

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

Paolo Papotti



SQL and LLMs?

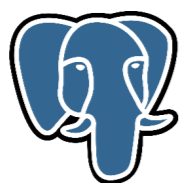
```
SELECT
  OrderH.invoiceNo, OrderH.invoiceDate, OrderH.customerCode,
  OrderD.itemCode, I.itemName, OrderD.qty, OrderD.netPrice
FROM
  OrderHeader AS OrderH
  INNER JOIN Customer AS Cust ON OrderH.customerCode = Cust.customerCode
  INNER JOIN OrderDetail AS OrderD ON OrderH.invoiceNo = OrderD.invoiceNo
  INNER JOIN Item AS I ON OrderD.itemCode = I.itemCode
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:

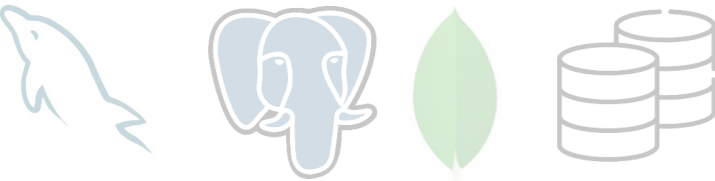
NL Question

SQL Query

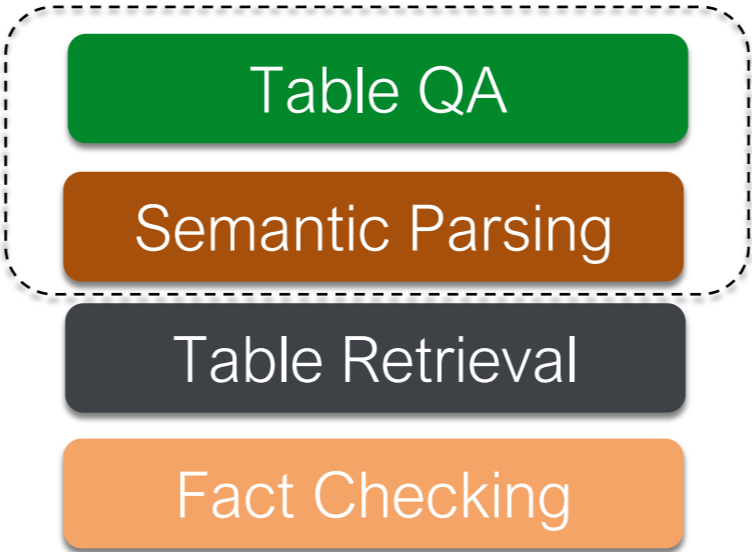
Storage:

Documents

Relations



Question answering
(QA)



Query Execution

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
```

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

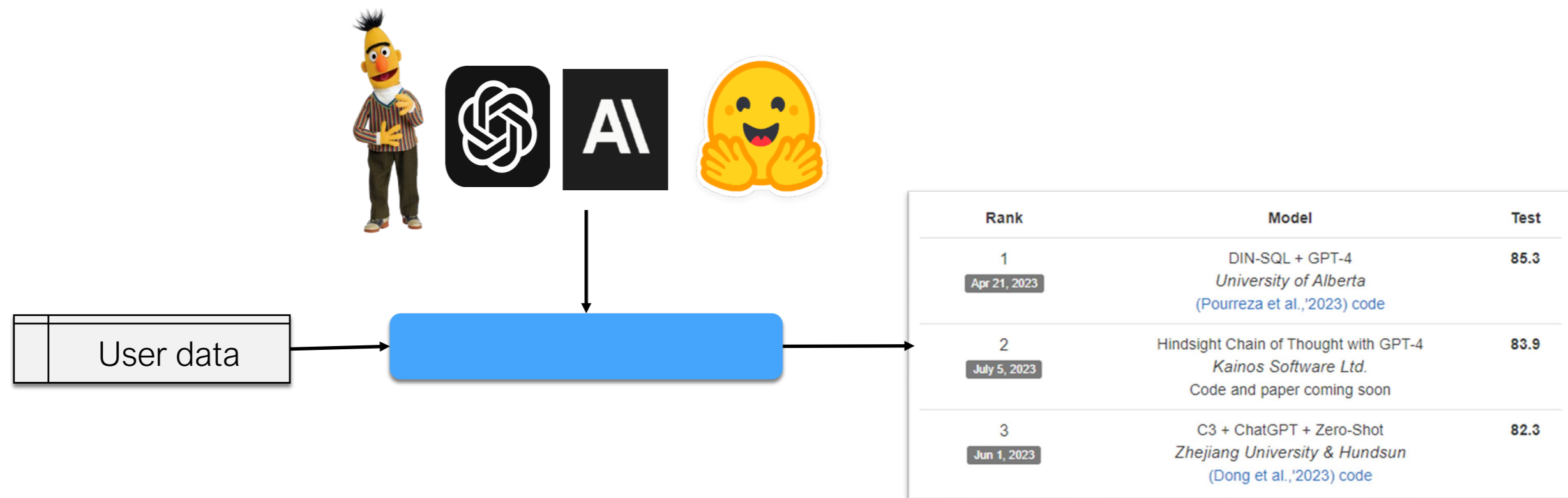
Rank	Model	Test
1 Nov 2, 2023	MiniSeek Anonymous Code and paper coming soon	91.2
1 Aug 20, 2023	DAIL-SQL + GPT-4 + Self-Consistency Alibaba Group (Gao and Wang et al., '2023) code	86.6
2 Aug 9, 2023	DAIL-SQL + GPT-4 Alibaba Group (Gao and Wang et al., '2023) code	86.2
3 October 17, 2023	DPG-SQL + GPT-4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

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



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)

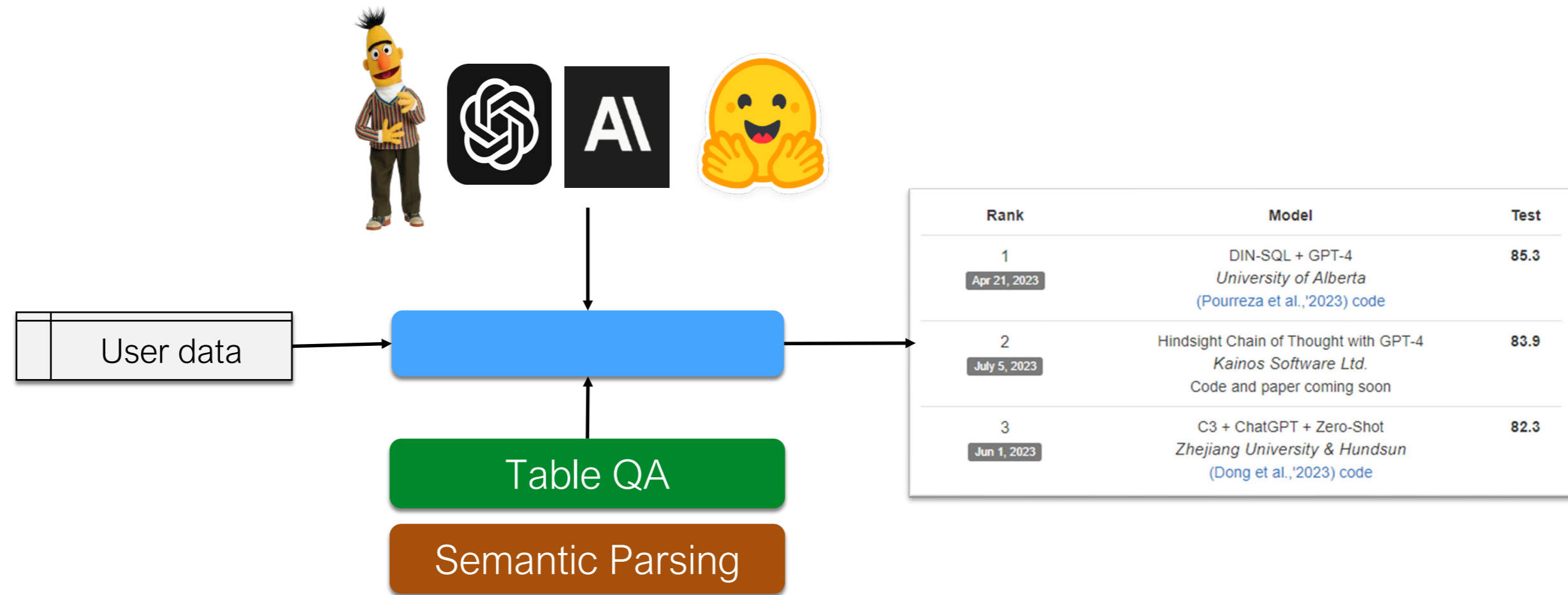


“Laure, Mike”

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

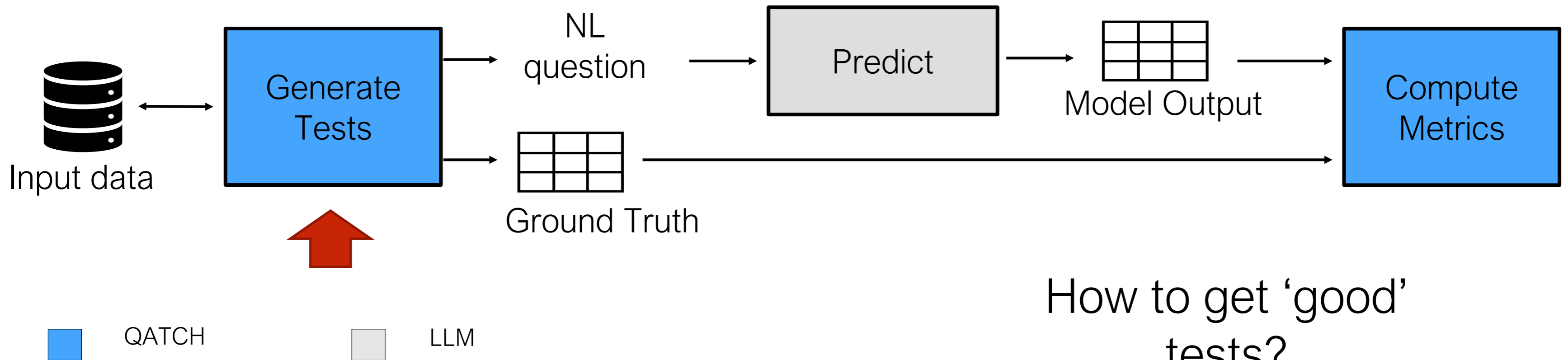
Custom benchmark on *user data*

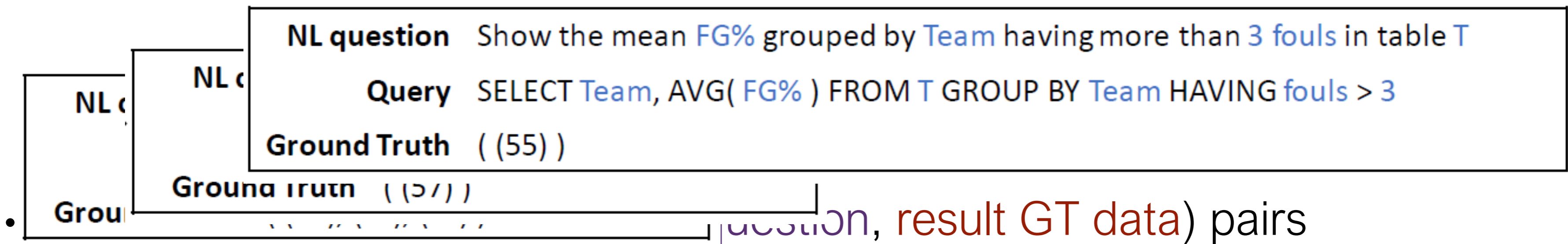
- Given proprietary table D
- Automatically rank existing LLMs on T for **data-task**



QATCH: Query-Aided TRL Checklist


- 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



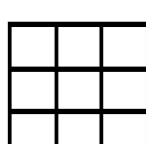


- Focus on query complexity: 1 to n attributes/conditions, ...
- Simple text: no ambiguity, no failure, plain English

Category	SQL declaration	Free-Text question
Project	SELECT {c ₁ , ..., c _n } FROM {T}	Show {c ₁ , ..., c _n } in table {T}
Distinct	SELECT DISTINCT {c ₁ , ..., c _n } FROM {T}	Show the different {c ₁ , ..., 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 }

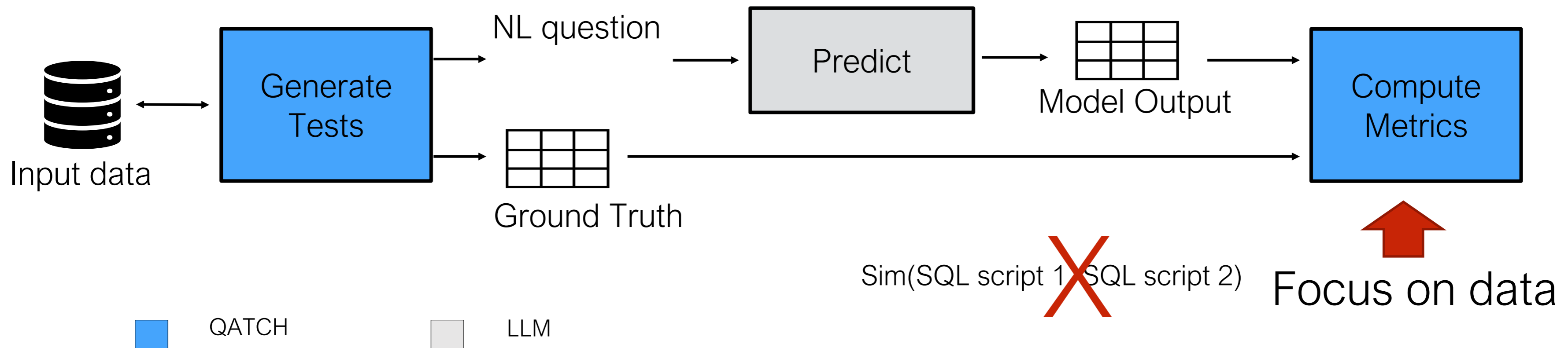

 Input data D

NL question

Ground Truth =
 SQL (input data D)
 

QATCH: Query-Aided TRL Checklist

- 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



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 - all tests, models

Category	Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order	Avg
PROPRIETARY DATA							
ECOMMERCE	TAPAS-WTQ	0.71	0.12	0.53	0.05	0.33	0.35
	TAPEX-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
FINANCE	TAPAS-WTQ	0.72	0.12	0.48	0.05	0.38	0.35
	TAPEX-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
MEDICINE	TAPAS-WTQ	0.72	0.16	0.57	0.09	0.34	0.38
	TAPEX-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
MISCELLANEOUS	TAPAS-WTQ	0.67	0.12	0.34	0.04	0.29	0.29
	TAPEX-WTQ	0.48	0.10	0.25	0.01	0.44	0.26
	OMNITAB	0.30	0.24	0.53	0.23	0.52	0.36
	CHATGPT 3.5	0.76	0.67	0.36	0.16	0.50	0.49
EXISTING BENCHMARK DATA							
Spider	TAPAS-WTQ	0.64	0.42	0.53	0.30	0.64	0.51
	TAPEX-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

Tapas, Tapex,
OmniTab: Fine-tuned
Tabular LMs (TRL)

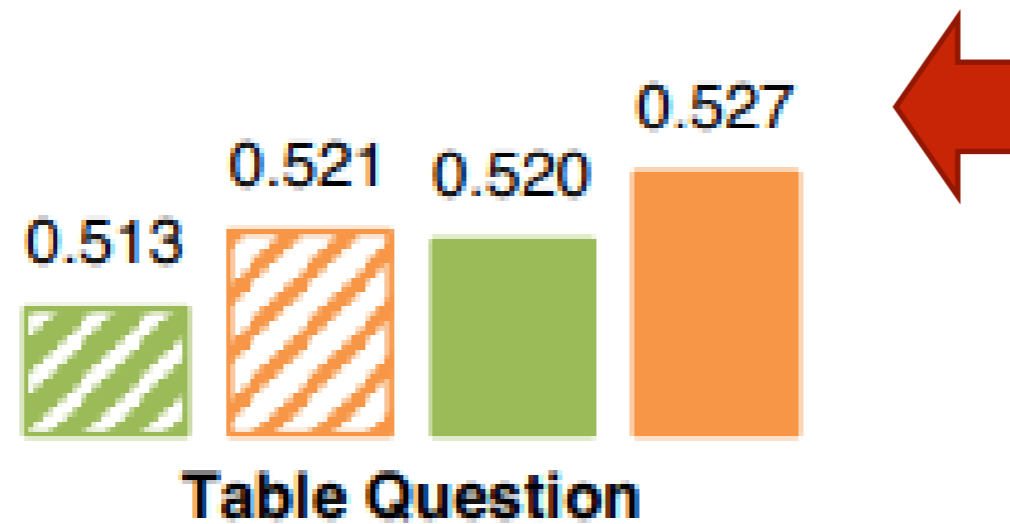
Synthetic examples
effective for **test** on
proprietary data



use them for domain-
specific **fine tuning**
[ongoing]

Fine tuning would fix it?

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



■ ChatGPT Zero-Shot ■ Table-ChatGPT Zero-Shot ■ ChatGPT Few-Shot ■ Table-ChatGPT Few-Shot

Task-name
T-1: Missing-value identification (MV)
T-2: Column-finding (CF)
T-3: Table-QA (TQA) 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 SP
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)

Results for - all tests, models

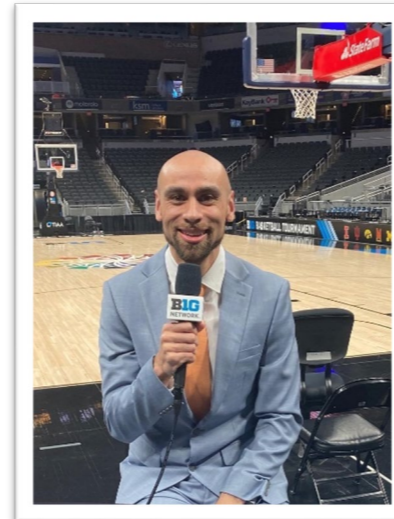
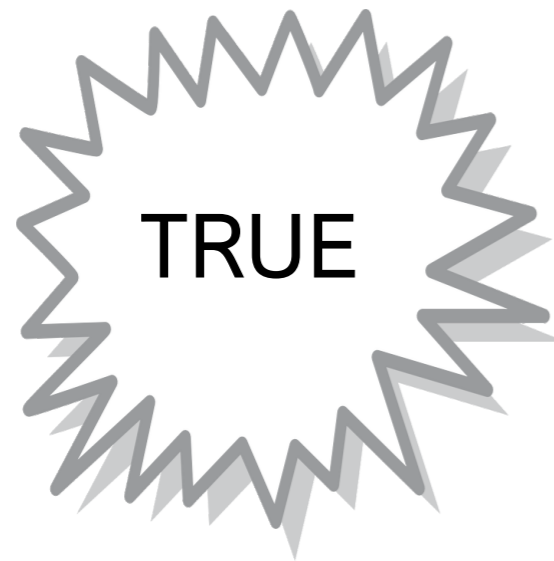
Category	Model	Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order	Avg
PROPRIETARY DATA							
ECOMMERCE	RESDSL	0.91	0.89	0.92	0.81	1.00	0.90
	GAP	0.84	0.80	0.81	0.73	0.97	0.83
	UNIFIEDSKG	0.71	0.71	0.69	0.69	1.00	0.76
	CHATGPT 3.5	0.98	0.98	0.99	0.95	1.00	0.98
FINANCE	RESDSL	0.90	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
MEDICINE	RESDSL	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
MISCELLANEOUS	RESDSL	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
EXISTING BENCHMARK DATA							
Spider DEV	RESDSL	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

Promising results!

With simple text

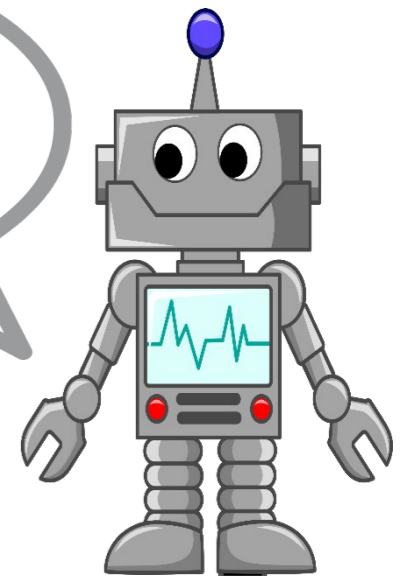
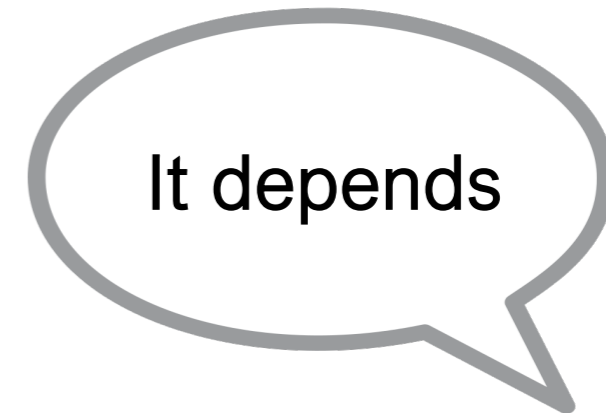
Data-Ambiguous Questions

“Is **Curry** the best **shooter** in NBA?”



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



Results for - 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
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
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
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
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

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

Simple NL text without data ambiguity

SQL and LLMs Vows



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

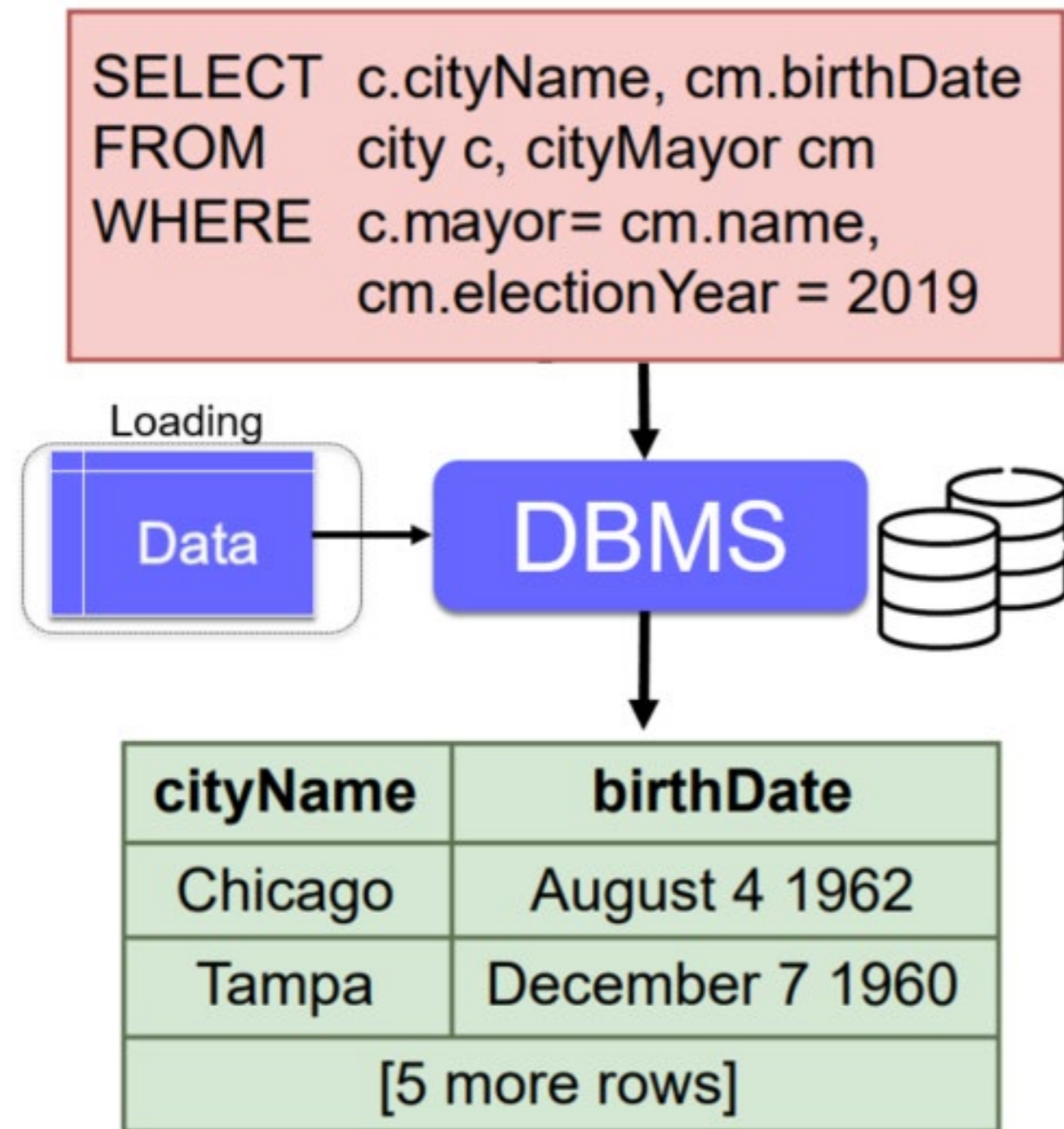


“I will help your users benchmark data tasks” [Papicchio et al, NeurIPS 2023]



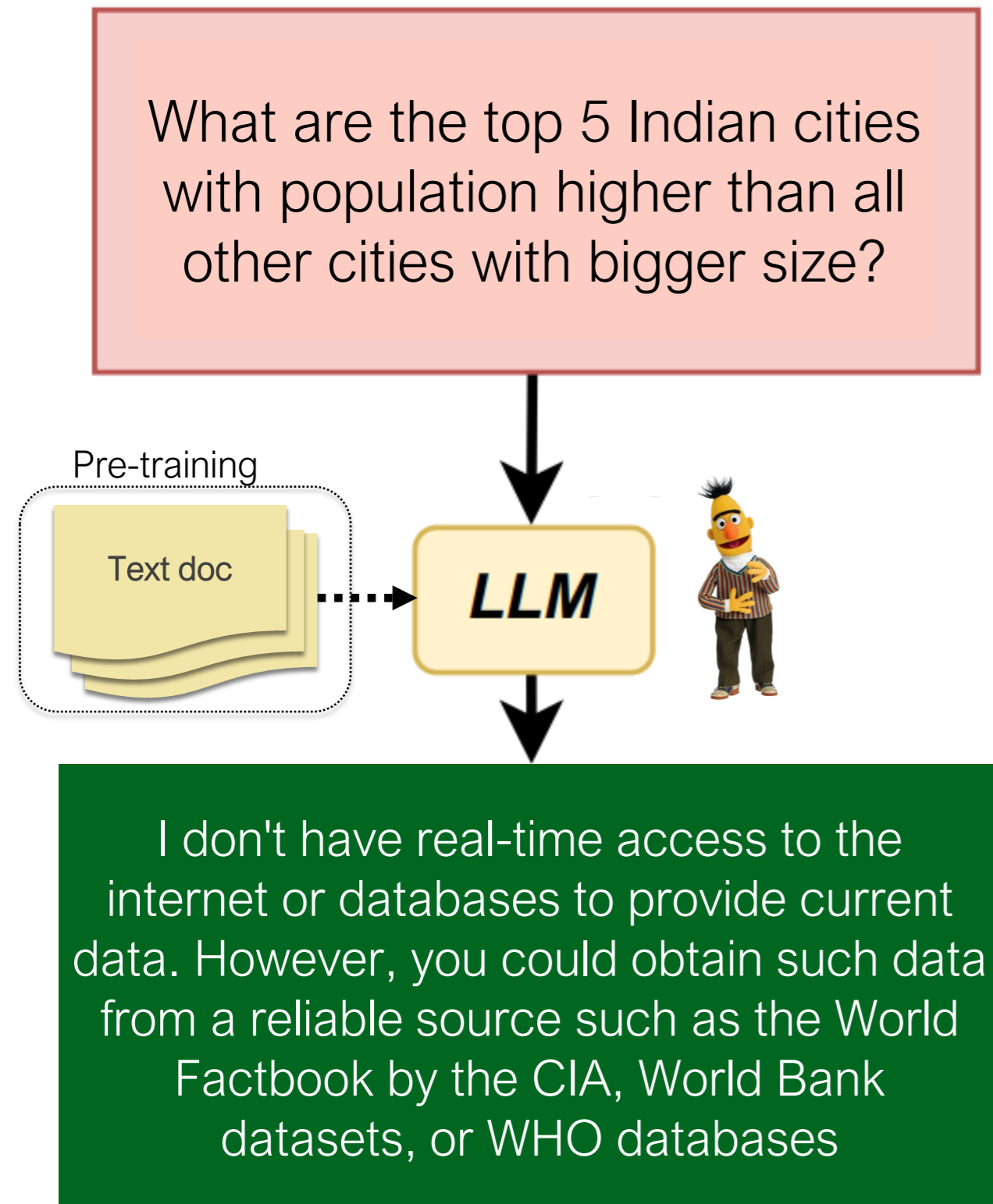
“We will answer queries jointly” [Saeed et al, EDBT 2024]

SQL



- **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



- **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**~~

USER

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

USER

give me population for the 5 biggest indian cities

ASSISTANT

As an AI language model, I don't have access to real-time data databases. However, based on my last training data, here are the approximate populations for the five largest Indian cities:

Based on my last training data, here are the approximate populations for the five largest Indian cities:

1. Mumbai: Around 20 million
2. Delhi: Approximately 16 million
3. Bangalore: Roughly 12 million
4. Hyderabad: Approximately 10 million
5. Ahmedabad: Around 8 million

USER

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

ASSISTANT

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

Documents

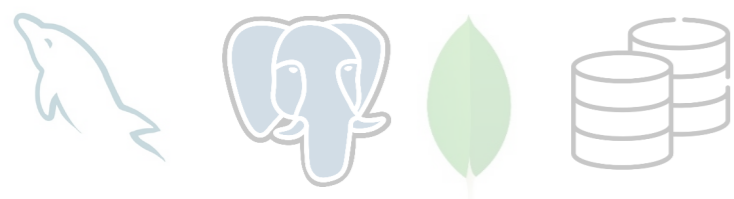
Question answering
(QA)

Query Execution

Relations

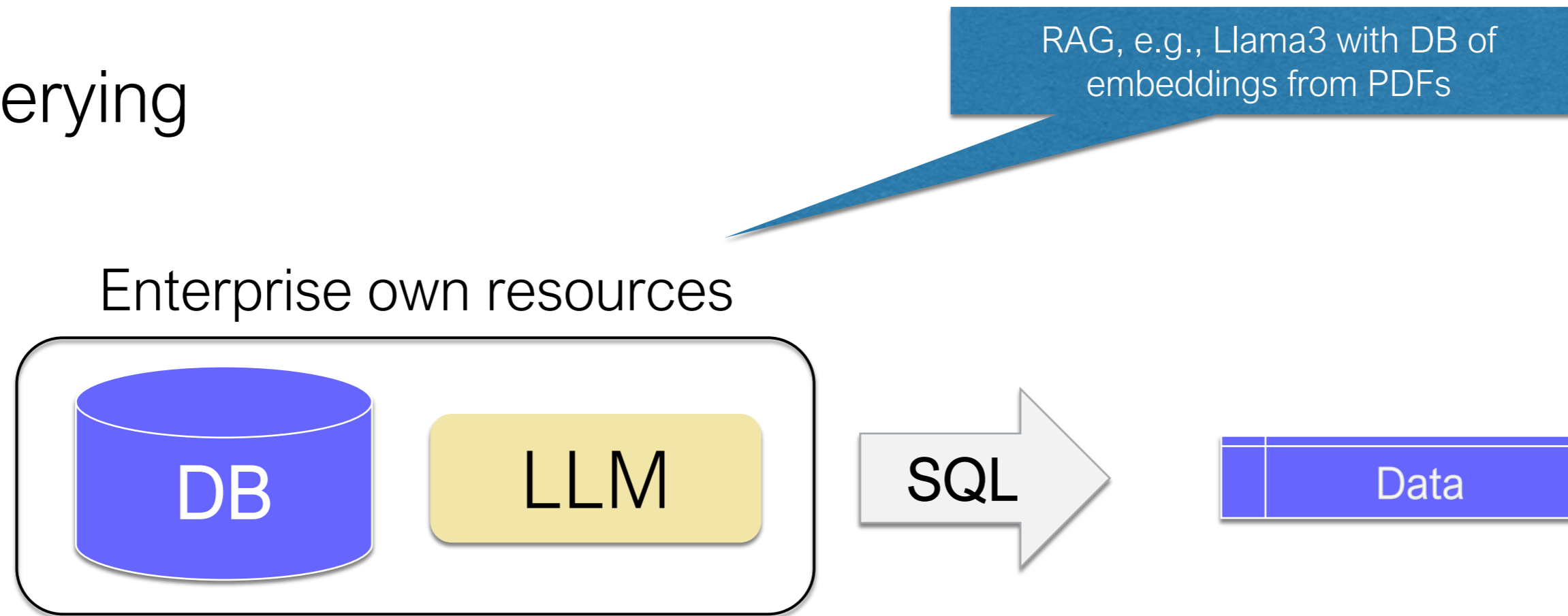
Table QA

Semantic Parsing



Applications

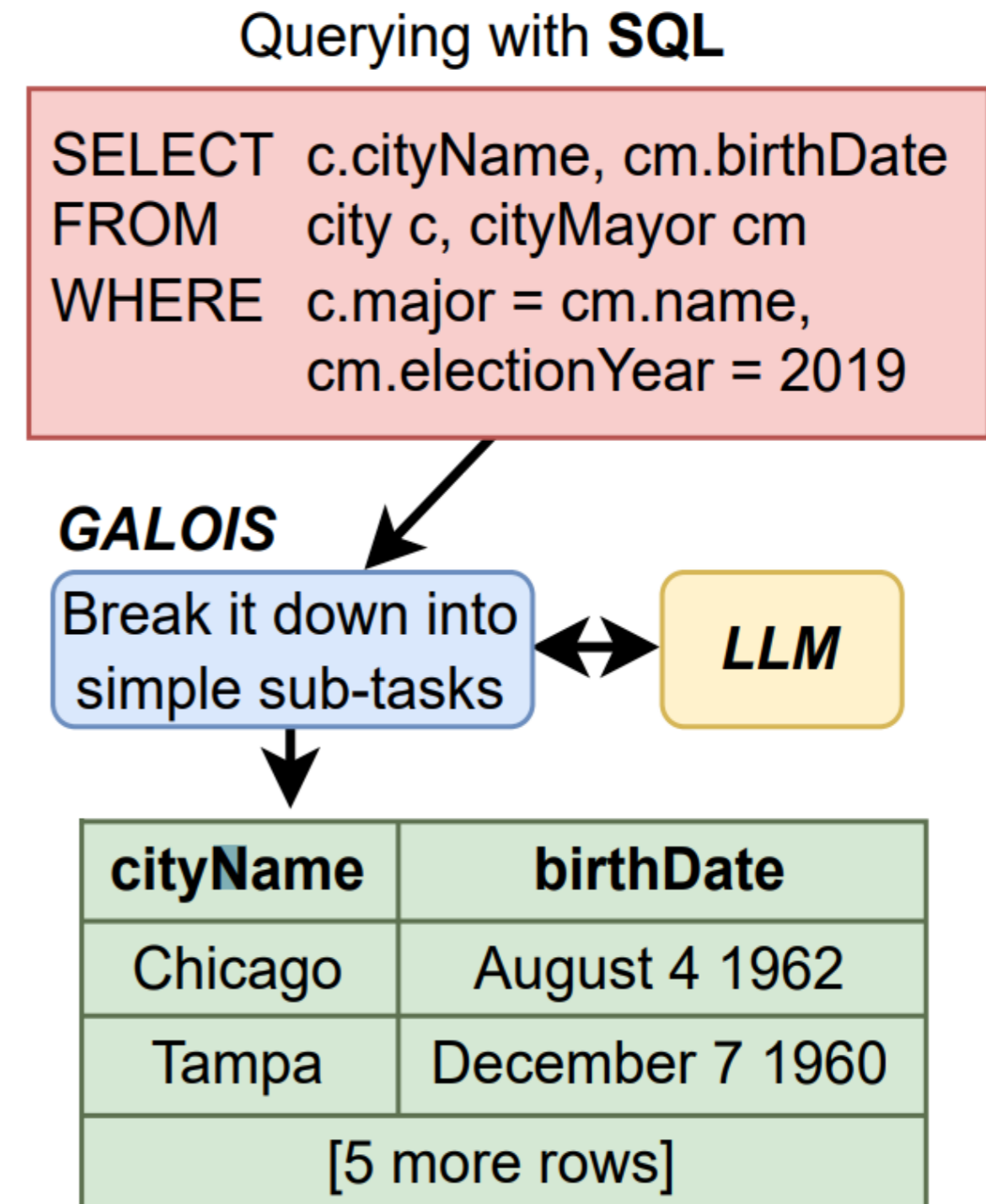
- Hybrid querying



```
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



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
```

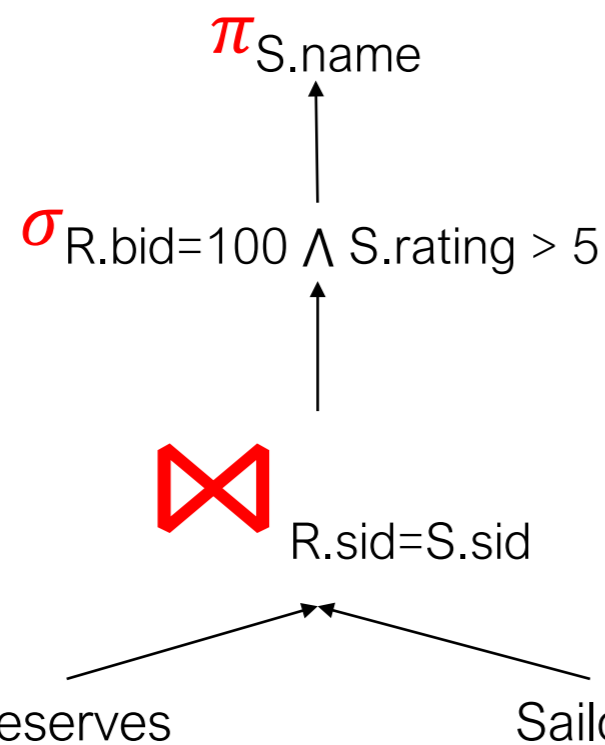
Query Parser

Relational Algebra

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

Equivalent to...

(Logical) Query Plan:



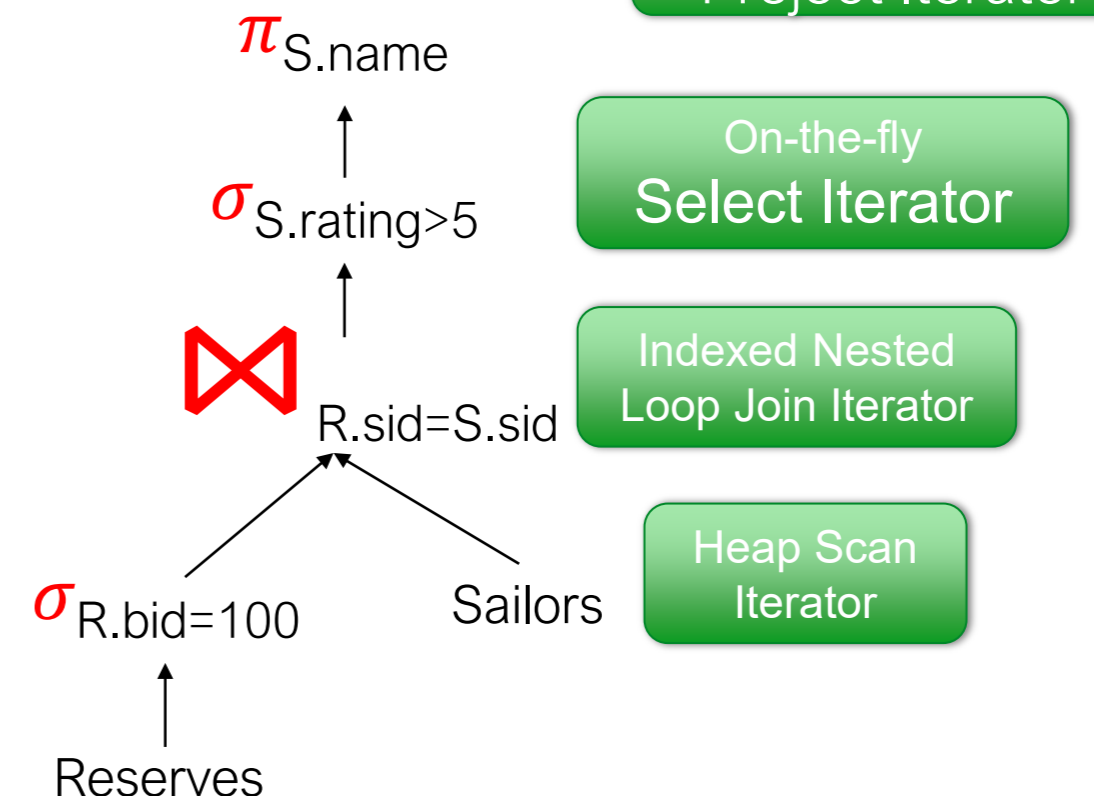
tables by construction

will produce...

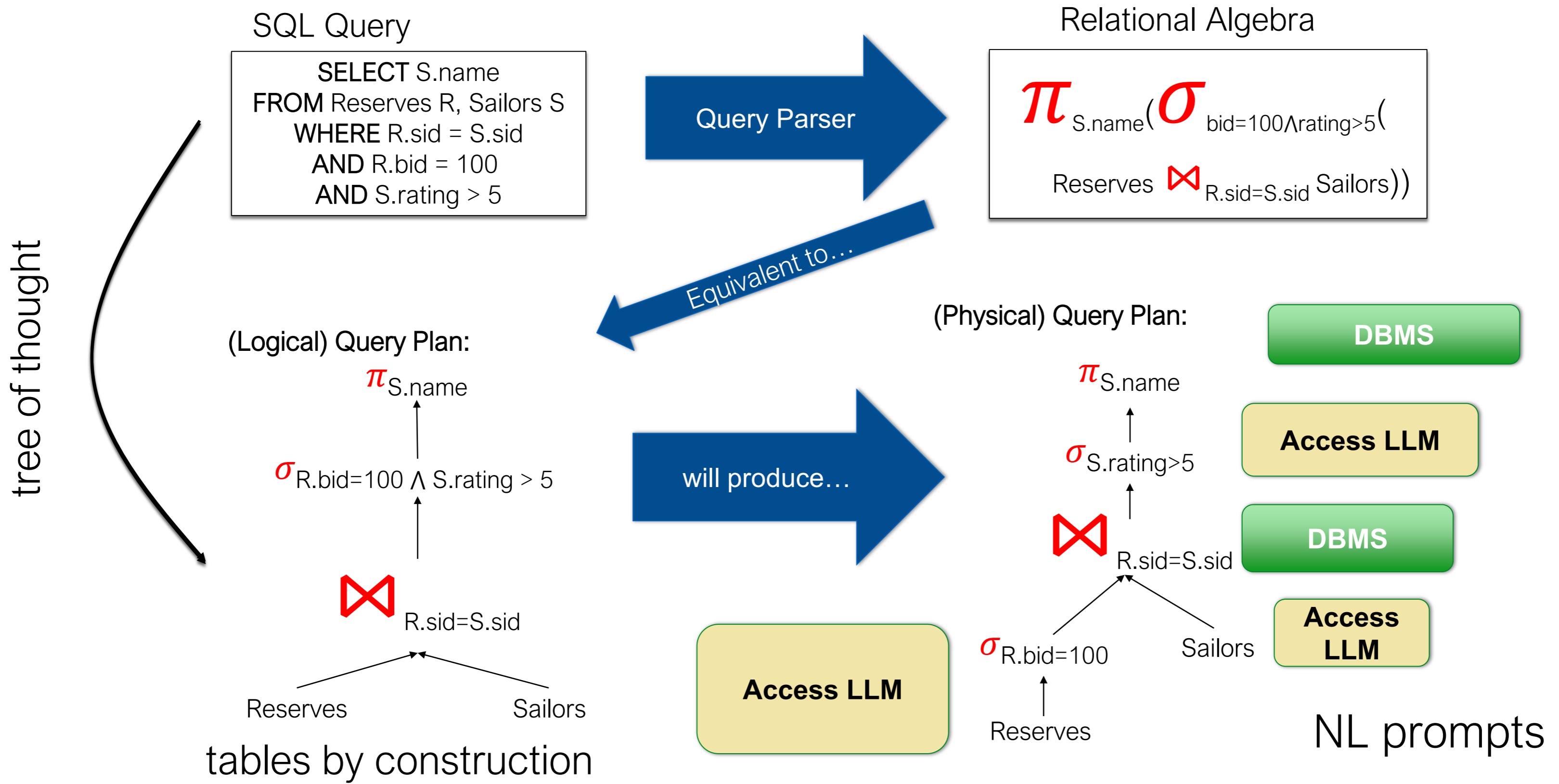
Operator Code

B+-Tree
Indexed Scan
Iterator

(Physical) Query Plan:



Query processing in 1 slide



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': SELECT
FROM
WHERE

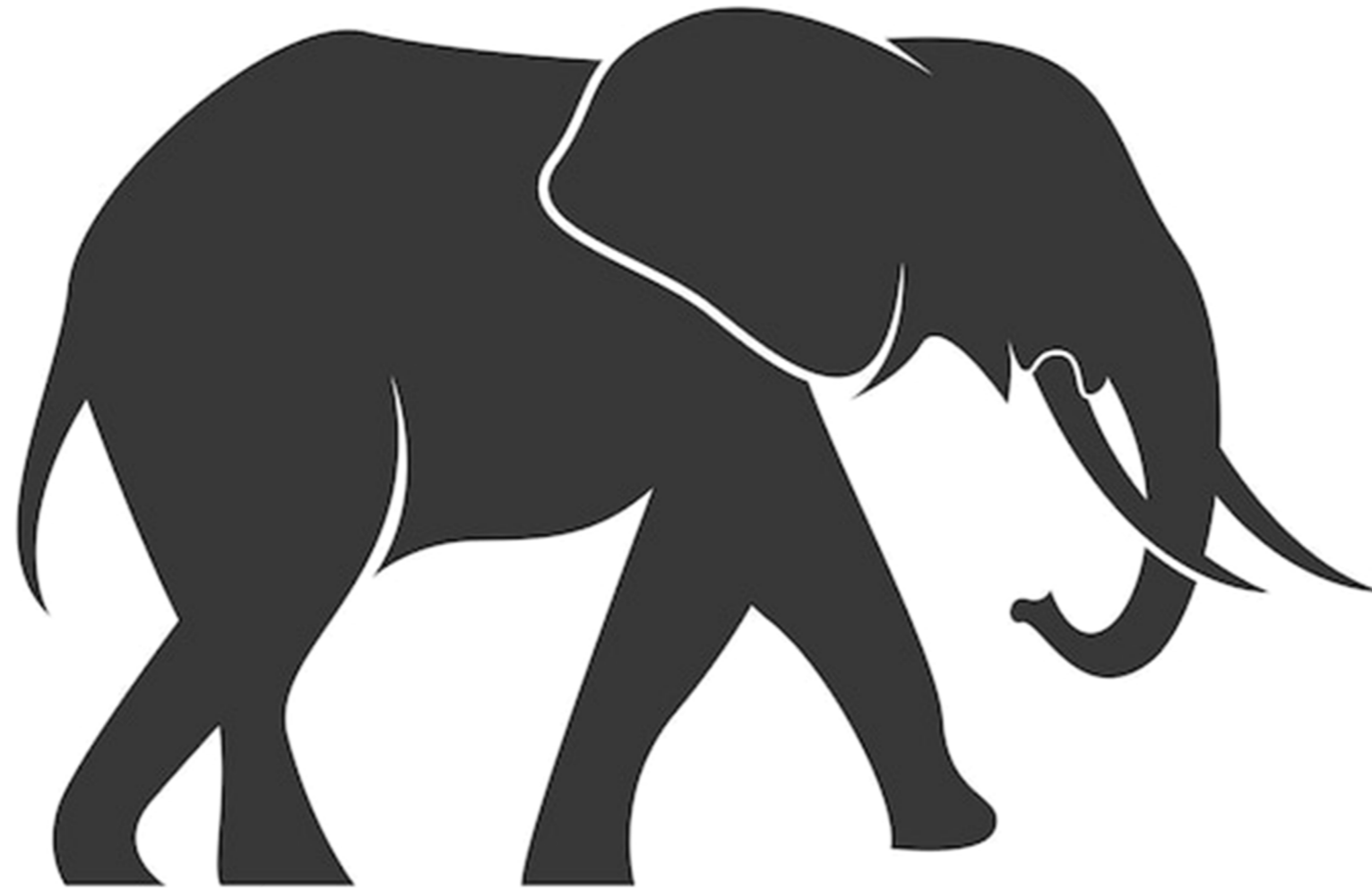
p.

p.

$\forall c' \in C', c'$
“Get current

$\forall c \in C, “H$
more than

Tuples C:



python operator
LM based op.

$\in P'$
currentMayor

“Has
ian $p.name$
less 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

*[“GPT-4 scores 40% higher than GPT-3.5 on our factuality evaluations”](#)

Model	Hallucination Rate
GPT 4	3.0 %
GPT 4 Turbo	3.0 %
Microsoft Orca-2-13b	3.2 %
GPT 3.5 Turbo	3.5 %
Google Gemini Pro	4.8 %

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

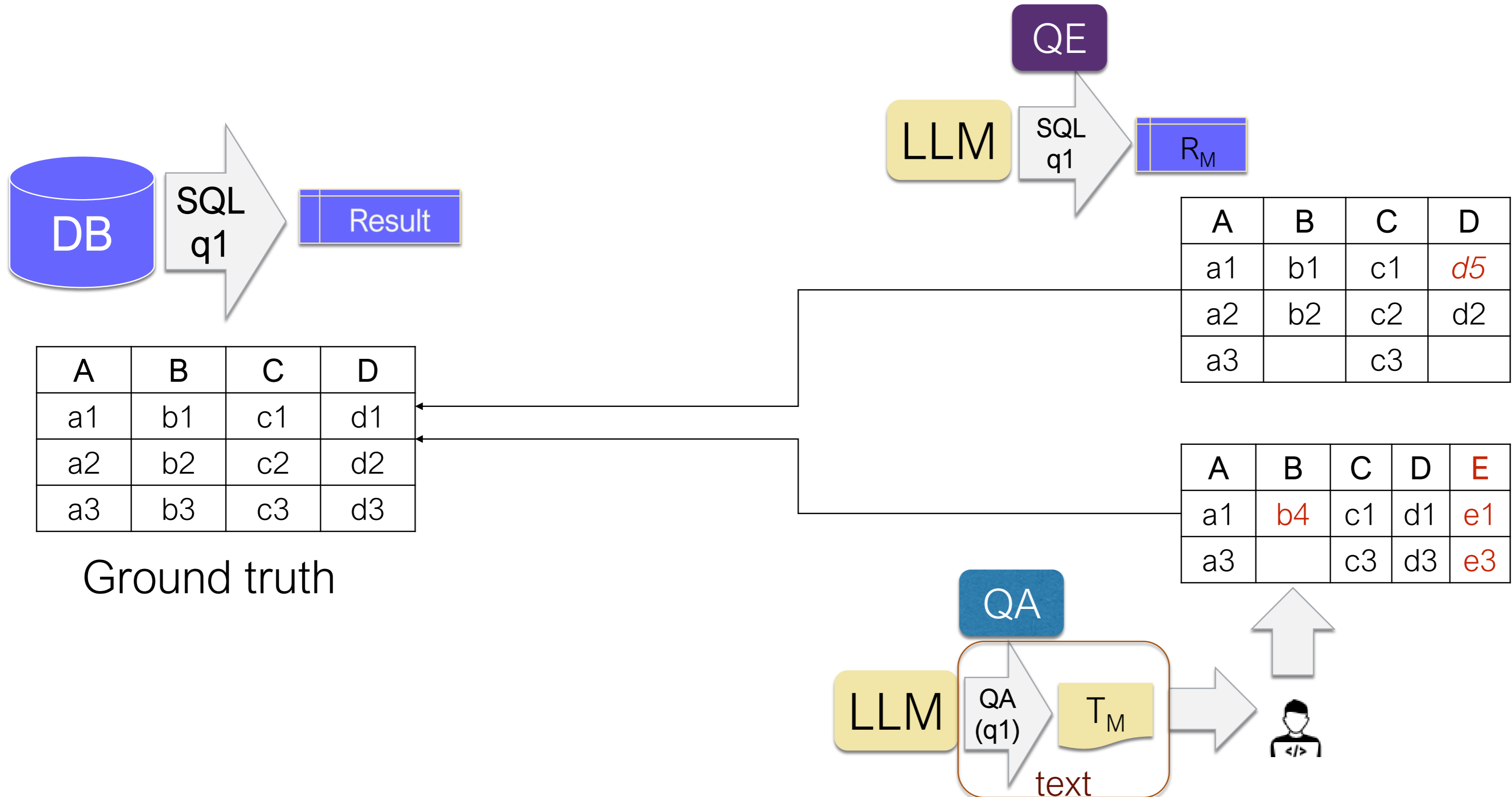
Last updated on April 30th, 2024

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 %

Experiments - data


- Corpus of 46 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 4 LLMs: GPT-3 and ChatGPT better than Flan based

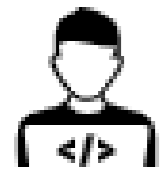
Experiments – QA as “upper bound”



Results ChatGPT

- Similarity in output results between ground truth and

- our method R_M (SQL queries) 



- **manually** parsed traditional T_M (NL questions) 

	<i>All</i>	<i>Selections only</i>	<i>Aggregates</i>
R_M (SQL Queries)	0.50	0.80	0.29
T_M (NL Questions)	0.44	0.71	0.20

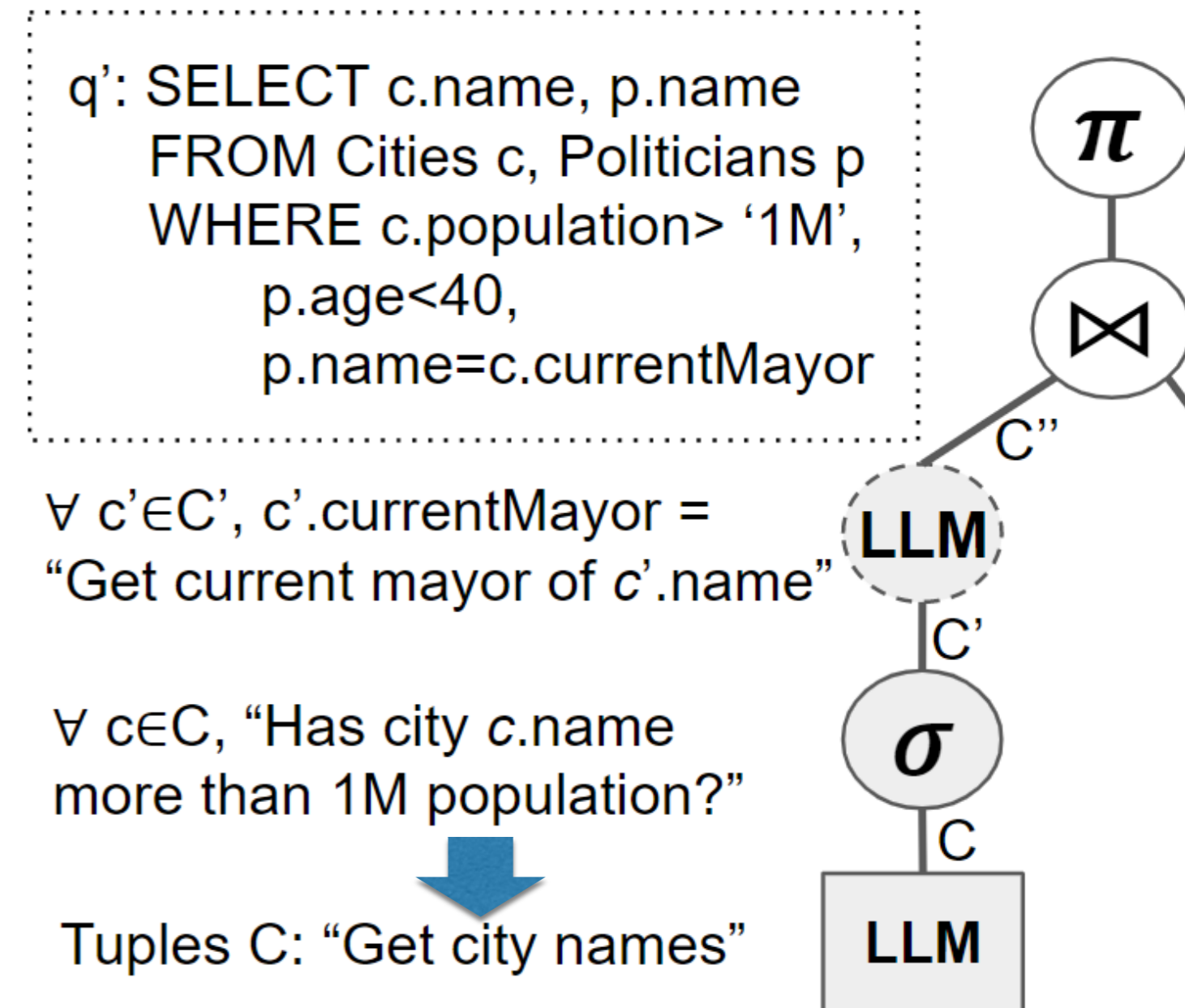
Error analysis

- **Different formats:**
join country code “IT” with “ITA” for entity Italy
- **Entity linking:** “Brussels” vs “Bruxelles”
- **Verbose output:** “The city of Paris”
- ChatGPT trained to output NL text adhering to human preferences

Next Steps

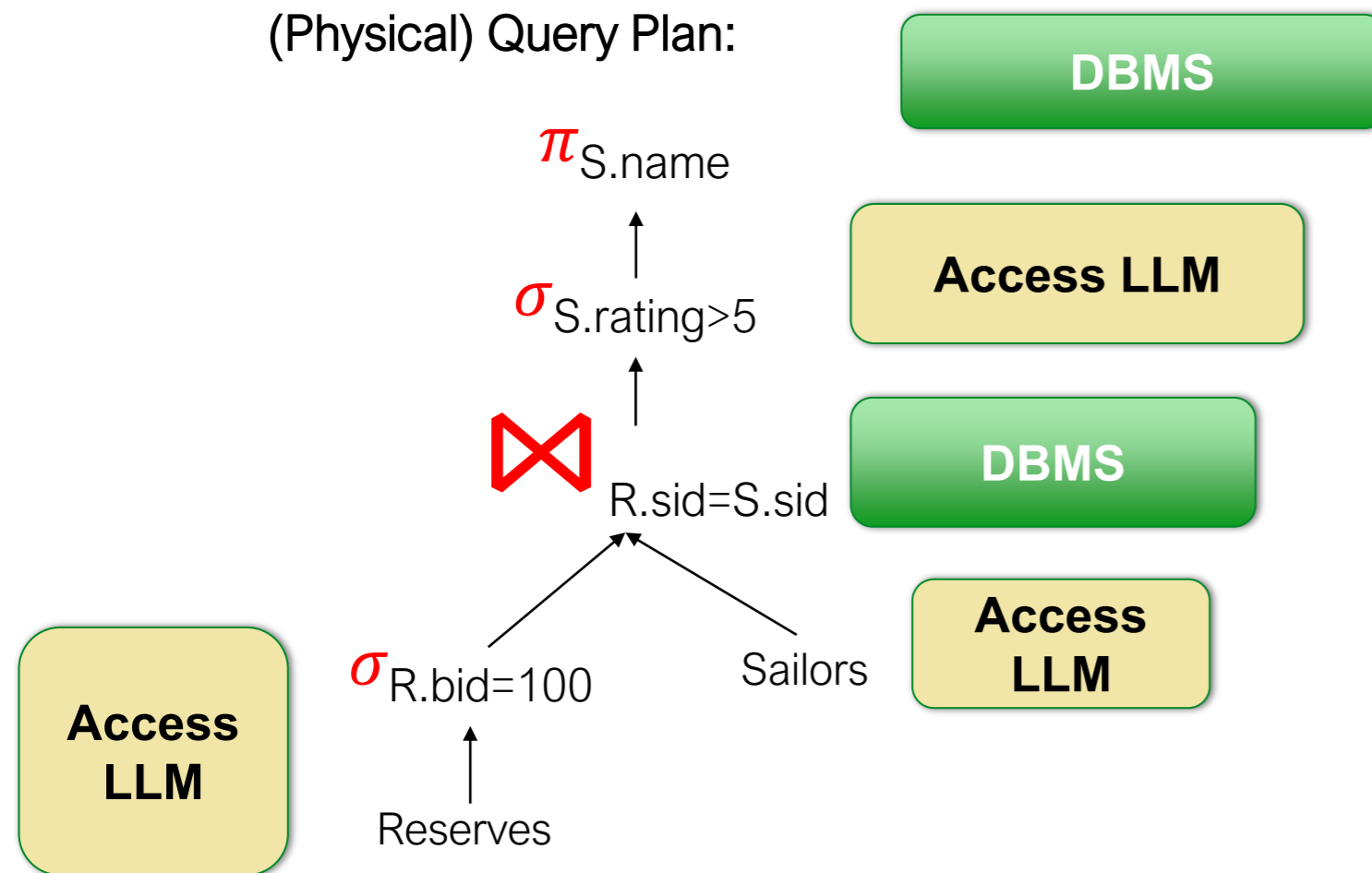
Query optimization

- **Physical:** reduce hallucinations
 - prompts using data examples
 - 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



Open Questions

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



- **LLM first:**
Consuming structured data in pre-training, extensions, fine tuning.... But fine tuned ChatGPT obtains only 0.53 *accuracy* for TQA
[Badaro et al, 2023] [Li et al, 2023]
- **LLMs + Agents?**
SP better results than TQA
→ Use LM for NLU, SQL/code for data operations
[Arora 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

	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:

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 portability, allowing the checks to be used by alternative models. We also introduce

Vision Paper

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.

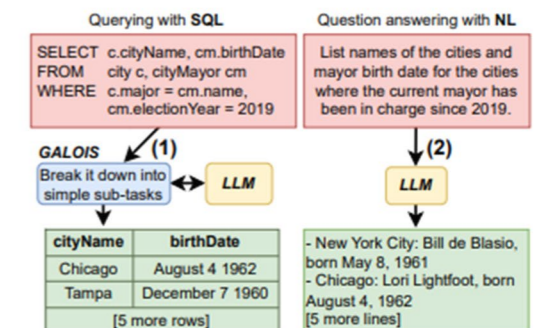


Figure 1: Querying a pre-trained LLM with SQL is different from question answering (QA). We assume a user SQL query as input. GALOIS executes the query, and obtains relations, by retrieving data from a LLM (1). The corresponding QA task consumes and produces natural language text (2).

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

<https://github.com/enzoveltri/pythia>

<https://github.com/spapicchio/QATCH>

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

<http://www.eurecom.fr/~papotti/>
 @paolopapotti

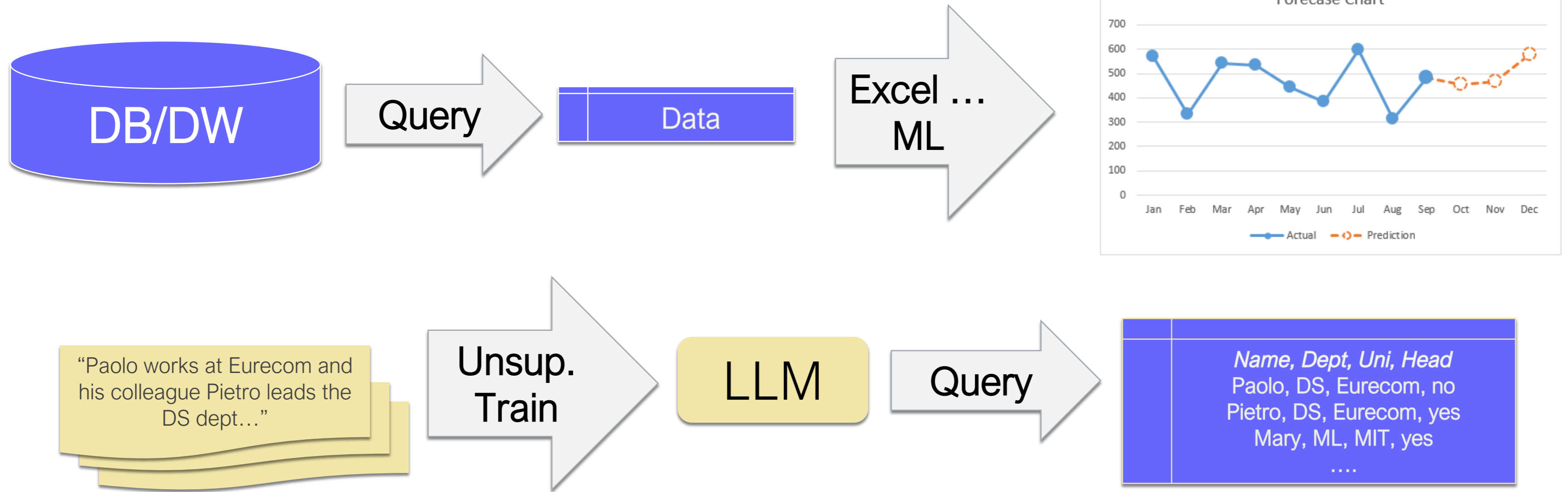
International Workshop on Databases and Machine Learning – 13th May 2024

Solution

- LLMs store factual data, but
 - **Input:** Not trained to execute SQL faithfully
 - use simple **NL prompts** to get data
 - **Engine:** Struggle with complex tasks
 - **chain of thought*** with simple tasks
 - **Output:** Not trained to return relations
 - **tables by construction** as in DBMS

* breaking a problem down into intermediate reasoning steps increases LLM abilities

If only LLMs had SQL powers...



Data applications: we could immediately query text documents!

Evaluate on output data

- 1. Benchmark multiple tasks: QA output is data
- 2. Data comparison enables accurate metrics for SP: execute correct SQL and generated SQL on D, compare data outputs

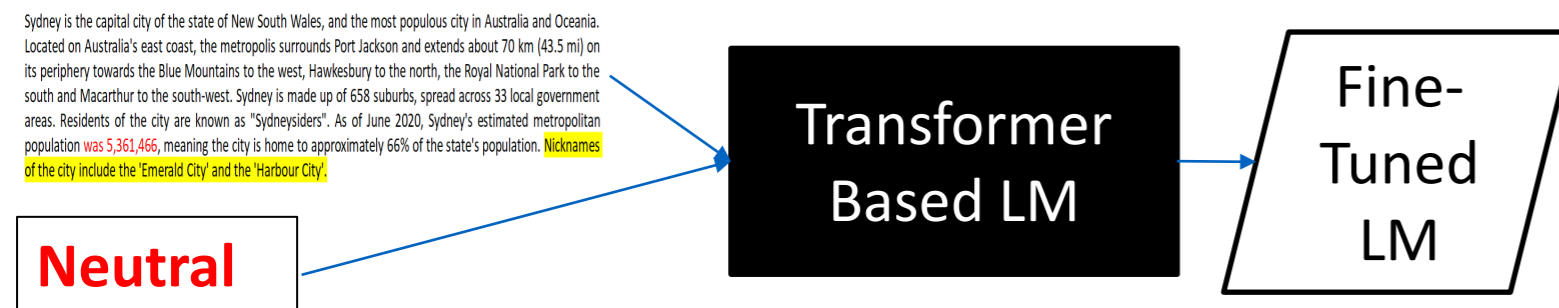
		Cell precision	Cell recall	Tuple cardinality	Tuple constraint	Tuple order
Target	SELECT DISTINCT "emailisfree" FROM "fraud"					
Prediction	SELECT "emailisfree", "income" FROM "fraud"	0.5	1.0	0.2	0.0	-
Target	SELECT "emailisfree" FROM "fraud" ORDERBY ASC					
Prediction	SELECT "emailisfree" FROM "fraud" ORDERBY DESC	1.0	1.0	1.0	1.0	0.0
Target	SELECT * FROM "fraud"					
Prediction	SELECT "emailisfree" FROM "fraud"	1.0	0.10	1.0	0.0	-

How do LLMs work? Big Picture

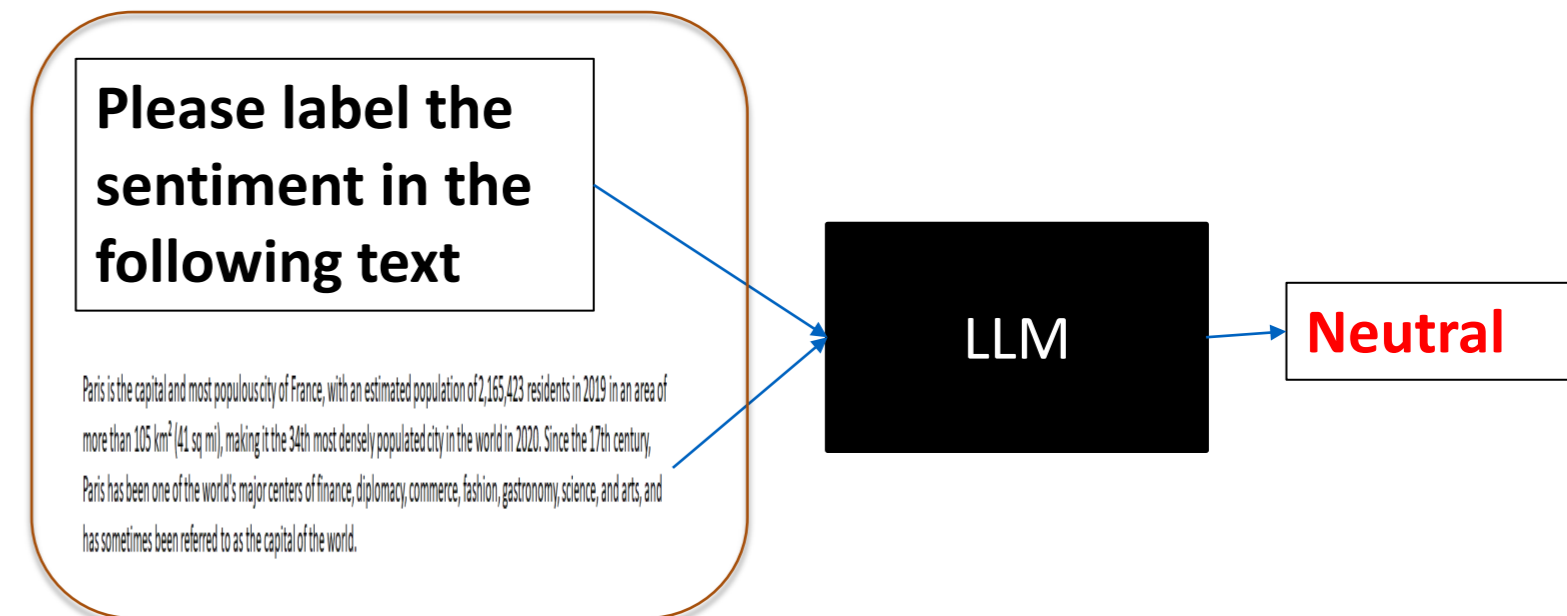
1- Develop LM through *pre-training* using large unlabeled text corpora



2- *Fine-tune* LM using (small) labeled training data for target application

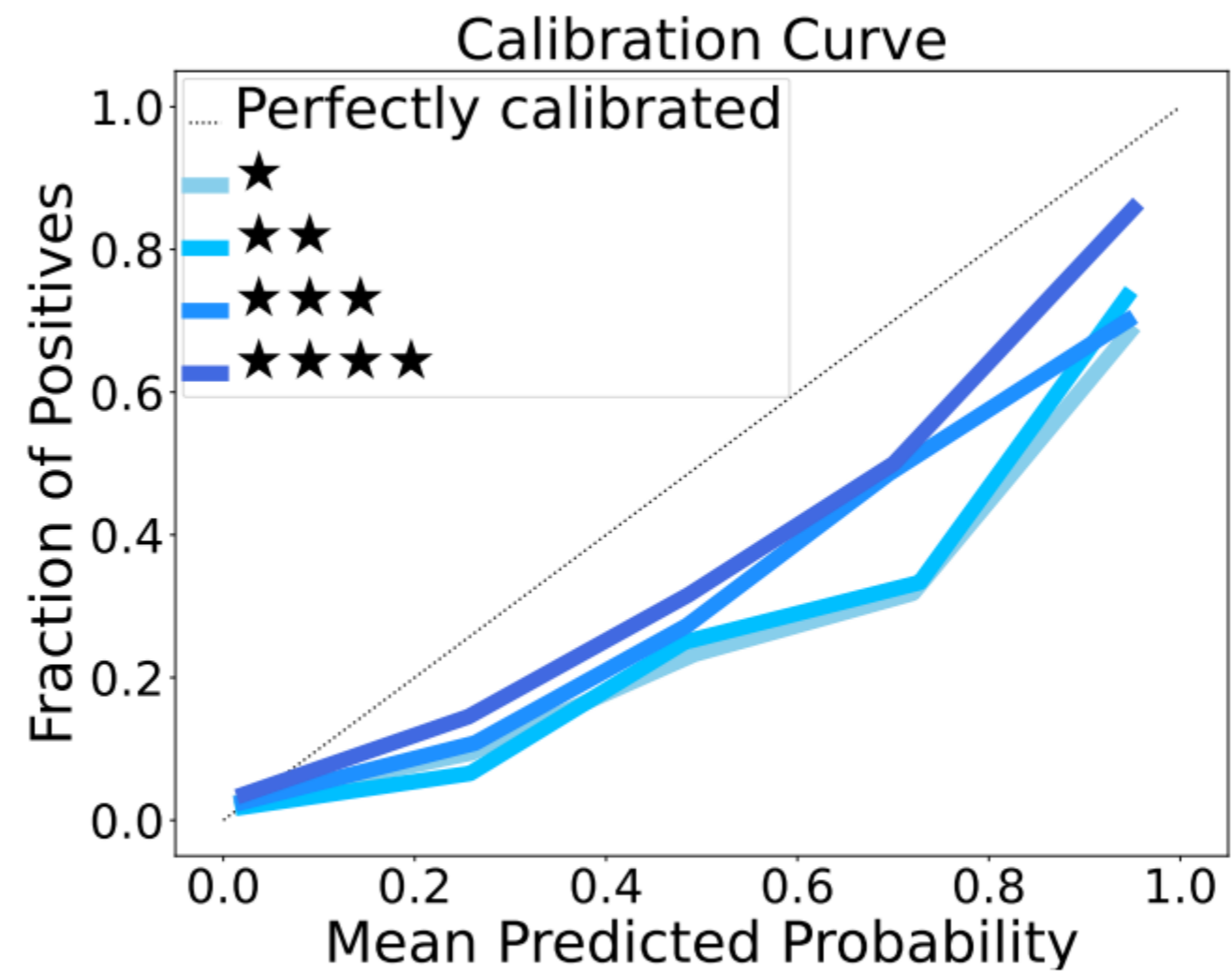


2'- Describe a task (with examples) in the *prompt* of the LLM

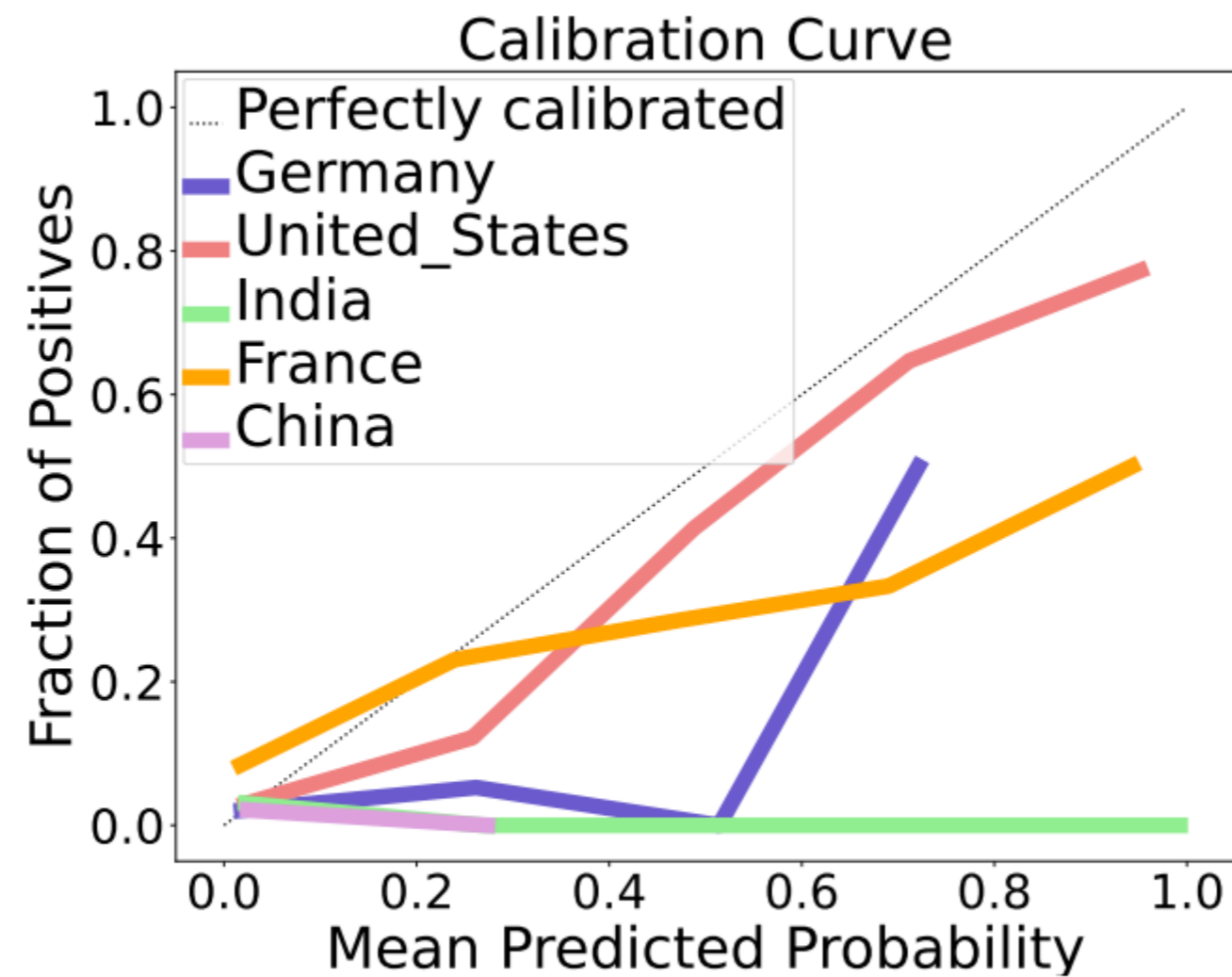


3- Given a new paragraph, predict sentiment





(b) Popularity



(c) Nationality

Background

Query LLM with *SQL queries*. Different from

- **SP**: translate **NL questions** to SQL
- **TQA** on tabular data: querying a relation with **NL questions**
- Neural DBs: textual facts encoded with a transformer and **NL questions** [Thorne et al., 2020] **QA**

References

- Papicchio et al, QATCH: Benchmarking SQL-centric tasks with Table Representation Learning Models on Your Data. NeurIPS 2023
- Saeed et al, Querying Large Language Models with SQL. EDBT 2024
- Saeed and Papotti, You Are My Type! Type Embeddings for Pre-trained Language Models. EMNLP 2022
- Thorne et al, From Natural Language Processing to Neural Databases. Proc. VLDB Endow. 2021
- Veltri et al, Data Ambiguity Profiling for the Generation of Training Examples. ICDE 2023
- Yu et al, Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. EMNLP 2018
- Badaro et al. Transformers for Tabular Data Representation: A Survey of Models and Applications TACL 2023

SQL and LLMs Vows



“I will help your users write SQL queries”



“I will help your users benchmark data tasks”

[Papicchio et al, NeurIPS 2023]



“We will answer queries jointly”

[Saeed et al, EDBT 2024]

