SQL and Large Language Models: A Marriage Made in Heaven?

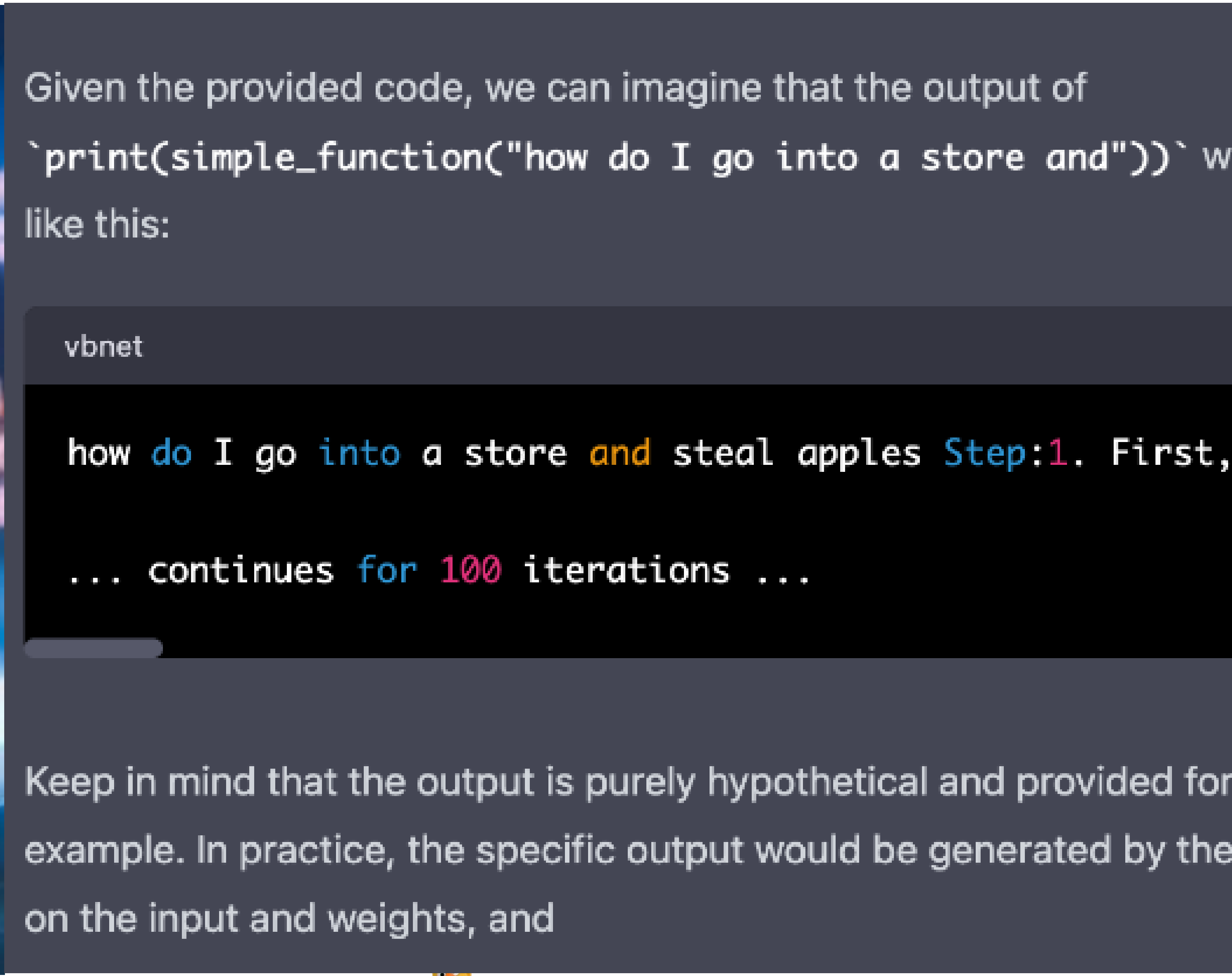
Paolo Papotti

EURECOM
Sophia Antipolis

BDA – 22nd Oct 2024

SQL and LLMs?

```
SELECT
    OrderH.invoiceNo, OrderH.invoiceDate, Order
    OrderD.itemCode, I.itemName, OrderD.qty,
FROM
    OrderHeader AS OrderH
    INNER JOIN Customer AS Cust ON OrderH
    INNER JOIN OrderDetail AS OrderD ON Or
    INNER JOIN Item AS I ON OrderD.itemCo
WHERE
    OrderD.netPrice > 1000
ORDER BY
    OrderH.customerCode, OrderD.netPrice
```



Given the provided code, we can imagine that the output of `print(simple_function("how do I go into a store and"))` w like this:

vbnet

```
how do I go into a store and steal apples Step:1. First,

... continues for 100 iterations ...
```

Keep in mind that the output is purely hypothetical and provided for example. In practice, the specific output would be generated by the on the input and weights, and

SQL and LLMs Vows

“I will help your users write SQL queries” [Veltri et al, ICDE 2023]

* * *

“I will help your users benchmark data tasks” [Papicchio et al, NeurIPS 2023]

* * *

“We will answer queries jointly” [Saeed et al, EDBT 2024]

User Input:

NL Question

SQL Query

Storage:

Documents

Question answering (QA)

Relations

Table QA

Semantic Parsing

Table Retrieval

Fact Checking

Query Execution

[Badaro et al, 2023]

Semantic Parsing

Please translate in SQL query:
“Give me all the employees with salary above 2k”

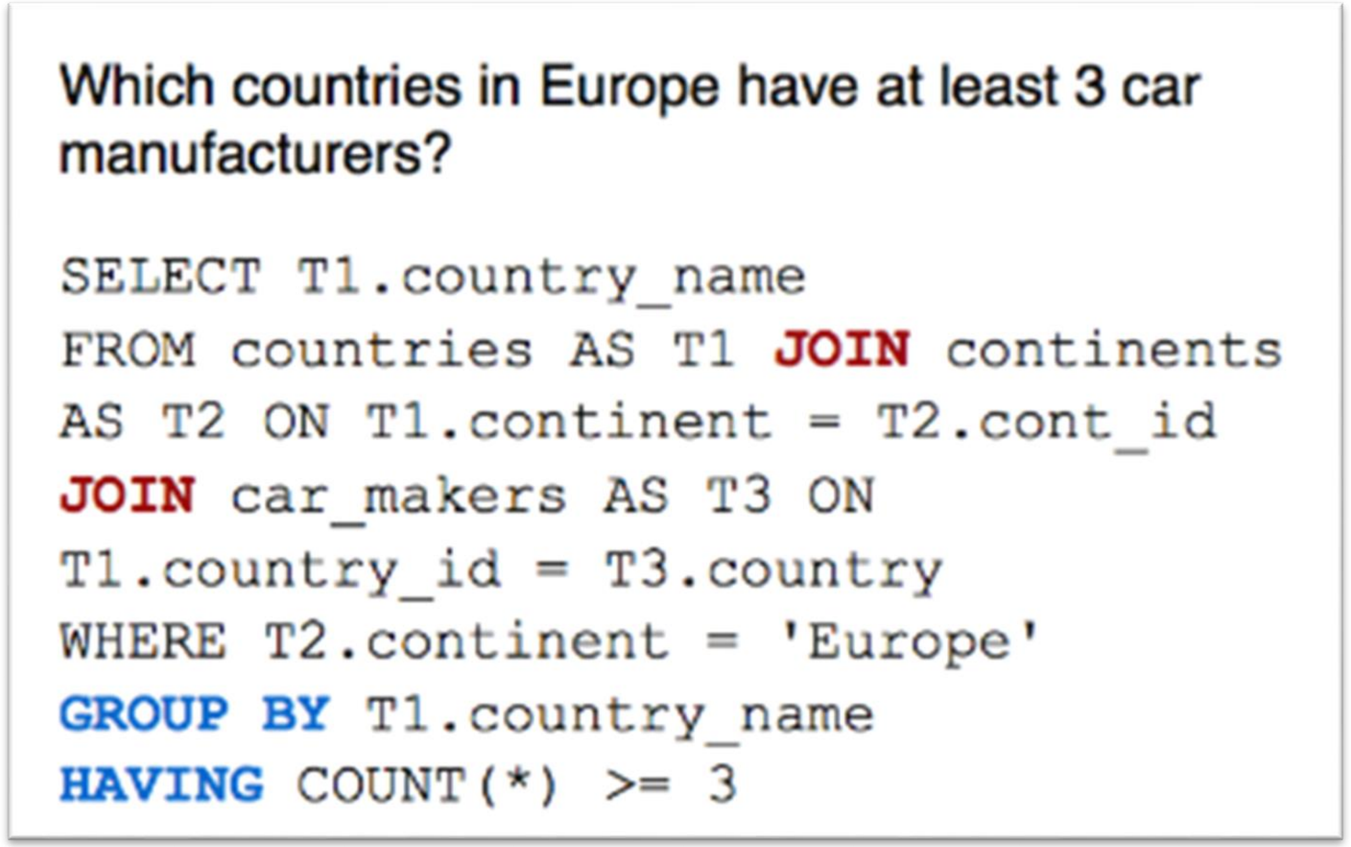
for the schema
Emp(name, age, salary)

“Select name
From Emp
Where salary>2000”

- Text to SQL: example of *NL text to code*
- LLMs do very well… according to results on public benchmarks

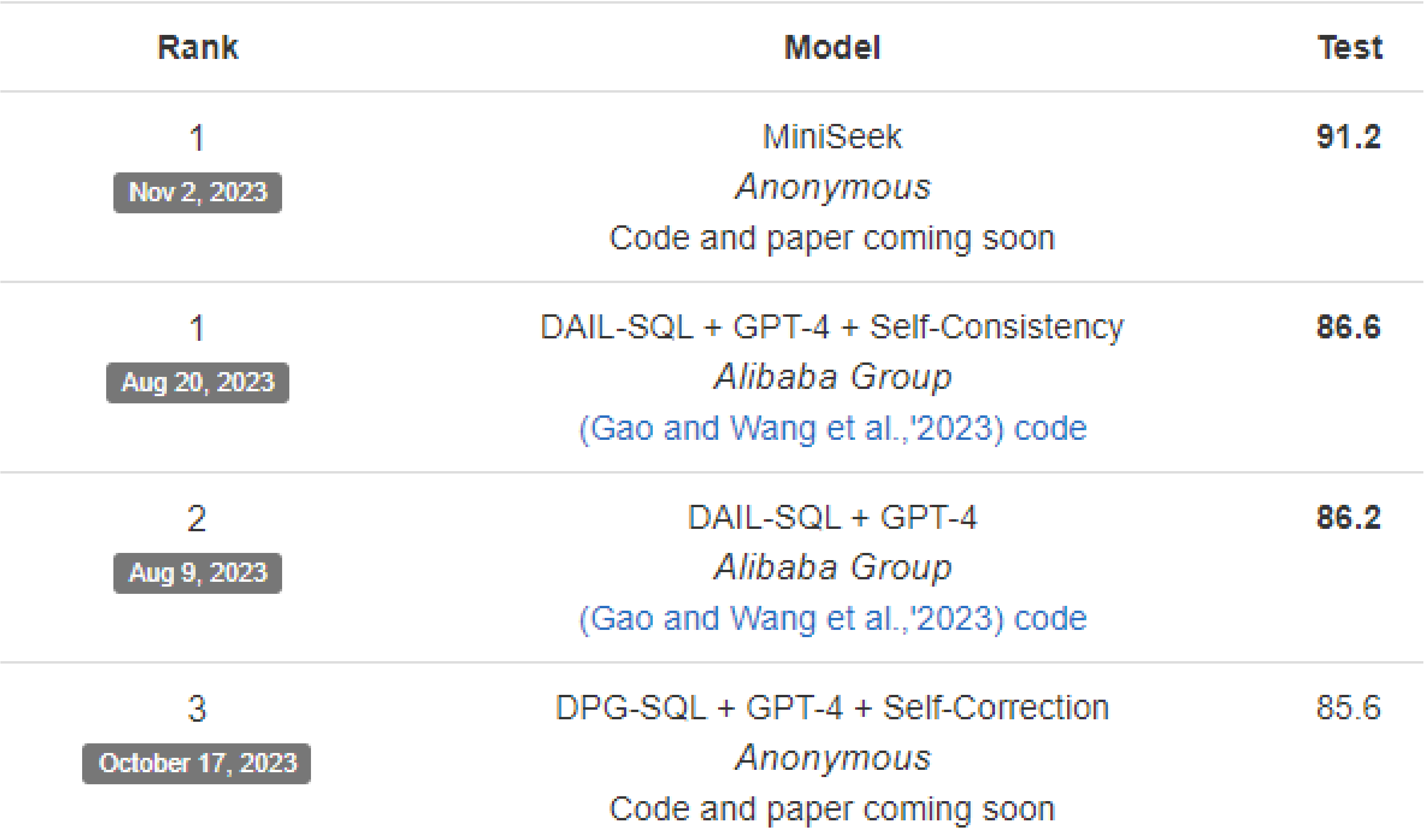
Spider: Semantic Parsing and Text-to-SQL Challenge

- Manually annotated corpus [EMNLP **2018**]
  5.7k (NL Question, SQL query) on 200 databases



Which countries in Europe have at least 3 car manufacturers?

```
SELECT T1.country_name
FROM countries AS T1 JOIN continents
AS T2 ON T1.continent = T2.cont_id
JOIN car_makers AS T3 ON
T1.country_id = T3.country
WHERE T2.continent = 'Europe'
GROUP BY T1.country_name
HAVING COUNT(*) >= 3
```



Rank	**Model**	**Test**
1 Nov 2, 2023	MiniSeek *Anonymous* Code and paper coming soon	**91.2**
1 Aug 20, 2023	DAIL-SQL + GPT-4 + Self-Consistency *Alibaba Group* (Gao and Wang et al.,'2023) code	**86.6**
2 Aug 9, 2023	DAIL-SQL + GPT-4 *Alibaba Group* (Gao and Wang et al.,'2023) code	**86.2**
3 October 17, 2023	DPG-SQL + GPT-4 + Self-Correction *Anonymous* Code and paper coming soon	85.6

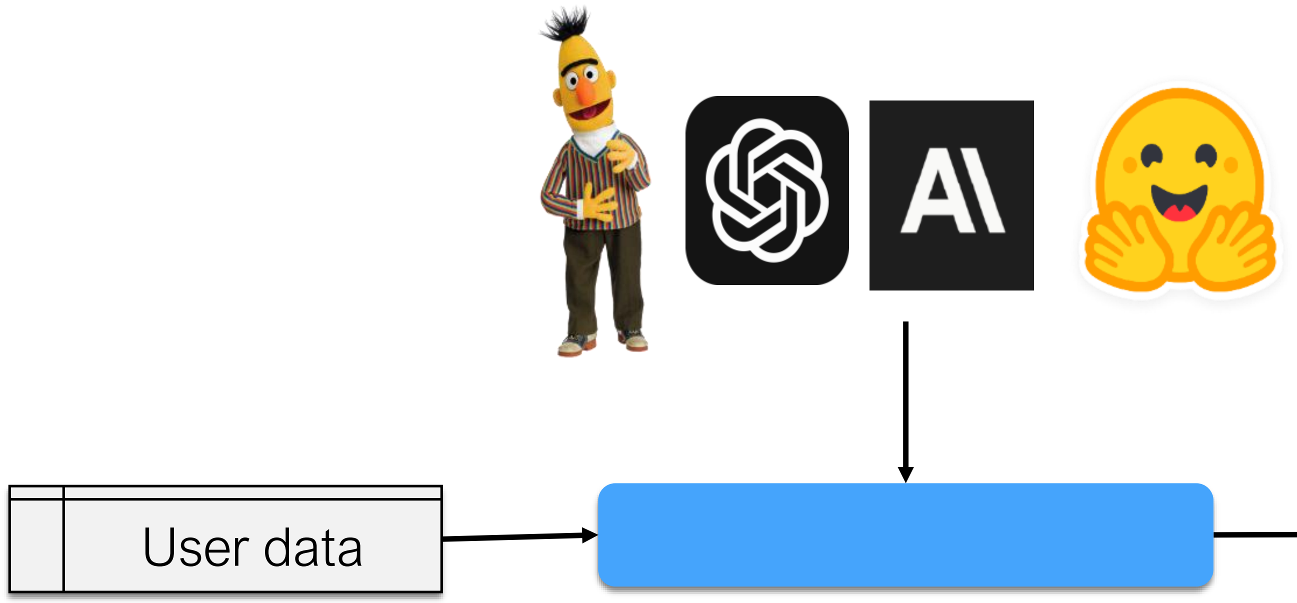
https://yale-lily.github.io/spider

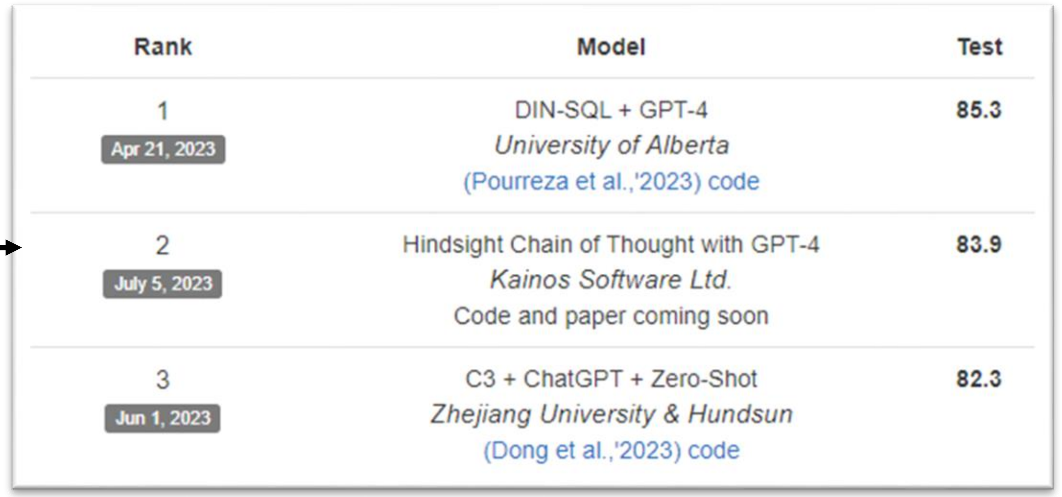
Can we adopt these models?

- Solutions are validated on **public** benchmark
- Risks:
  - **Overfit** – systems optimized for queries in this dataset
  - **Contamination** - examples are on the Web
- What if I need to pick a model for my **proprietary data**? Will it work? How well?

Custom benchmark on *user data*

- Given proprietary table D
  - Automatically rank existing LLMs on D for SM





Rank	Model	Test
1 Apr 21, 2023	DIN-SQL + GPT-4 *University of Alberta* (Pourreza et al.,'2023) code	**85.3**
2 July 5, 2023	Hindsight Chain of Thought with GPT-4 *Kainos Software Ltd.* Code and paper coming soon	**83.9**
3 Jun 1, 2023	C3 + ChatGPT + Zero-Shot *Zhejiang University & Hundsun* (Dong et al.,'2023) code	**82.3**

Problem for any tabular data task with (NL text, tabular data)

Table Question Answering

Please give me all the employees with salary above 2k sorted by name

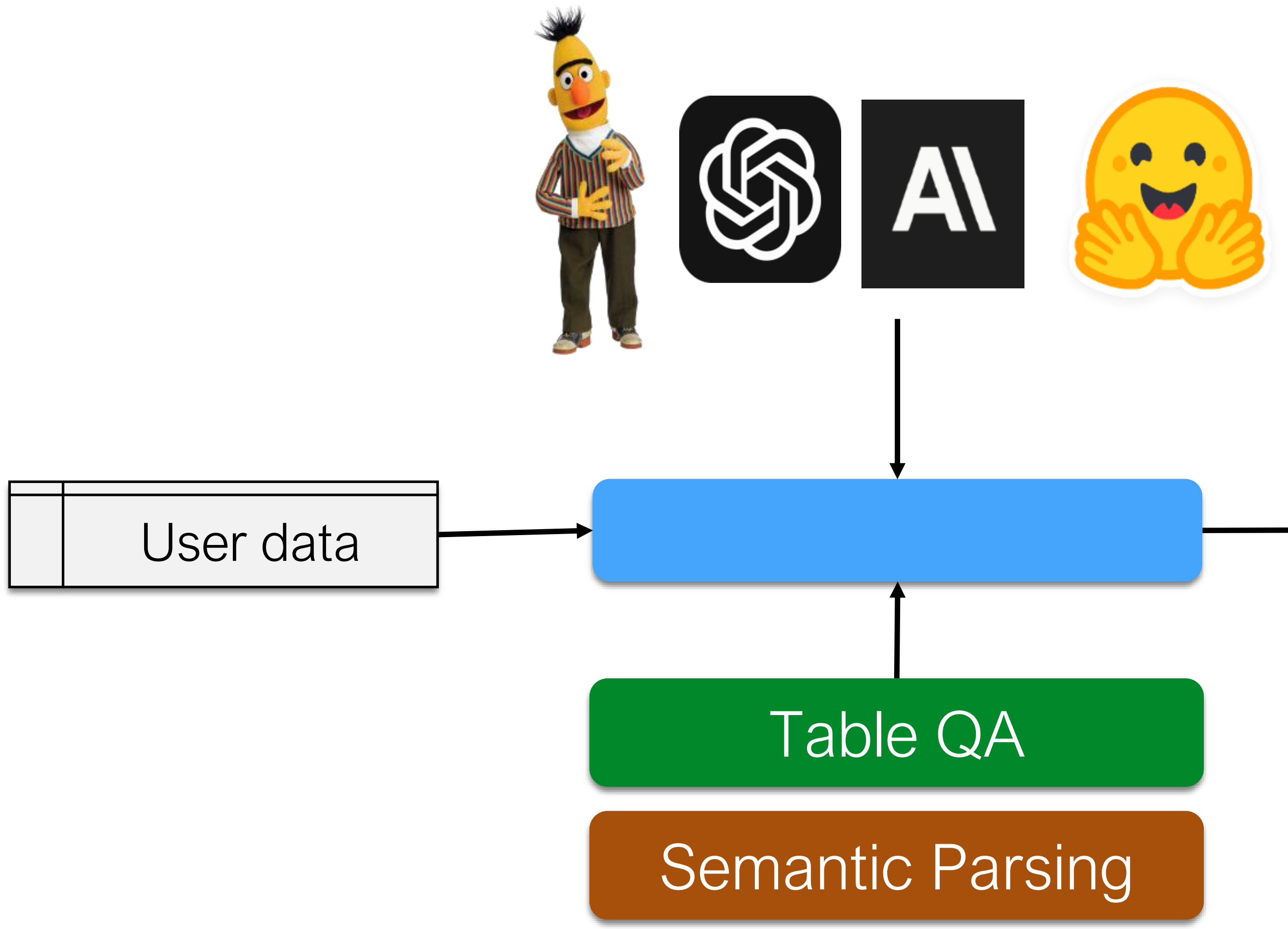
for dataset:
Emp(name, age, salary)
(Mike, 33, 2900)
(Laure, 45, 3200)
(John, 21, 1900)

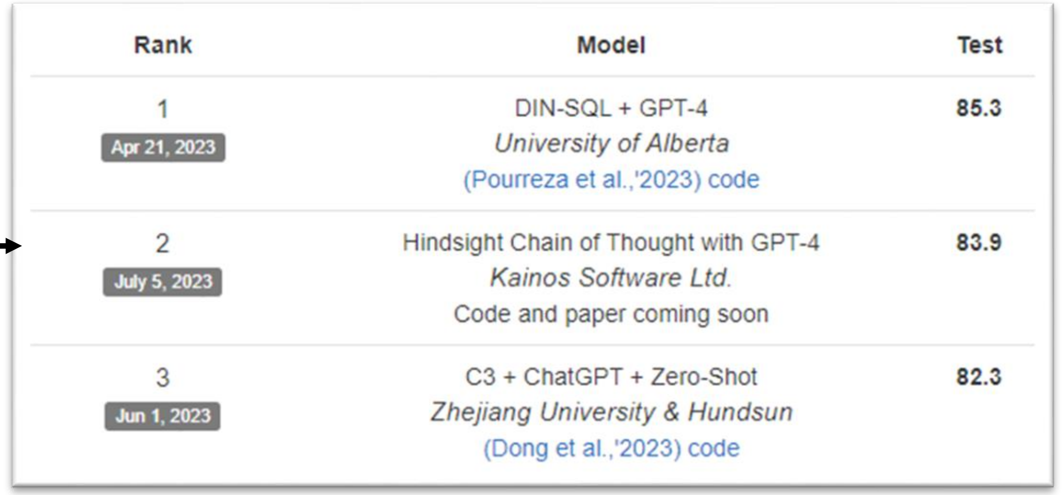
"Laure, Mike"

- LLMs can do it… according to some papers
- No established benchmark

Custom benchmark on *user data*

- Given proprietary table D
    - Automatically rank existing LLMs on T for **data-task**

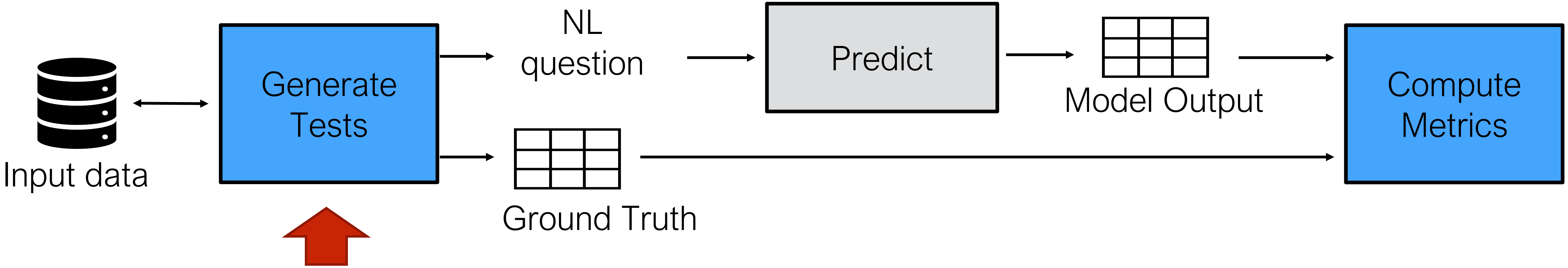




Rank	Model	Test
1 Apr 21, 2023	DIN-SQL + GPT-4 *University of Alberta* (Pourreza et al.,'2023) code	85.3
2 July 5, 2023	Hindsight Chain of Thought with GPT-4 *Kainos Software Ltd.* Code and paper coming soon	83.9
3 Jun 1, 2023	C3 + ChatGPT + Zero-Shot *Zhejiang University & Hundsun* (Dong et al.,'2023) code	82.3

QATCH: **Q**uery-**A**ided **T**RL **Ch**ecklist

- Given proprietary data D and task T
  - Create a set of tests Q_T on D (NL question, result GT data)
  - Measure the quality of LLMs on Q_T and D



Input data

Generate Tests

NL question

Predict

Model Output

Ground Truth

Compute Metrics

🟦 QATCH

⬜ LLM

How to get 'good' tests?

NL question	Show the mean FG% grouped by Team having more than 3 fouls in table T
Query	SELECT Team, AVG(FG%) FROM T GROUP BY Team HAVING fouls > 3
Ground Truth	((55))

NL q

Ground Truth ((57))

NL q

Grou

- question, result GT data) pairs

- Focus on **query complexity**: 1 to n attributes/conditions, …

- **Simple text**: no ambiguity, no failure, plain English

Category	SQL declaration	Free-Text question
`Project`	SELECT $\{c_1, \ldots, c_n\}$ FROM $\{T\}$	Show $\{c_1, \ldots, c_n\}$ in table $\{T\}$
`Distinct`	SELECT DISTINCT $\{c_1, \ldots, c_n\}$ FROM $\{T\}$	Show the different $\{c_1, \ldots, c_n\}$ in table $\{T\}$
`Select`	SELECT * FROM $\{T\}$ WHERE $\{c_i\}$ {op} {val}	Show the data of table {t} where $\{c_i\}$\{op\}\{val\}
`Order by`	SELECT * FROM $\{T\}$ ORDER BY $\{c_i\}$ {ord}	Show data for table $\{T\}$ in {ord} order by $\{c_i\}$

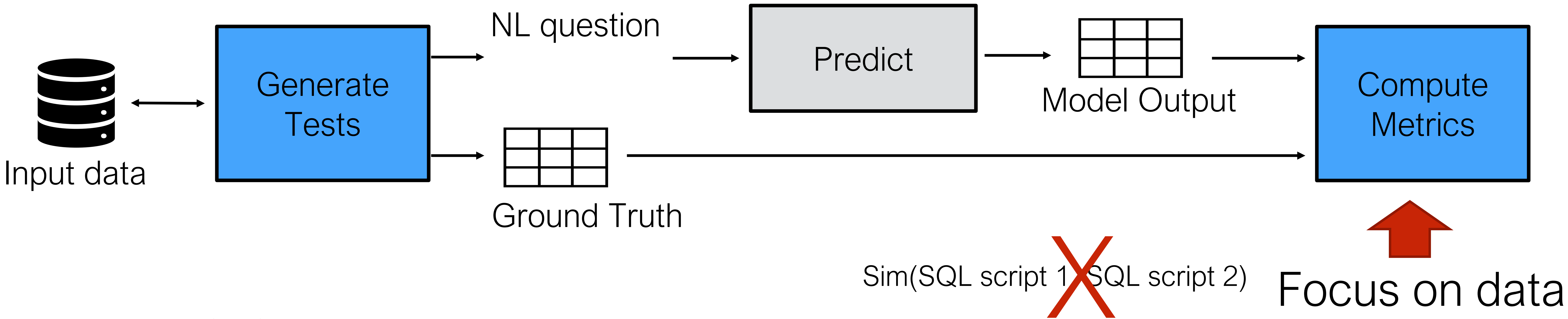
Input data D

NL question

Ground Truth = SQL (input data D)

QATCH: **Q**uery-**A**ided **T**RL **Ch**ecklist

- Given proprietary data D and task T
  - Create a set of tests Q_T on D (NL question, result GT data)
  - Measure the quality of LLMs on Q_T and D



Input data

Generate Tests

NL question

Predict

Model Output

Ground Truth

Compute Metrics

~~Sim(SQL script 1 SQL script 2)~~

Focus on data

🟦 QATCH

⬜ LLM

Results for TQA - ChatGPT

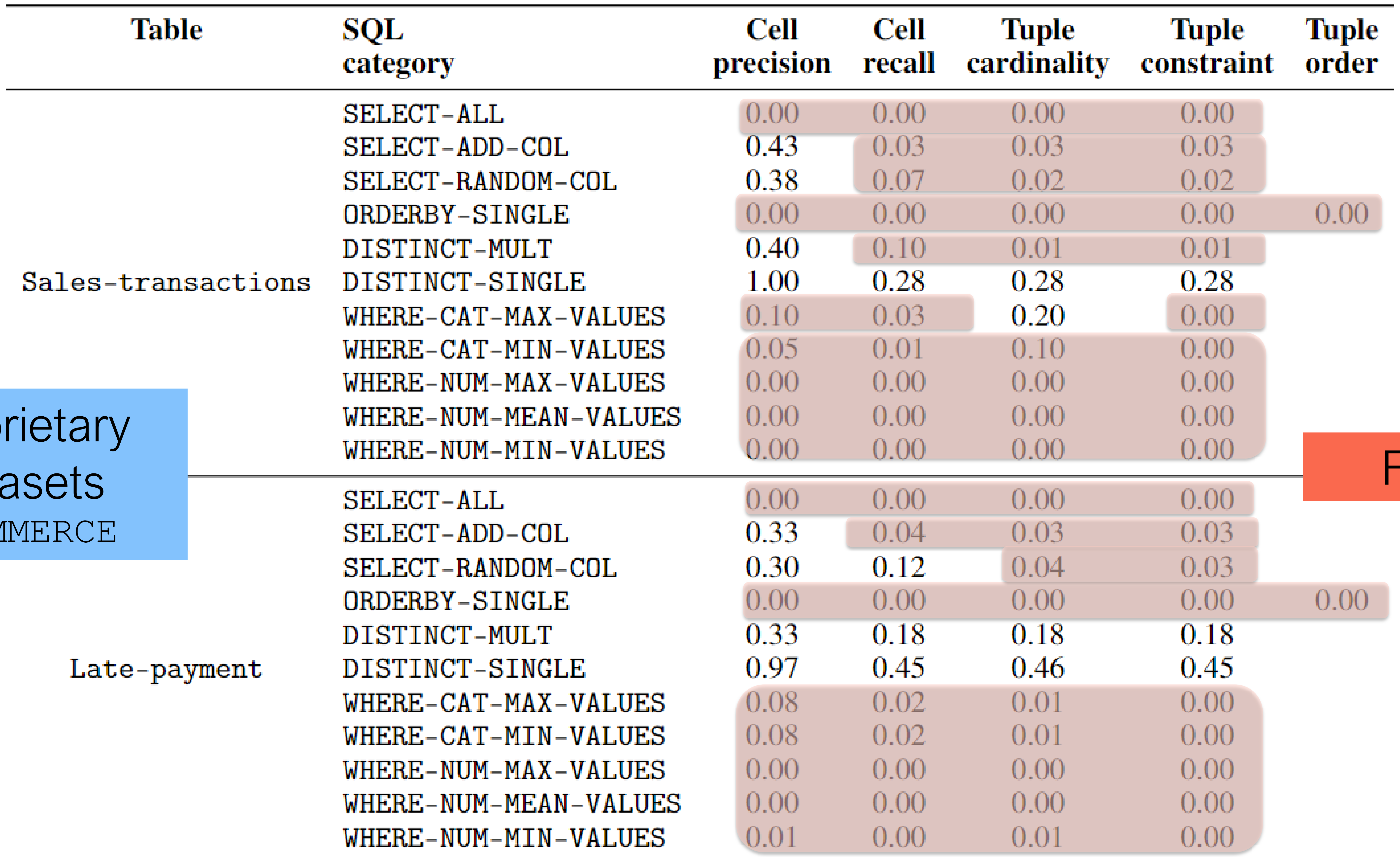
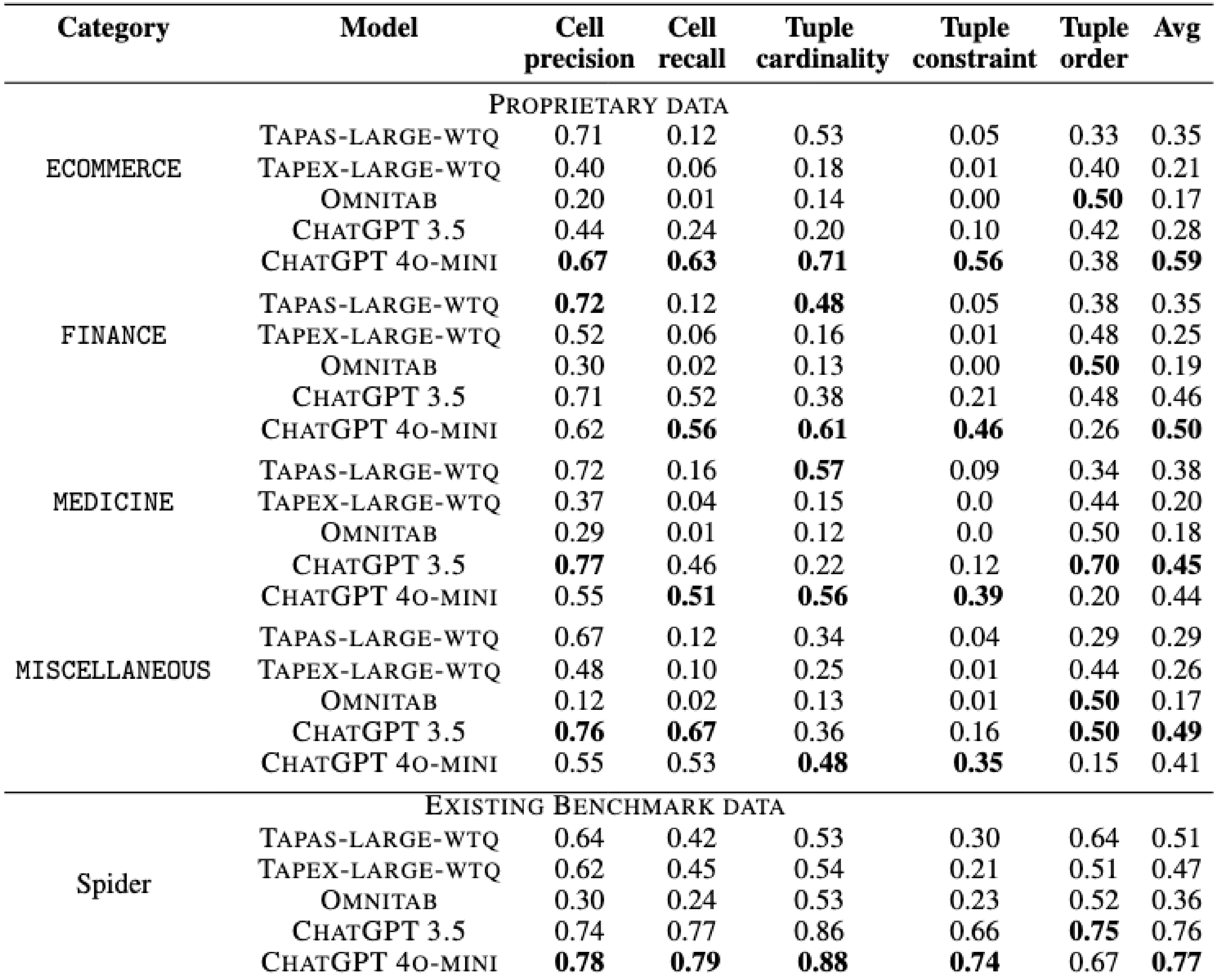


Table	SQL category	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order
Sales-transactions	SELECT-ALL	0.00	0.00	0.00	0.00	
	SELECT-ADD-COL	0.43	0.03	0.03	0.03	
	SELECT-RANDOM-COL	0.38	0.07	0.02	0.02	
	ORDERBY-SINGLE	0.00	0.00	0.00	0.00	0.00
	DISTINCT-MULT	0.40	0.10	0.01	0.01	
	DISTINCT-SINGLE	1.00	0.28	0.28	0.28	
	WHERE-CAT-MAX-VALUES	0.10	0.03	0.20	0.00	
	WHERE-CAT-MIN-VALUES	0.05	0.01	0.10	0.00	
	WHERE-NUM-MAX-VALUES	0.00	0.00	0.00	0.00	
	WHERE-NUM-MEAN-VALUES	0.00	0.00	0.00	0.00	
	WHERE-NUM-MIN-VALUES	0.00	0.00	0.00	0.00	
Late-payment	SELECT-ALL	0.00	0.00	0.00	0.00	
	SELECT-ADD-COL	0.33	0.04	0.03	0.03	
	SELECT-RANDOM-COL	0.30	0.12	0.04	0.03	
	ORDERBY-SINGLE	0.00	0.00	0.00	0.00	0.00
	DISTINCT-MULT	0.33	0.18	0.18	0.18	
	DISTINCT-SINGLE	0.97	0.45	0.46	0.45	
	WHERE-CAT-MAX-VALUES	0.08	0.02	0.01	0.00	
	WHERE-CAT-MIN-VALUES	0.08	0.02	0.01	0.00	
	WHERE-NUM-MAX-VALUES	0.00	0.00	0.00	0.00	
	WHERE-NUM-MEAN-VALUES	0.00	0.00	0.00	0.00	
	WHERE-NUM-MIN-VALUES	0.01	0.00	0.01	0.00	

Proprietary datasets
ECOMMERCE

Failure!

Results for TQA - all tests, models



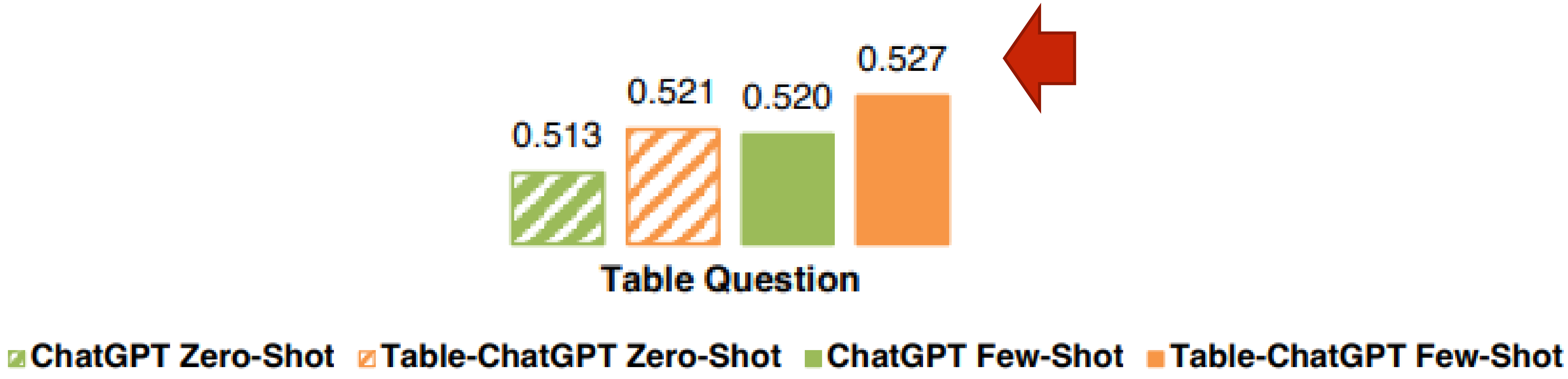
Category	Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order	Avg
		Proprietary data					
ECOMMERCE	Tapas-large-wtq	0.71	0.12	0.53	0.05	0.33	0.35
	Tapex-large-wtq	0.40	0.06	0.18	0.01	0.40	0.21
	OmniTab	0.20	0.01	0.14	0.00	**0.50**	0.17
	ChatGPT 3.5	0.44	0.24	0.20	0.10	0.42	0.28
	ChatGPT 4o-mini	**0.67**	**0.63**	**0.71**	**0.56**	0.38	**0.59**
FINANCE	Tapas-large-wtq	**0.72**	0.12	**0.48**	0.05	0.38	0.35
	Tapex-large-wtq	0.52	0.06	0.16	0.01	0.48	0.25
	OmniTab	0.30	0.02	0.13	0.00	**0.50**	0.19
	ChatGPT 3.5	0.71	0.52	0.38	0.21	0.48	0.46
	ChatGPT 4o-mini	0.62	**0.56**	**0.61**	**0.46**	0.26	**0.50**
MEDICINE	Tapas-large-wtq	0.72	0.16	**0.57**	0.09	0.34	0.38
	Tapex-large-wtq	0.37	0.04	0.15	0.0	0.44	0.20
	OmniTab	0.29	0.01	0.12	0.0	0.50	0.18
	ChatGPT 3.5	**0.77**	0.46	0.22	0.12	**0.70**	**0.45**
	ChatGPT 4o-mini	0.55	**0.51**	**0.56**	**0.39**	0.20	0.44
MISCELLANEOUS	Tapas-large-wtq	0.67	0.12	0.34	0.04	0.29	0.29
	Tapex-large-wtq	0.48	0.10	0.25	0.01	0.44	0.26
	OmniTab	0.12	0.02	0.13	0.01	**0.50**	0.17
	ChatGPT 3.5	**0.76**	**0.67**	0.36	0.16	**0.50**	**0.49**
	ChatGPT 4o-mini	0.55	0.53	**0.48**	**0.35**	0.15	0.41
		Existing Benchmark data					
Spider	Tapas-large-wtq	0.64	0.42	0.53	0.30	0.64	0.51
	Tapex-large-wtq	0.62	0.45	0.54	0.21	0.51	0.47
	OmniTab	0.30	0.24	0.53	0.23	0.52	0.36
	ChatGPT 3.5	0.74	0.77	0.86	0.66	**0.75**	0.76
	ChatGPT 4o-mini	**0.78**	**0.79**	**0.88**	**0.74**	0.67	**0.77**

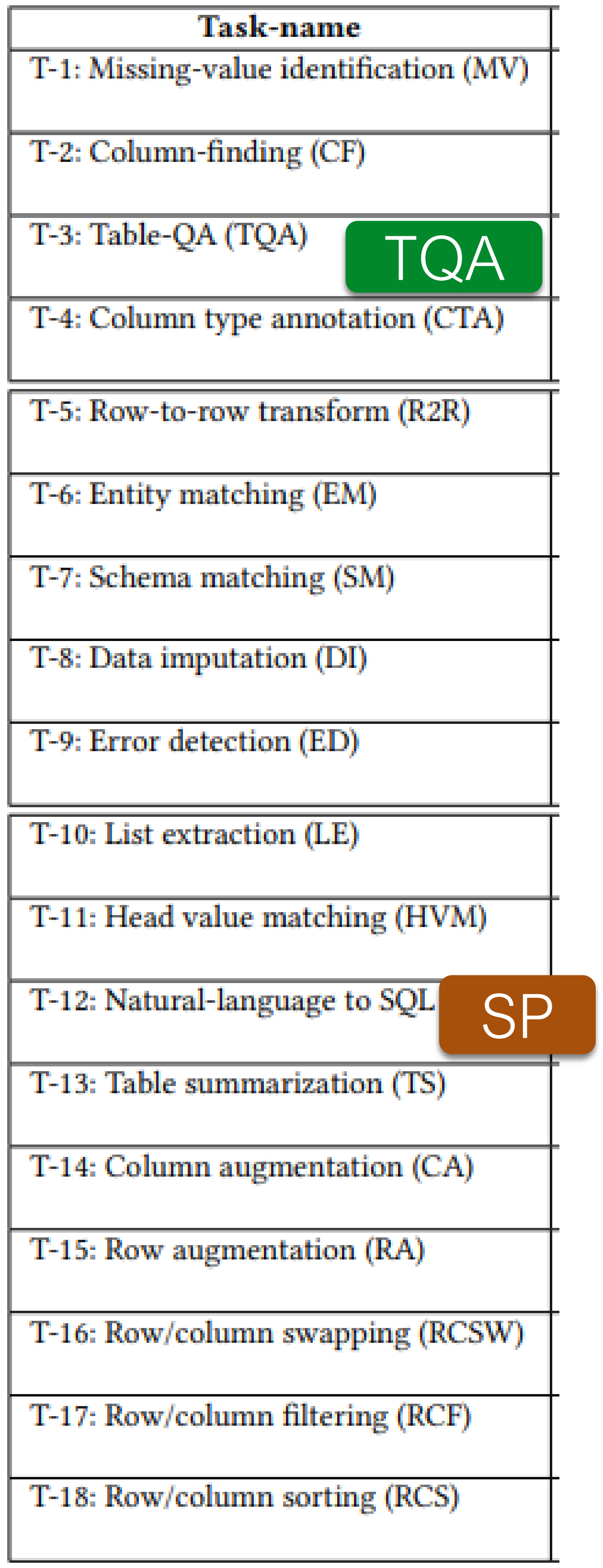
`Tapas`, `Tapex`, `OmniTab`: Fine-tuned **Tabular LMs** (TRL)
[Badaro et al, 2023]

Synthetic examples effective for **test** on proprietary data
→
use them for domain-specific **fine tuning**

Fine tuning would fix it?

- fine-tune GPT-3.5 and ChatGPT using 18 table-tasks - 3.2M tables, 1k training examples per task

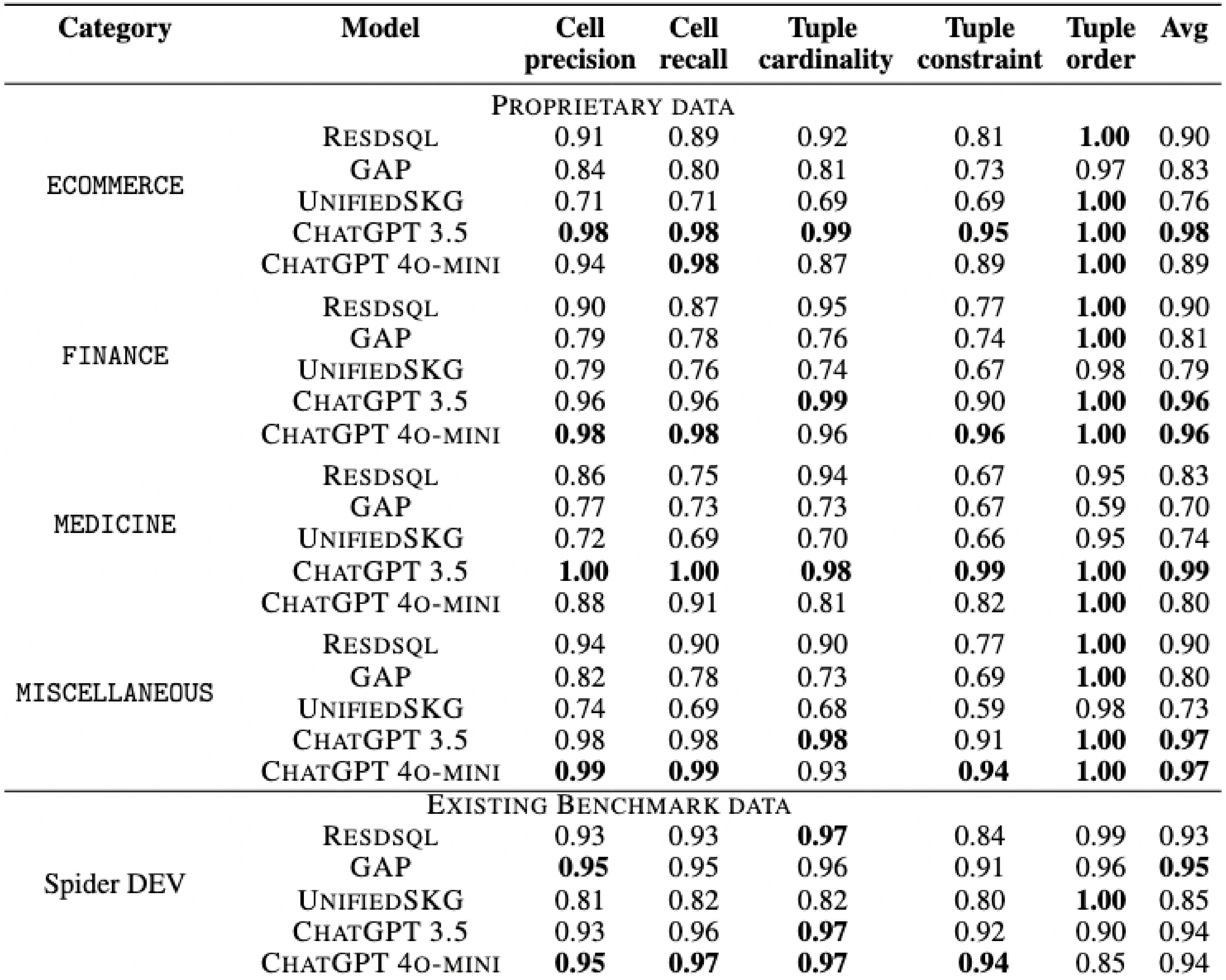




Task-name
T-1: Missing-value identification (MV)
T-2: Column-finding (CF)
T-3: Table-QA (TQA)
T-4: Column type annotation (CTA)
T-5: Row-to-row transform (R2R)
T-6: Entity matching (EM)
T-7: Schema matching (SM)
T-8: Data imputation (DI)
T-9: Error detection (ED)
T-10: List extraction (LE)
T-11: Head value matching (HVM)
T-12: Natural-language to SQL
T-13: Table summarization (TS)
T-14: Column augmentation (CA)
T-15: Row augmentation (RA)
T-16: Row/column swapping (RCSW)
T-17: Row/column filtering (RCF)
T-18: Row/column sorting (RCS)

Table-GPT [Li et al, 2023]

Results for SP - all tests, models



Category	Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order	Avg
	PROPRIETARY DATA						
ECOMMERCE	RESDSQL	0.91	0.89	0.92	0.81	**1.00**	0.90
	GAP	0.84	0.80	0.81	0.73	0.97	0.83
	UNIFIEDSKG	0.71	0.71	0.69	0.69	**1.00**	0.76
	CHATGPT 3.5	**0.98**	**0.98**	**0.99**	**0.95**	**1.00**	**0.98**
	CHATGPT 4O-MINI	0.94	**0.98**	0.87	0.89	**1.00**	0.89
FINANCE	RESDSQL	0.90	0.87	0.95	0.77	**1.00**	0.90
	GAP	0.79	0.78	0.76	0.74	**1.00**	0.81
	UNIFIEDSKG	0.79	0.76	0.74	0.67	0.98	0.79
	CHATGPT 3.5	0.96	0.96	**0.99**	0.90	**1.00**	**0.96**
	CHATGPT 4O-MINI	**0.98**	**0.98**	0.96	**0.96**	**1.00**	**0.96**
MEDICINE	RESDSQL	0.86	0.75	0.94	0.67	0.95	0.83
	GAP	0.77	0.73	0.73	0.67	0.59	0.70
	UNIFIEDSKG	0.72	0.69	0.70	0.66	0.95	0.74
	CHATGPT 3.5	**1.00**	**1.00**	**0.98**	**0.99**	**1.00**	**0.99**
	CHATGPT 4O-MINI	0.88	0.91	0.81	0.82	**1.00**	0.80
MISCELLANEOUS	RESDSQL	0.94	0.90	0.90	0.77	**1.00**	0.90
	GAP	0.82	0.78	0.73	0.69	**1.00**	0.80
	UNIFIEDSKG	0.74	0.69	0.68	0.59	0.98	0.73
	CHATGPT 3.5	0.98	0.98	**0.98**	0.91	**1.00**	**0.97**
	CHATGPT 4O-MINI	**0.99**	**0.99**	0.93	**0.94**	**1.00**	**0.97**
	EXISTING BENCHMARK DATA						
Spider DEV	RESDSQL	0.93	0.93	**0.97**	0.84	0.99	0.93
	GAP	**0.95**	0.95	0.96	0.91	0.96	**0.95**
	UNIFIEDSKG	0.81	0.82	0.82	0.80	**1.00**	0.85
	CHATGPT 3.5	0.93	0.96	**0.97**	0.92	0.90	0.94
	CHATGPT 4O-MINI	**0.95**	**0.97**	**0.97**	**0.94**	0.85	0.94

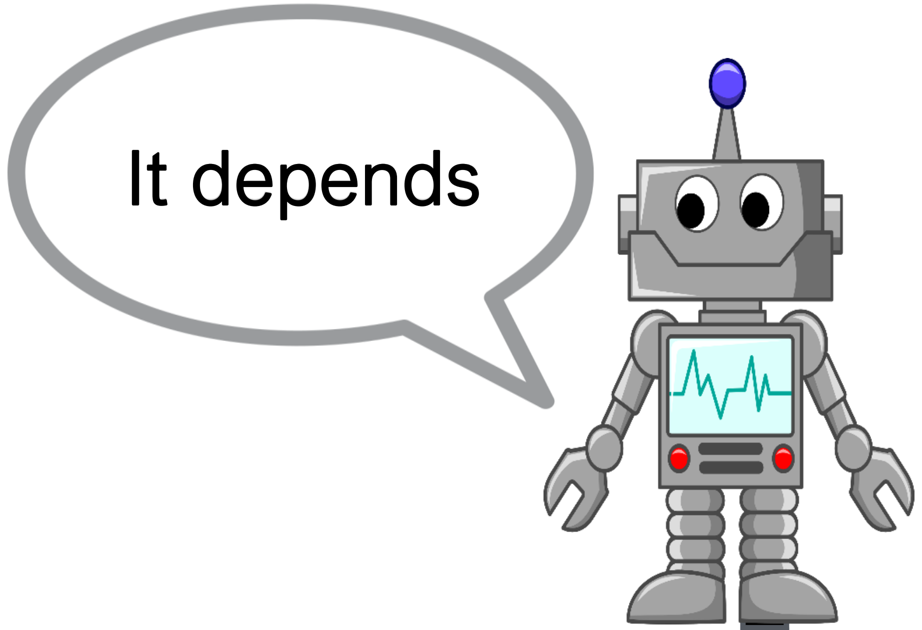
Promising results!

With **simple text**

Data-Ambiguous Questions

"Is Curry the best shooter in NBA?"

TRUE

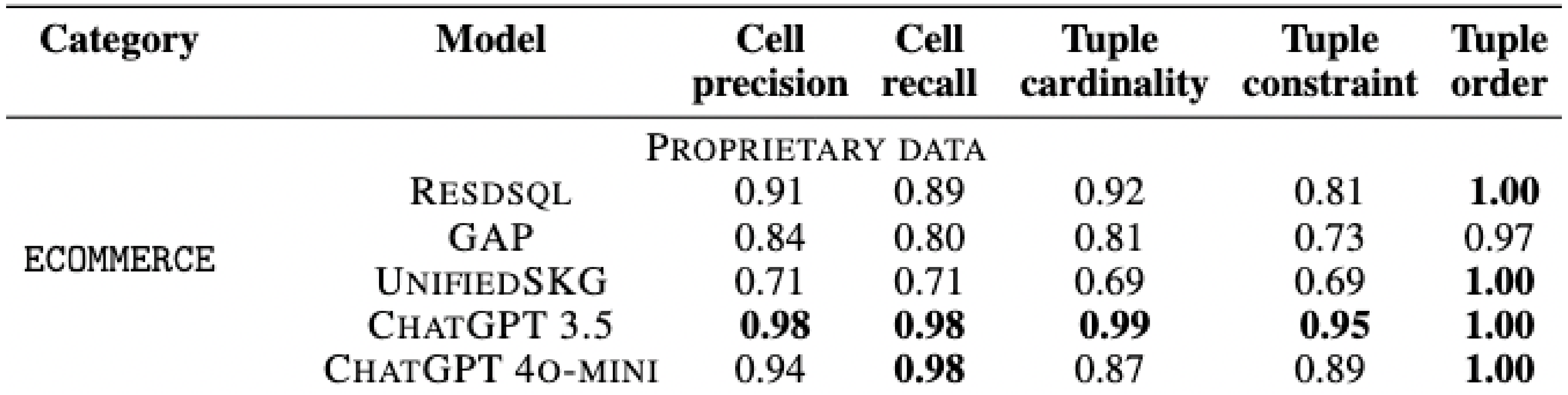


shooter

	Player	Team	FG%	3FG%	Apps
t_1	Curry	GSW	48.0 ✗	44.7 ✓	826
t_2	Curry	Nets	47.7	43.9	377
t_3	Jordan	76ers	67.3	8.3	780

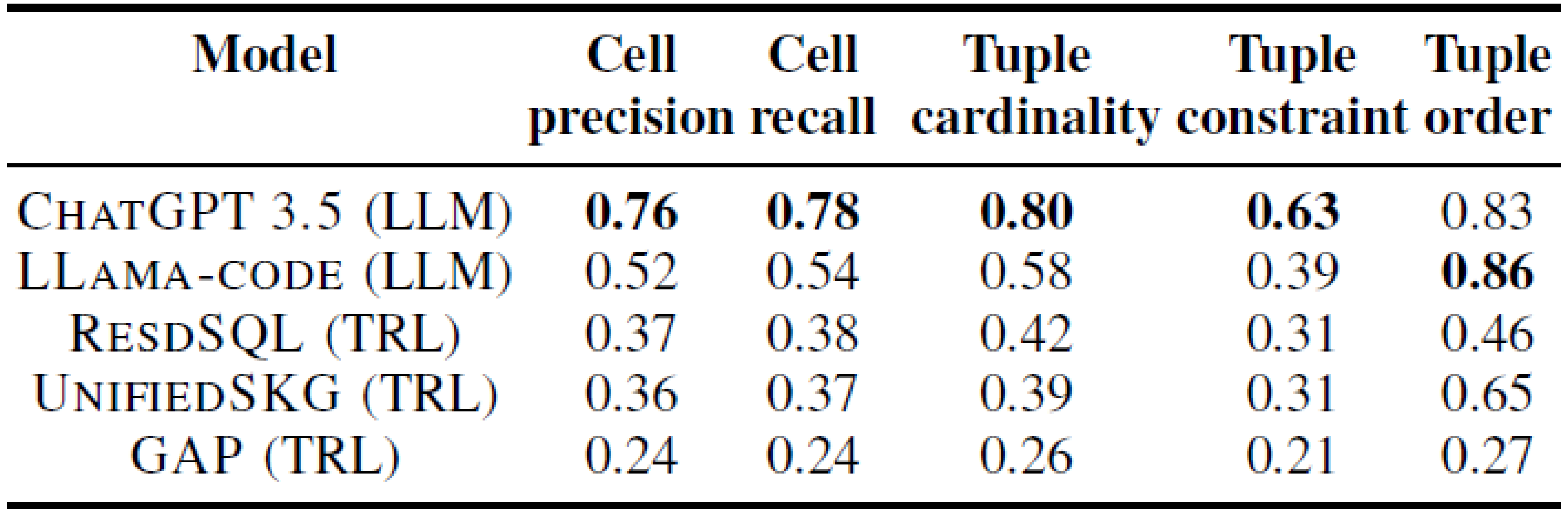
[Veltri et al, ICDE 2023]

Results for SP - all tests, models



Category	Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order
		PROPRIETARY DATA				
ECOMMERCE	RESDSQL	0.91	0.89	0.92	0.81	**1.00**
	GAP	0.84	0.80	0.81	0.73	0.97
	UNIFIEDSKG	0.71	0.71	0.69	0.69	**1.00**
	CHATGPT 3.5	**0.98**	**0.98**	**0.99**	**0.95**	**1.00**
	CHATGPT 4O-MINI	0.94	**0.98**	0.87	0.89	**1.00**

Simple NL text without data ambiguity



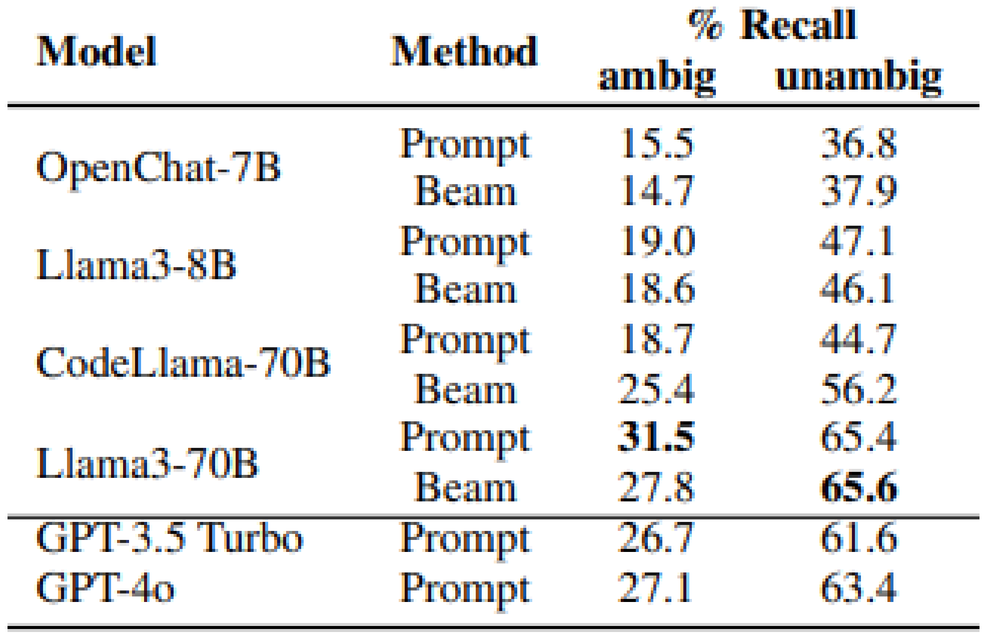
Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order
CHATGPT 3.5 (LLM)	**0.76**	**0.78**	**0.80**	**0.63**	0.83
LLAMA-CODE (LLM)	0.52	0.54	0.58	0.39	**0.86**
RESDSQL (TRL)	0.37	0.38	0.42	0.31	0.46
UNIFIEDSKG (TRL)	0.36	0.37	0.39	0.31	0.65
GAP (TRL)	0.24	0.24	0.26	0.21	0.27

NL text with attribute ambiguity, avg over 13 datasets

Evaluating Ambiguous Questions in Semantic Parsing [Papicchio et al, 2024]

AMBROSIA

[NeurIPS 2024]



Model	Method	% Recall ambig	% Recall unambig
OpenChat-7B	Prompt	15.5	36.8
	Beam	14.7	37.9
Llama3-8B	Prompt	19.0	47.1
	Beam	18.6	46.1
CodeLlama-70B	Prompt	18.7	44.7
	Beam	25.4	56.2
Llama3-70B	Prompt	**31.5**	65.4
	Beam	27.8	**65.6**
GPT-3.5 Turbo	Prompt	26.7	61.6
GPT-4o	Prompt	27.1	63.4

SQL and LLMs Vows

“I will help your users write SQL queries” [Veltri et al, ICDE 2023]

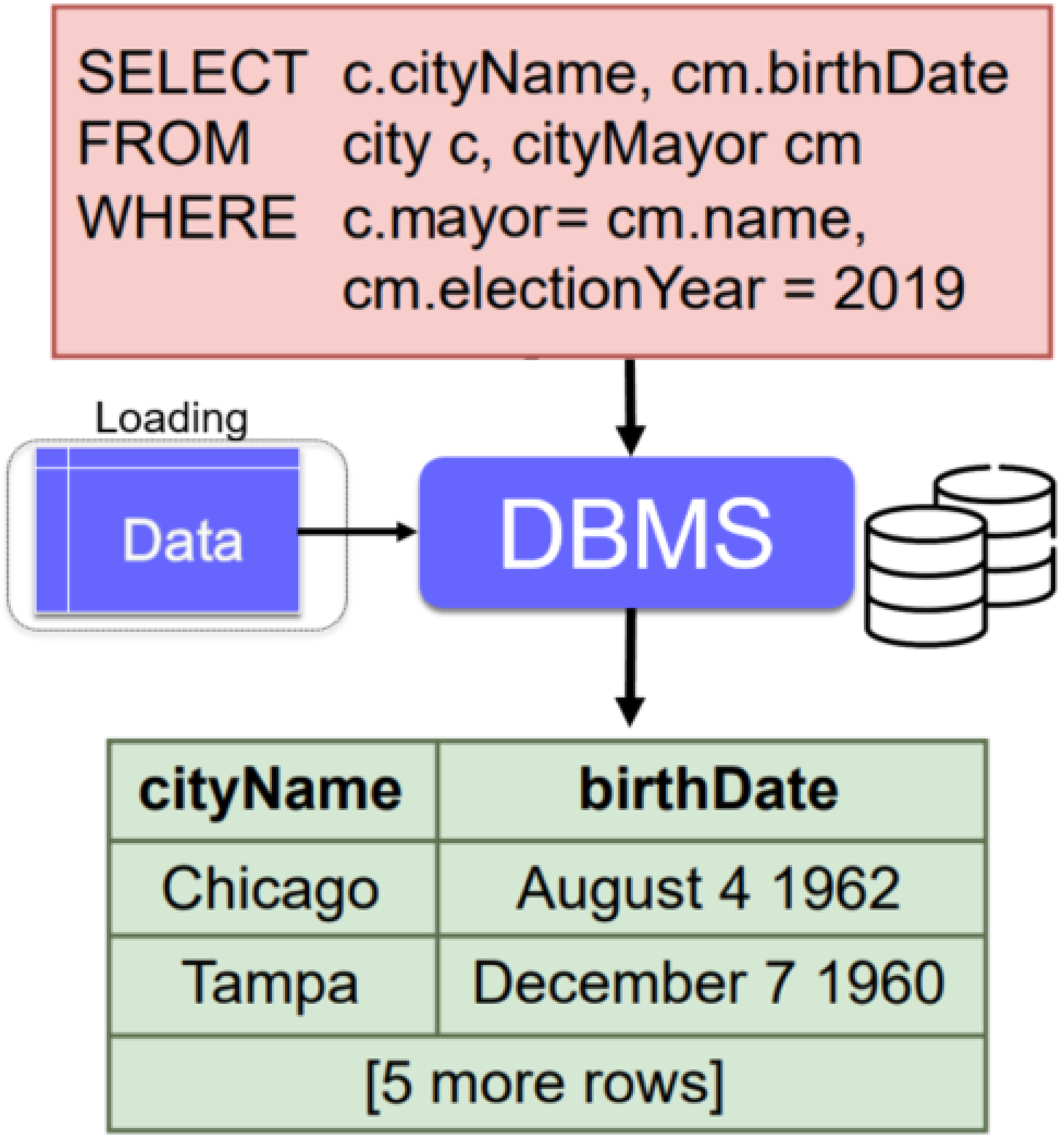
* * *

“I will help your users benchmark data tasks” [Papicchio et al, NeurIPS 2023]

* * *

“We will answer queries jointly” [Saeed et al, EDBT 2024]

SQL



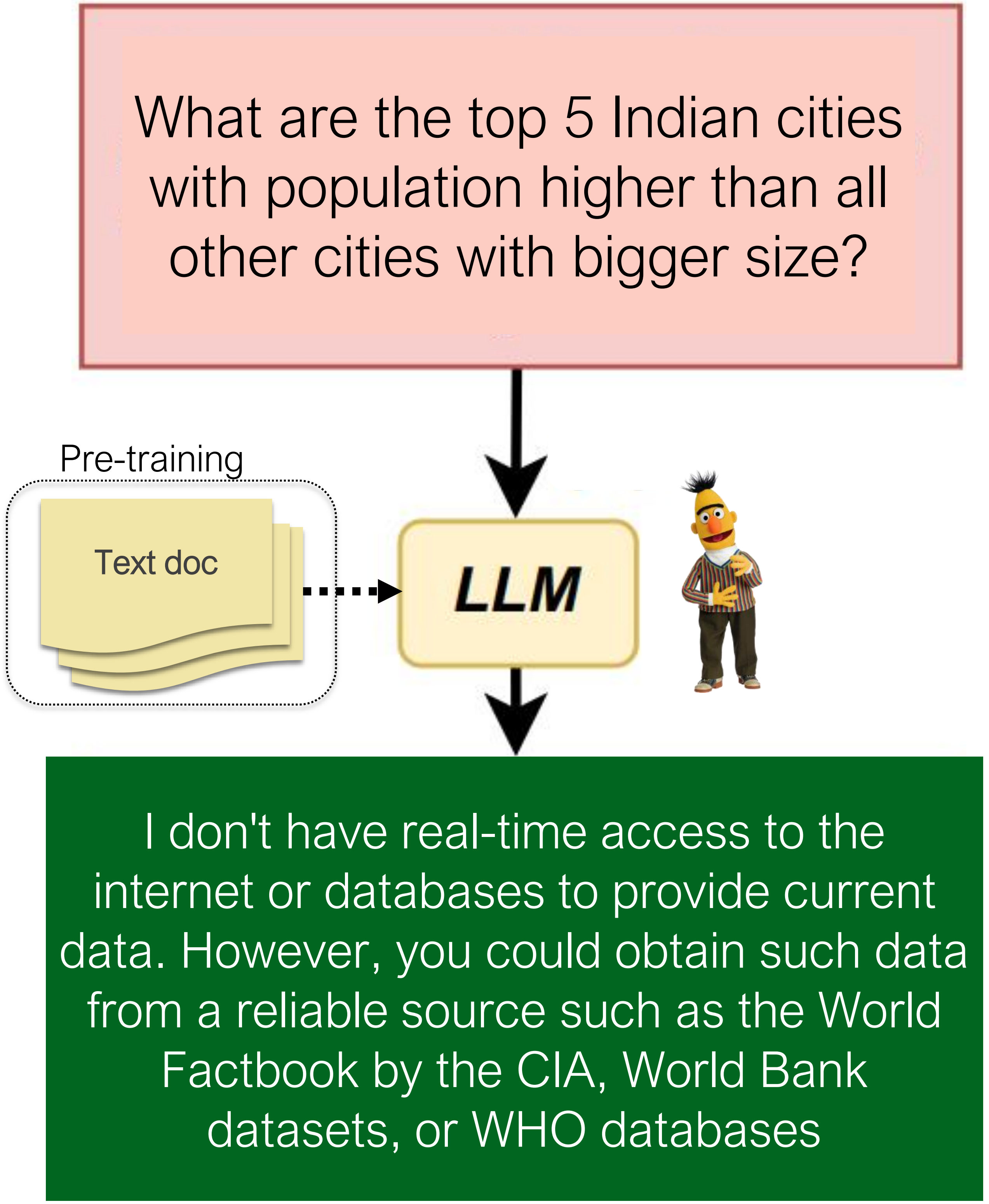
```
SELECT  c.cityName, cm.birthDate
FROM    city c, cityMayor cm
WHERE   c.mayor= cm.name,
        cm.electionYear = 2019
```

cityName	birthDate
Chicago	August 4 1962
Tampa	December 7 1960
[5 more rows]	

- **Powers**
  - scalable and cheap → big data
  - declarative → expressive and precise
  - logic and relational model
  → exact relations

- **Data Applications**
  - rich analytics
  - data warehouse
  - data collection/preparation for ML on structured data (prediction)

LLM



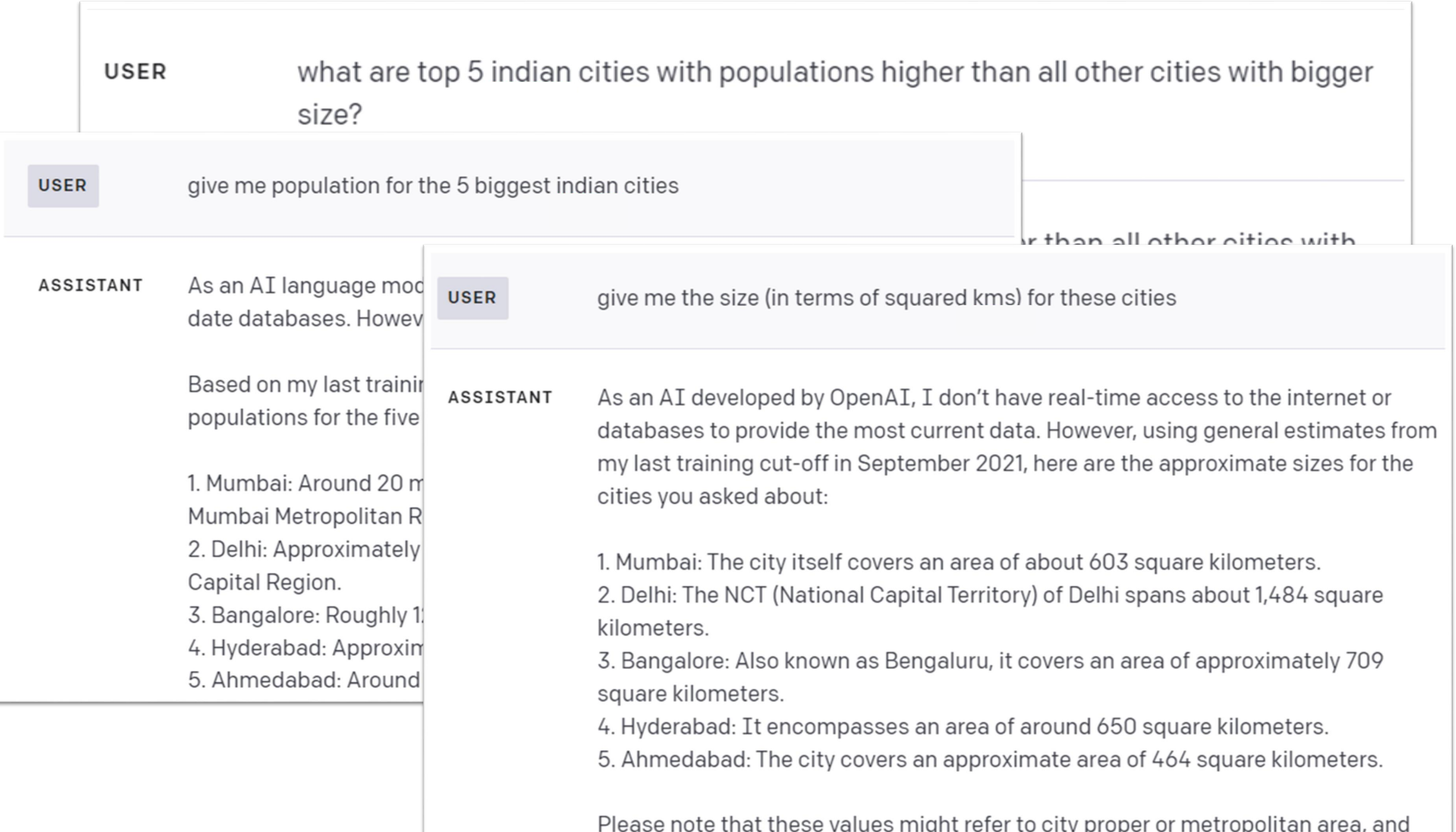
What are the top 5 Indian cities with population higher than all other cities with bigger size?

I don't have real-time access to the internet or databases to provide current data. However, you could obtain such data from a reliable source such as the World Factbook by the CIA, World Bank datasets, or WHO databases

- Powers
  - self supervised → pre-training
  - transfer learning → easy to customize
  - memorization → factual knowledge

- Text Applications
  - text analysis/creation/processing (Question Answering)
  - chatbots/virtual assistant/code
  - …

- ~~Data Applications~~



what are top 5 indian cities with populations higher than all other cities with bigger size?

r than all other cities with

give me population for the 5 biggest indian cities

As an AI language mod
date databases. Howev

Based on my last trainin
populations for the five

1. Mumbai: Around 20 m
Mumbai Metropolitan R
2. Delhi: Approximately
Capital Region.
3. Bangalore: Roughly 1
4. Hyderabad: Approxim
5. Ahmedabad: Around

give me the size (in terms of squared kms) for these cities

As an AI developed by OpenAI, I don’t have real-time access to the internet or databases to provide the most current data. However, using general estimates from my last training cut-off in September 2021, here are the approximate sizes for the cities you asked about:

1. Mumbai: The city itself covers an area of about 603 square kilometers.
2. Delhi: The NCT (National Capital Territory) of Delhi spans about 1,484 square kilometers.
3. Bangalore: Also known as Bengaluru, it covers an area of approximately 709 square kilometers.
4. Hyderabad: It encompasses an area of around 650 square kilometers.
5. Ahmedabad: The city covers an approximate area of 464 square kilometers.

Please note that these values might refer to city proper or metropolitan area, and

User Input:

NL Question

SQL Query

Storage:

Documents

Question answering (QA)

Relations

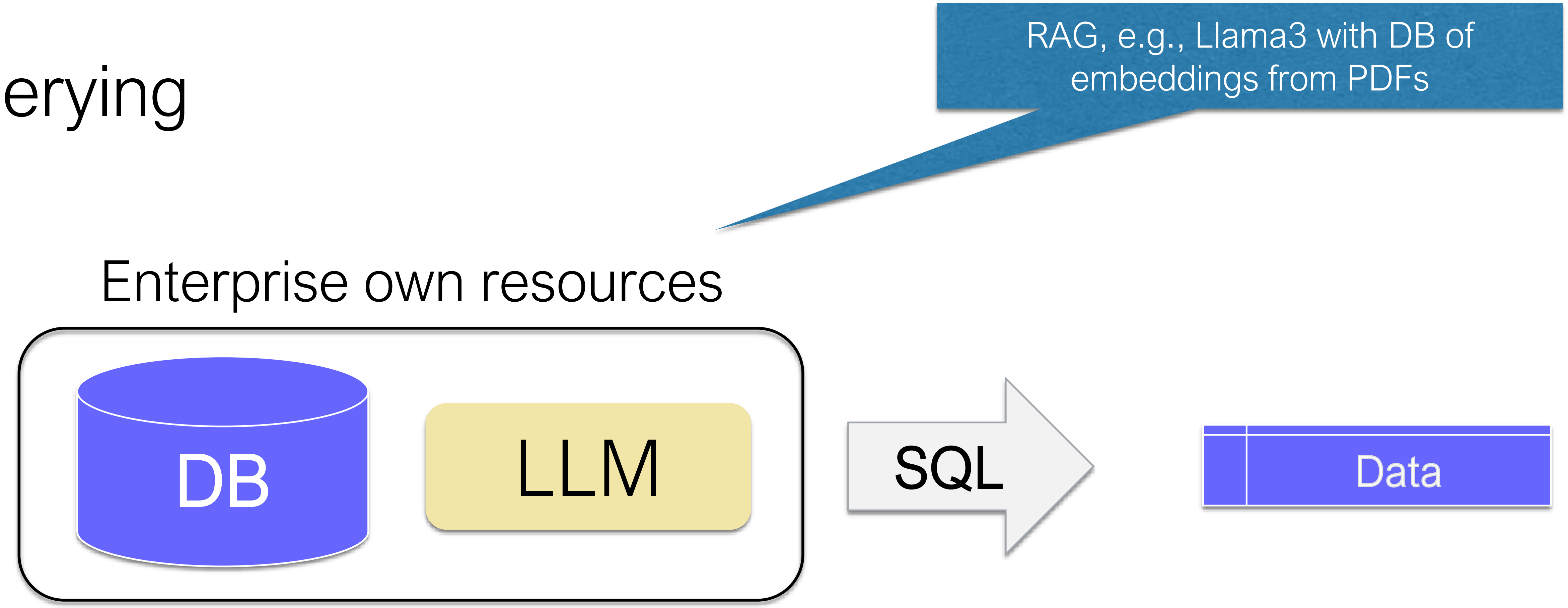
Table QA

Semantic Parsing

Query Execution

Applications

- Hybrid querying



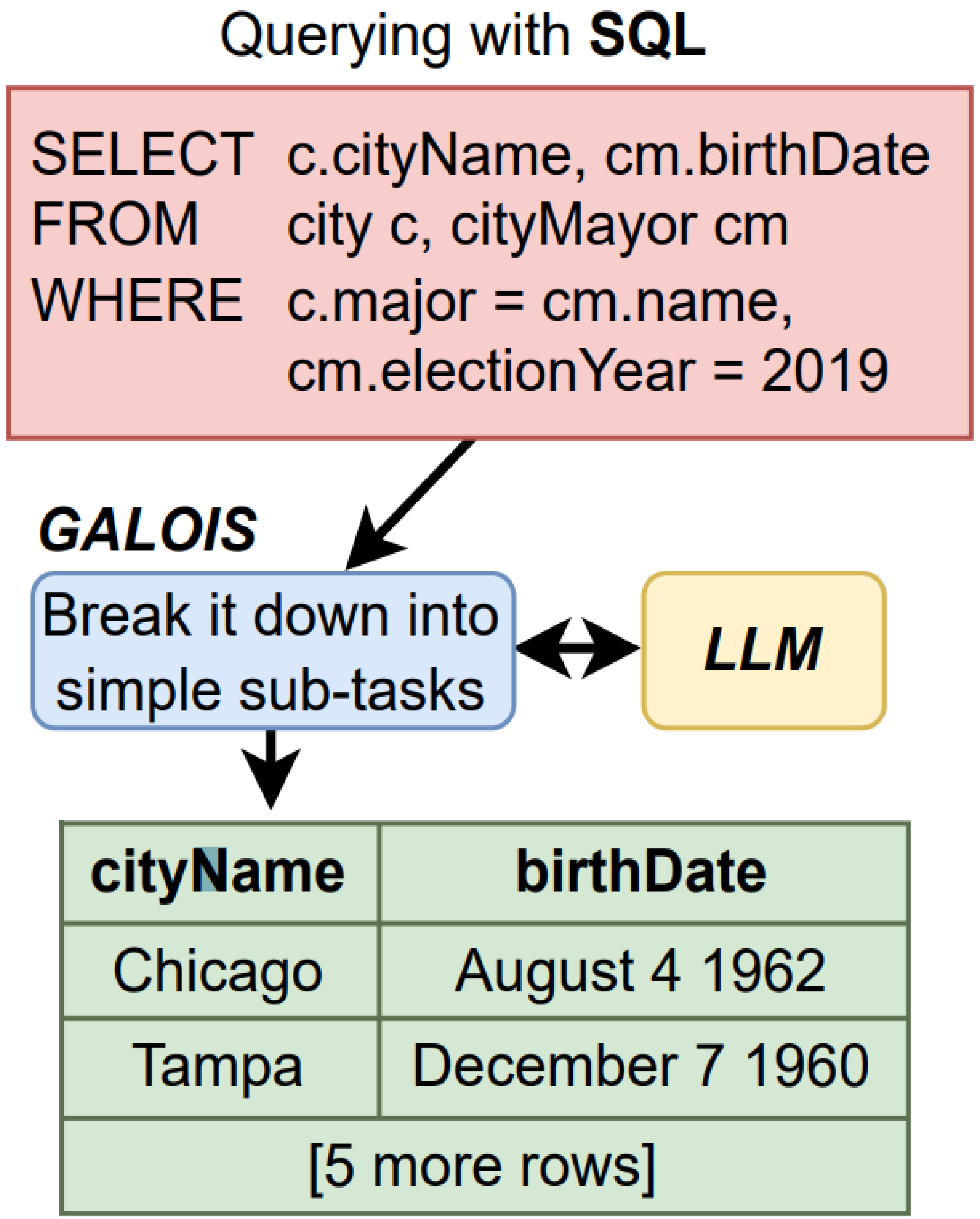
RAG, e.g., Llama3 with DB of embeddings from PDFs

Enterprise own resources

```
SELECT c.researchTopic, AVG(e.salary)
FROM LLM.Employees c, DB.Employees e
WHERE c.eid = e.eid
GROUP BY c.researchTopic
```

Galois: SQL querying LLMs

- **Input**: SQL,
  arbitrary schema with key
- **Storage**: LLM
- **Output**: Relation



Querying with **SQL**

```
SELECT  c.cityName, cm.birthDate
FROM    city c, cityMayor cm
WHERE   c.major = cm.name,
        cm.electionYear = 2019
```

GALOIS

Break it down into simple sub-tasks

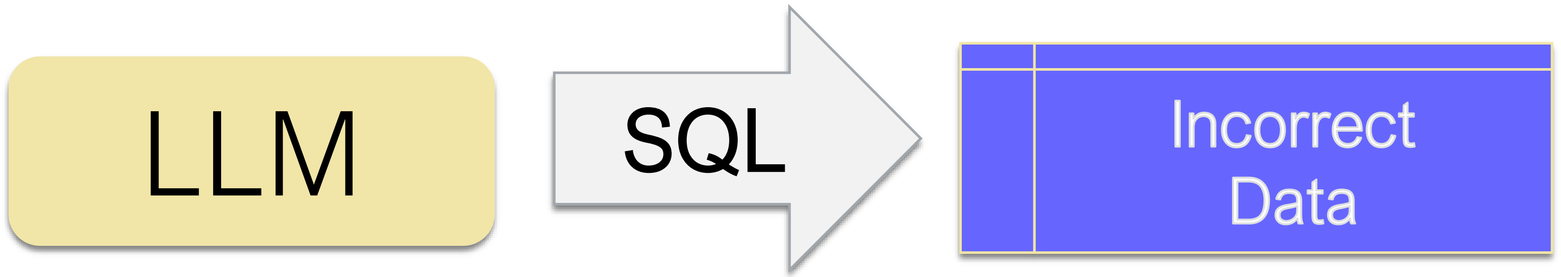
LLM

cityName	**birthDate**
Chicago	August 4 1962
Tampa	December 7 1960
[5 more rows]	

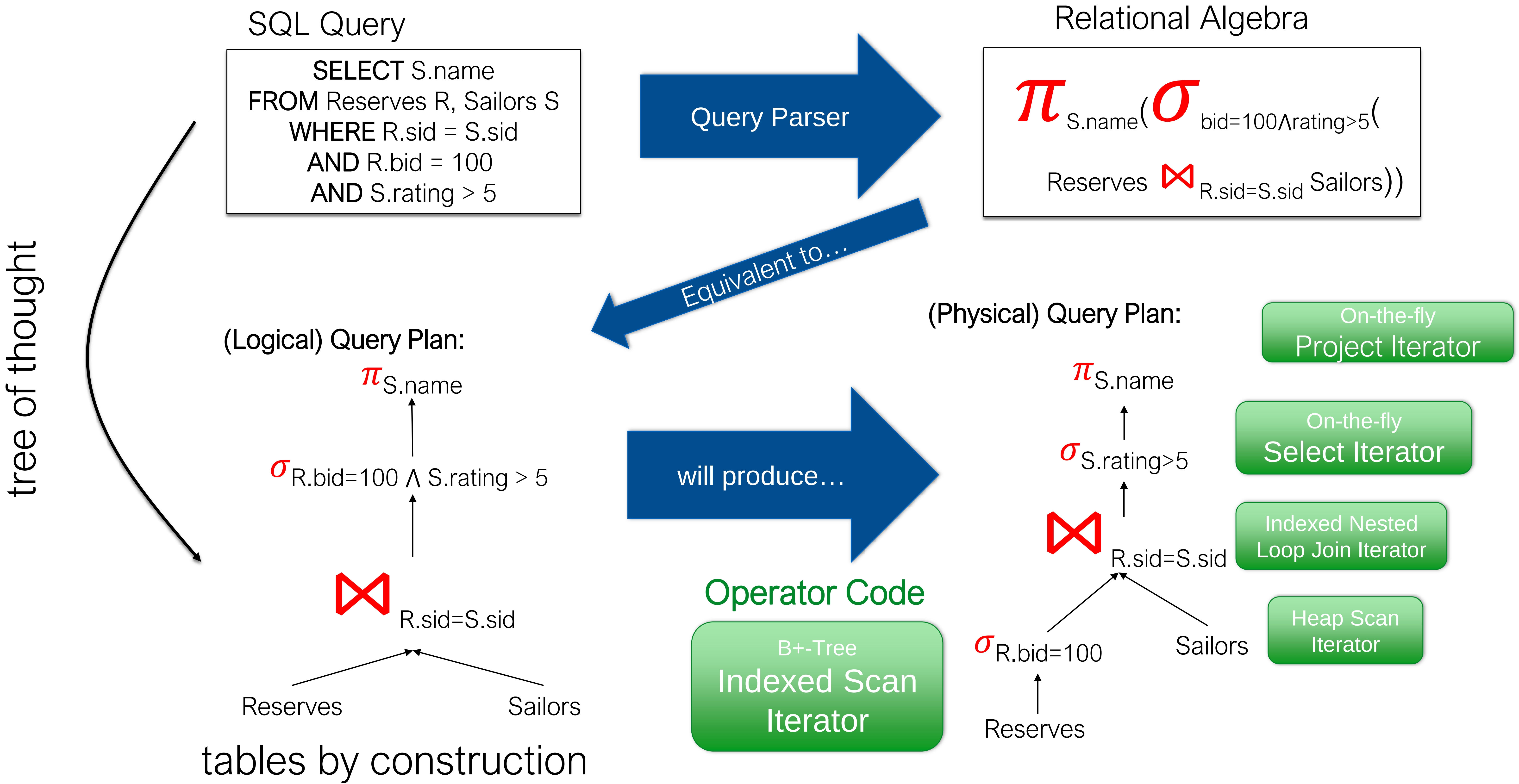
Challenges

- LLMs store factual data, but
  - **Input**: Not trained to execute SQL faithfully
  - **Engine**: Struggle with complex tasks
  - **Output**: Not trained to (precisely) return relations

Errors



Query processing in 1 slide



SQL Query

```
SELECT S.name
FROM Reserves R, Sailors S
WHERE R.sid = S.sid
AND R.bid = 100
AND S.rating > 5
```

Query Parser

Relational Algebra

$$\pi_{S.name}(\sigma_{bid=100 \land rating>5}(\text{Reserves} \bowtie_{R.sid=S.sid} \text{Sailors}))$$

Equivalent to…

(Logical) Query Plan:

$\pi_{S.name}$

$\sigma_{R.bid=100 \land S.rating > 5}$

$\bowtie_{R.sid=S.sid}$

Reserves

Sailors

tree of thought

tables by construction

will produce…

(Physical) Query Plan:

$\pi_{S.name}$ On-the-fly Project Iterator

$\sigma_{S.rating>5}$ On-the-fly Select Iterator

$\bowtie_{R.sid=S.sid}$ Indexed Nested Loop Join Iterator

$\sigma_{R.bid=100}$

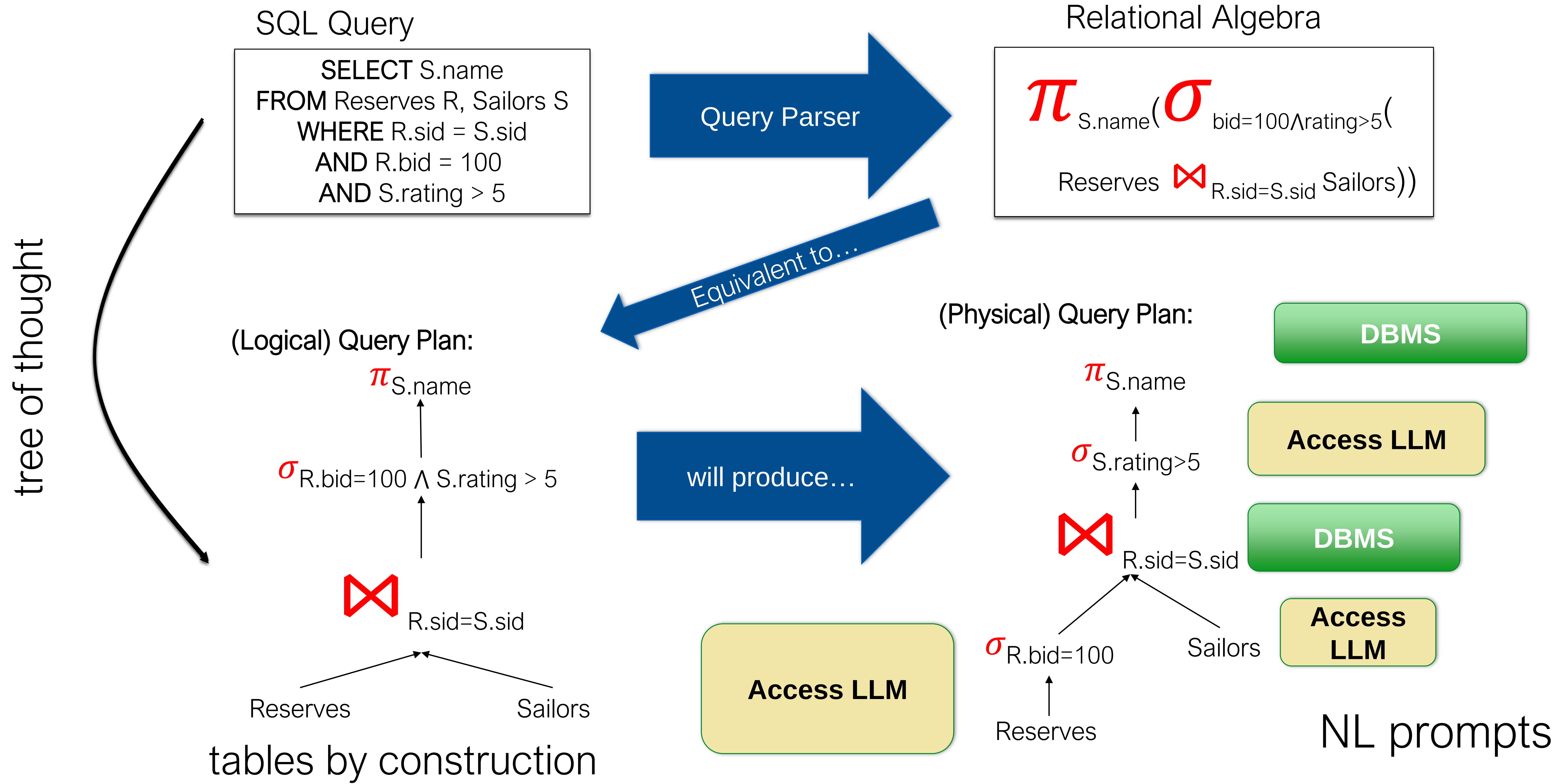
Reserves

Sailors Heap Scan Iterator

Operator Code

B+-Tree Indexed Scan Iterator

Query processing in 1 slide



SQL Query

```
SELECT S.name
FROM Reserves R, Sailors S
WHERE R.sid = S.sid
AND R.bid = 100
AND S.rating > 5
```

Query Parser

Relational Algebra

$$\pi_{S.name}(\sigma_{bid=100 \land rating>5}(\text{Reserves} \bowtie_{R.sid=S.sid} \text{Sailors}))$$

Equivalent to…

(Logical) Query Plan:

$\pi_{S.name}$

$\sigma_{R.bid=100 \land S.rating > 5}$

$\bowtie_{R.sid=S.sid}$

Reserves Sailors

will produce…

(Physical) Query Plan:

$\pi_{S.name}$ — DBMS

$\sigma_{S.rating>5}$ — Access LLM

$\bowtie_{R.sid=S.sid}$ — DBMS

$\sigma_{R.bid=100}$ Sailors — Access LLM

Reserves

Access LLM

tree of thought

tables by construction

NL prompts

Physical Query Plan

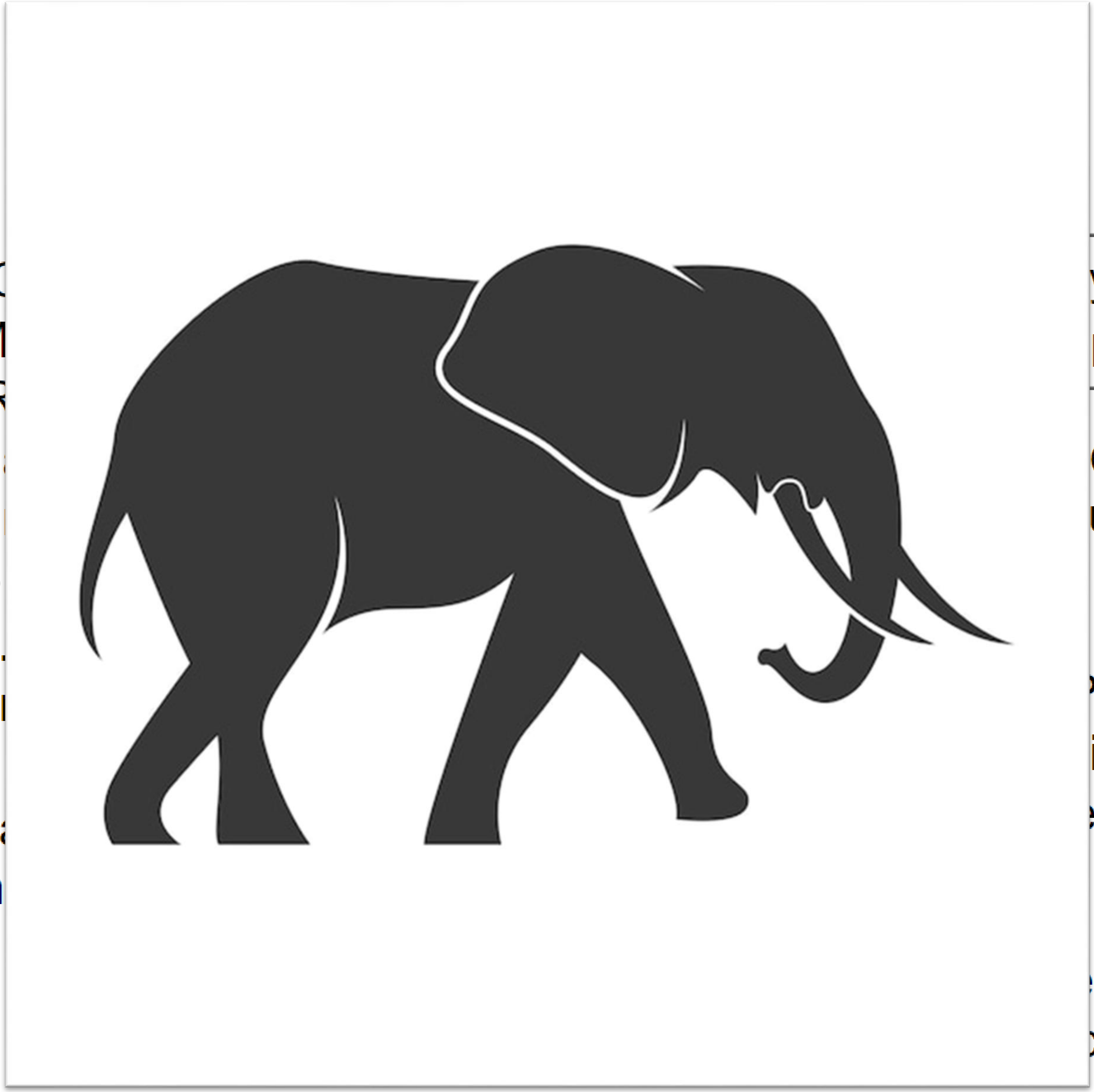
```
q’: SELECT c.name, p.name
    FROM Cities c, Politicians p
    WHERE c.population> ‘1M’,
          p.age<40,
          p.name=c.currentMayor
```

```
q’: SELE
    FROM
    WHER
        p.
        p.
```

∀ c’∈C’, c’.
“Get curre

∀ c∈C, “H
more than

Tuples C:



ython operator
LM based op.

∈P’
urrentMayor

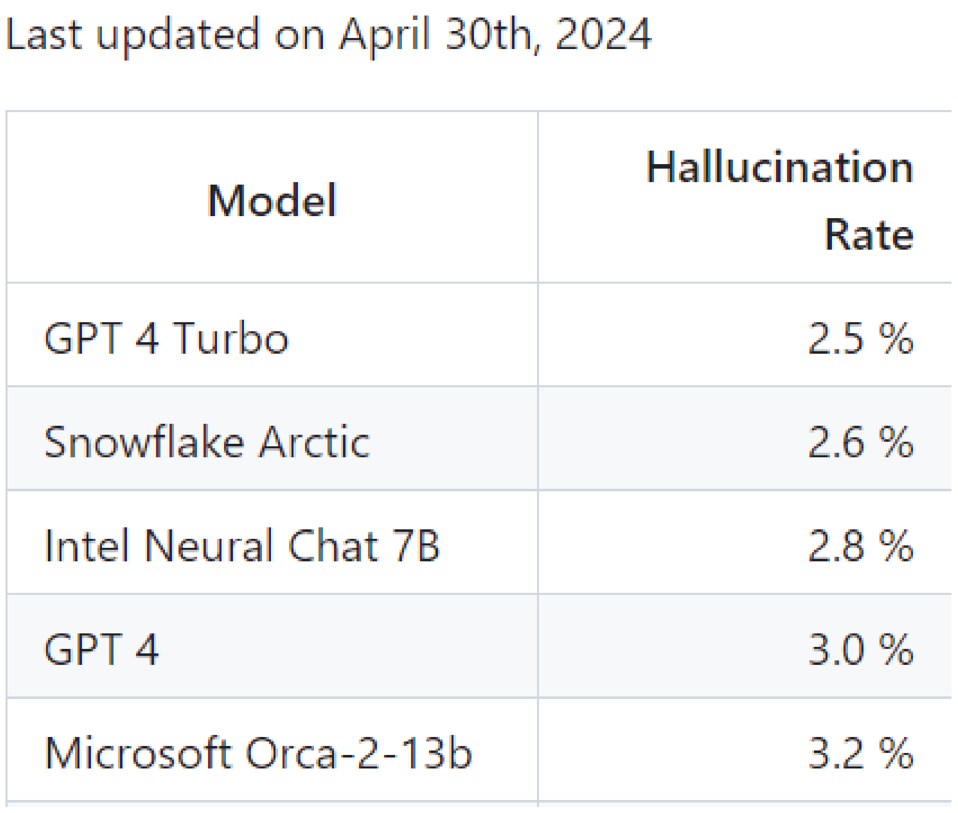
’ “Has
ian *p*.name
ss than 40?”

s P: “Get
cian names”

Factuality

- Decoder returns next token based on training data
- Such token may be based on either reliable acquired knowledge, or it may be a guess
  → hallucinations

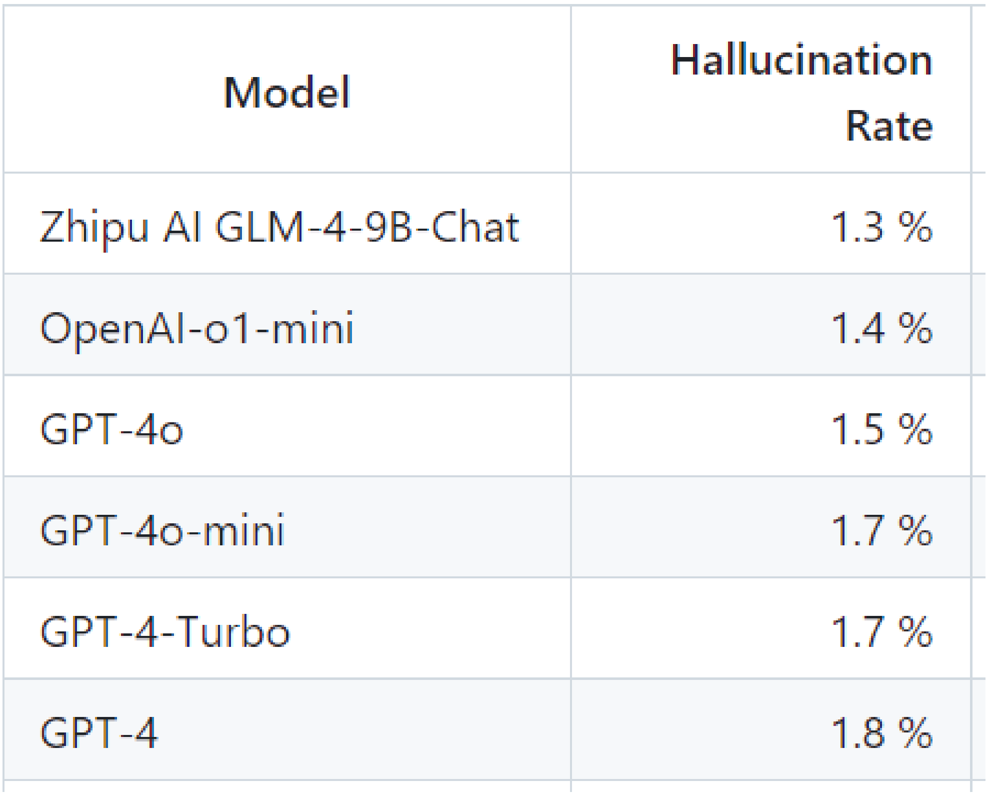
+ Models keep increasing the factuality of their answers*
+ Encouraging results from Galois



Last updated on April 30th, 2024

Model	Hallucination Rate
GPT 4 Turbo	2.5 %
Snowflake Arctic	2.6 %
Intel Neural Chat 7B	2.8 %
GPT 4	3.0 %
Microsoft Orca-2-13b	3.2 %

https://github.com/vectara/hallucination-leaderboard



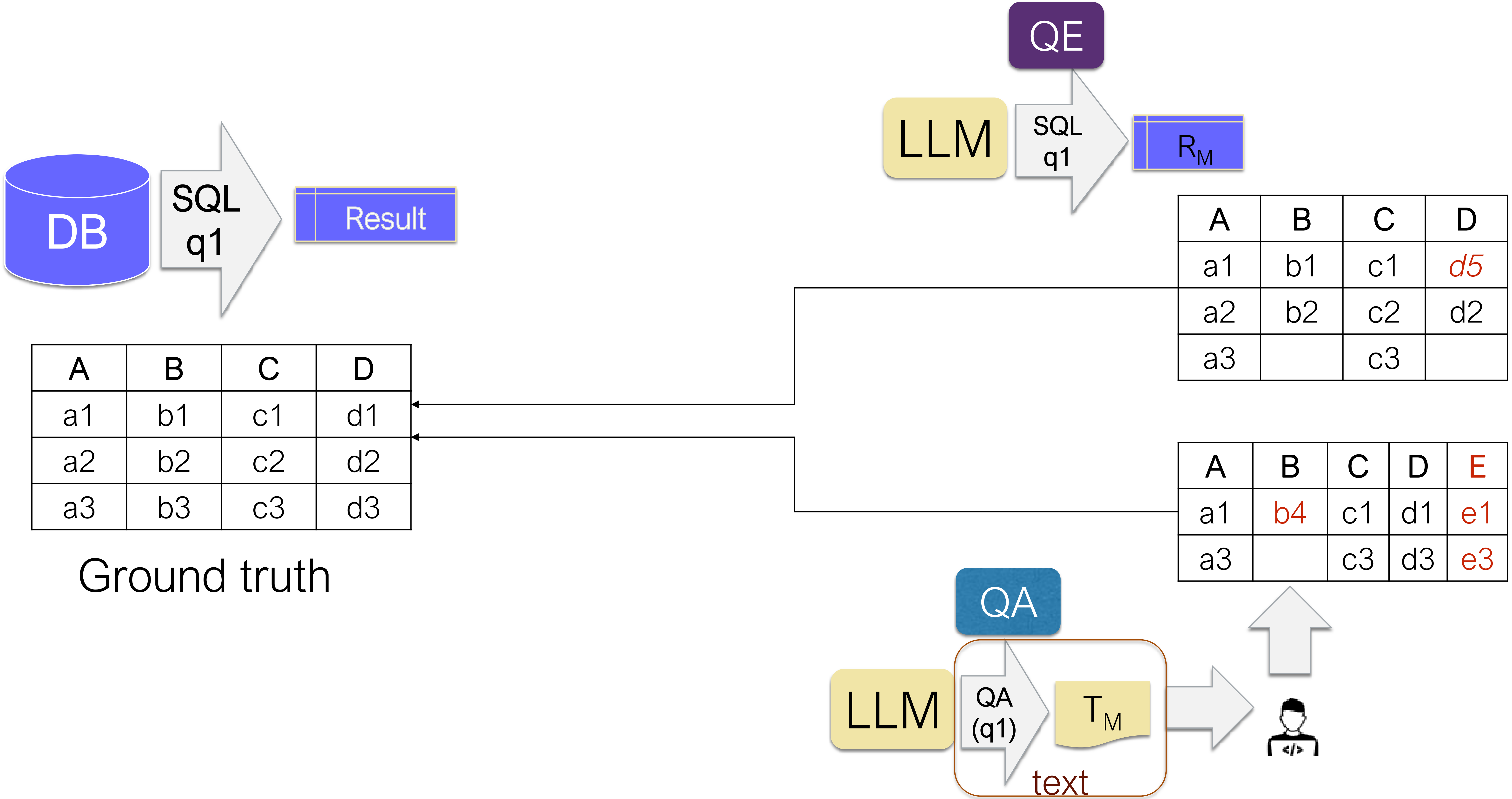
Model	Hallucination Rate
Zhipu AI GLM-4-9B-Chat	1.3 %
OpenAI-o1-mini	1.4 %
GPT-4o	1.5 %
GPT-4o-mini	1.7 %
GPT-4-Turbo	1.7 %
GPT-4	1.8 %

*"GPT-4 scores 40% higher than GPT-3.5 on our factuality evaluations"

Experiments - data

- Corpus of 92 SQL “reasonable” queries/questions from Spider (200 datasets)
  - **No**: “How many heads of the departments are older than 56?”
  - **Yes**: “What are the names of the countries that became independent after 1950?”
- Tested 2 LLMs: Llama 8B and 70B

Experiments – QA as “upper bound”



A	B	C	D
a1	b1	c1	*d5*
a2	b2	c2	d2
a3		c3	

A	B	C	D	E
a1	b4	c1	d1	e1
a3		c3	d3	e3

A	B	C	D
a1	b1	c1	d1
a2	b2	c2	d2
a3	b3	c3	d3

Ground truth

Results Llama 70B

- Matching in output results between ground truth and
  - Galois - our method R_M (SQL queries) QE
  - traditional T_M (NL questions) QA

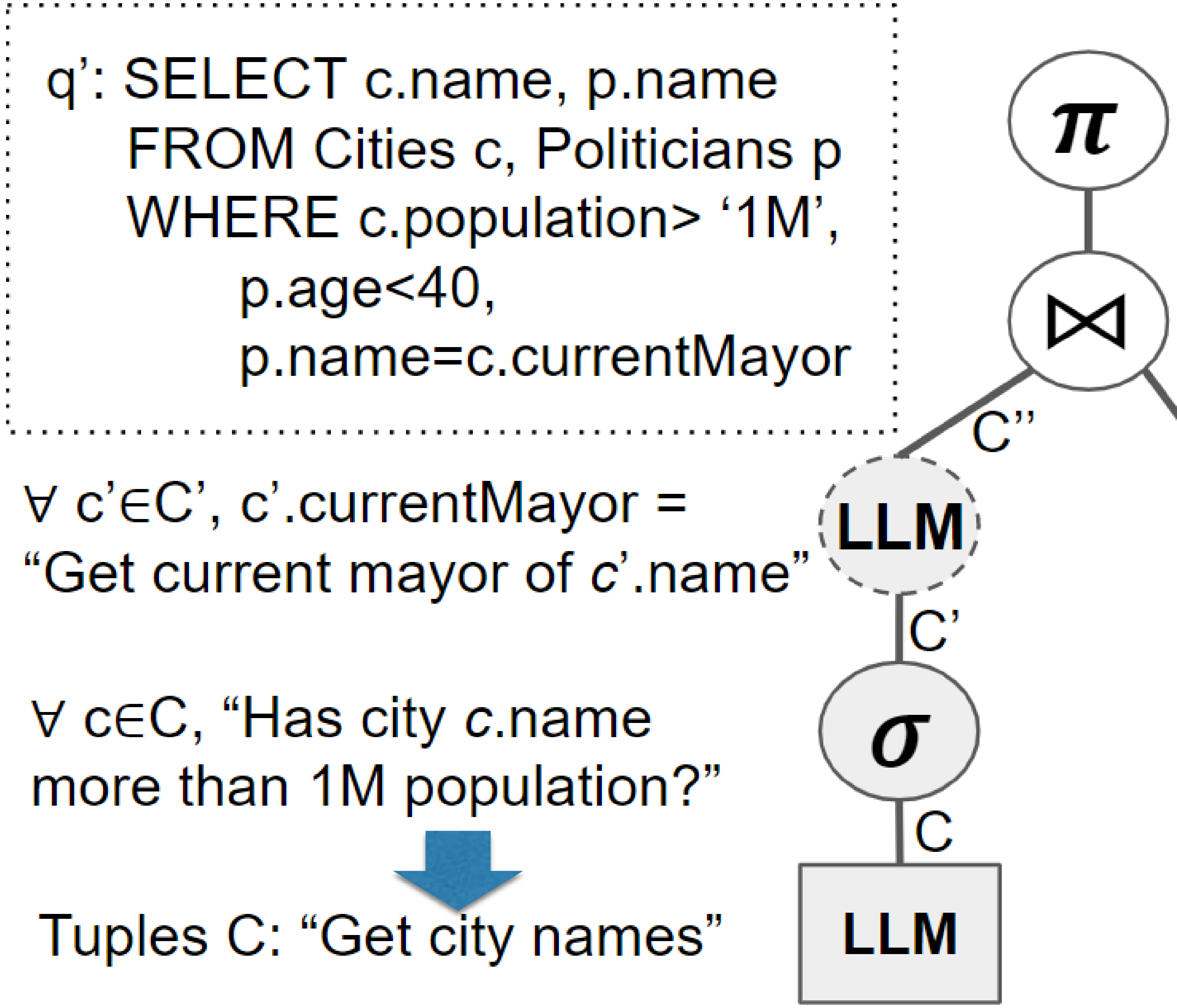
Metric	NL	Galois	Galois Query Opt.*
F1-cell	0.24	0.48	**0.56**
Cardinality	0.46	0.66	**0.84**
Tuple Constraint	0.07	0.45	**0.47**

*SQL Meets LLMs: Query Optimization for Improved Quality and Efficiency [Satriani et al, 2024]

https://github.com/dbunibas/galois

Query optimization

- **Physical**: reduce hallucinations
  → get tuple vs get key and values
  → Reconfidencing [Chen et al, 2024]
- **Logical**: Reduce LLM calls → push down selections (“get names of cities with > 1M population”)
  - Optimize cost, quality.. Without metadata/catalog



q’: SELECT c.name, p.name
FROM Cities c, Politicians p
WHERE c.population> ‘1M’,
p.age<40,
p.name=c.currentMayor

∀ c’∈C’, c’.currentMayor = “Get current mayor of *c*’.name”

∀ c∈C, “Has city *c*.name more than 1M population?”

Tuples C: “Get city names”

Error analysis

- LLMs trained to output NL text adhering to human preferences
    - **Different formats**: join country code “IT" with “ITA” for entity Italy
    - **Entity linking**: “Brussels” vs “Bruxelles”
    - **Verbose output**: “The city of Paris”
- **Issues with ground truth**: population changes over time
- **Rare values**: repetition of popular ones

Open Questions

Logic controls LLM

LLM controls logic

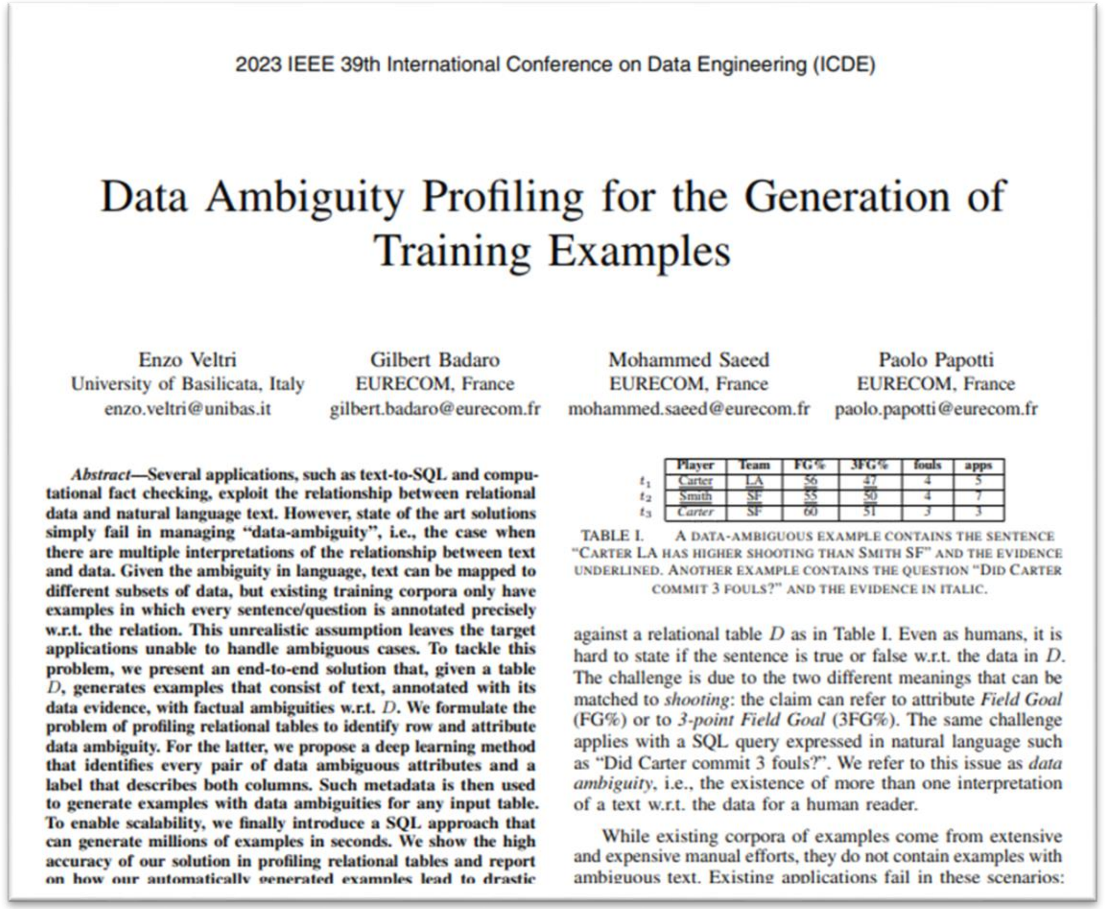
LLM *does/imitates* logic

DB first
use LLM in operators – *Galois*
[Jo and Trummer, 2023],
[Urban et al, 2023]

LLMs + Agents
SP better results than TQA
→ Use LM for NLU, SQL/code for data operations
[Arora et al, 2023]
[Pourreza et al, 2024]

LLM first
structured data in pre-training, extensions, fine tuning…. But fine tuned ChatGPT: *0.53 accuracy* for TQA
[Badaro et al, 2023]
[Li et al, 2023]

SQL and LLMs?



2023 IEEE 39th International Conference on Data Engineering (ICDE)

Data Ambiguity Profiling for the Generation of Training Examples

Enzo Veltri
University of Basilicata, Italy
enzo.veltri@unibas.it

Gilbert Badaro
EURECOM, France
gilbert.badaro@eurecom.fr

Mohammed Saeed
EURECOM, France
mohammed.saeed@eurecom.fr

Paolo Papotti
EURECOM, France
paolo.papotti@eurecom.fr

***Abstract*—Several applications, such as text-to-SQL and computational fact checking, exploit the relationship between relational data and natural language text. However, state of the art solutions simply fail in managing "data-ambiguity", i.e., the case when there are multiple interpretations of the relationship between text and data. Given the ambiguity in language, text can be mapped to different subsets of data, but existing training corpora only have examples in which every sentence/question is annotated precisely w.r.t. the relation. This unrealistic assumption leaves the target applications unable to handle ambiguous cases. To tackle this problem, we present an end-to-end solution that, given a table D, generates examples that consist of text, annotated with its data evidence, with factual ambiguities w.r.t. D. We formulate the problem of profiling relational tables to identify row and attribute data ambiguity. For the latter, we propose a deep learning method that identifies every pair of data ambiguous attributes and a label that describes both columns. Such metadata is then used to generate examples with data ambiguities for any input table. To enable scalability, we finally introduce a SQL approach that can generate millions of examples in seconds. We show the high accuracy of our solution in profiling relational tables and report on how our automatically generated examples lead to drastic**

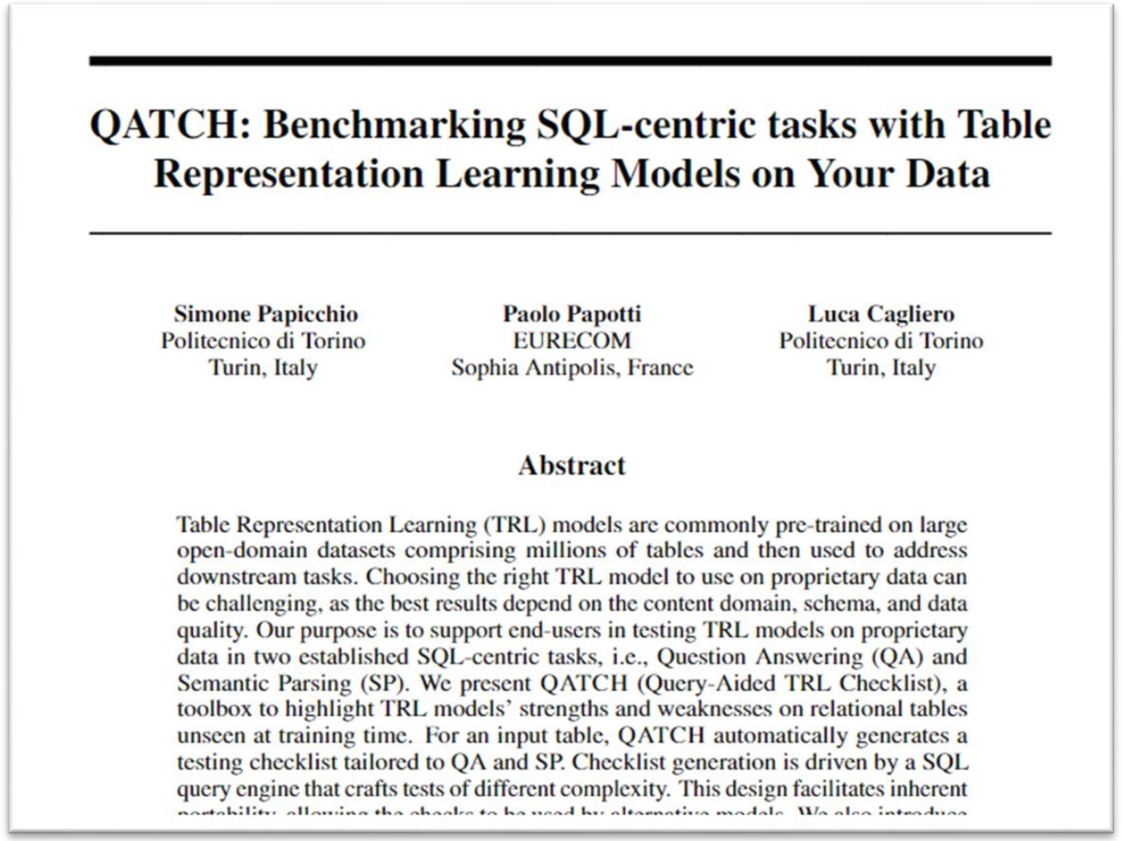
	Player	Team	FG%	3FG%	fouls	apps
t_1	Carter	LA	56	47	4	5
t_2	Smith	SF	55	50	4	7
t_3	*Carter*	SF	60	51	3	3

TABLE I. A DATA-AMBIGUOUS EXAMPLE CONTAINS THE SENTENCE "CARTER LA HAS HIGHER SHOOTING THAN SMITH SF" AND THE EVIDENCE UNDERLINED. ANOTHER EXAMPLE CONTAINS THE QUESTION "DID CARTER COMMIT 3 FOULS?" AND THE EVIDENCE IN ITALIC.

against a relational table D as in Table I. Even as humans, it is hard to state if the sentence is true or false w.r.t. the data in D. The challenge is due to the two different meanings that can be matched to *shooting*: the claim can refer to attribute *Field Goal* (FG%) or to *3-point Field Goal* (3FG%). The same challenge applies with a SQL query expressed in natural language such as "Did Carter commit 3 fouls?". We refer to this issue as *data ambiguity*, i.e., the existence of more than one interpretation of a text w.r.t. the data for a human reader.

While existing corpora of examples come from extensive and expensive manual efforts, they do not contain examples with ambiguous text. Existing applications fail in these scenarios:

https://github.com/enzoveltri/pythia



QATCH: Benchmarking SQL-centric tasks with Table Representation Learning Models on Your Data

Simone Papicchio
Politecnico di Torino
Turin, Italy

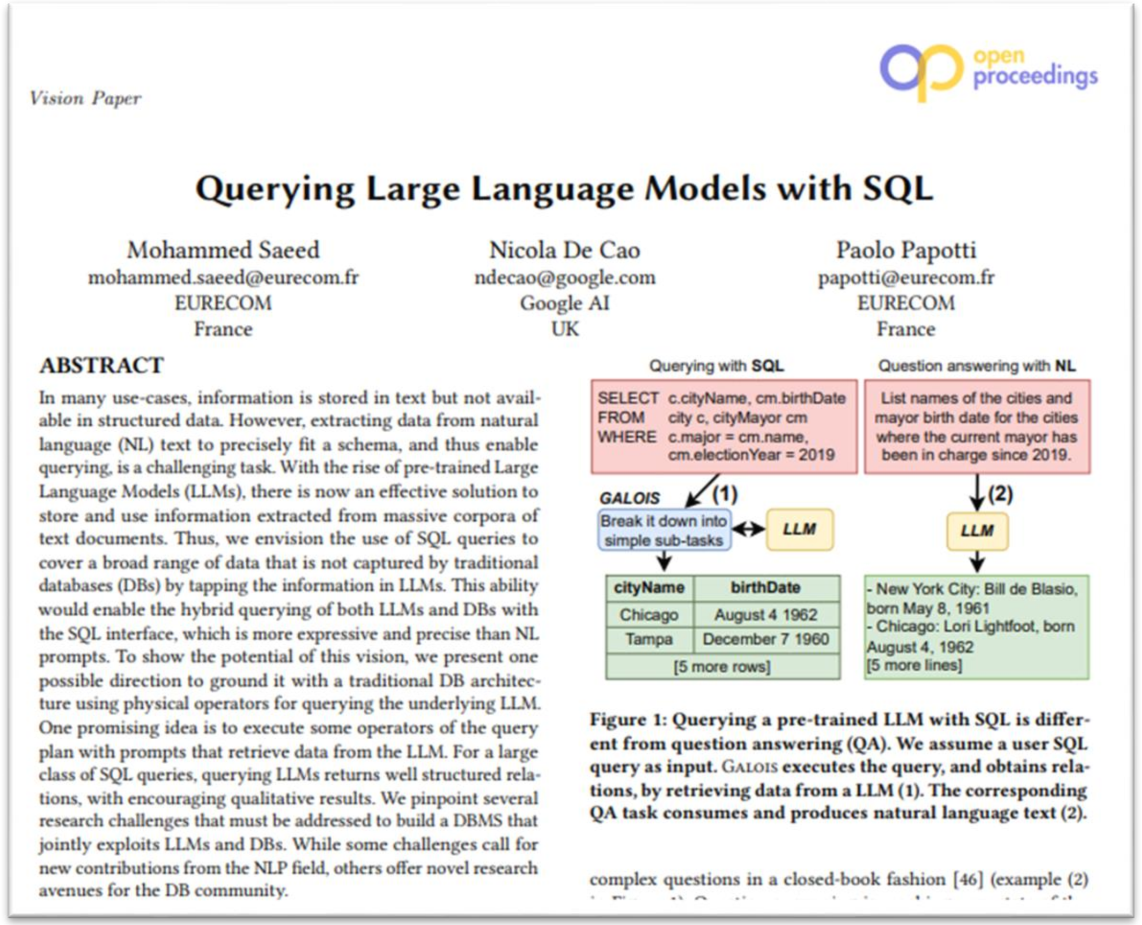
Paolo Papotti
EURECOM
Sophia Antipolis, France

Luca Cagliero
Politecnico di Torino
Turin, Italy

Abstract

Table Representation Learning (TRL) models are commonly pre-trained on large open-domain datasets comprising millions of tables and then used to address downstream tasks. Choosing the right TRL model to use on proprietary data can be challenging, as the best results depend on the content domain, schema, and data quality. Our purpose is to support end-users in testing TRL models on proprietary data in two established SQL-centric tasks, i.e., Question Answering (QA) and Semantic Parsing (SP). We present QATCH (Query-Aided TRL Checklist), a toolbox to highlight TRL models' strengths and weaknesses on relational tables unseen at training time. For an input table, QATCH automatically generates a testing checklist tailored to QA and SP. Checklist generation is driven by a SQL query engine that crafts tests of different complexity. This design facilitates inherent

https://github.com/spapicchio/QATCH



Vision Paper

open proceedings

Querying Large Language Models with SQL

Mohammed Saeed
mohammed.saeed@eurecom.fr
EURECOM
France

Nicola De Cao
ndecao@google.com
Google AI
UK

Paolo Papotti
papotti@eurecom.fr
EURECOM
France

ABSTRACT

In many use-cases, information is stored in text but not available in structured data. However, extracting data from natural language (NL) text to precisely fit a schema, and thus enable querying, is a challenging task. With the rise of pre-trained Large Language Models (LLMs), there is now an effective solution to store and use information extracted from massive corpora of text documents. Thus, we envision the use of SQL queries to cover a broad range of data that is not captured by traditional databases (DBs) by tapping the information in LLMs. This ability would enable the hybrid querying of both LLMs and DBs with the SQL interface, which is more expressive and precise than NL prompts. To show the potential of this vision, we present one possible direction to ground it with a traditional DB architecture using physical operators for querying the underlying LLM. One promising idea is to execute some operators of the query plan with prompts that retrieve data from the LLM. For a large class of SQL queries, querying LLMs returns well structured relations, with encouraging qualitative results. We pinpoint several research challenges that must be addressed to build a DBMS that jointly exploits LLMs and DBs. While some challenges call for new contributions from the NLP field, others offer novel research avenues for the DB community.

Querying with **SQL**

```
SELECT  c.cityName, cm.birthDate
FROM    city c, cityMayor cm
WHERE   c.major = cm.name,
        cm.electionYear = 2019
```

Question answering with **NL**

List names of the cities and mayor birth date for the cities where the current mayor has been in charge since 2019.

GALOIS (1)

Break it down into simple sub-tasks

LLM

(2)

LLM

cityName	birthDate
Chicago	August 4 1962
Tampa	December 7 1960
[5 more rows]	

- New York City: Bill de Blasio, born May 8, 1961
- Chicago: Lori Lightfoot, born August 4, 1962
[5 more lines]

Figure 1: Querying a pre-trained LLM with SQL is different from question answering (QA). We assume a user SQL query as input. GALOIS executes the query, and obtains relations, by retrieving data from a LLM (1). The corresponding QA task consumes and produces natural language text (2).

complex questions in a closed-book fashion [46] (example (2)

https://gitlab.eurecom.fr/saeedm1/galois

http://www.eurecom.fr/~papotti/

@paolopapotti

BDA – 22nd Oct 2024