DataFrame QA: Privacy-Preserving Data Analysis via Schema Metadata

Coby McClellan, Data Analytics '28 | Professor: John MacCormick

Replication & Extension of Ye, Du & Wang (2024) • arXiv:2401.15463

93% pass@1 | **28% fewer tokens** | **Zero data exposed** | **45 pairs generated**

How DataFrame QA works

The LLM receives only schema metadata — never the actual data values.

Input is defined as a tuple:

$$(S, H, C, Q)$$

S = supplementary info (assumptions, constraints)
H = $\{h_1, h_2, ..., h_n\}$ dataframe column headers
C = $\{c_1, c_2, ..., c_n\}$ column data types
Q = natural language question

The LLM generates a Pandas query P′:

$$P' = f(S, H, C, Q)$$

Executed in a sandbox to get the answer:

$$A' = execute(P', df)$$

Compared against ground truth:

$$A = execute(P, df)$$

Success measured by how closely A′ ≈ A using the pass@1 metric.

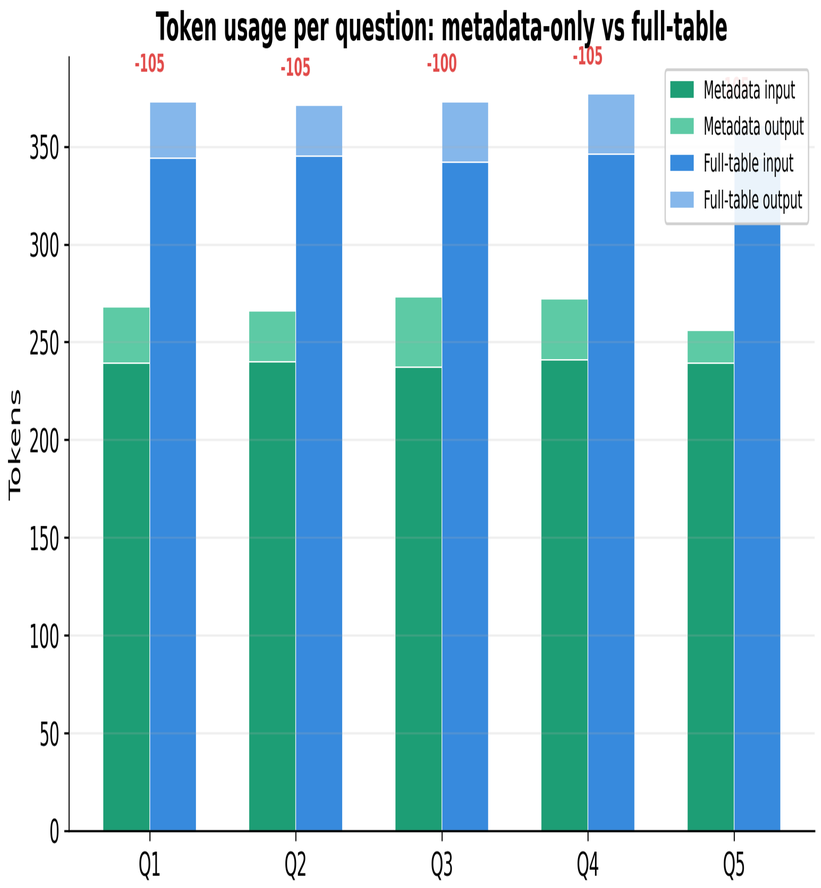
The problem with table QA today

Traditional table QA embeds the entire table into the LLM prompt. This creates three issues:

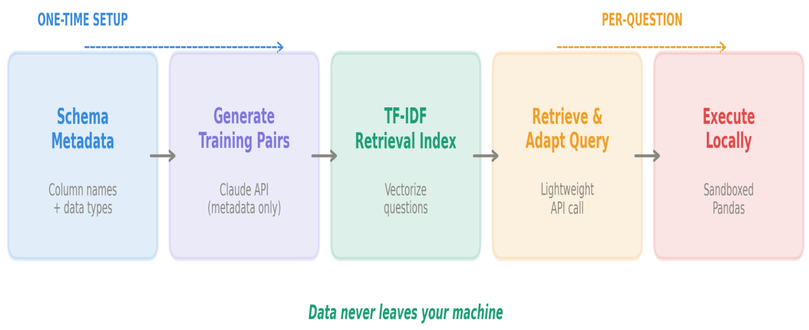
1. Token costs explode with table size
A 1,000-row table can use 15,000+ tokens per query

2. Context windows overflow
Large datasets exceed 4K–8K limits entirely

3. Sensitive data is exposed to the API
Medical, financial, and personal data leaves your machine



Retrieval-based DataFrame QA pipeline



Token savings: 28% on small data

With only 7 rows, metadata-only already saves 105 tokens per question. The key: this gap grows linearly with table size.

At 1,000 rows, full-table needs 15,000+ tokens. Metadata-only stays at ~240.

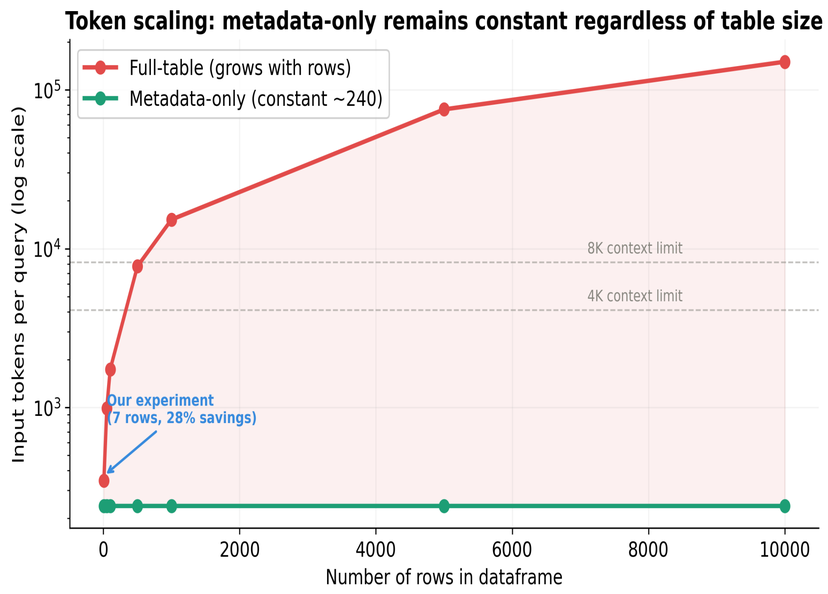
~240	~345+
tokens/query Metadata-only	tokens/query Full-table

Scaling: constant cost at any size

The core advantage becomes clear at scale. Metadata-only input tokens stay flat at ~240 regardless of table size, while full-table grows without bound.

A 10,000-row table would need ~150,000 input tokens with the traditional approach. Metadata-only: still 240 tokens.

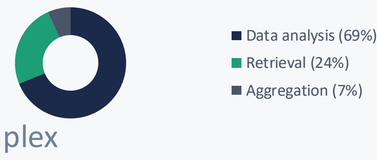
Rows	Full-table	Metadata
7	345	240
1,000	15,240	240
10,000	150,240	240



Extension: training data from metadata

We generated 45 question/query training pairs across 3 domains using only column names and types — no real data needed.

Domains: medical, automotive, sports
Roles: provider, client , data owner
Total cost: 7,880 tokens (~$0.02)



69% of generated pairs involved complex multi-step analysis (groupby, correlation, pivot tables) — not just simple lookups.

Retrieval-based prediction system

We built a retrieval system that reuses generated training pairs as templates:

1. **New question comes in**
2. **TF-IDF finds most similar training pair**
3. **Retrieved query adapted via lightweight API call (~200 tokens)**
4. **Adapted code executes locally**

Key takeaways

Schema metadata alone is enough to:

- Answer dataframe questions (93% pass@1
- without seeing a single data value)

- Generate training data across domains
- (45 pairs, 3 domains, 7,880 tokens)

- Build retrieval systems that reduce API
- costs while preserving full data privacy

References

[1] Ye, J., Du, M., & Wang, G. (2024).
"DataFrame QA: A Universal LLM Framework on DataFrame Question Answering Without Data Exposure." arXiv:2401.15463

[2] Zhong, V., Xiong, C., & Socher, R. (2017).
"Seq2SQL: Generating Structured Queries from Natural Language Using Reinforcement Learning." arXiv:1709.00103 (WikiSQL)

[3] Anthropic. (2025).
Claude API Documentation. platform.claude.com/docs

[4] Chen, M. et al. (2021).
"Evaluating Large Language Models Trained on Code." arXiv:2107.03374 (pass@1 metric)

SCAN ME

GitHub: github.com/cobymcclellan/llm-dataframe-qa | Paper: arxiv.org/abs/2401.15463 | API: Claude Sonnet 4 (Anthropic) | Framework: Python, Pandas, scikit-learn