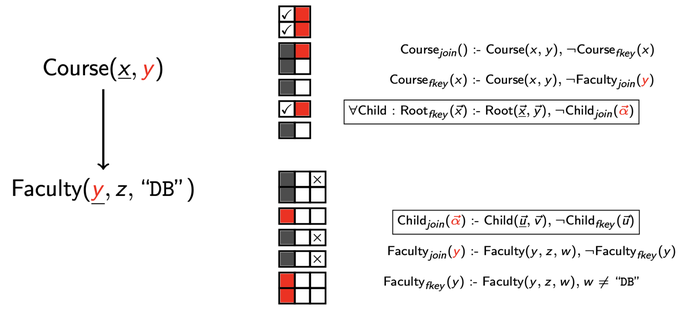
## LinCQA and the Rewriting

LinCQA is a query rewriter that takes as input a SQL query, output its first-order rewriting.

```
python3 lincqa.py -i <input.sql> -o <output.sql>
```

Using PPJT, the consistent answers can be computed in a bottom-up fashion.

$$q() \text{:-} \mathsf{Course}(\underline{x}, y), \mathsf{Faculty}(\underline{y}, z, \text{"DB"}).$$




For queries with projection:

```
SELECT DISTINCT A1, A2 FROM T WHERE A3 = 42
```

we use PPJT for each potential answer $(a, b)\ldots$:

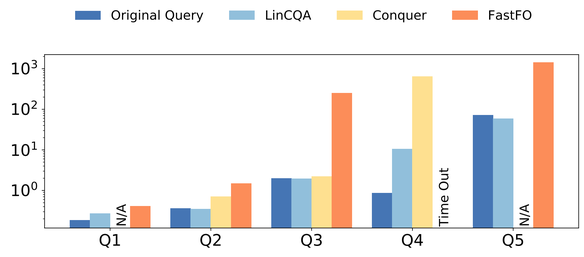
```
SELECT DISTINCT 1 FROM T WHERE A3 = 42 AND A1 = a AND A2 = b
```

if **yes**, then $(a, b)$ is a consistent answer.

## Experiments

We used a 400GB StackOverflow dataset (among others) on SQL Server.

| Table | # of rows | inconsistencyRatio | blockSize | # of Attributes |
|---|---|---|---|---|
| Users | 14M | 0% | 1 | 14 |
| Posts | 53M | 0% | 1 | 20 |
| PostHistory | 141M | 0.001% | 4 | 9 |
| Badges | 40M | 0.58% | 941 | 4 |
| Votes | 213M | 30.9% | 1441 | 6 |




| | | | | | |
|---|---|---|---|---|---|
| $Q(\mathbf{db}^{!})$ | 27578 | 145 | 38320 | 3925 | 1250 |
| $\mathsf{CQA}_Q(\mathbf{db}^{!})$ | 27578 | 145 | 38320 | 3925 | 1245 |

| Acyclic $q$ | LinCQA [SIGMOD'23] | Yannakakis [VLDB'81] |
|---|---|---|
| Boolean $q$ | $O(N)$ | $O(N)$ |
| Projection $q$ | $O(N \cdot \lvert\text{OUT}_{\text{inconsistent}}\rvert)$ | $O(N \cdot \lvert\text{OUT}\rvert)$ |
| full $q$ (SELECT *) | $O(N + \lvert\text{OUT}_{\text{consistent}}\rvert)$ | $O(N + \lvert\text{OUT}\rvert)$ |

*Consistent answers can be computed with no asymptotic overhead*