



Natural Language to SQL: State of the Art and Open Problems

Yuyu Luo, Nan Tang
HKUST(GZ)



Guoliang Li
Tsinghua University



Ju Fan
Renmin Univ. of China

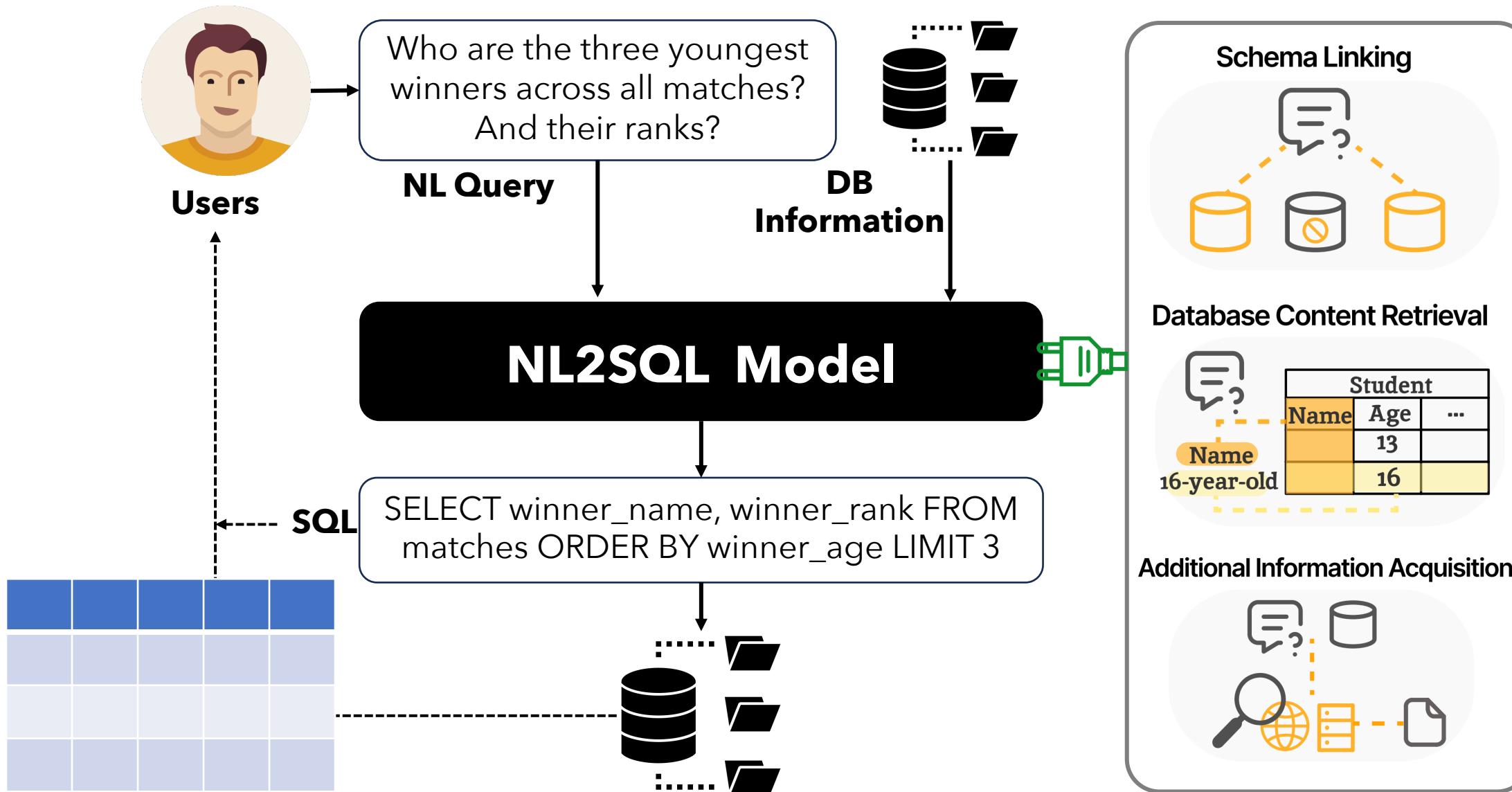


Chengliang Chai
Beijing Institute of Tech.



https://github.com/HKUSTDial/NL2SQL_Handbook
<https://github.com/HKUSTDial/awesome-data-agents>
yuyuluo@hkust-gz.edu.cn

NL2SQL (Text-to-SQL): Bridges Humans and Databases



Task Challenges

C1. Ambiguous NL Query

C2. Requiring Domain Knowledge

C3. Complex Database Schema



Natural Language Query:

Find the **names** of all **customer** who checked out **books** on exactly 3 different **genres** on **Labor Day in 2023**.

C3 Database:

Customer		
CustomerId	Name	...
1	John Doe	...

Book				
BookId	Title	LiteraryGenre	SubjectGenre	...
1	The Great Gatsby	Novel	Magic	...
2	1984	Science Fiction	Dystopian	...
3	To Kill a Mockingbird	Novel	Social Commentary	...

BookOrder			
CustomerId	BookId	OrderDate	...
1	1	2023-05-01	01/05/23
1	2	2023-05-15	15/05/23

- Table Linking
- Columns Linking
- Database Content
- Additional Information
- Foreign Key

Additional Information: Note that **Labor Day stand for May 1**

C2

Task Challenges

C1. Ambiguous NL Query

C2. Requiring Domain Knowledge

C3. Complex Database Schema

C4. Multiple Possible SQL Queries



Natural Language Query:

Find the **names** of all **customer** who checked out **books** on exactly 3 different **genres** on **Labor Day in 2023**.

C3 Database:

Customer		
CustomerId	Name	...
1	John Doe	...

Book			
BookId	Title	LiteraryGenre	SubjectGenre
1	The Great Gatsby	Novel	Mystery
2	1984	Novel	Science Fiction
3	To Kill a Mockingbird	Novel	Classic
4	The Catcher in the Rye	Novel	Young Adult
5	Pride and Prejudice	Novel	Romance
6	The Hobbit	Fantasy	Science Fiction
7	The Lord of the Rings	Fantasy	Science Fiction
8	The Da Vinci Code	Mystery	Religious
9	The Shining	Horror	Psychological Thriller
10	The Silence of the Lambs	Horror	Psychological Thriller
11	The Godfather	Crime	Family Drama
12	The Godfather: Part II	Crime	Family Drama
13	The Godfather: Part III	Crime	Family Drama
14	The Godfather: The Deceit	Crime	Family Drama
15	The Godfather: The Death	Crime	Family Drama
16	The Godfather: The Return	Crime	Family Drama
17	The Godfather: The Final Chapter	Crime	Family Drama
18	The Godfather: The Last Chapter	Crime	Family Drama
19	The Godfather: The Final Chapter	Crime	Family Drama
20	The Godfather: The Last Chapter	Crime	Family Drama
21	The Godfather: The Final Chapter	Crime	Family Drama
22	The Godfather: The Last Chapter	Crime	Family Drama
23	The Godfather: The Final Chapter	Crime	Family Drama
24	The Godfather: The Last Chapter	Crime	Family Drama
25	The Godfather: The Final Chapter	Crime	Family Drama
26	The Godfather: The Last Chapter	Crime	Family Drama
27	The Godfather: The Final Chapter	Crime	Family Drama
28	The Godfather: The Last Chapter	Crime	Family Drama
29	The Godfather: The Final Chapter	Crime	Family Drama
30	The Godfather: The Last Chapter	Crime	Family Drama
31	The Godfather: The Final Chapter	Crime	Family Drama
32	The Godfather: The Last Chapter	Crime	Family Drama
33	The Godfather: The Final Chapter	Crime	Family Drama
34	The Godfather: The Last Chapter	Crime	Family Drama
35	The Godfather: The Final Chapter	Crime	Family Drama
36	The Godfather: The Last Chapter	Crime	Family Drama
37	The Godfather: The Final Chapter	Crime	Family Drama
38	The Godfather: The Last Chapter	Crime	Family Drama
39	The Godfather: The Final Chapter	Crime	Family Drama
40	The Godfather: The Last Chapter	Crime	Family Drama
41	The Godfather: The Final Chapter	Crime	Family Drama
42	The Godfather: The Last Chapter	Crime	Family Drama
43	The Godfather: The Final Chapter	Crime	Family Drama
44	The Godfather: The Last Chapter	Crime	Family Drama
45	The Godfather: The Final Chapter	Crime	Family Drama
46	The Godfather: The Last Chapter	Crime	Family Drama
47	The Godfather: The Final Chapter	Crime	Family Drama
48	The Godfather: The Last Chapter	Crime	Family Drama
49	The Godfather: The Final Chapter	Crime	Family Drama
50	The Godfather: The Last Chapter	Crime	Family Drama
51	The Godfather: The Final Chapter	Crime	Family Drama
52	The Godfather: The Last Chapter	Crime	Family Drama
53	The Godfather: The Final Chapter	Crime	Family Drama
54	The Godfather: The Last Chapter	Crime	Family Drama
55	The Godfather: The Final Chapter	Crime	Family Drama
56	The Godfather: The Last Chapter	Crime	Family Drama
57	The Godfather: The Final Chapter	Crime	Family Drama
58	The Godfather: The Last Chapter	Crime	Family Drama
59	The Godfather: The Final Chapter	Crime	Family Drama
60	The Godfather: The Last Chapter	Crime	Family Drama
61	The Godfather: The Final Chapter	Crime	Family Drama
62	The Godfather: The Last Chapter	Crime	Family Drama
63	The Godfather: The Final Chapter	Crime	Family Drama
64	The Godfather: The Last Chapter	Crime	Family Drama
65	The Godfather: The Final Chapter	Crime	Family Drama
66	The Godfather: The Last Chapter	Crime	Family Drama
67	The Godfather: The Final Chapter	Crime	Family Drama
68	The Godfather: The Last Chapter	Crime	Family Drama
69	The Godfather: The Final Chapter	Crime	Family Drama
70	The Godfather: The Last Chapter	Crime	Family Drama
71	The Godfather: The Final Chapter	Crime	Family Drama
72	The Godfather: The Last Chapter	Crime	Family Drama
73	The Godfather: The Final Chapter	Crime	Family Drama
74	The Godfather: The Last Chapter	Crime	Family Drama
75	The Godfather: The Final Chapter	Crime	Family Drama
76	The Godfather: The Last Chapter	Crime	Family Drama
77	The Godfather: The Final Chapter	Crime	Family Drama
78	The Godfather: The Last Chapter	Crime	Family Drama
79	The Godfather: The Final Chapter	Crime	Family Drama
80	The Godfather: The Last Chapter	Crime	Family Drama
81	The Godfather: The Final Chapter	Crime	Family Drama
82	The Godfather: The Last Chapter	Crime	Family Drama
83	The Godfather: The Final Chapter	Crime	Family Drama
84	The Godfather: The Last Chapter	Crime	Family Drama
85	The Godfather: The Final Chapter	Crime	Family Drama
86	The Godfather: The Last Chapter	Crime	Family Drama
87	The Godfather: The Final Chapter	Crime	Family Drama
88	The Godfather: The Last Chapter	Crime	Family Drama
89	The Godfather: The Final Chapter	Crime	Family Drama
90	The Godfather: The Last Chapter	Crime	Family Drama
91	The Godfather: The Final Chapter	Crime	Family Drama
92	The Godfather: The Last Chapter	Crime	Family Drama
93	The Godfather: The Final Chapter	Crime	Family Drama
94	The Godfather: The Last Chapter	Crime	Family Drama
95	The Godfather: The Final Chapter	Crime	Family Drama
96	The Godfather: The Last Chapter	Crime	Family Drama
97	The Godfather: The Final Chapter	Crime	Family Drama
98	The Godfather: The Last Chapter	Crime	Family Drama
99	The Godfather: The Final Chapter	Crime	Family Drama
100	The Godfather: The Last Chapter	Crime	Family Drama

BookOrder			
CustomerId	BookId	OrderDate	...
1	1	01/05/23	...
2	2	01/05/23	...

Additional Information: Note that **Labor Day stand for May 1**

C2

C1 SQL:

```
SELECT Name
FROM Customer
NATURAL JOIN BookOrder NATURAL JOIN Book
WHERE OrderDate = '01/05/23'
GROUP BY CustomerId, Name
HAVING COUNT(DISTINCT SubjectGenre) = 3;
```

LiteraryGenre

C1

C2 SELECT Name

```
FROM Customer
WHERE CustomerId = (SELECT CustomerId
FROM BookOrder NATURAL JOIN Book
WHERE OrderDate = '01/05/23'
GROUP BY CustomerId
HAVING COUNT(DISTINCT SubjectGenre)=3);
```

LiteraryGenre

Task Challenges



Natural Language Query:

Find the **names** of all **customer** who checked out **books** on exactly 3 different **genres** on **Labor Day in 2023**.

C1. Ambiguous NL Query

C2. Requiring Domain Knowledge

C3. Complex Database Schema

C4. Multiple Possible SQL Queries

C5. Database Schema Dependency

C6. Database Domain Adaption

Account			
AcId	AccName	Password	CustomerId

Book			
BookId	Title	LiteraryGenre	SubjectGenre
		Novel	Magic
	

Customer		
CustomerId	CName	...

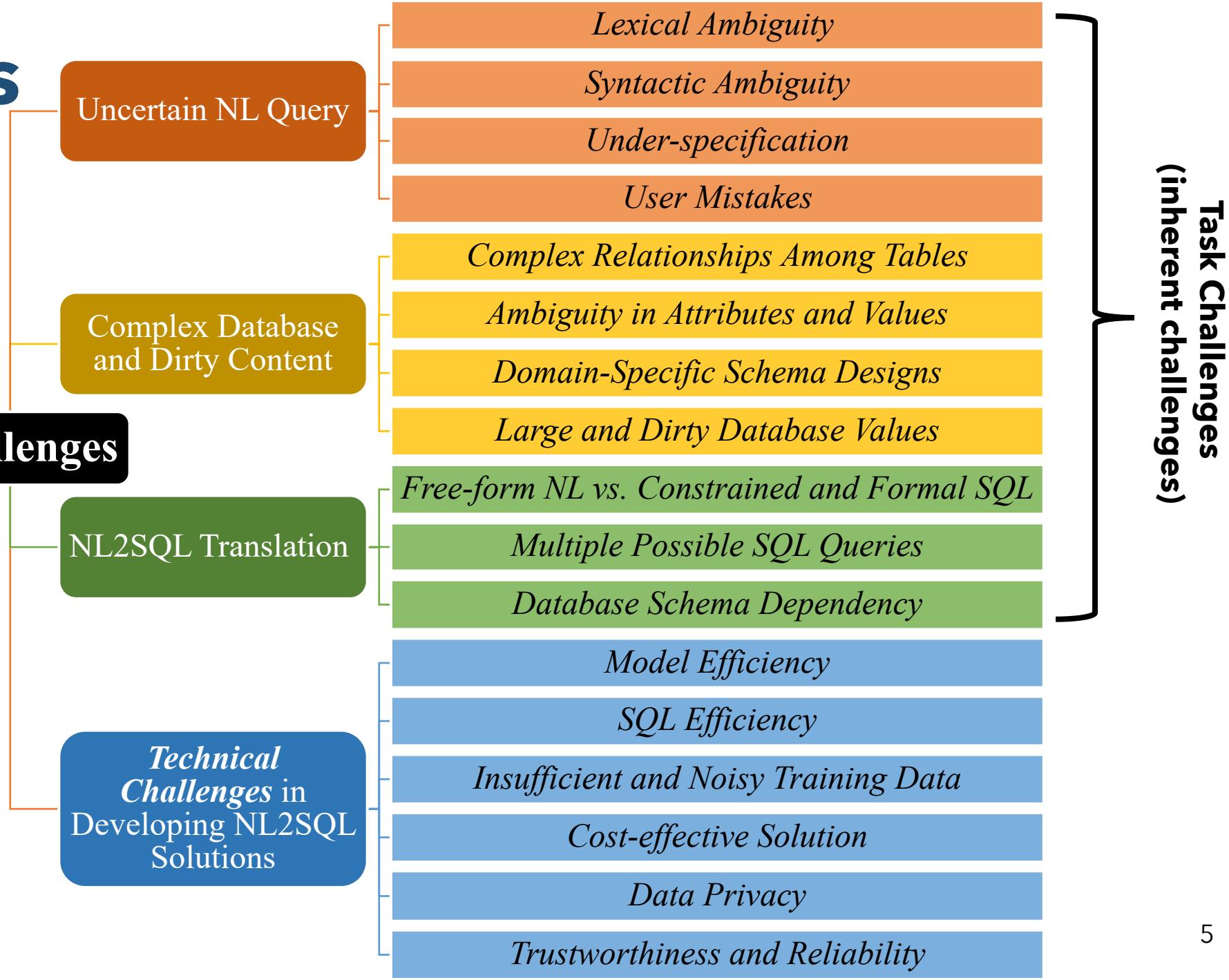
BookOrder		
BookId	AcId	...

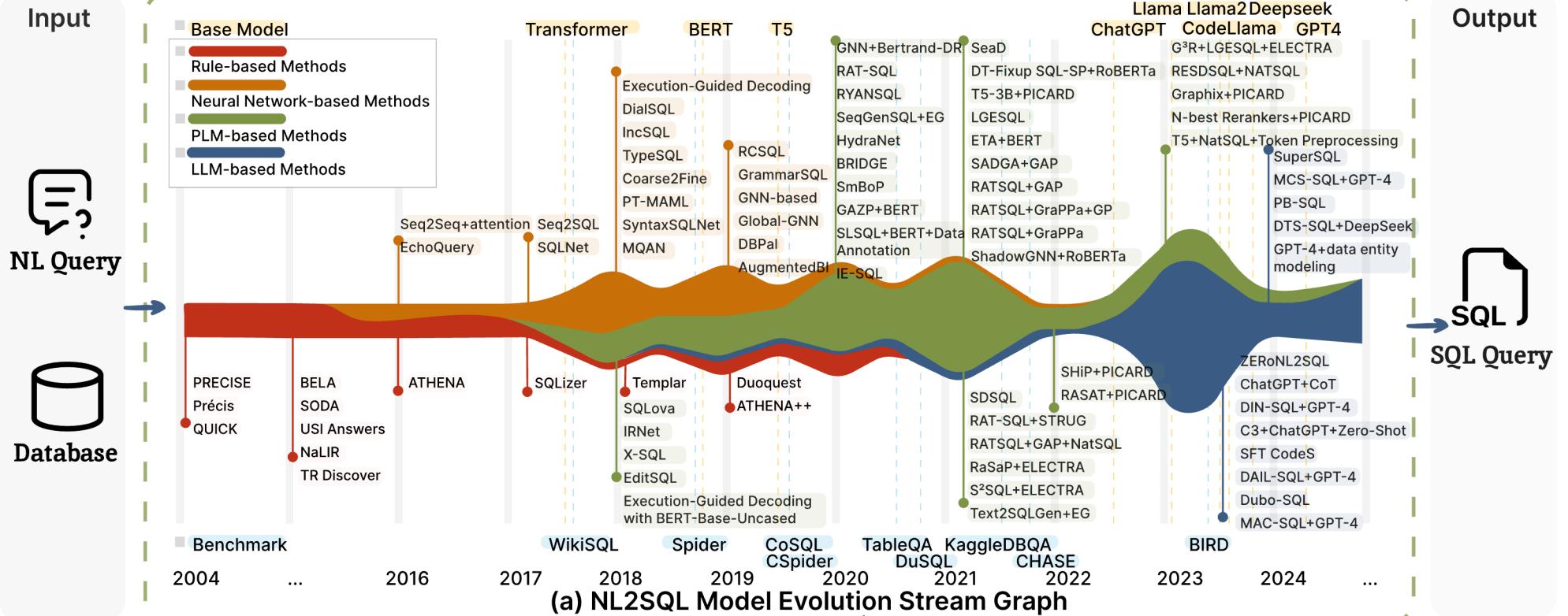
Additional Information: Note that **Labor Day** stand for May 1

```
SELECT CName
FROM Customer
NATURAL JOIN Account
NATURAL JOIN BookOrder
NATURAL JOIN Book
WHERE OrderDate = 'May 1st 2023'
GROUP BY CustomerId, CName
HAVING COUNT(DISTINCT SubjectGenre) = 3;
```

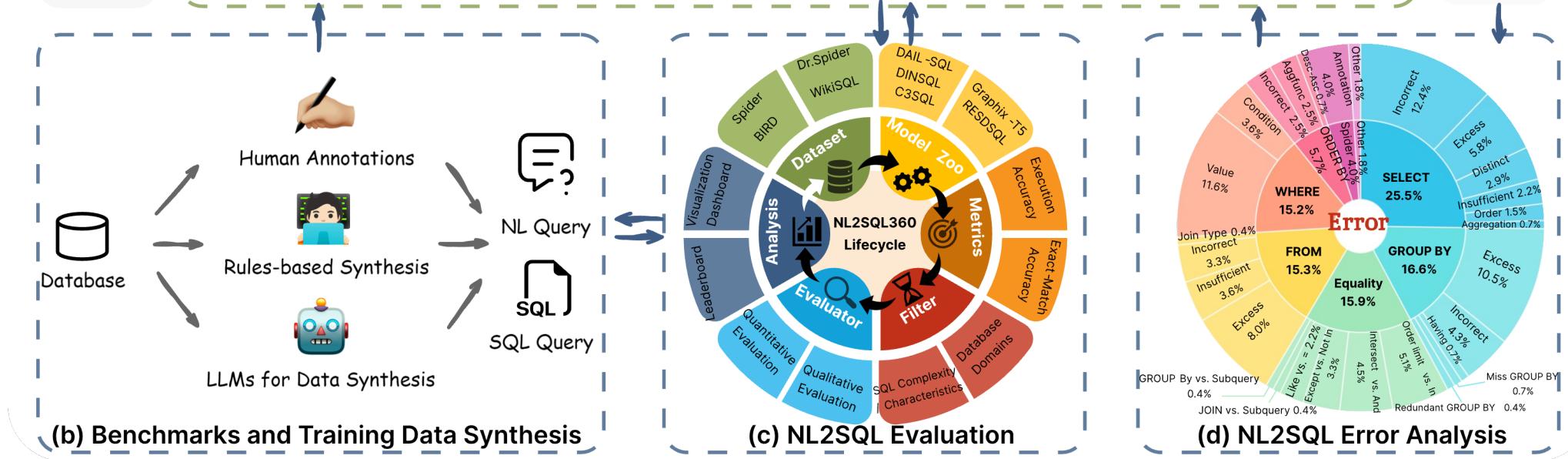
Challenges

NL2SQL Challenges

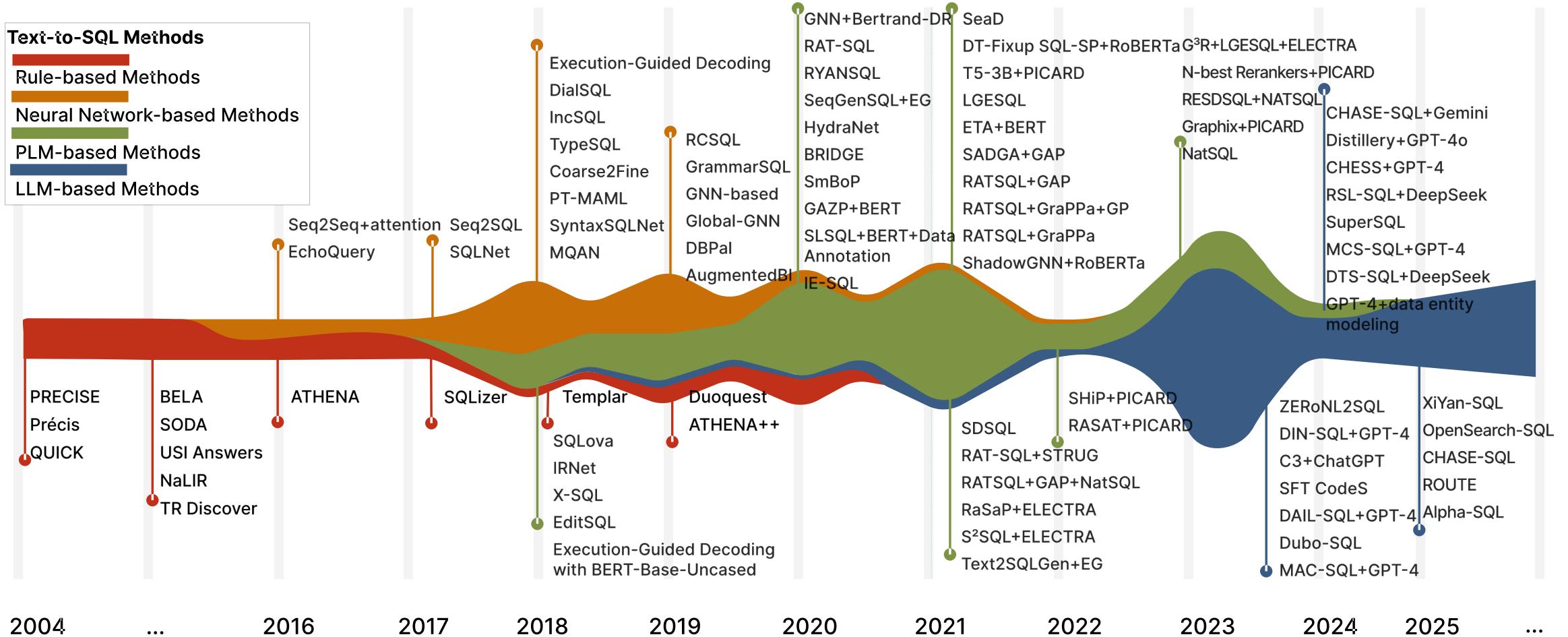




(a) NL2SQL Model Evolution Stream Graph



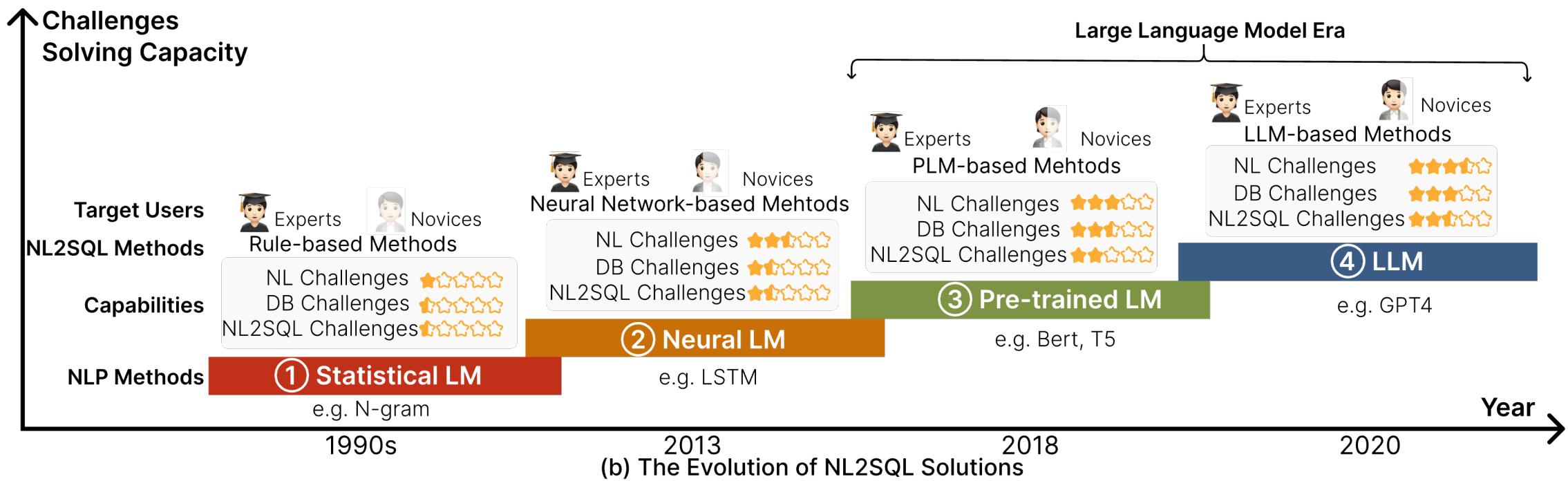
Where Are We?



Where Are We?

Type	Level	★	★★	★★★	★★★★	★★★★★
NL Challenges	Token-level Recognition	Synonym Recognition	Semantic Understanding	Domain Knowledge Query	Multi-turn Dialogues	Recognition
DB Challenges	Single-table Queries	Simple Multiple Tables	Multiple Tables with Complex Schema	Massive Tables and Values	Real-world Databases	
NL2SQL Challenges	Single-table SQL	Multi-table SQL	Advanced SQL Feature Support	Adapting to Changed Schema	Efficient SQL Generation	

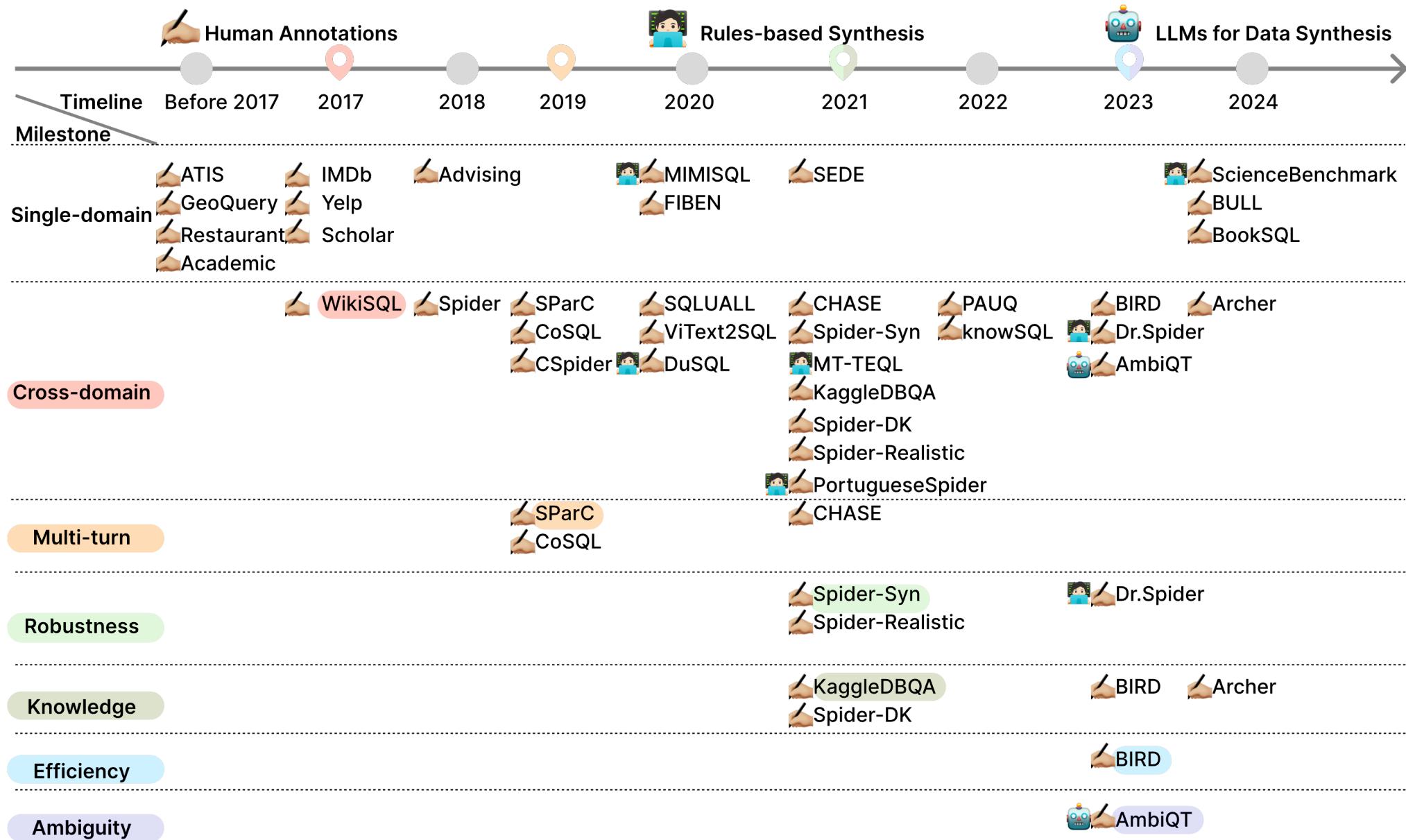
(a) The Definition of Challenges Levels



(b) The Evolution of NL2SQL Solutions

Figure: The Evolution of NL2SQL Solutions from the Perspective of Language Models.

An Overview of NL2SQL Benchmarks



NL2SQL Benchmark Discussion & Insights

- **From the Redundancy Measure perspective**

- We observe a trend from early datasets to recent ones where datasets have grown in size, including increases in the number of questions and unique queries.

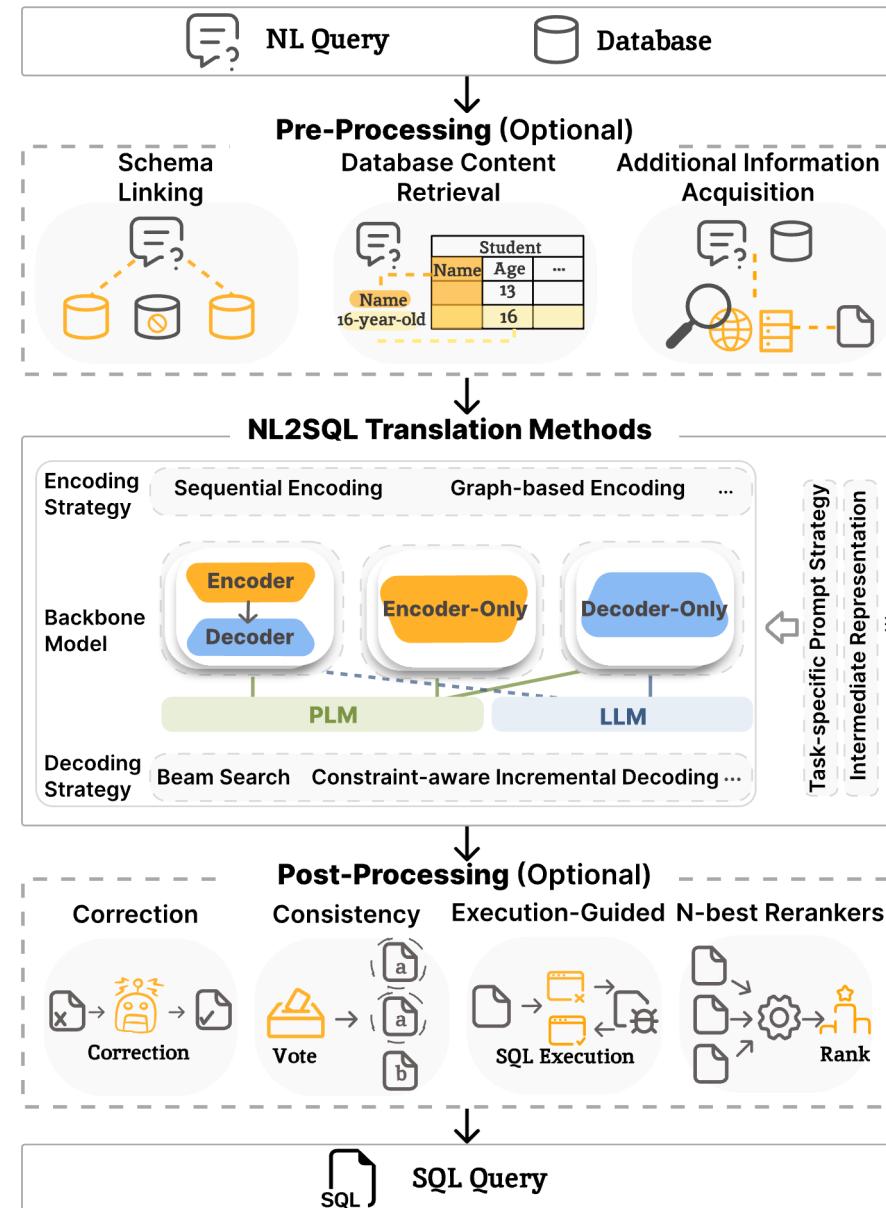
- **From the Database Complexity perspective**

- The number of databases (and tables) in datasets correlates with the tasks (e.g., Single-domain vs. Robustness) they serve.

- **From the Query Complexity perspective**

- Recent datasets show a growing emphasis on Scalar Functions and Mathematical Computations in SQL queries, which introduces challenges in SQL generation structure not seen in earlier datasets.

Tutorial Roadmap



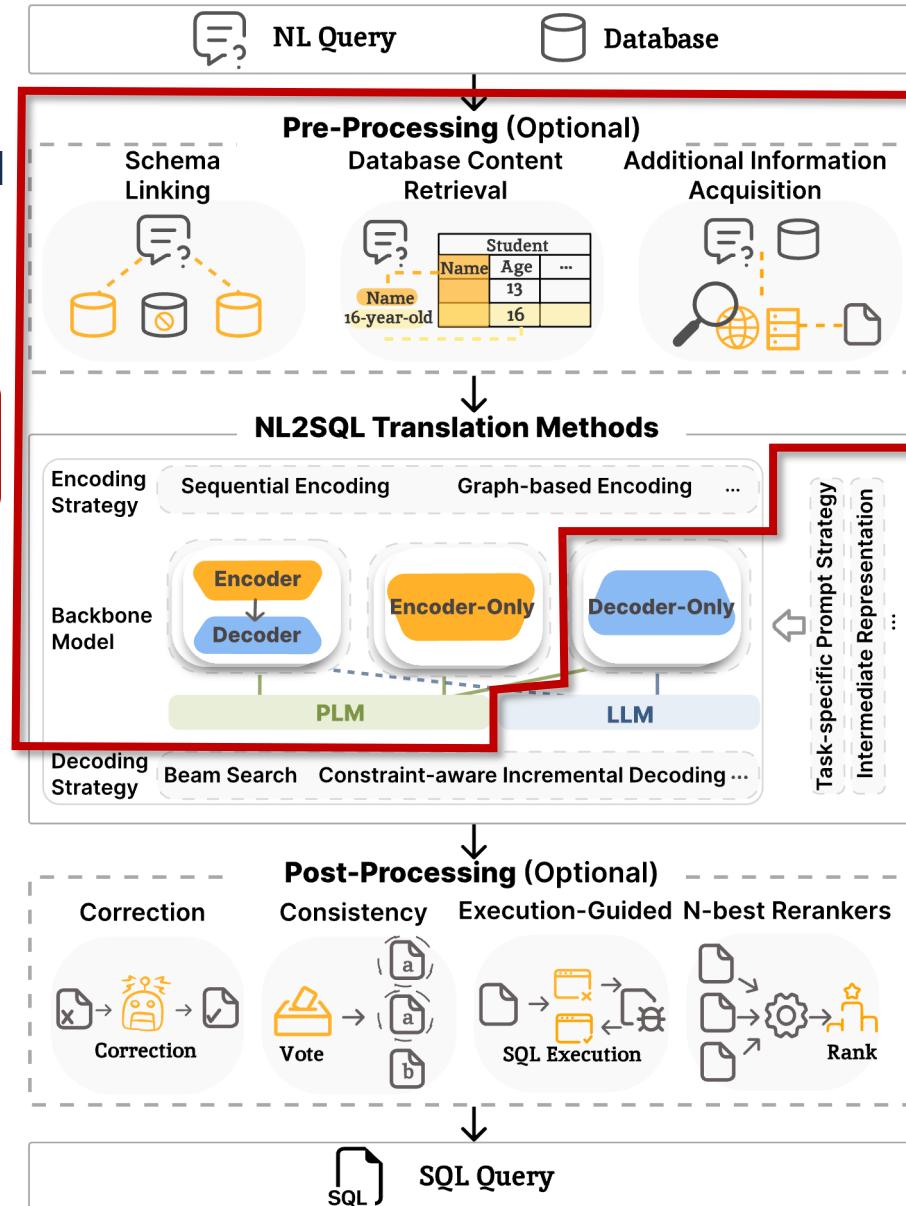
Tutorial Roadmap

NL2SQL Solutions with PLMs and LLMs

Q1: How to design prompts and train PLMs/LLMs for NL2SQL?

- Prompt Settings: Few-shot/Zero-shot
- Training: SFT / RL

Q2: How effective are the core pre-processing techniques?



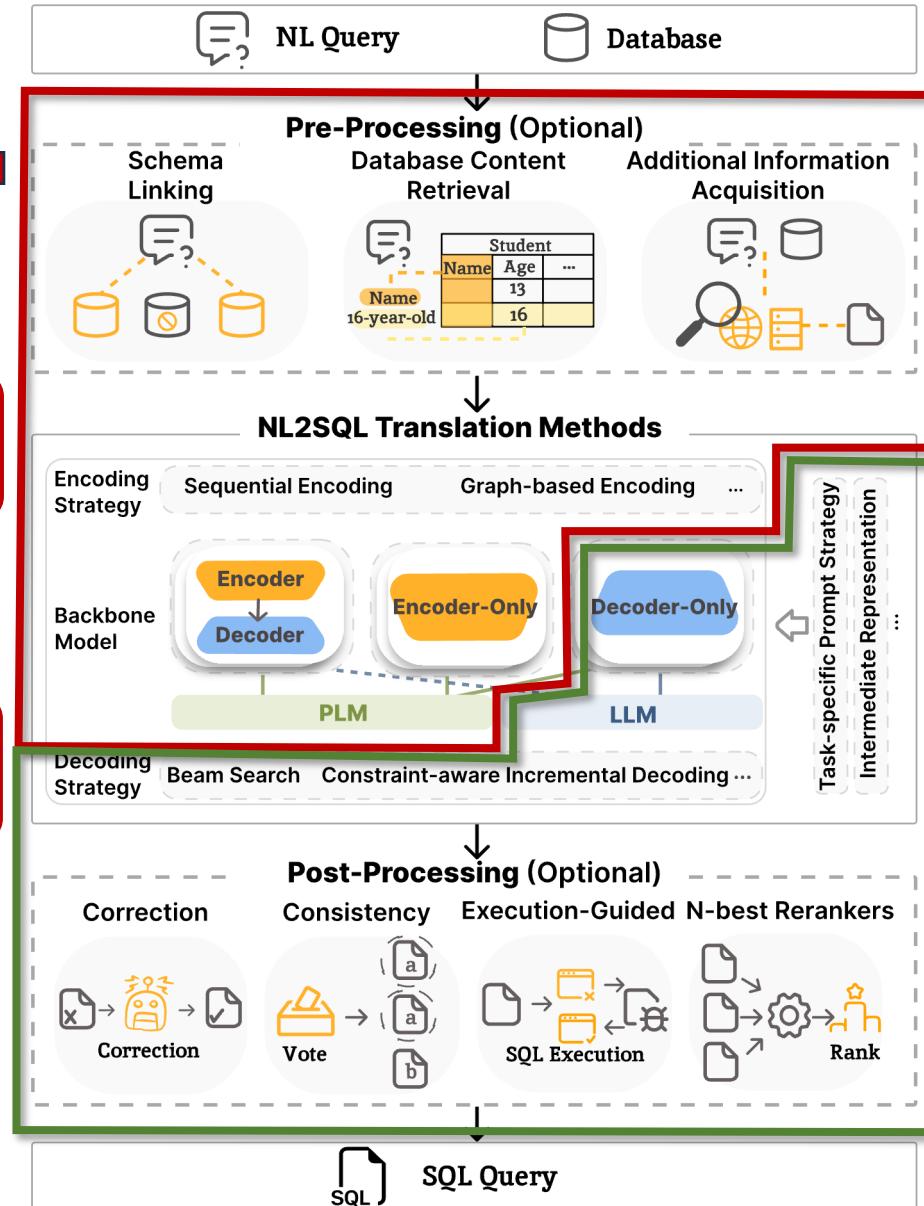
Tutorial Roadmap

NL2SQL Solutions
with PLMs and LLMs

Q1: How to design prompts and train PLMs/LLMs for NL2SQL?

- Prompt Settings: Few-shot/Zero-shot
- Training: SFT / RL

Q2: How effective are the core pre-processing techniques?



Q3: How can we build a robust NL2SQL Agent with LLMs?

Q4: From NL2SQL Agents to Data Agents: Where are we going?

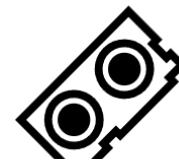
→ **NL2SQL Solutions with (LLM) Agents**

Tutorial Outline

- Problem Definition, Preliminaries, Benchmarks
- **NL2SQL Solutions with PLMs and LLMs**
- NL2SQL Solutions with LLM Agents
- Open Problems

NL2SQL Solutions with PLMs and LLMs

- Rather than categorizing existing solutions by the specific PLMs or LLMs they employ, we classify them according to **the practical considerations of different applications**.
- **Consideration #1:** The resources or costs required to develop NL2SQL
 - Computational resources (e.g., GPUs) for training
 - The monetary cost of calling LLMs (e.g., GPT) APIs



Model	Resources
RESDSQL + NatSQL	A100*1
CodeS	A800*8
Granite-20B-Code	A100*8+H100*8



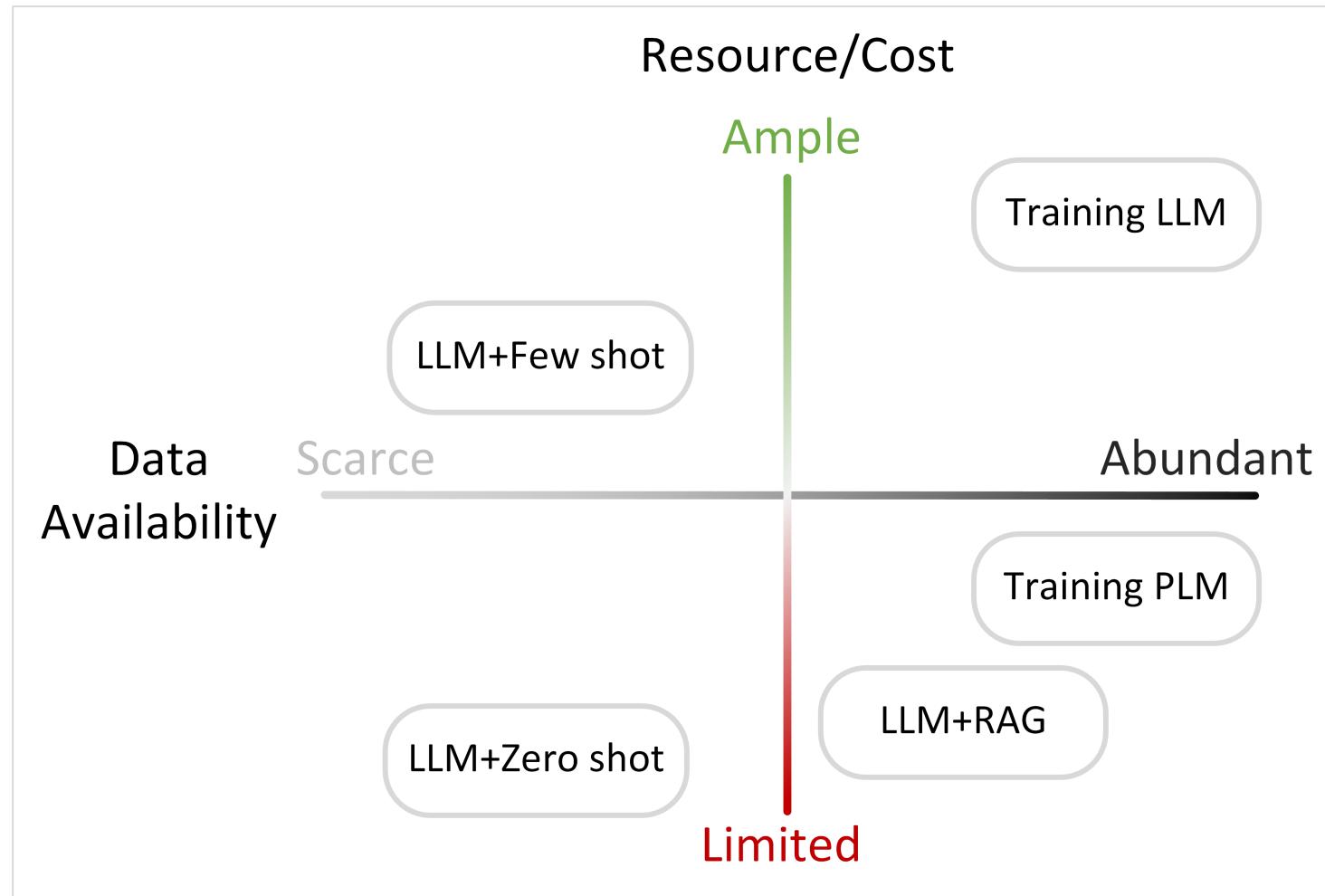
Model	Input	Output
GPT-3.5-turbo	\$0.50 / 1M tokens	\$0.50 / 1M tokens
gpt-4o	\$5 / 1M tokens	\$15 / 1M tokens

NL2SQL Solutions with PLMs and LLMs

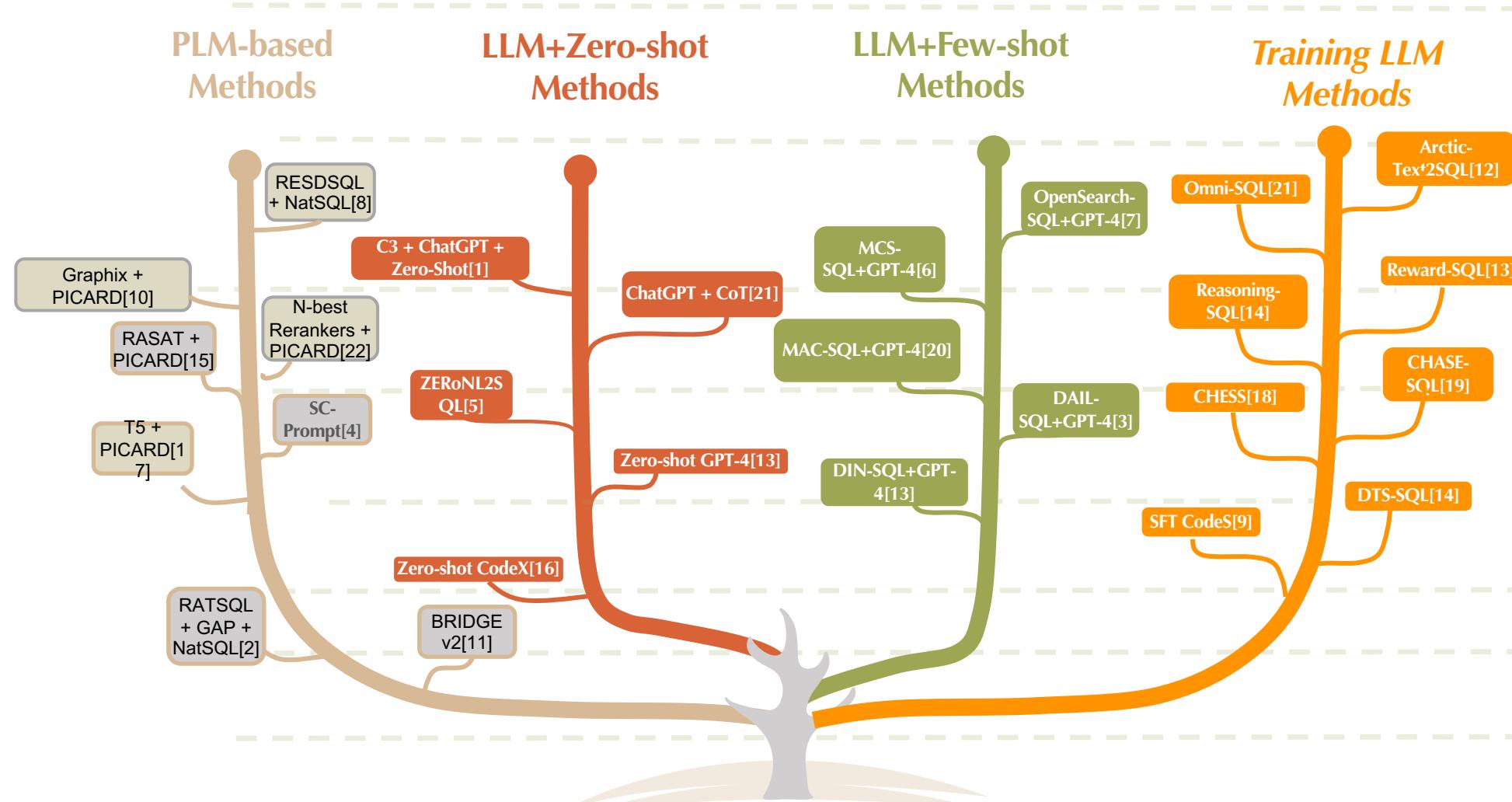
- Rather than categorizing existing solutions by the specific PLMs or LLMs they employ, we classify them according to **the practical considerations of different applications**.
- **Consideration #2:** The amount of data required for training NL2SQL
 - E.g., the CodeS model collects:
 - SQL-related data (11GB), NL-to-code data (6GB), and NL-related data (4.5GB)
 - E.g., the existing benchmarks paid much efforts to collect annotated data
 - Spider has 10,181 NL-SQL pairs
 - BIRD has 12,751 NL-SQL pairs

Categorization of Existing Studies

- We categorize the existing studies of NL2SQL Solutions with PLMs and LLMs based on two dimensions: (1) **Resources/Cost**; (2) **Data availability**



Categorization of Existing Studies



Few-Shot NL2SQL

- **Basic Idea**
 - Utilizing the **in-context learning** capability of LLMs to generate SQL queries from a few **demonstration examples**.
- **Key Characteristics**
 - Requirement of a handful of examples → Reduction of annotation costs
- **Technical Challenges**
 - How to represent the structure of the underlying database
 - How to select and organize the demonstration examples

Few-Shot NL2SQL

- **DAIL-SQL**, by Alibaba
 - **Database Representation:** representing database schema as CREATE TABLE statements with complete primary/foreign key information
 - **Example Selection:** combining question similarity and SQL query similarity, prioritizing examples with both similar questions and similar SQL structures
 - **Example Organization:** only preserving question-to-SQL mappings while removing token-expensive database schema from examples

```
1 Below is an instruction that describes a task, paired
2 ↴ with an input that provides further context. Write a
3 ↴ response that appropriately completes the request.
4
5
6
7
8
9
10
11
```

1 ## Instruction:
2 Write a sql to answer the question "How many continents
3 ↴ are there?"
4
5 ## Input:
6 continents(ContId, Continent)
7 countries(CountryId, CountryName, Continent)
8
9 ## Response:
10 SELECT

```
1 /* Some example questions and corresponding SQL queries
2 ↴ are provided based on similar problems: */
3 /* Answer the following: How many authors are there? */
4 SELECT count(*) FROM authors
5
6 /* Answer the following: How many farms are there?. */
7 SELECT count(*) FROM farm
8 ${TARGET_QUESTION}
```

Zero-shot NL2SQL

- **Zero-shot NL2SQL**
 - A practical scenario for NL2SQL is that oftentimes, for a **new test environment**, **annotated NL-SQL pairs** are time-consuming and labor-intensive to acquire, and thus is not available
- Existing approaches may not perform well in this zero-shot NL2SQL setting, as the new test environments may be very different
 - **New databases:** an NL2SQL model trained on the Spider benchmark may not perform well for domain-specific (e.g., academic or financial) databases
 - **New linguistic phenomena:** varying linguistic phenomena (e.g., abbreviations, synonyms, etc.) in the test environments

Can we have a NL2SQL model generalizable to new test environments

Limitation of Existing Solutions

- The LM-based approaches to NL2SQL fall into two categories
 - **Pre-trained language models (PLMs)** such as BART and T5
 - **Large language models (LLMs)** such as GPT and PaLM
- PLM-based methods (e.g., T5) may have limited *generalizability* in natural language reasoning in the zero-shot setting

(a) A Text Question Q

Which course has the highest score for the student named timothy ward?

(b) Snippets of a Database D

Course	id	course	teacher
001	math	jordy wu	
...	

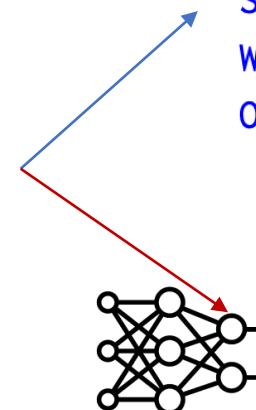
Student	id	given_name	last_name	score	course
	1	timmy	ward	92	math

(c) The Ground-truth SQL Query S w.r.t. Q

```
SELECT course FROM Student  
WHERE given_name = 'timmy' AND last_name = 'ward'  
ORDER BY score LIMIT 1;
```

(d) An SQL query S' translated by an SLM

```
SELECT course FROM Student  
WHERE given_name = 'timothy ward'  
ORDER BY score LIMIT 1;
```



Limitation of Existing Solutions

- The LM-based approaches to NL2SQL fall into two categories
 - **Pre-trained language models (PLMs)** such as BART and T5
 - **Large language models (LLMs)** such as GPT and PaLM
- LLMs (e.g., gpt-3.5-turbo-0613) are capable of NL reasoning, but may not achieve precise **alignment** on schema and data value due to “hallucination”

(a) A Text Question Q

Which course has the highest score for the student named timothy ward?

(b) Snippets of a Database D

Course	id	course	teacher
001	math	jordy wu	
...	

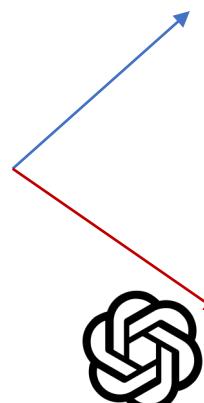
Student	id	given_name	last_name	score	course
	1	timmy	ward	92	math

(c) The Ground-truth SQL Query S w.r.t. Q

```
SELECT course FROM Student  
WHERE given_name = 'timmy' AND last_name = 'ward'  
ORDER BY score LIMIT 1;
```

(e) An SQL query S'' translated by an LLM

```
SELECT Course.course, Student.score  
FROM Student JOIN Course ON Student.id = Course.id  
WHERE given_name = 'timothy' AND last_name = 'ward'  
ORDER BY score LIMIT 1;
```



Limitations of Existing Solutions

- A systematic **error analysis** that illustrates insights into limitations of the fine-tuned T5 and vanilla GPT-3.5

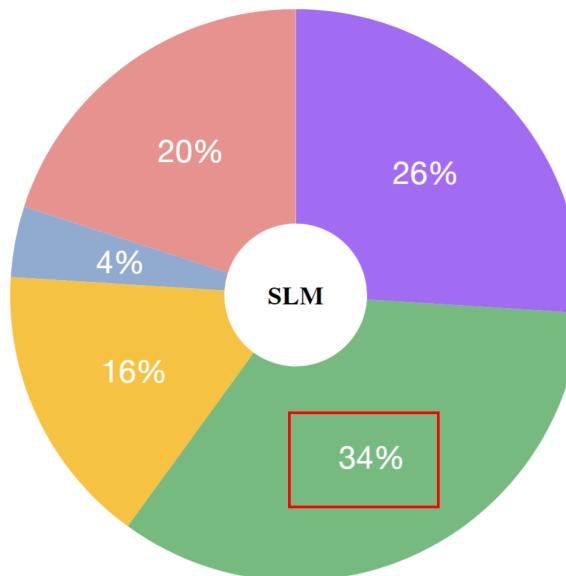
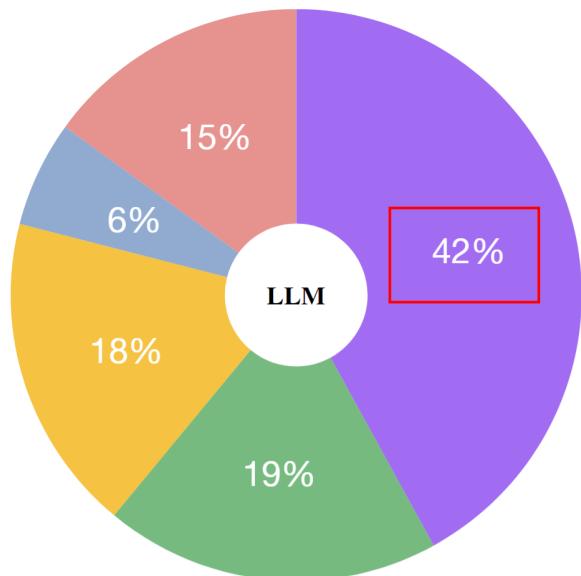
● table/column selection

● conditions

● keywords

● invalid

● others



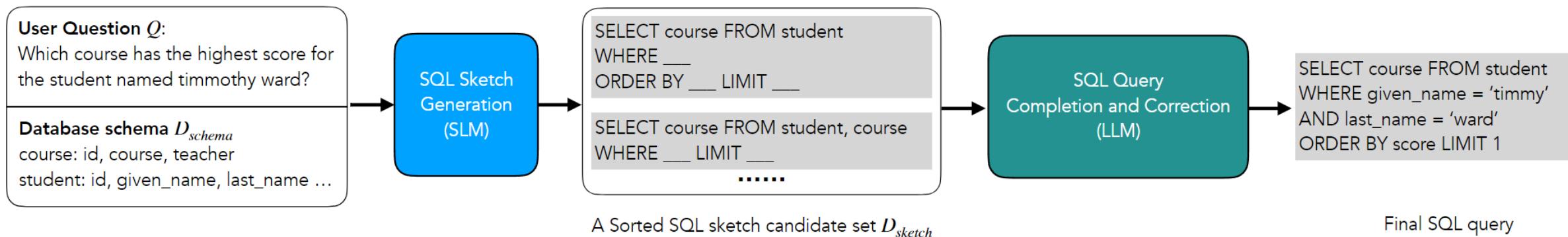
LLMs	PLMs
Complex NL Reasoning	Schema Alignment
Schema Alignment	Complex NL Reasoning

200 error examples sampled from Dr.Spider (GPT-3.5 and T5 respectively)

Can we combine PLMs and LLMs to solve Zero-shot NL2SQL?

The ZeroNL2SQL Framework

- ZeroNL2SQL breaks down the NL2SQL task into smaller sub-tasks
 - **Sub-task 1: SQL Sketch Generation**
 - Utilizing PLMs to generate a **SQL sketch**, with **attributes** to SELECT, **tables** in FROM, and necessary **keywords** (e.g., ORDER BY) for composing the SQL query
 - **Sub-task 2: SQL Query Completion and Correction**
 - Utilizing LLMs to complete the **missing information** in the SQL sketch and generate complete SQL queries, e.g., aligning with **data values** from the database

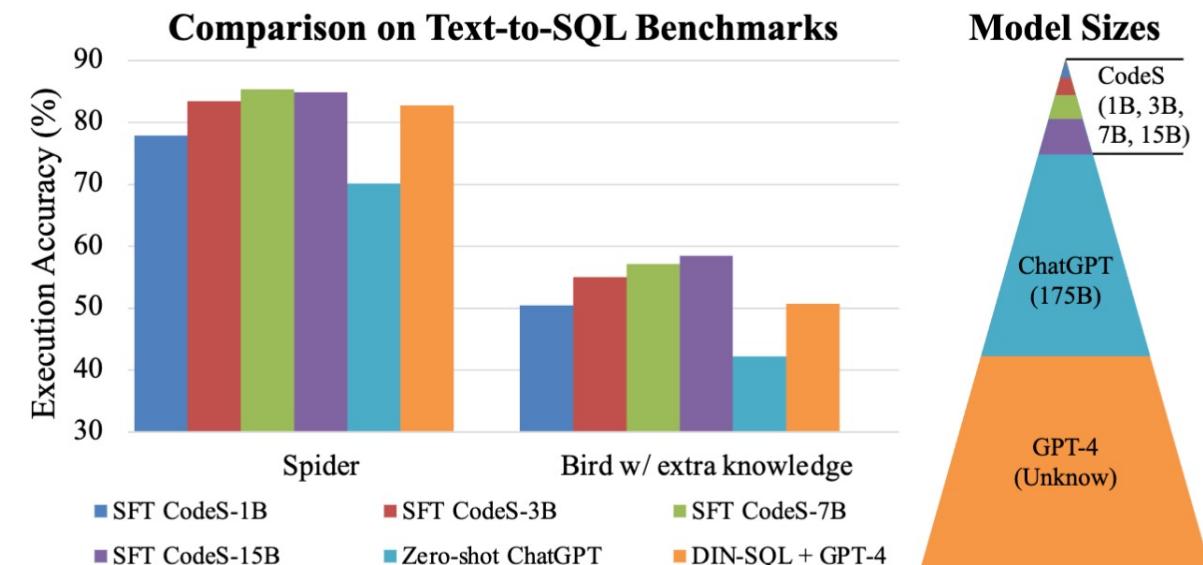


Training LLMs for NL2SQL

- **Basic Idea**
 - Training LLMs in two stages: (1) performing continual pre-training (**CPT**) on SQL-related corpora to strengthen SQL knowledge, and (2) conducting supervised fine-tuning (**SFT**) on curated NL2SQL datasets to specialize in SQL generation.
- **Key Characteristics**
 - **Enhanced SQL domain knowledge:** CPT injects rich understanding of SQL syntax and semantics.
 - **Superior reasoning capabilities:** SFT enables models to gain stronger ability to parse complex natural language and map it to SQL queries.

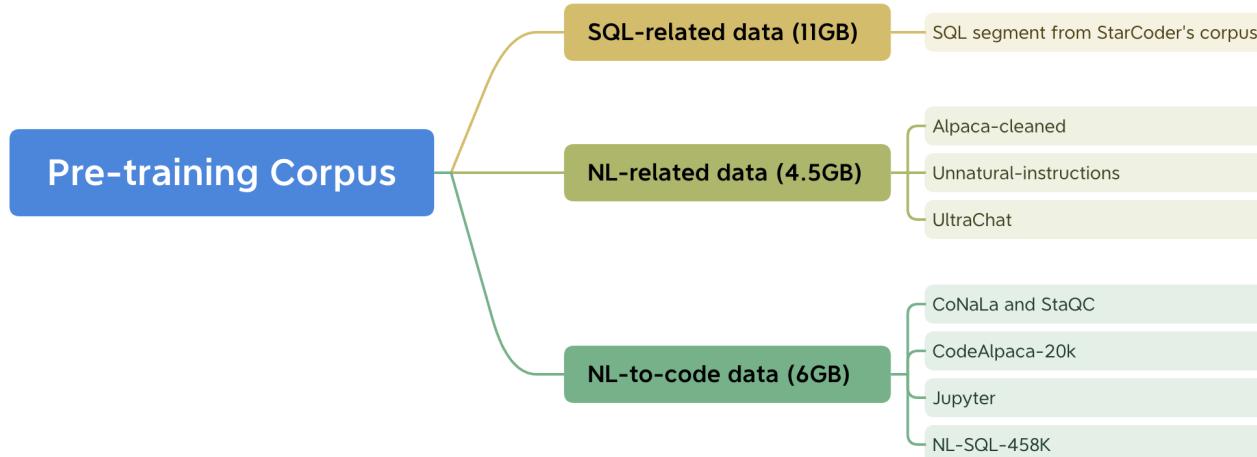
Training LLMs for NL2SQL

- **CodeS** proposes to develop a new text-to-SQL model built on **open-source** models.
- **Solution Overview:**
 - CodeS introduces a series of open-source language models (ranging from 1B to 15B parameters) specifically tailored for text-to-SQL tasks
 - Built on top of StarCoder, CodeS is further enhanced through CPT and SFT on a curated 21.5GB SQL-centric corpus.

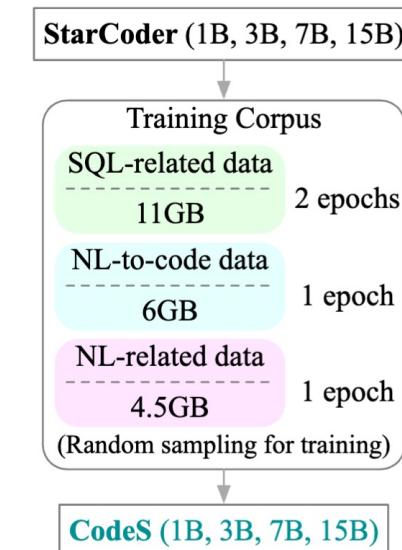


Data Collection for CPT and SFT

- **Curated CPT corpus:** 11GB SQL-related data, 6GB NL-to-code data, and 4.5GB NL-related data
- **SFT corpus:** NL-SQL-458K, containing 458K SQL queries paired with corresponding natural language questions
- **Enhanced capabilities:** improvements in both SQL generation and natural language understanding



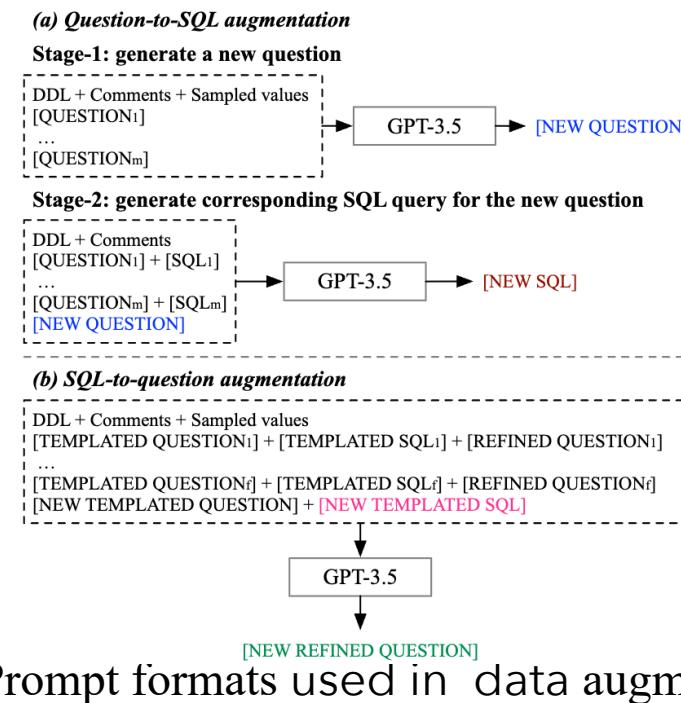
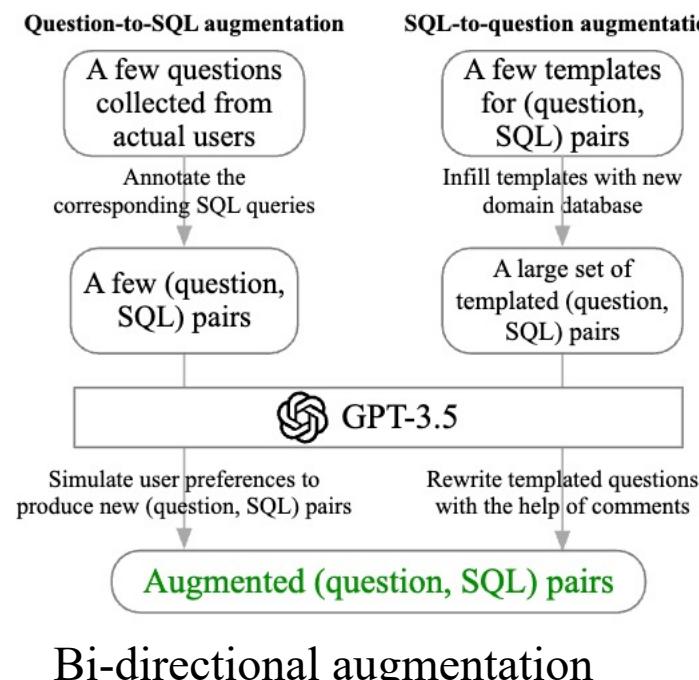
Step 1: Collect SQL-related corpus



Step 2: Incremental pre-training

Data Augmentation for SFT

- **Question-to-SQL:** starting from real user questions, manually annotate, and expanding using GPT-3.5
- **SQL-to-Question:** leveraging Spider-style templates, populating with new domain schemas, and refining via GPT-3.5
- **Enhanced capabilities:** rapid domain adaptation with minimal annotation effort



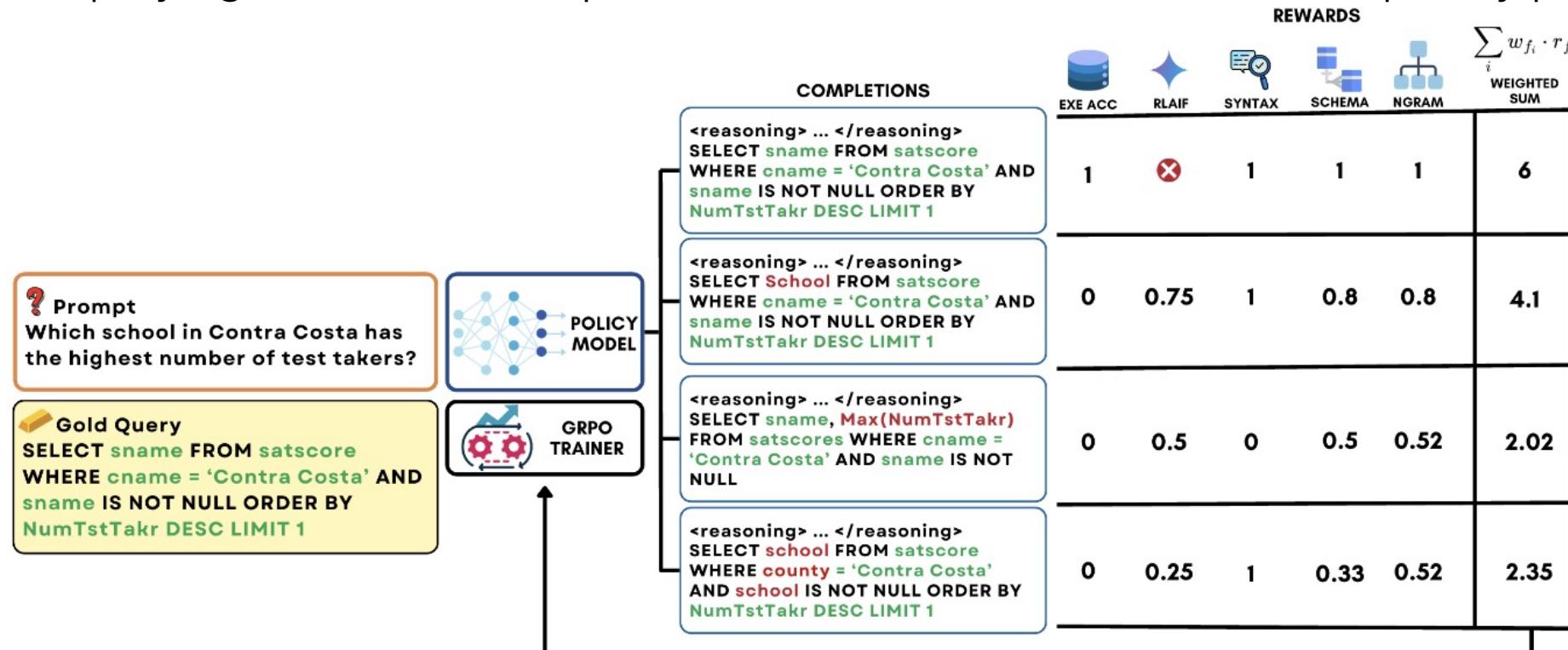
RL-based Training for NL2SQL

- Reinforcement learning based training for NL2SQL leverages **execution feedback** and **reasoning signals**, and applies techniques such as DPO, GRPO, and reward-based optimization to generate SQL queries.
- **Key Characteristics:**
 - **Stronger Reasoning:** RL fosters structured, step-by-step reasoning for better SQL generation
 - **Richer Feedback:** dedicated rewards overcome sparsity, guiding models more effectively
 - **Higher Accuracy & Generalization:** outperform larger models across benchmarks at lower cost

RL-based Training for NL2SQL

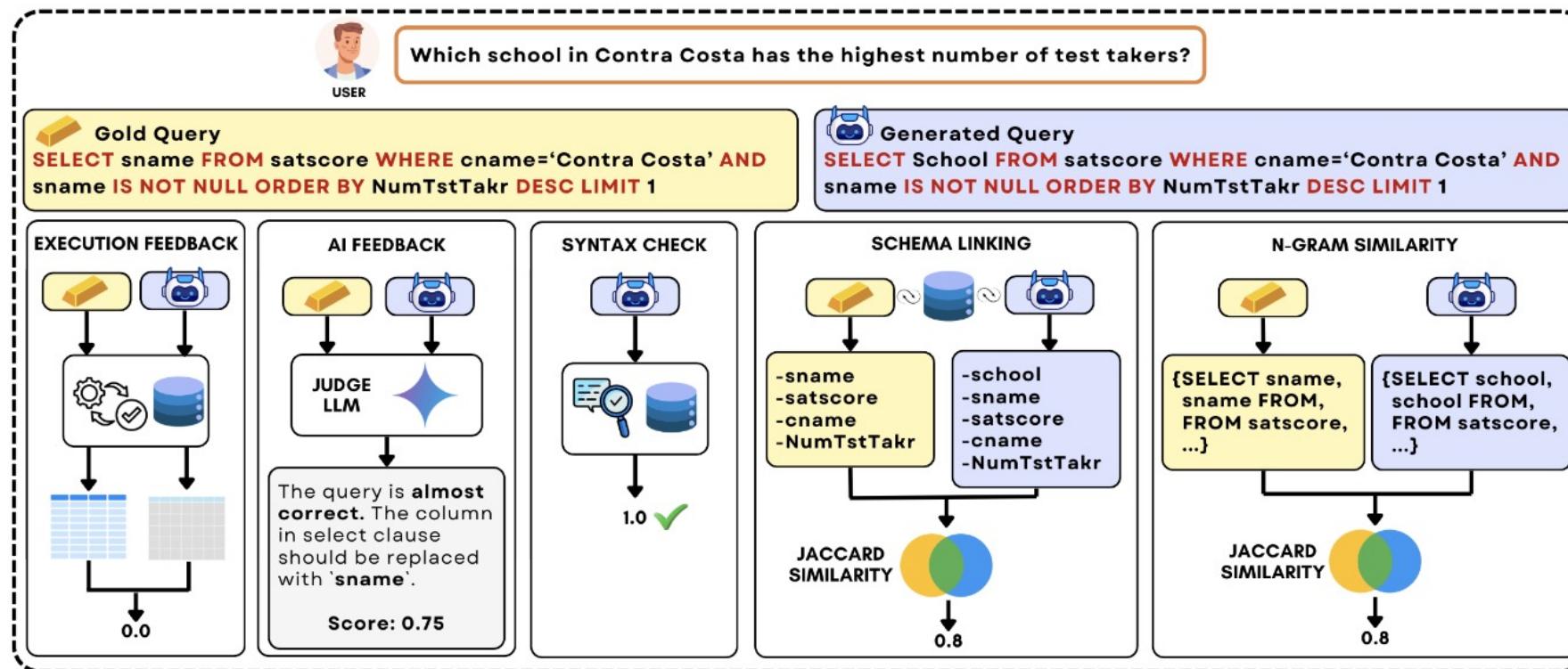
- **Reasoning-SQL**, RL-Enhanced NL2SQ with *Partial Rewards*

- Introducing the first RL-based framework for optimizing reasoning in LLMs for NL2SQL
- Leveraging Group Relative Policy Optimization (GRPO) for efficient and stable training
- Employing a novel suite of partial rewards to address the reward sparsity problem



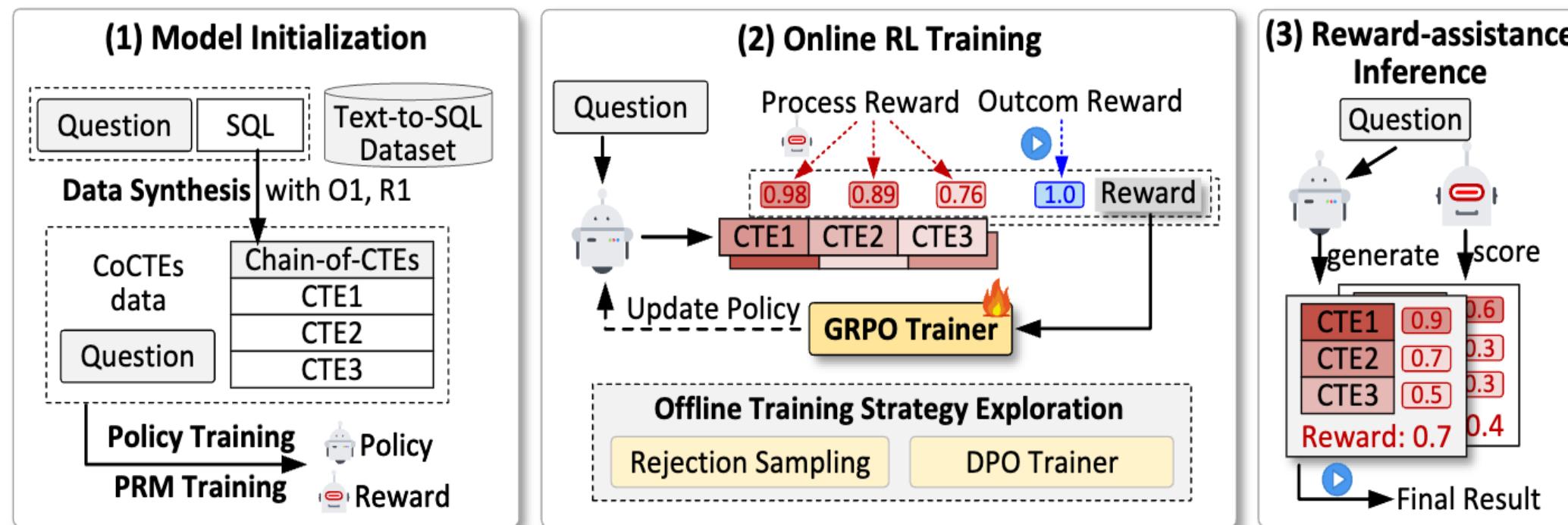
A Suite of Partial Rewards

- **Execution Accuracy Reward (RLEF):** Binary reward for correct SQL execution
- **LLM-as-a-Judge Reward (RLAIF):** AI feedback for queries with zero execution accuracy
- **Syntax Check Reward:** Positive score for syntactically valid and executable queries
- **Schema Linking Reward:** Jaccard similarity between schema items in candidate vs. gold queries
- **N-gram Similarity Reward:** Token-level overlap measurement using Jaccard similarity



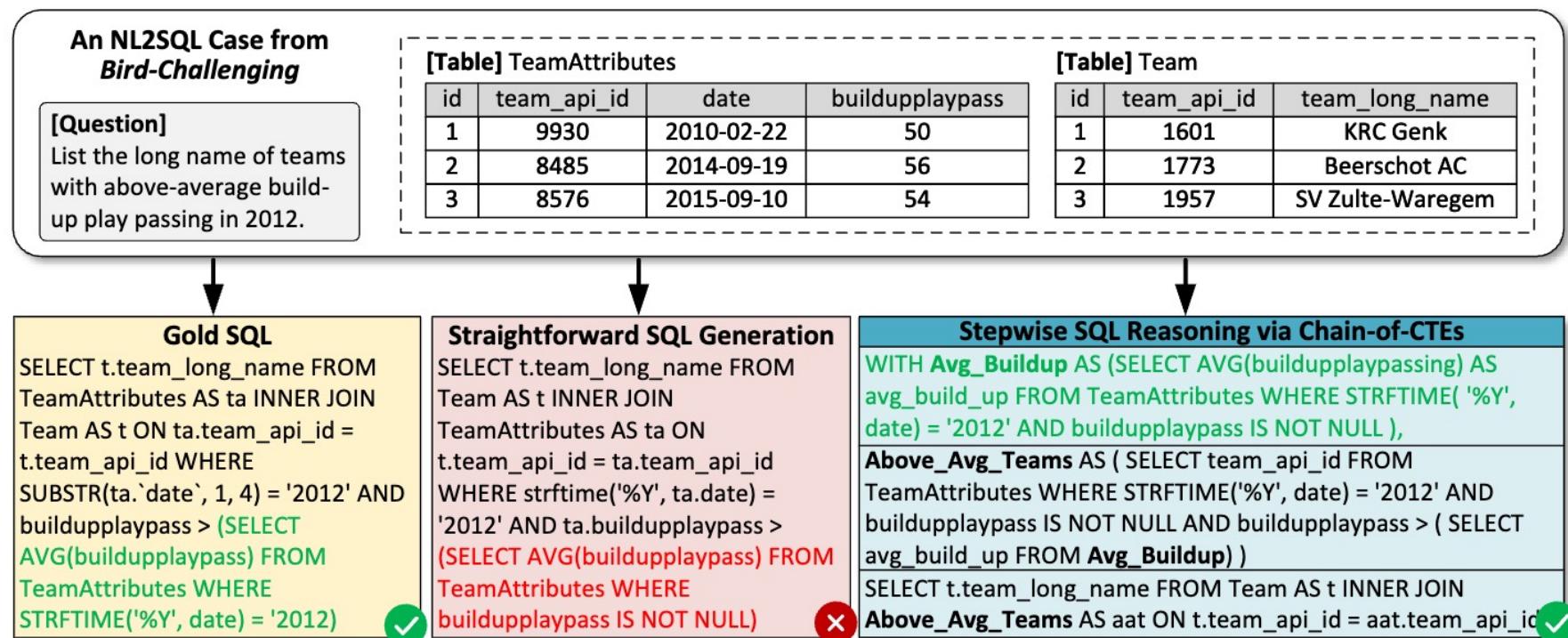
Process-Supervised Rewards for NL2SQL

- **Reward-SQL:** introducing Process Reward Models (PRMs) for NL2SQL
 - **PRM-Enhanced Test-Time Scaling:** Adopting PRMs for test-time scaling for NL2SQL
 - **GRPO-Integrated Training:** Incorporating PRMs into training via Group Relative Policy Optimization to further enhance reasoning capabilities



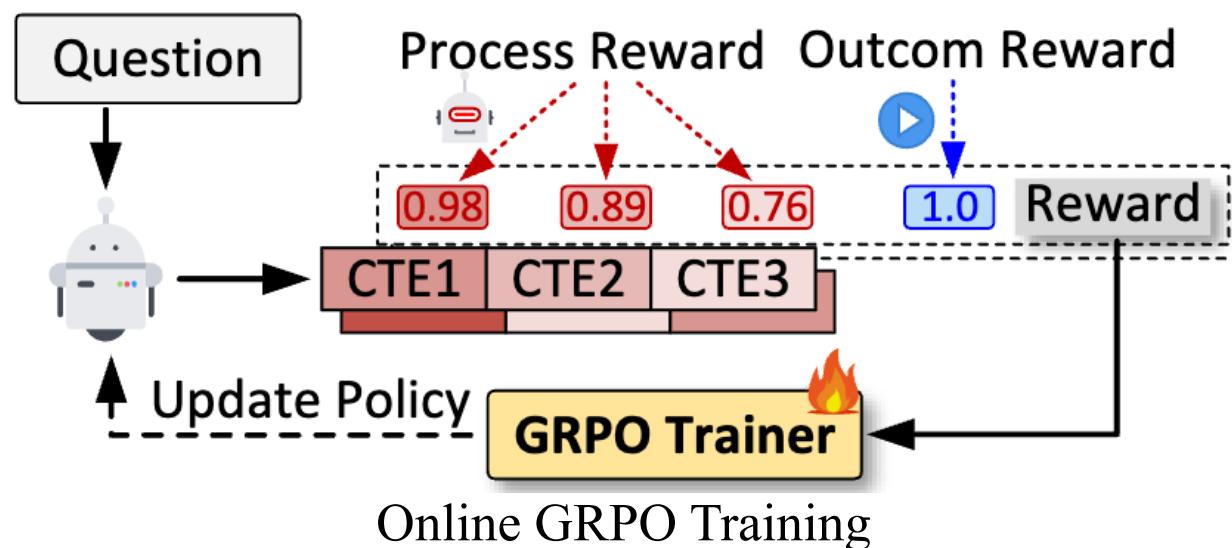
Process-Supervised Rewards for NL2SQL

- **SQL Query Decomposition:** Breaks complex queries into step-by-step Common Table Expressions (CTEs)
- **Step-Level Executability:** Each CTE produces concrete, verifiable intermediate results
- **PRM-Compatible Structure:** Enables fine-grained evaluation at each reasoning step



PRM-Involved GRPO Training

- **GRPO Model Update:** Leveraging GRPO to update the model with PRM preferences, maintaining consistency between training and inference distributions to further enhance test-time scaling capabilities.
- **Combined Reward Structure:** Process Reward (PR) + Outcome Reward (OR) for comprehensive feedback
- **Fine-Grained Advantages:** Step-level advantages reflecting both solution quality and internal step variations



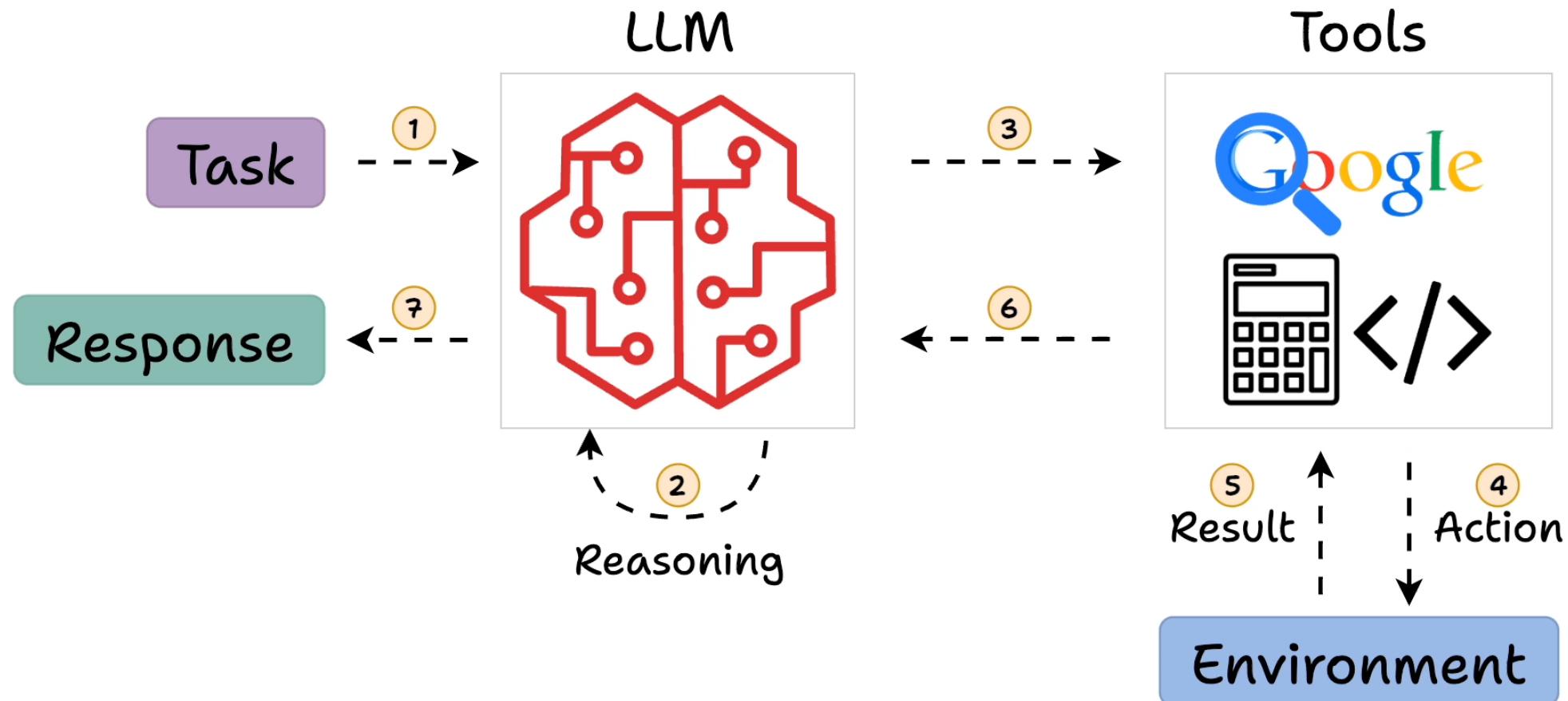
Takeaways

- **Architectural Simplification:** Text-to-SQL has evolved from complex multi-stage PLM pipelines to streamlined end-to-end training, with RL-based frameworks **eliminating auxiliary components** while achieving superior performance.
- **Escalating Data Demands:** Simplified architectures paradoxically require exponentially **more training data**, making synthetic data generation critical while demanding unprecedented quality and diversity for robust generalization.
- **Performance-Cost Trade-off:** State-of-the-art methods introduce substantial **computational overhead**, creating fundamental tensions between model performance and practical deployment in resource-constrained environments.

Tutorial Outline

- Problem Definition, Preliminaries, Benchmarks
- NL2SQL Solutions with PLMs and LLMs
- **NL2SQL Solutions with LLM Agents**
- Open Problems

What is the (Reasoning) Agent?



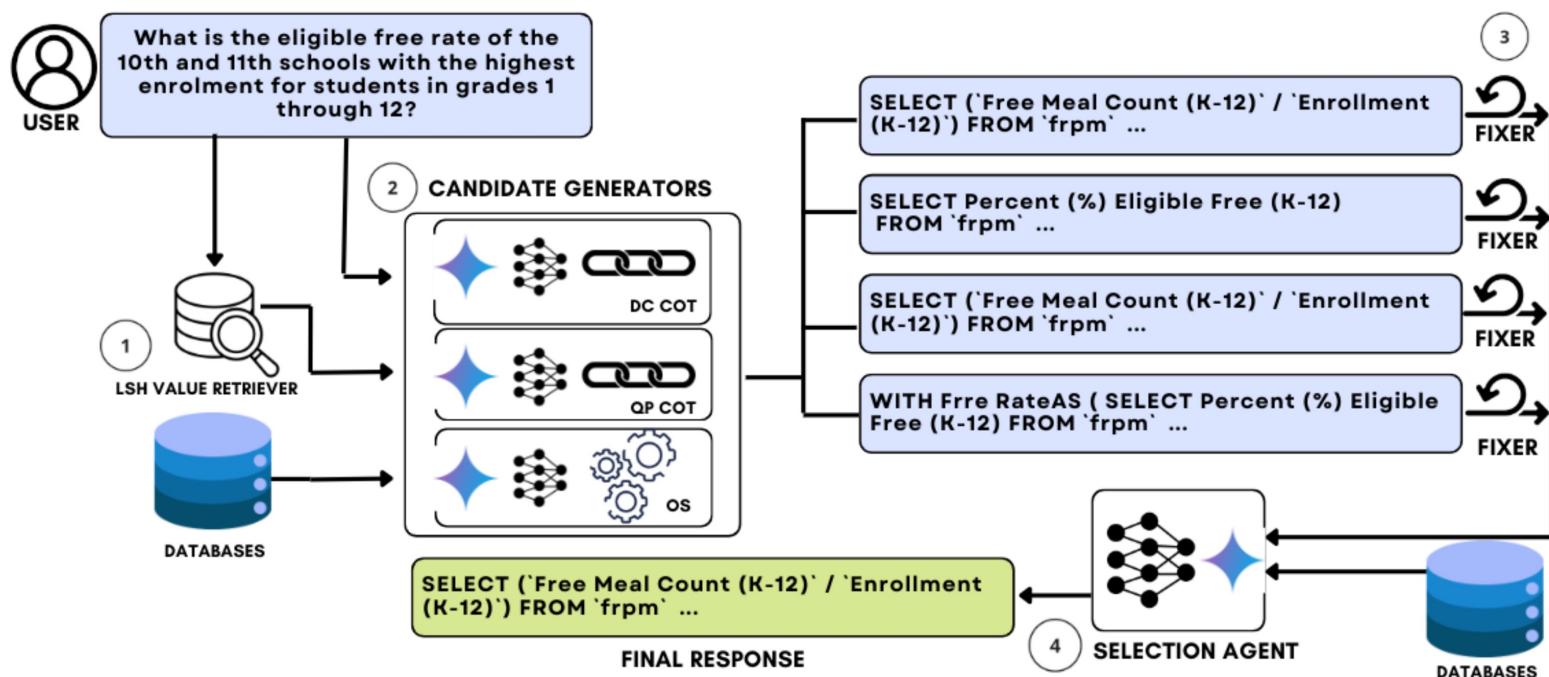


Where Are We?

- **CHASE-SQL** (*ICLR 2025, Google Cloud and Stanford*)

Closed-source
LLMs

- Utilizes the **MinHash LSH** to search for values related to the user query
- Multiple prompting strategies to **generate various candidate SQL queries** using LLMs, and corrects SQL queries with execution errors through prompting LLMs.
- Employs an **SQL selection agent** fine-tuned specifically for the database to select the final SQL from multiple candidates.

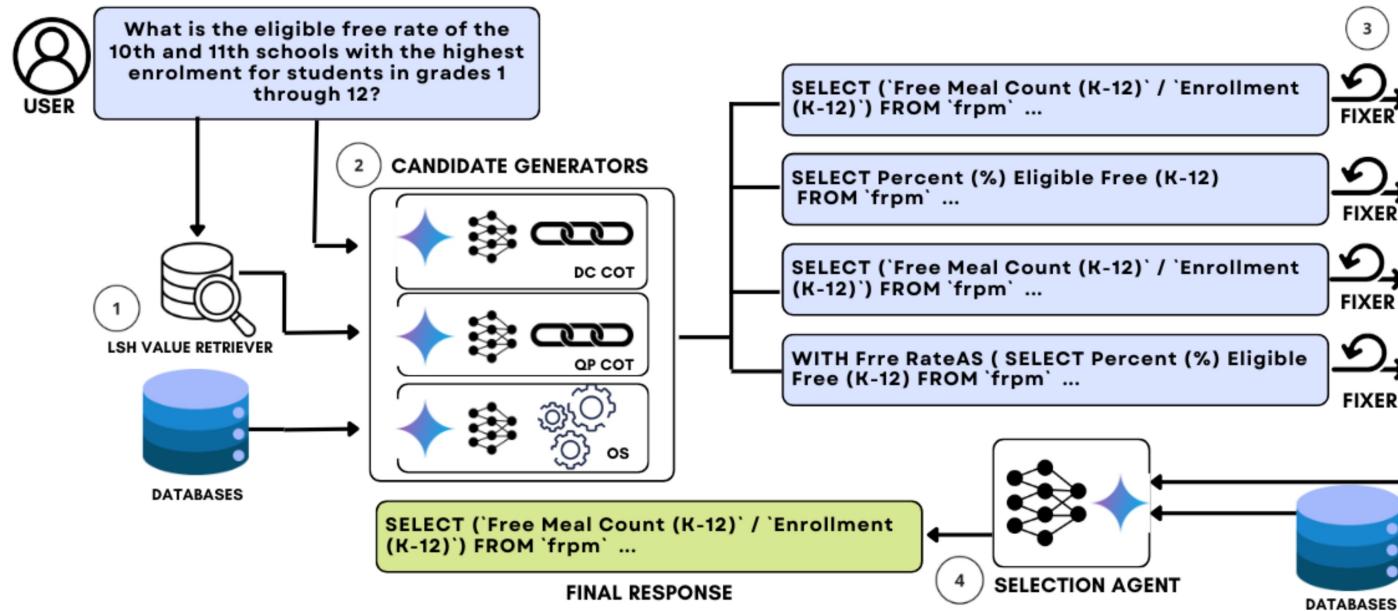




Where Are We?

- **CHASE-SQL** (ICLR 2025, Google Cloud and Stanford)

Closed-source
LLMs



Key Limitations:

- **Reliance on closed-source large models**
 - High cost (**0.6 USD/query**), making it difficult to widely deploy in real-world industrial scenarios.
- **SQL selection agent requires fine-tuning**
 - The **Google** team *fine-tuned* the **Gemini-1.5-Flash** model specifically.
 - Limited flexibility due to reliance on domain-specific data.
- **Predefined and Fixed Reasoning Workflows**

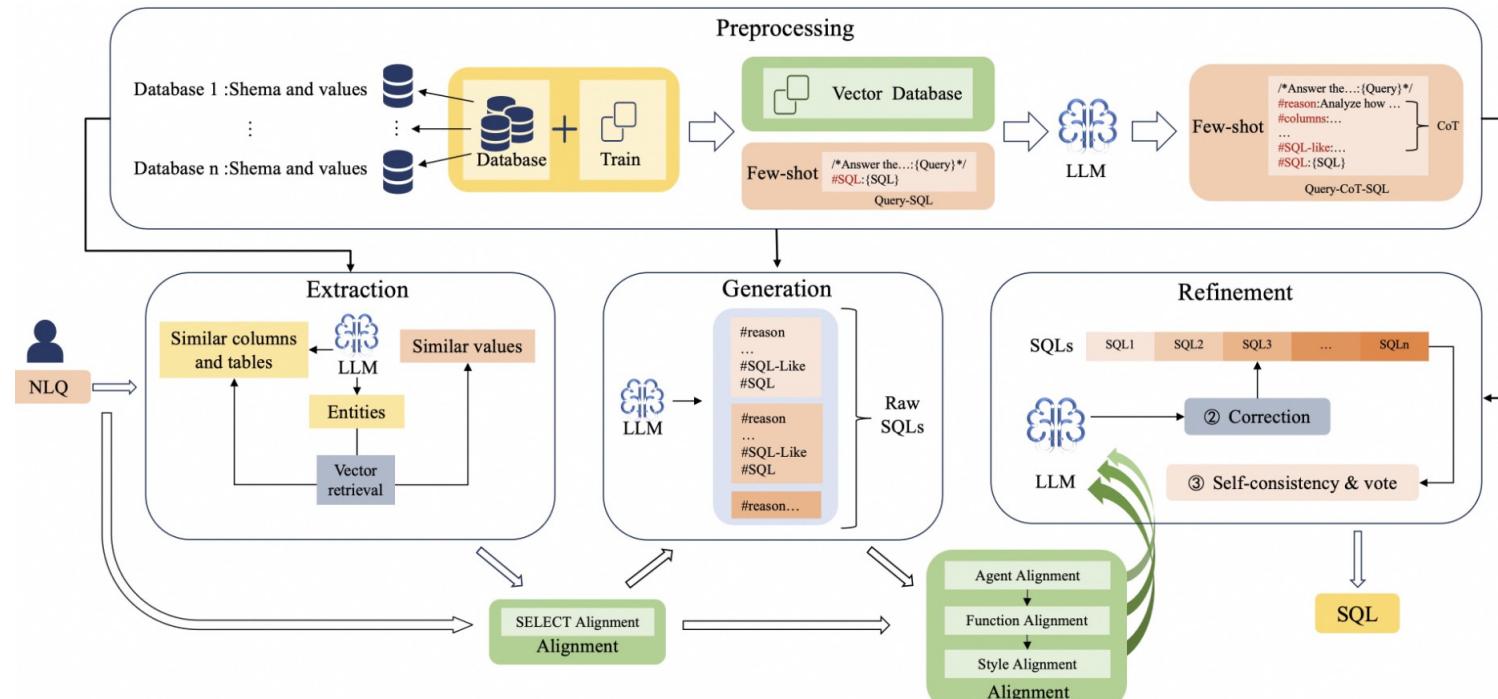


Where Are We?

- **OpenSearch** (SIGMOD 25, Alibaba)

**Closed-source
LLMs**

- **Modular Architecture:** Divides the task into four stages (Preprocessing, Extraction, Generation, and Refinement) and adds an Alignment module to ensure consistency between steps.
- **Intermediate Language:** A custom language named SQL-Like is designed to structure the model's reasoning process.

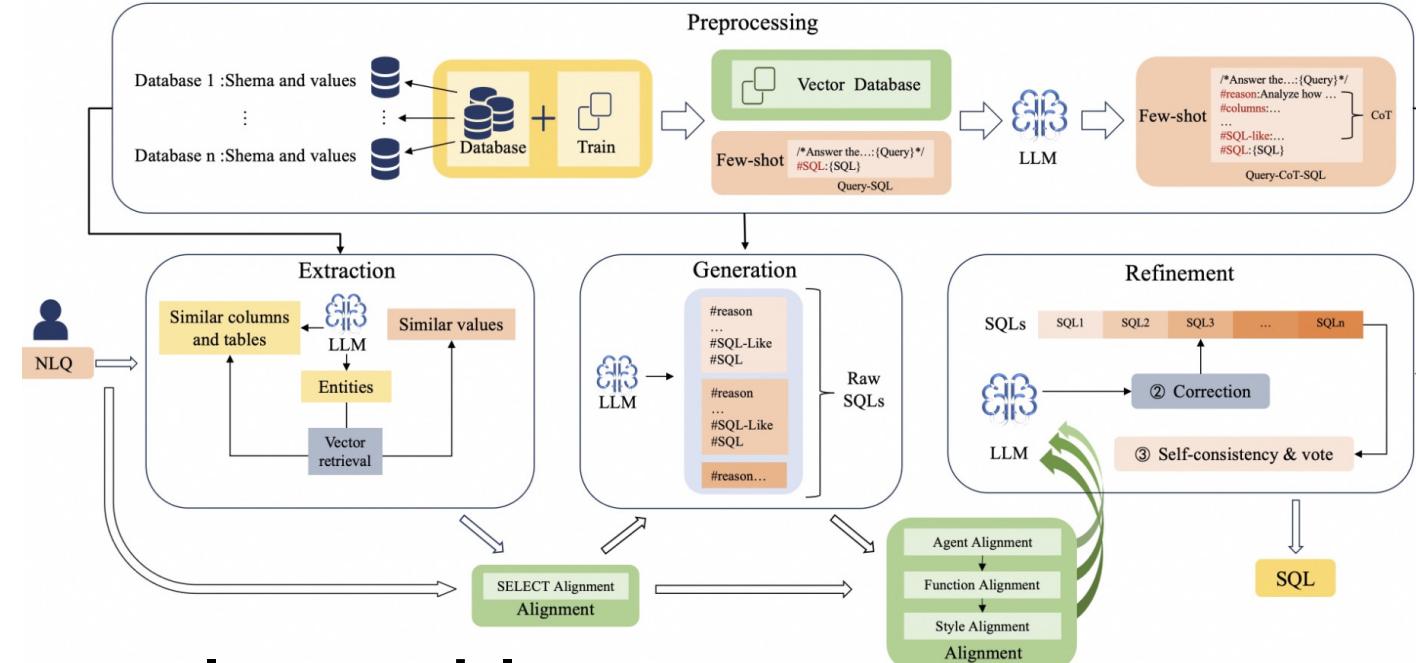


Where Are We?



- **OpenSearch** (SIGMOD 25, Alibaba)

**Closed-source
LLMs**



Key Limitations:

- **Reliance on closed-source large models**
 - Privacy Risks
- **Rigidity of the Alignment Module:**
 - The alignment mechanism enforces consistency but risks over-constraining SQL generation and limiting adaptability across scenarios.
- **Predefined and fixed reasoning workflows**

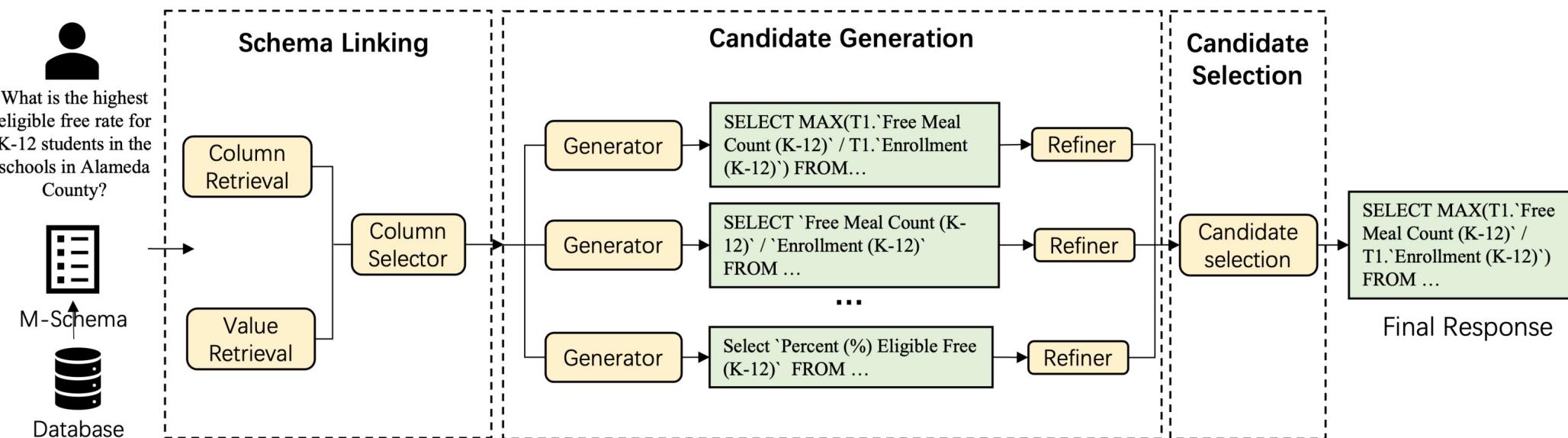
Where Are We?



- **XiYan-SQL** (SIGMOD 25, Alibaba)

- **M-Schema**: Uses column and value retrieval to select relevant schema items from DBs.
- **Fine-tunes a base LLM** on SQL-specific data, then **creates multiple specialized SQL-generation** models by fine-tuning with diverse Text-to-SQL syntax datasets.
- Employs a **fine-tuned SQL selection** model to choose the best SQL from predictions made by multiple generators.

Open-source LLMs

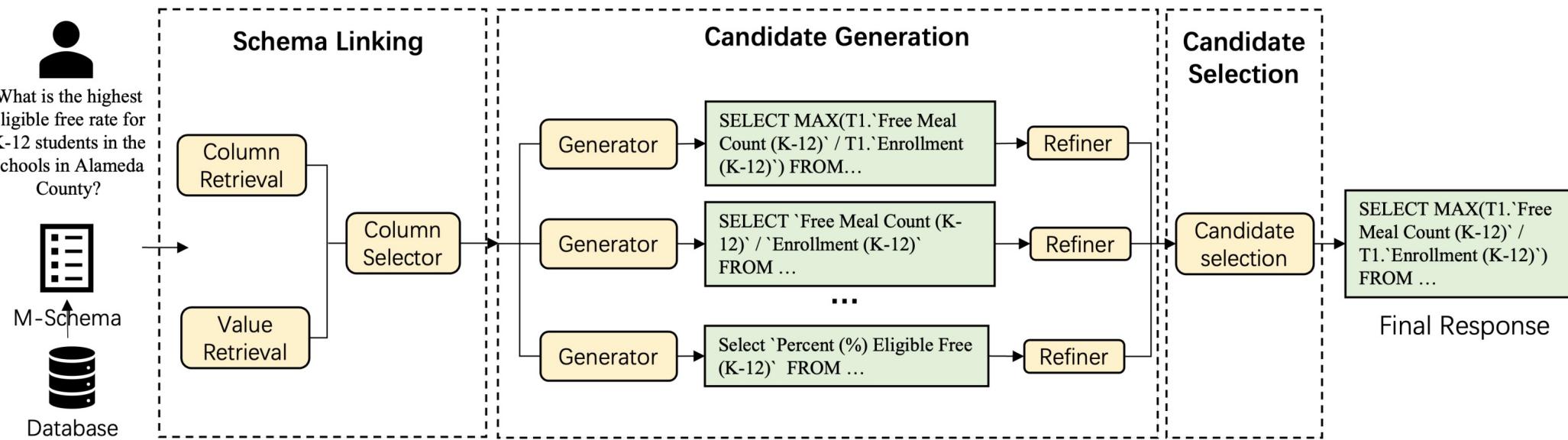


Where Are We?



- **XiYan-SQL** (SIGMOD 25, Alibaba)

Open-source LLMs



Key Limitations:

- High dependency on extensive **domain-specific data**.
- Significant **costs** associated with **fine-tuning** multiple models.
- **Difficulty** in rapid **adaptation** and **generalization** across varied scenarios.
- **Predefined and Fixed Reasoning Workflows**.

Key Takeaways



Closed-source LLMs for Text-to-SQL:

- High **inference API cost** limits practical deployments.
- Potential data privacy concerns for sensitive applications.



Open-source LLMs for Text-to-SQL:

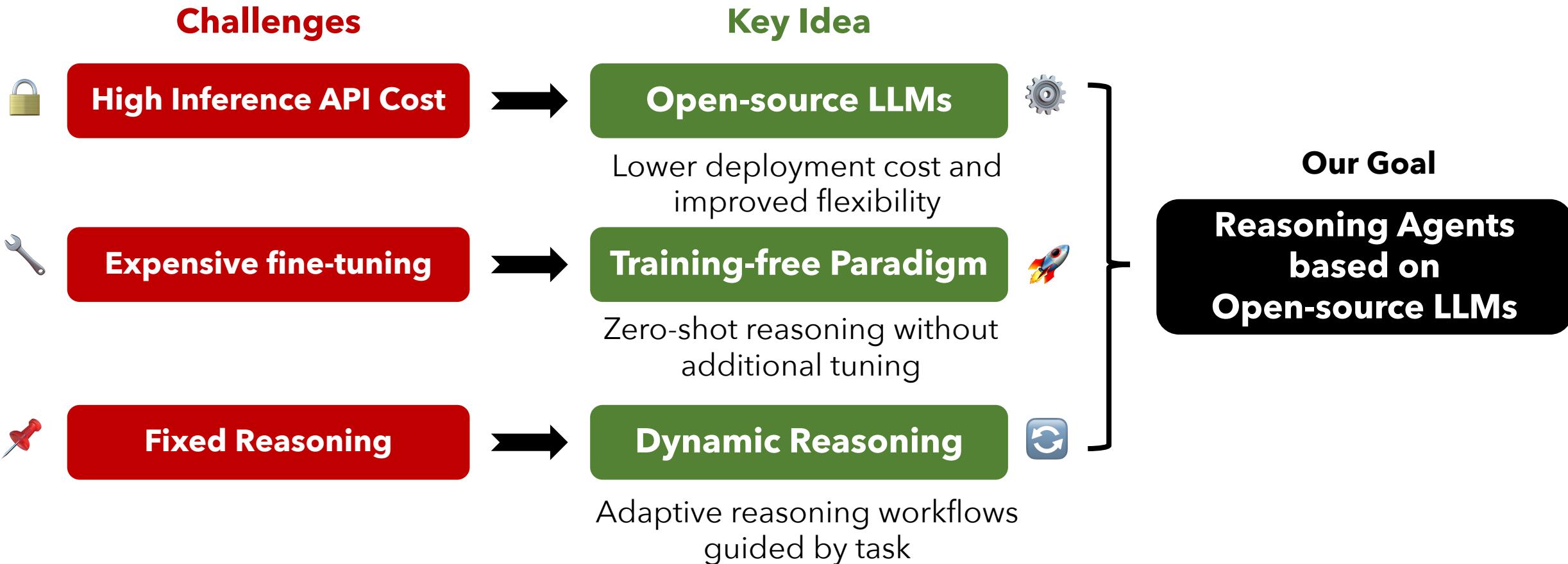
- Dependence on **extensive domain-specific data** for **model fine-tuning**.
- Limited generalization capability across different use cases.



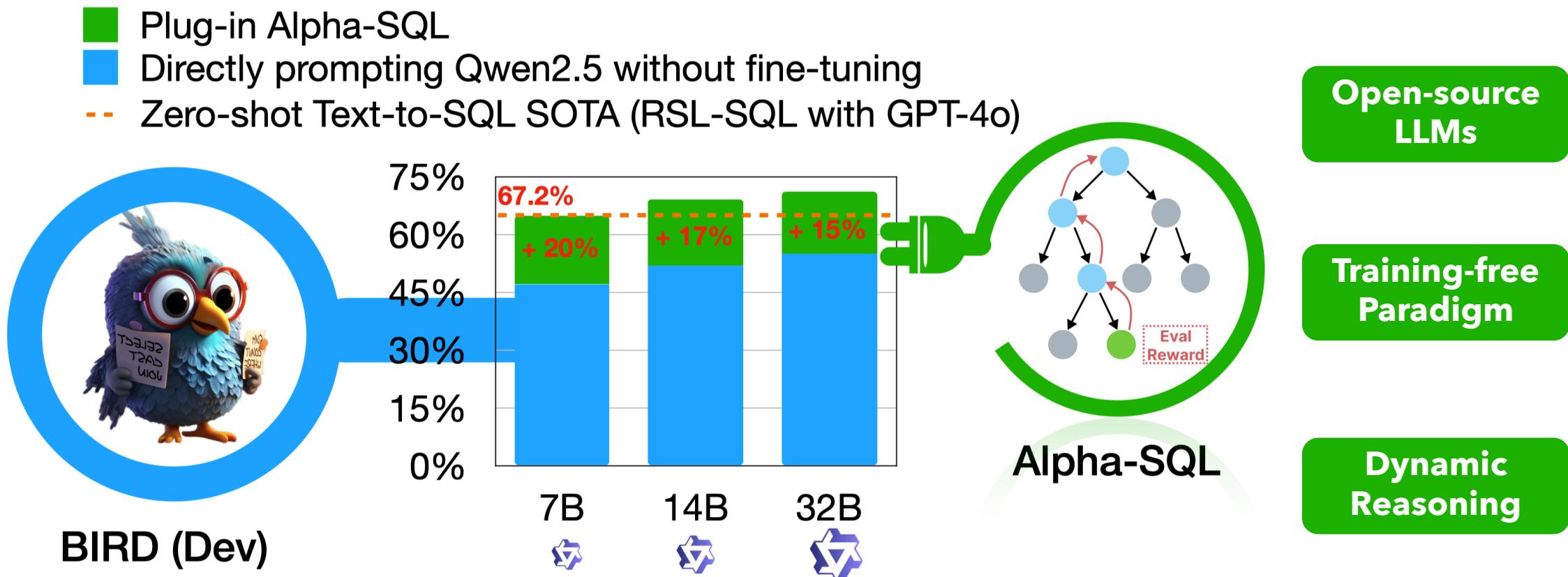
Common Limitations in Existing Solutions:

- Predefined and fixed reasoning workflows restrict adaptability.
- Domain adaptation and generalization across DB and text queries

Where Are We Going?



Alpha-SQL: A Plug-and-Play NL2SQL Reasoning Framework



NL2SQL Human Workflow

Step-1 NL Understanding



Find the number of dog pets that are raised by female student

Step-2 Schema Linking and Database Content Retrieval

Pets			
PetID	PetType	PetAge	...
	Dog		

Student			
StuID	Sex	Age	...
	F		

Has_Pet		
PetID	StuID	...

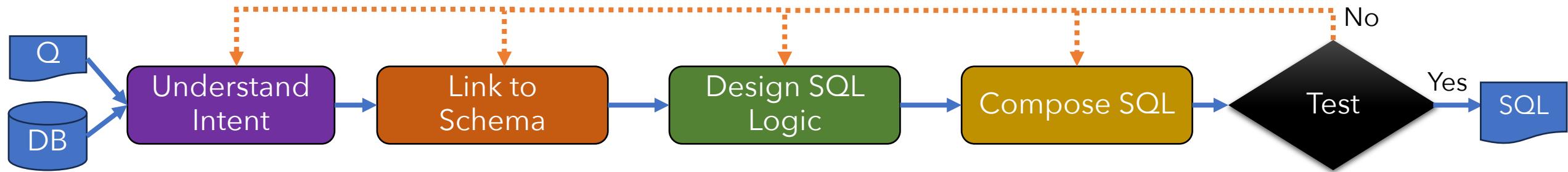
Step-3 Translating the NL Intent into the SQL

```
Select count(*) FROM student AS T1 JOIN has_pet AS T2 ON T1.stuid=T2.stuid  
JOIN pets AS T3 ON T2.petid=T3.petid WHERE T1.sex='F' AND T3.pettype='Dog'
```

- Table Linking
- Columns Linking
- Database Content
- Foreign Key

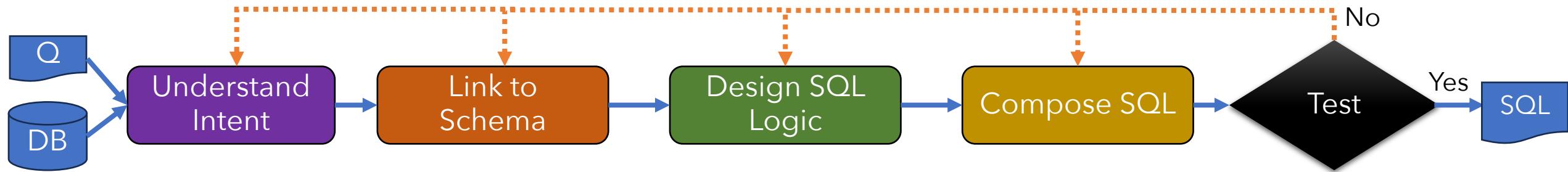
Task Formulation: Mimic Human Experts

- Human Expert Workflow for Text-to-SQL

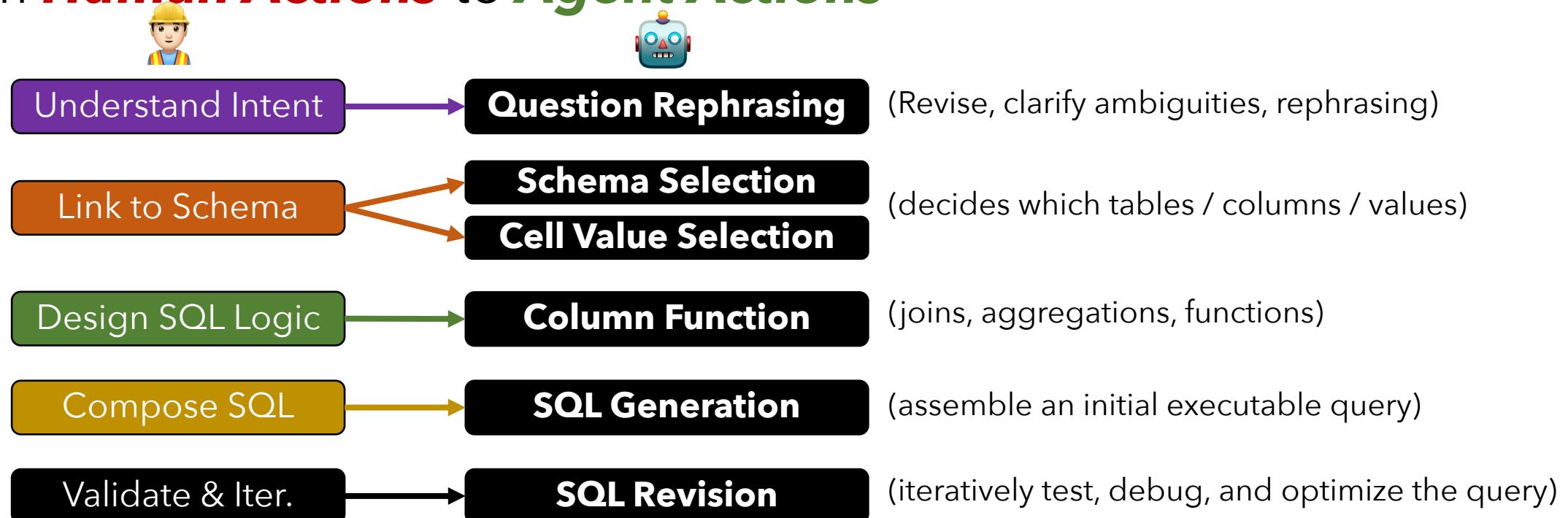


Task Formulation: Mimic Human Experts

- Human Expert Workflow for Text-to-SQL

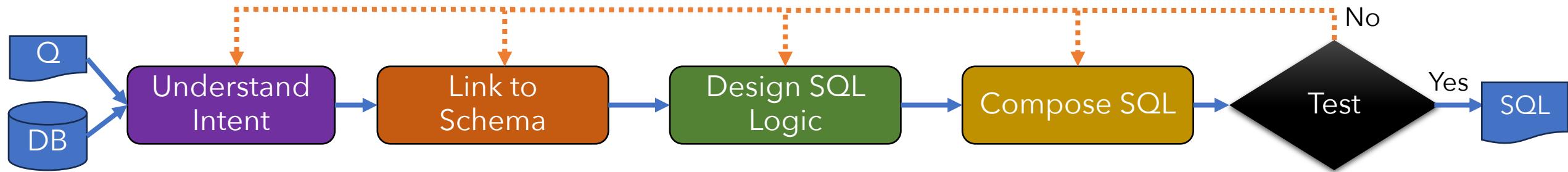


- From **Human Actions** to **Agent Actions**

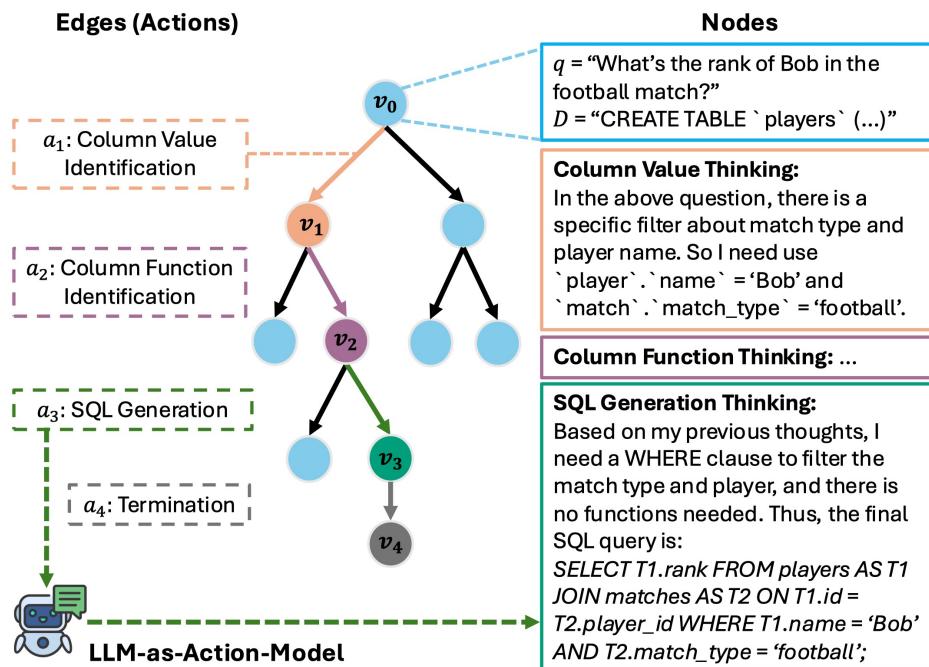


Task Formulation: Mimic Human Experts

- Human Expert Workflow for Text-to-SQL



- From the **Fixed** Action to **Dynamic** Actions



Tree-based Search:

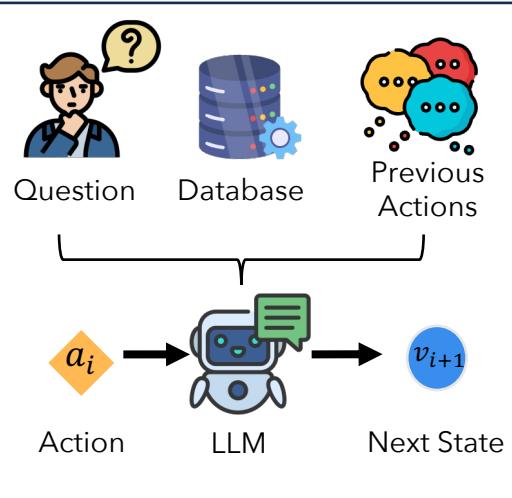
- Each **edge** corresponds to an **agentic action** in the query construction process,
- Each **node** represents a **reasoning state** at a specific step, and
- Each **path** corresponds **to a sequence of SQL construction actions** for Text-to-SQL task.

Text-to-SQL as a Tree-based Search Problem

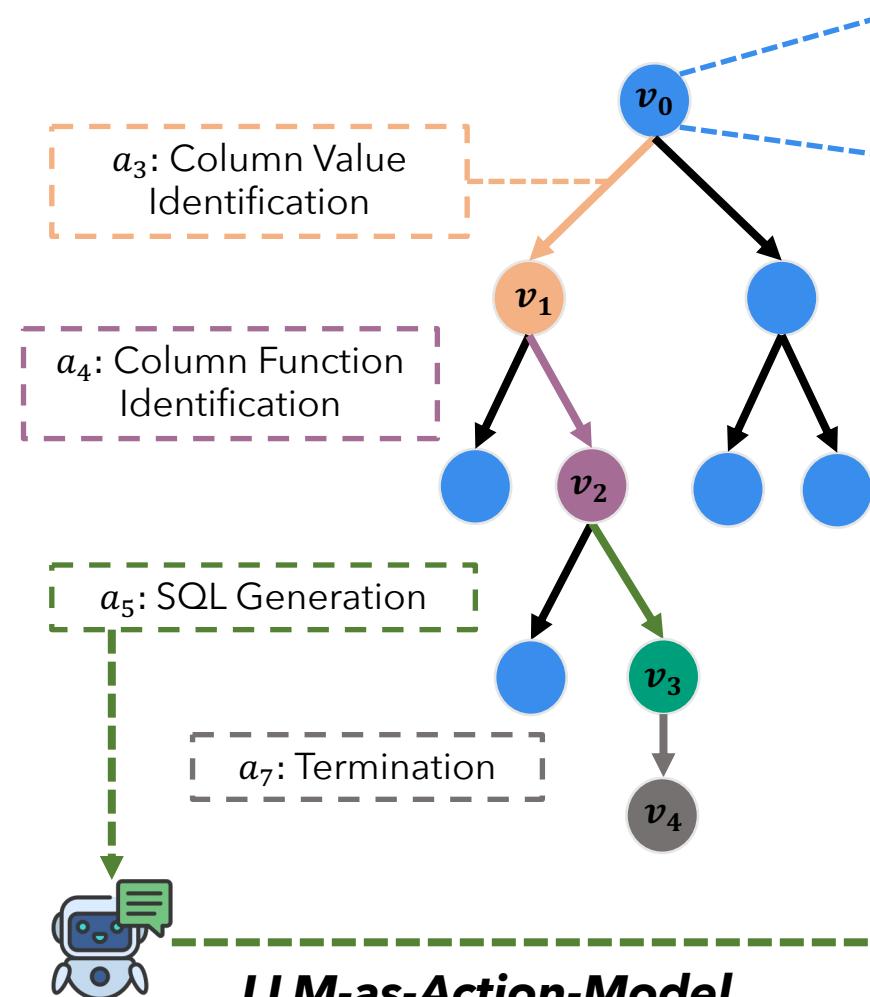
Action Space

- a_1 Rephrase Question
- a_2 Schema Selection
- a_3 Column Value Identification
- a_4 Column Function Identification
- a_5 SQL Generation
- a_6 SQL Revision
- a_7 Termination

LLM-as-Action-Model



Edges (Actions)



Nodes (Reasoning States)

q = "What's the rank of Bob in the football match?"
 D = "CREATE TABLE `players` (...)"

Column Value Thinking:

In the above question, there is a specific filter about match type and player name. So I need use `player`.`name` = 'Bob' and `match`.`match_type` = 'football'.

Column Function Thinking: ...

SQL Generation Thinking:

Based on my previous thoughts, I need a WHERE clause to filter the match type and player, and there is no functions needed. Thus, the final SQL query is:

```
SELECT T1.rank FROM players AS T1  
JOIN matches AS T2 ON T1.id =  
T2.player_id WHERE T1.name = 'Bob'  
AND T2.match_type = 'football';
```



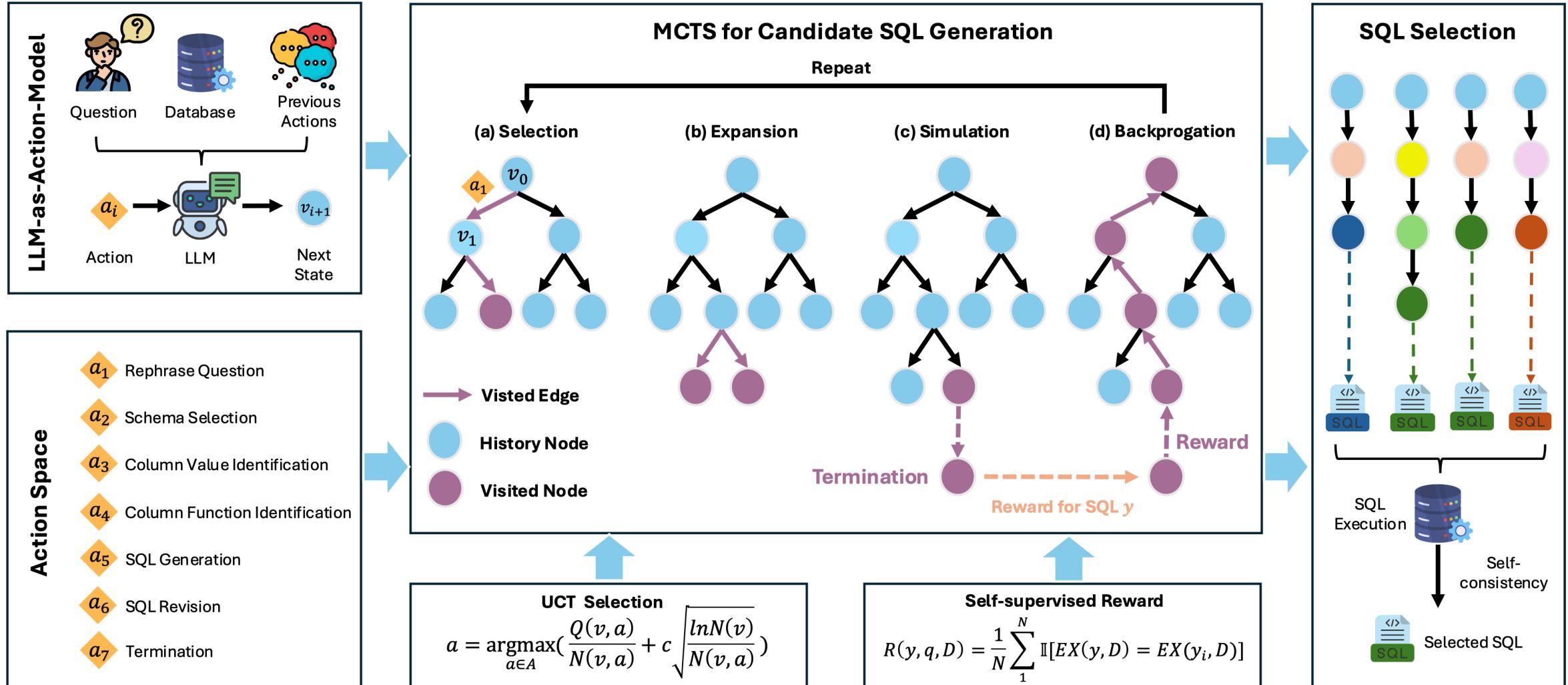
Output

Text-to-SQL as a Tree-based Search Problem

- **Q1**: How to select the next action (edge)?
- **Q2**: How to effectively navigate the vast search space?
- **Q3**: How to evaluate the quality of the candidate SQL queries?

- Q1 & Q2**
- Monte Carlo Tree Search (MCTS) addresses this by balancing *exploration* (testing *uncertain* actions) and *exploitation* (choosing actions *likely to yield good results*)
- Q3**
- We need a *self-supervised* reward function since our goal is to avoid reliance on labeled data
 - Resampling the LLMs M times to compute the self-consistent scores

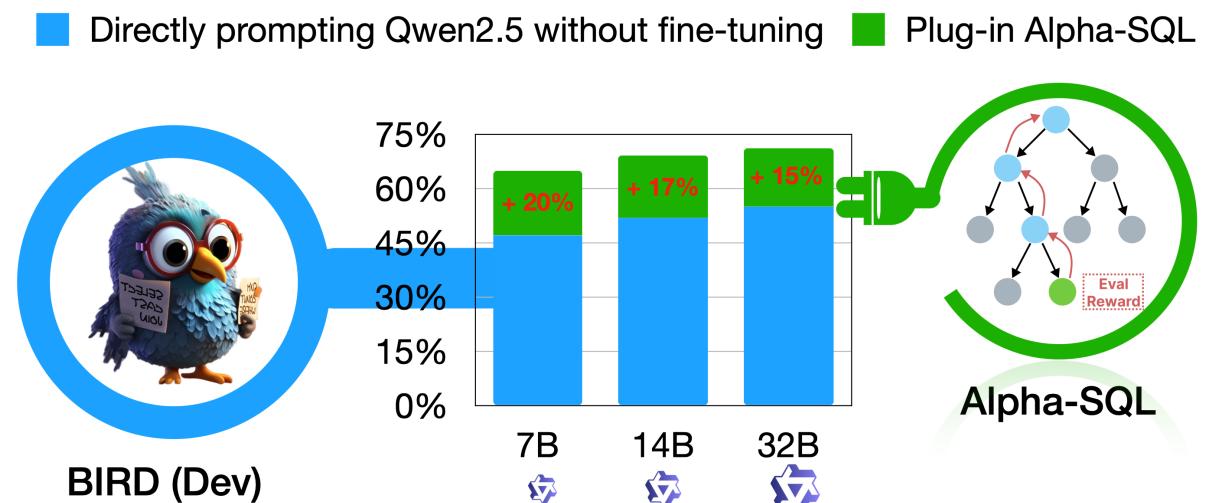
Alpha-SQL Solution Overview



Alpha-SQL: Plug-and-Play Capabilities

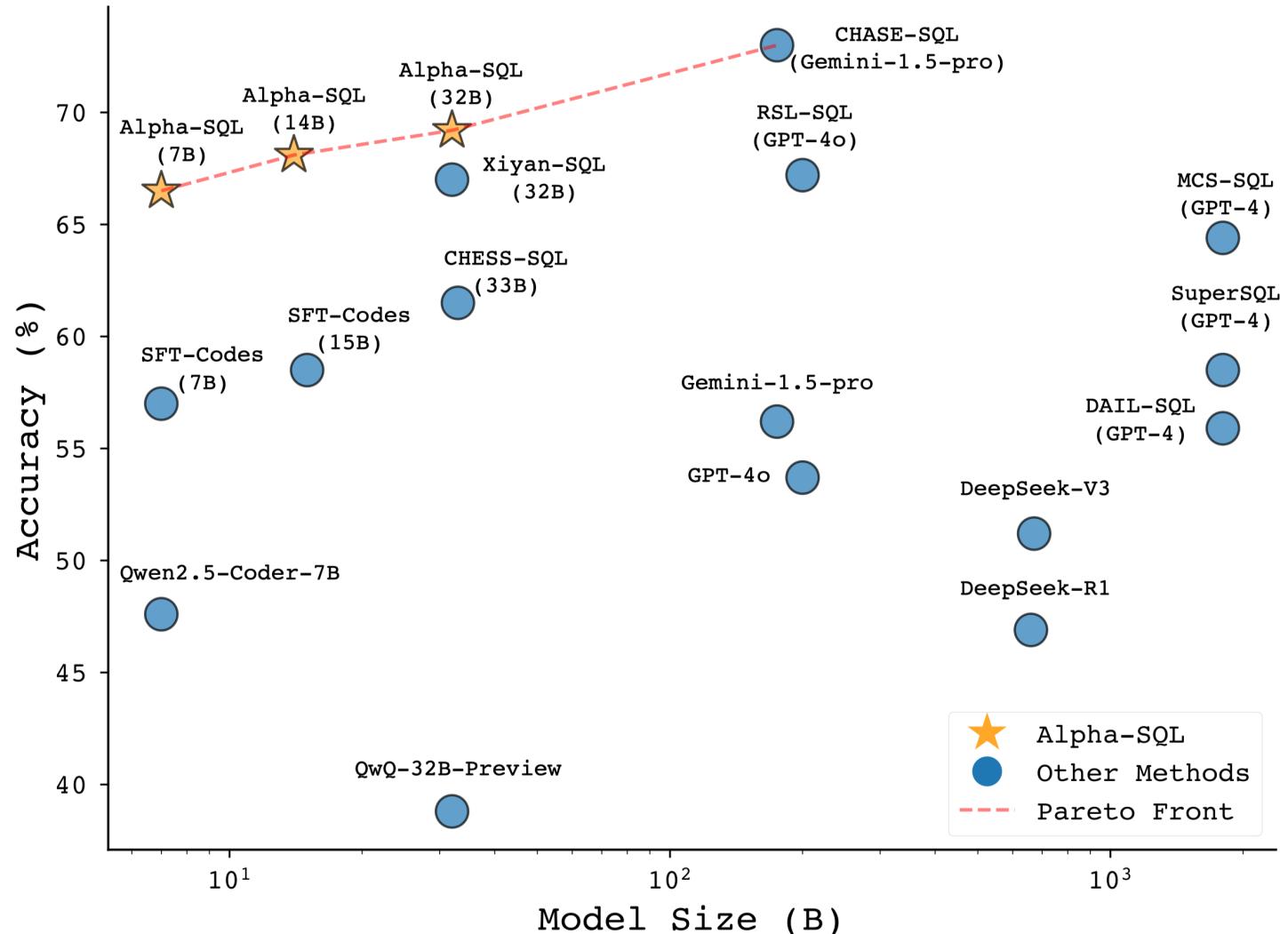
Table 4. Comparison with Baseline LLMs on the SDS dataset.

Model	Accuracy (%)
Deepseek-V3	51.2
GPT-4o	53.7
Gemini-1.5-Pro	56.2
QwQ-32B-Preview	38.8
DeepSeek-R1	50.3
Gemini-2.0-Flash-Thinking-Exp	60.8
Qwen2.5-Coder-7B	47.6
+ Alpha-SQL (Ours)	64.6 (↑ 17.0)
Phi-4	43.5
+ Alpha-SQL (Ours)	60.0 (↑ 16.5)



Performance-Scale Trade-off Analysis

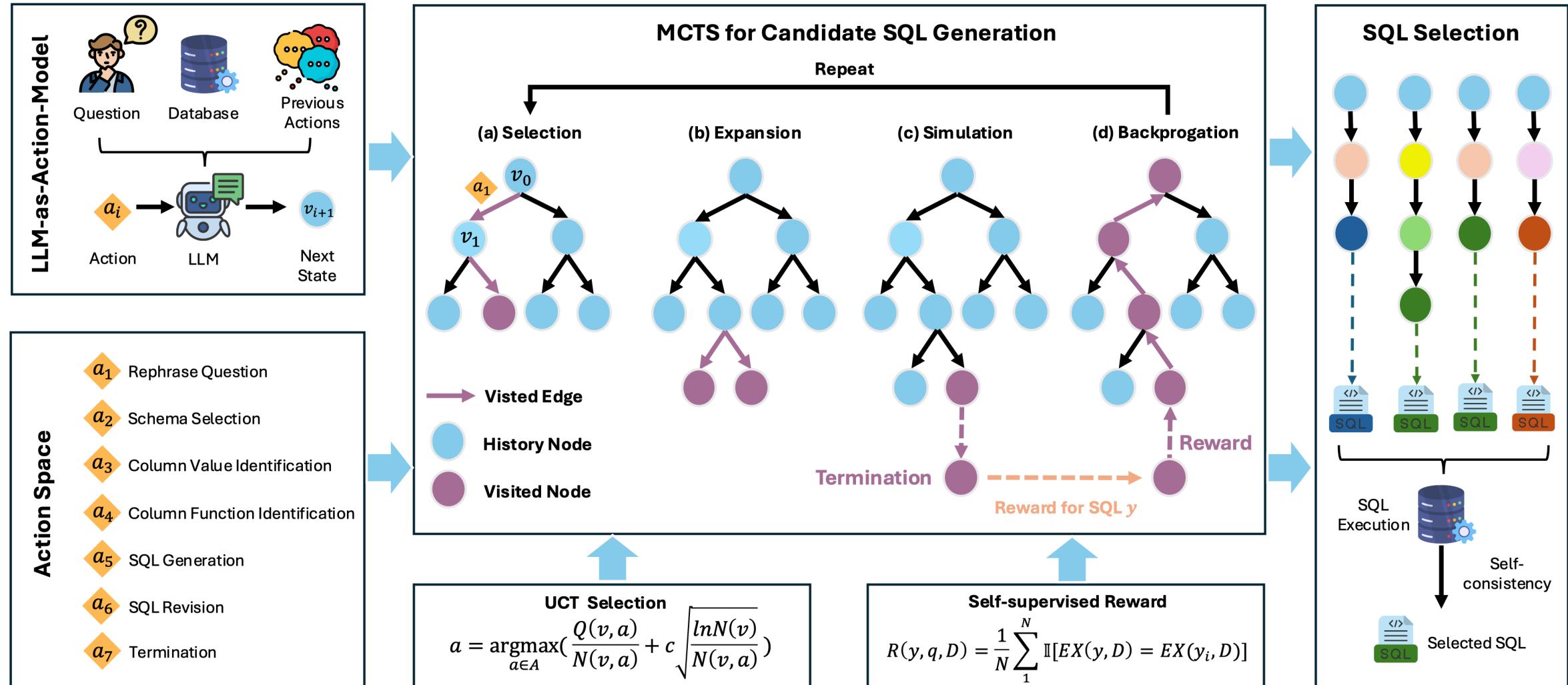
Agents: Small LLMs, Big Gains



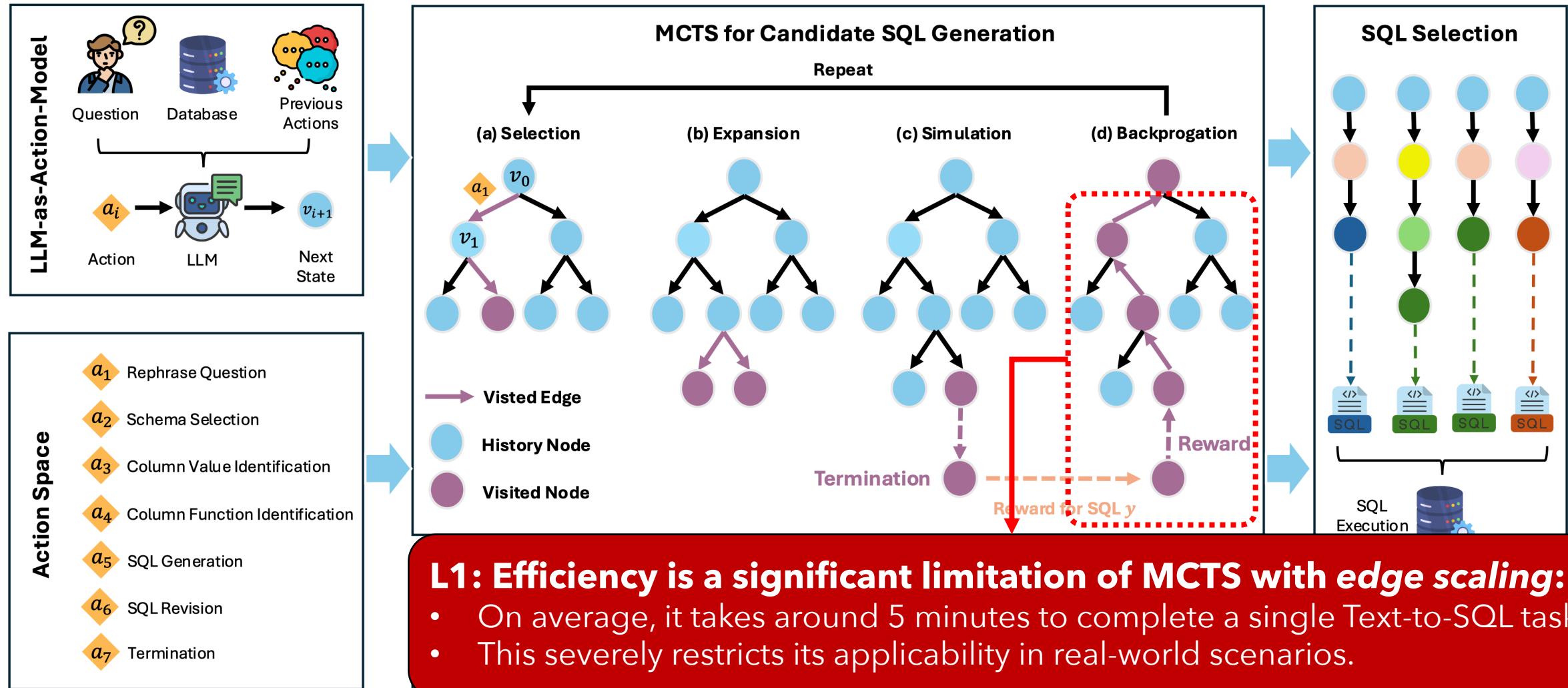
Tutorial Outline

- Problem Definition, Preliminaries, Benchmarks
- NL2SQL Solutions with PLMs and LLMs
- NL2SQL Solutions with LLM Agents
- **Open Problems**

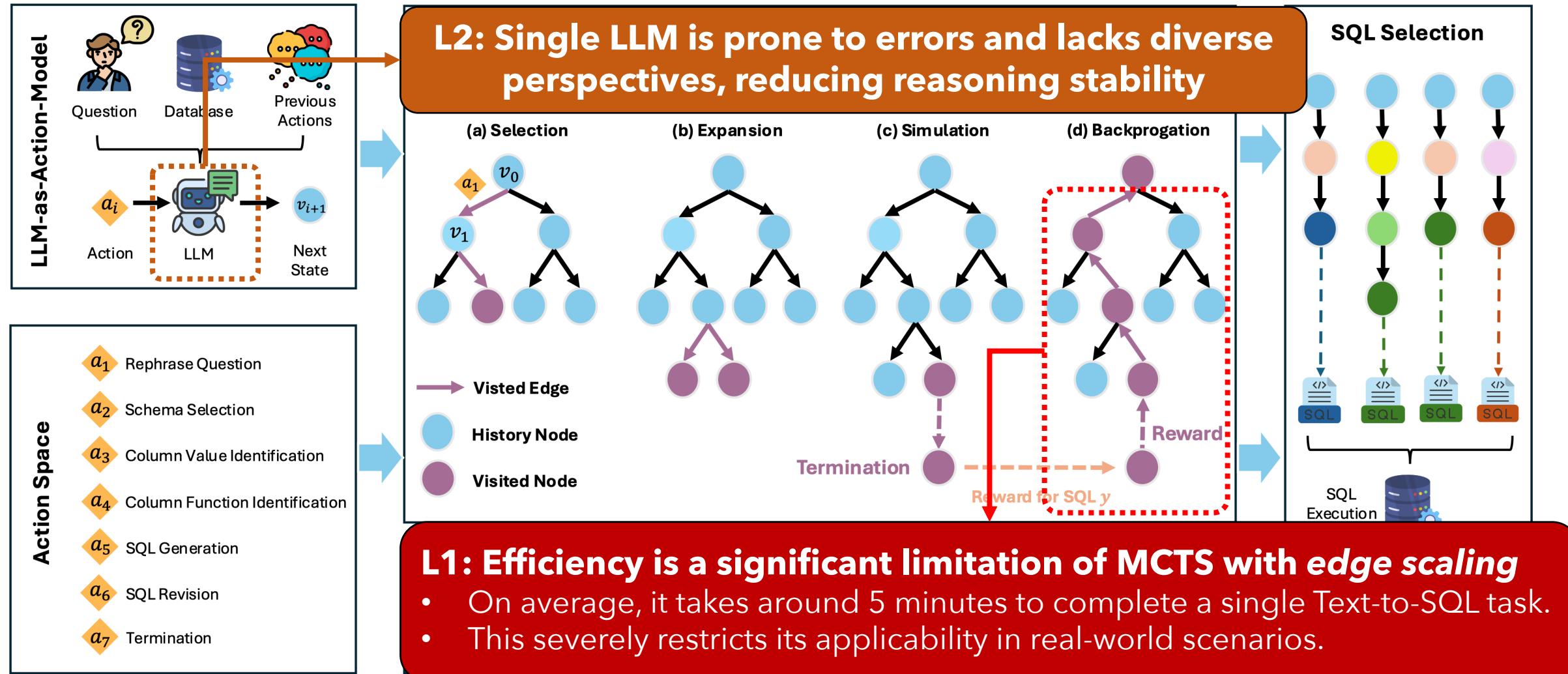
Rethinking: Limitations of NL2SQL Agent



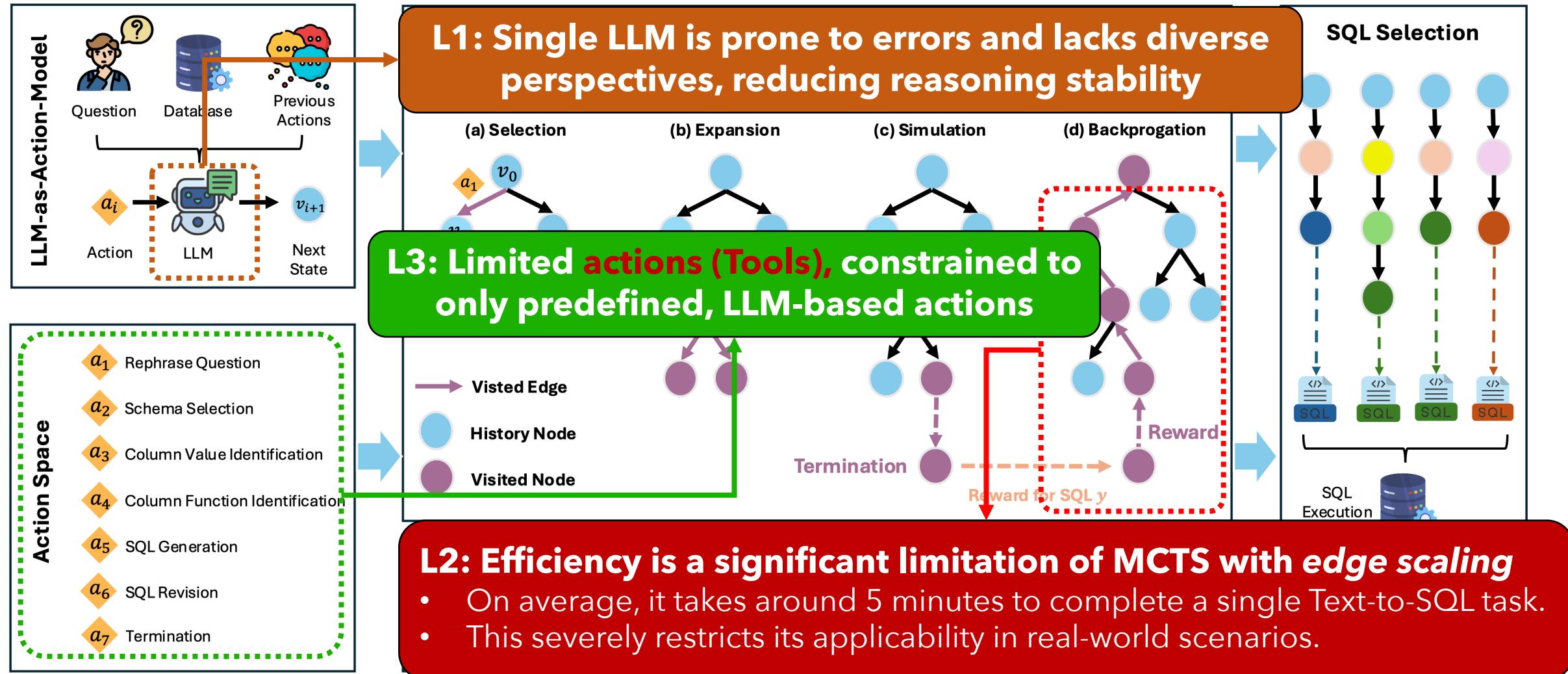
Rethinking: Limitations of NL2SQL Agent



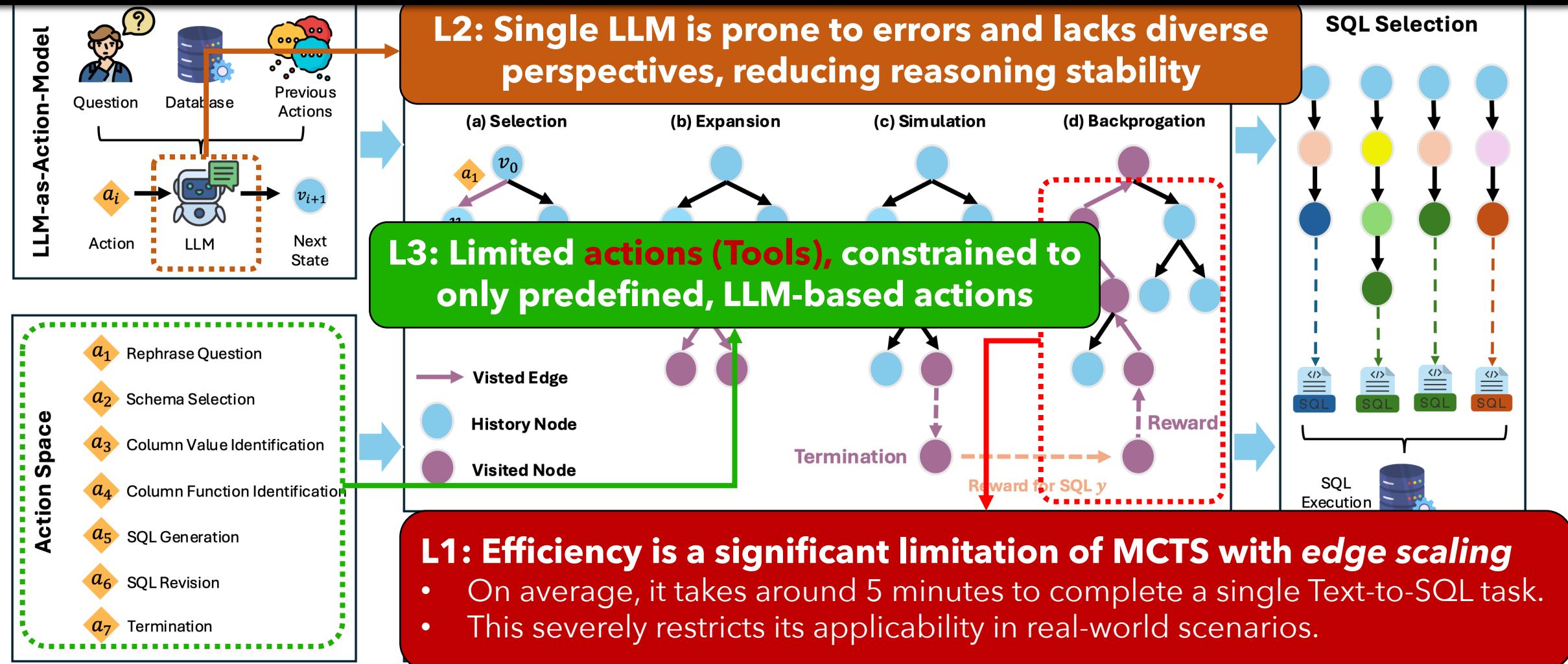
Rethinking: Limitations of NL2SQL Agent



Rethinking: Limitations of NL2SQL Agent



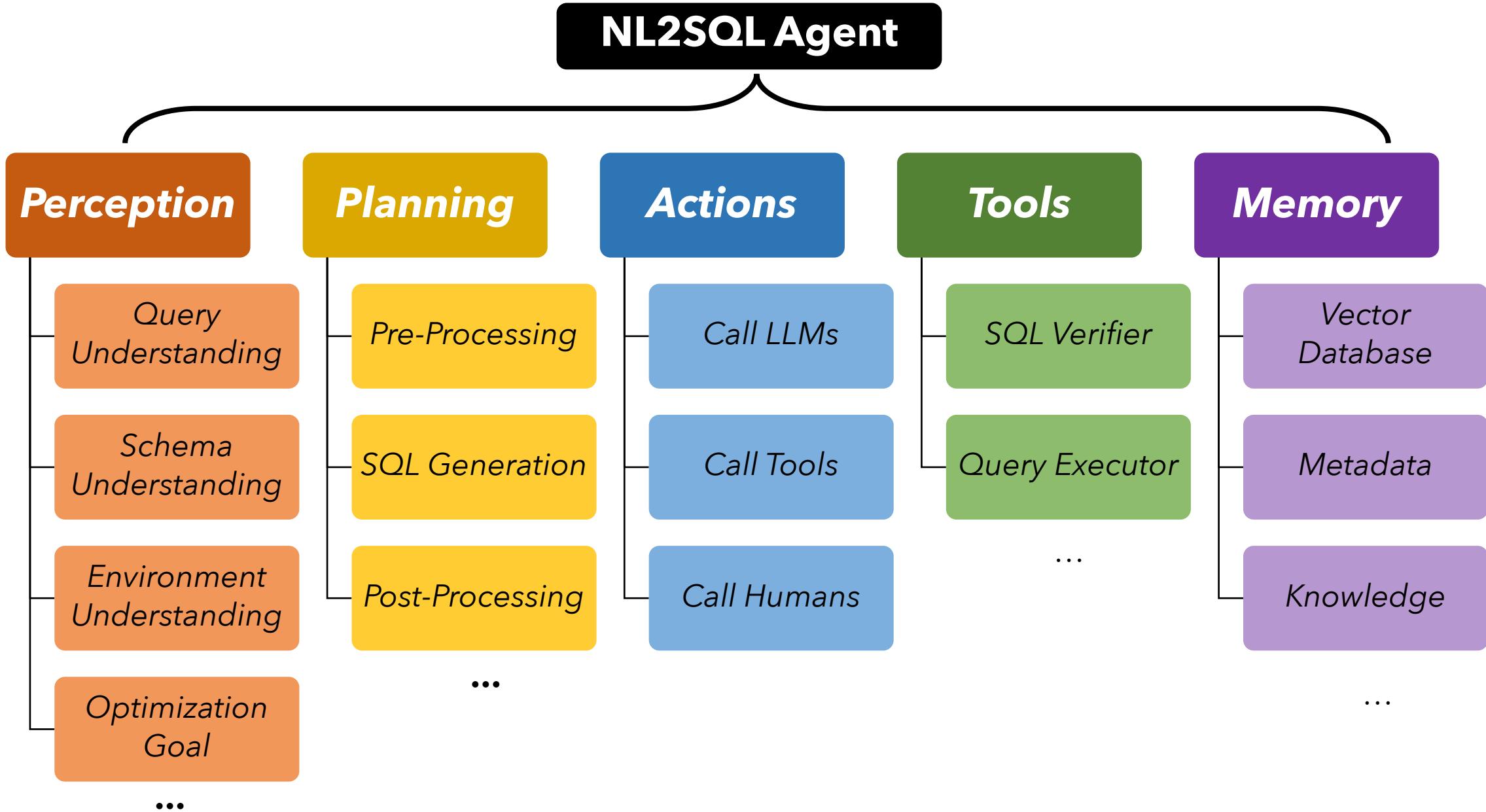
These limitations highlight the need for more diverse actions, efficient reasoning, and richer memory



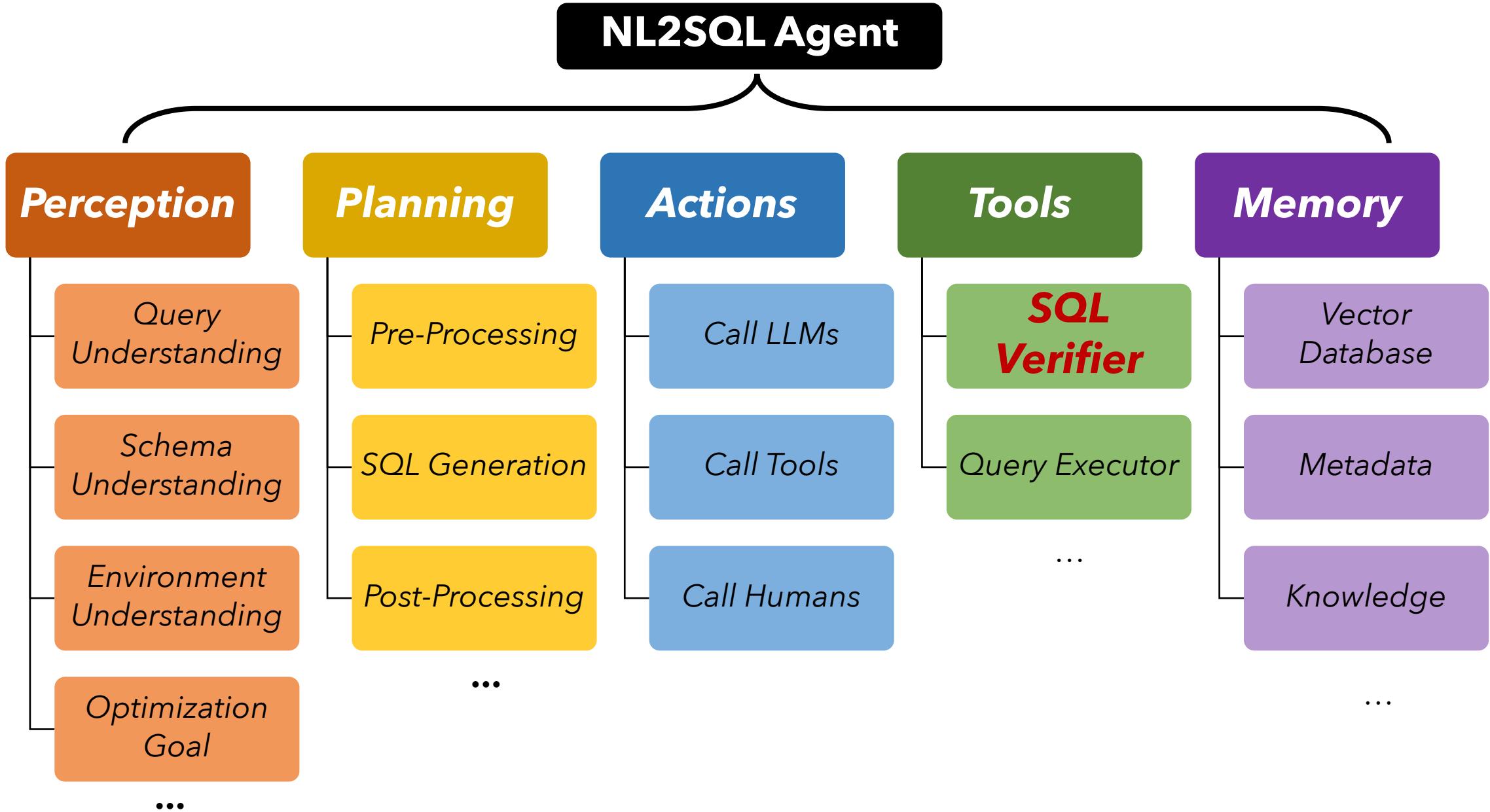
What Alpha-SQL Reveals About NL2SQL Agents

(Alpha-SQL) Limitation	Opportunity Axis	Design Lever (what to change)
L1. Efficiency bottleneck (MCTS is slow)	Planning	Adaptive search budgets, routing by query difficulty, test-time compute allocation
	Tools	Early pruning via validators / partial execution; cost-aware candidates
	Memory	Cache schema/context/results; reuse prior plans
L2. Reasoning instability / low diversity	Actions	Multi-agent/committee, self-consistency, <i>human-as-an-agent</i> for disambiguation
	Perception	Better query & schema understanding (scope detection, value grounding)
L3. Limited actions & tool use	Tools	Add retrievers, value lookups, execution-guided rewrite, SQL checkers
	Memory	(1) Long context + vector memory for task state & user prefs; (2) Metadata Management and Schema Interpretation

Opportunities for NL2SQL Agents: Five Key Aspects

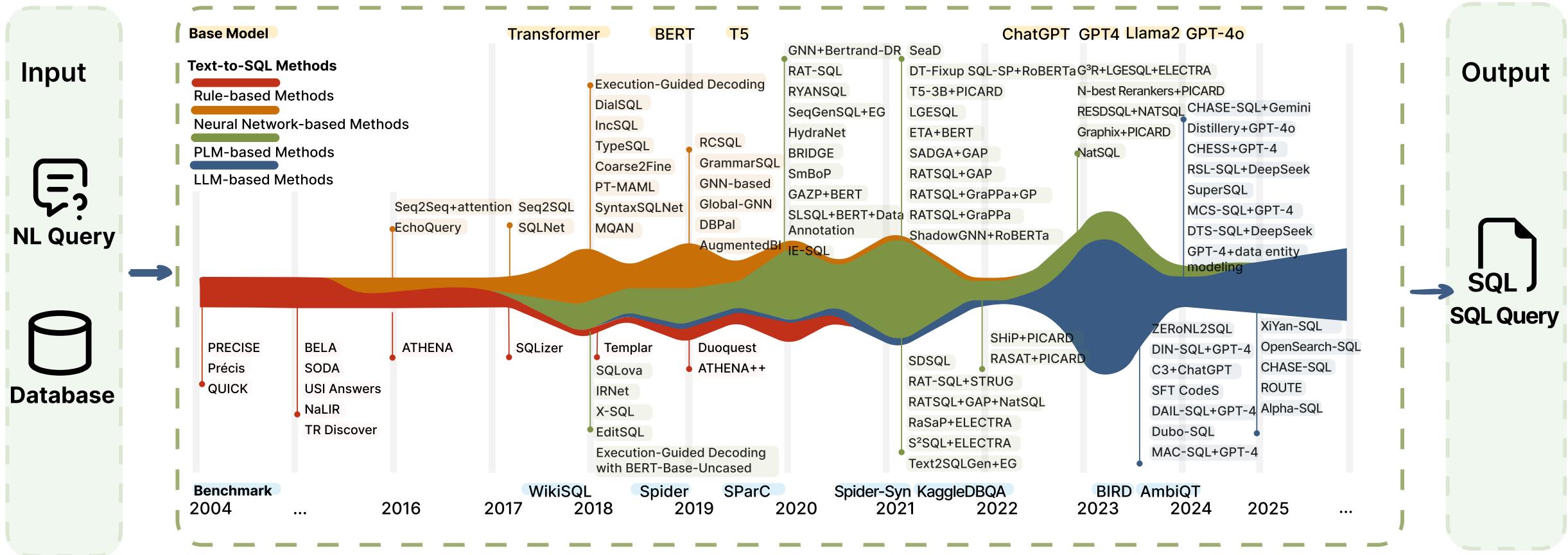


Opportunities for NL2SQL Agents: Five Key Aspects





Is your model reliable?
You can't achieve 100% accuracy.



😔 Execution ACC ~75%



BIRD-SQL

A Big Bench for Large-Scale Database Grounded Text-to-SQLs

About BIRD

Page Views 177553

BIRD (Big Bench for LaRge-scale Database Grounded Text-to-SQL Evaluation) represents a pioneering, cross-domain dataset that examines the impact of extensive database contents on text-to-SQL parsing. BIRD contains over 12,751 unique question-SQL pairs, 95 big databases with a total size of 33.4 GB. It also covers more than 37 professional domains, such as blockchain, hockey, healthcare and education, etc.

Paper

Code

Mini-Dev (500)

BIRD-CRITIC 1.0 (SQL)

Leaderboard - Execution Accuracy (EX)

Model	Code	Size	Oracle Knowledge	Dev (%)	Test (%)
Human Performance <i>Data Engineers + DB Students</i>			✓		92.96
1 Contextual-SQL <i>Contextual AI</i> Feb 27, 2025	UNK		✓	73.50	75.63
2 XIYan-SQL <i>Alibaba Cloud</i> [Yifu Liu et al. '24] Dec 17, 2024	[link]	UNK	✓	73.34	75.63
3 CHASE-SQL + Gemini <i>Google Cloud</i> [Pourreza et al. '24] Nov 24, 2024	UNK		✓	74.46	74.79
4 DSAIR + GPT-4o <i>AT&T - CDO</i> Nov 11, 2024	UNK		✓	74.32	74.12
5 ExSL + granite-34b-code					

<https://bird-bench.github.io/>

😔 Execution ACC ~30%

Spider 2.0

Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows

ICLR 2025 Oral

About Spider 2.0

Spider 2.0 is an evaluation framework comprising 632 real-world text-to-SQL workflow problems derived from enterprise-level database use cases. The databases in Spider 2.0 are sourced from real data applications, often containing over 1,000 columns and stored in local or cloud database systems such as BigQuery and Snowflake. This challenge calls for models to interact with complex SQL workflow environments, process extremely long contexts, perform intricate reasoning, and generate multiple SQL queries with diverse operations, often exceeding 100 lines, which goes far beyond traditional text-to-SQL challenges.

↗ Paper

🔗 Code

🐦 Twitter

↗ Submit

Leaderboard

Spider 2.0-Snow Spider 2.0-lite Spider 2.0

[Spider 2.0-Snow](#) is a self-contained text-to-SQL task that includes well-prepared database metadata and documentation, includes 547 examples, all hosted on [Snowflake](#), which offers participants free quotas. If you want to test performance on [a single SQL dialect](#), don't hesitate to use [Spider 2.0-Snow](#).

Rank	Method	Score
1 Jan 28, 2025	ReFoCE + o1-preview <i>Hao AI Lab x Snowflake</i> [Deng et al. '25]	31.26
2 Nov 30, 2024	Spider-Agent + o1-preview	23.58

<https://spider2-sql.github.io/>

Types of Errors That Require Verification



NL Query

What are the ids of high school students who do not have friends?

Predict SQL
Syntax Error

Database
highschooler
id name ...

friend
student_id friend_id ...

ooler

sult

umn: id

Predict SQL
Semantic Error

SELECT id
FROM highschooler

INTERSECT
SELECT student_id
FROM friend;

Execution result

DBMS:
1247, 1304,
1316 ...

Large proportion
Difficult to Detect

Gold Answer:



Gold SQL

SELECT id
FROM highschooler
EXCEPT
SELECT student_id
FROM friend;



Execution result

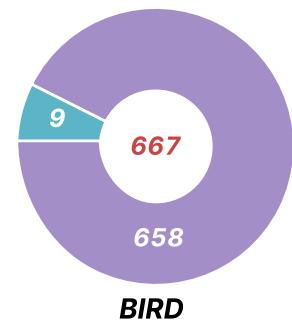
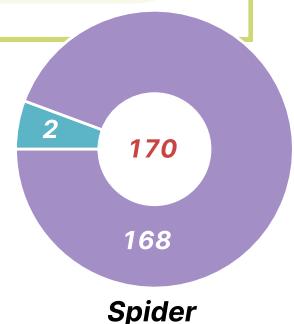


DBMS: Less proportion
1025, 1641 Easy to Detect

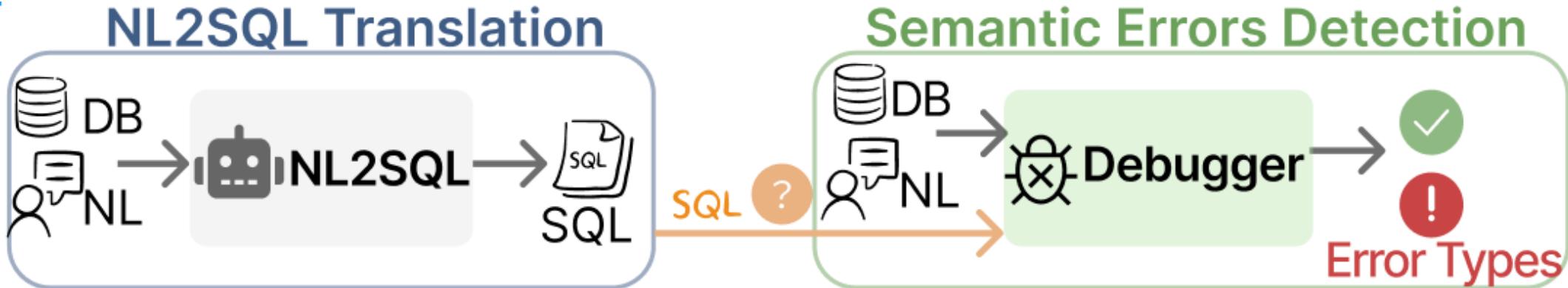
Total Errors

Syntax Errors

Semantic Errors



Semantic Errors Detection



Question:

List all students and their course grades. (including students who haven't taken any courses)

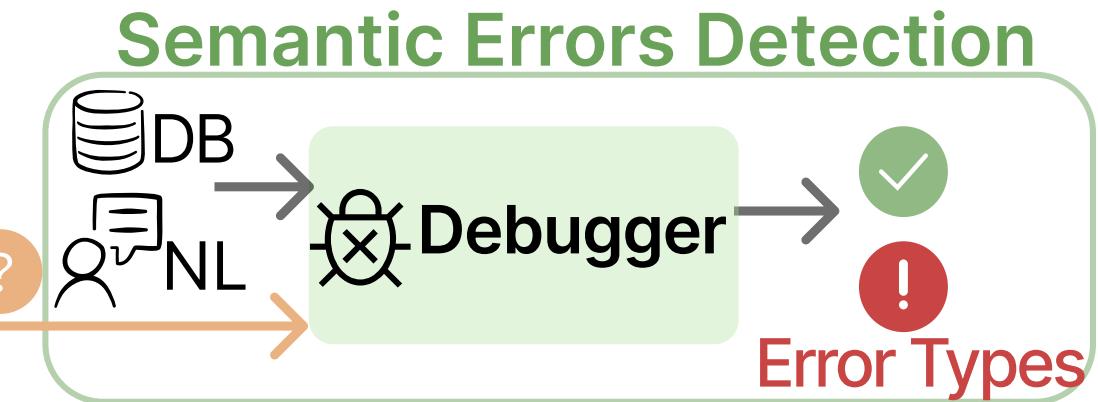
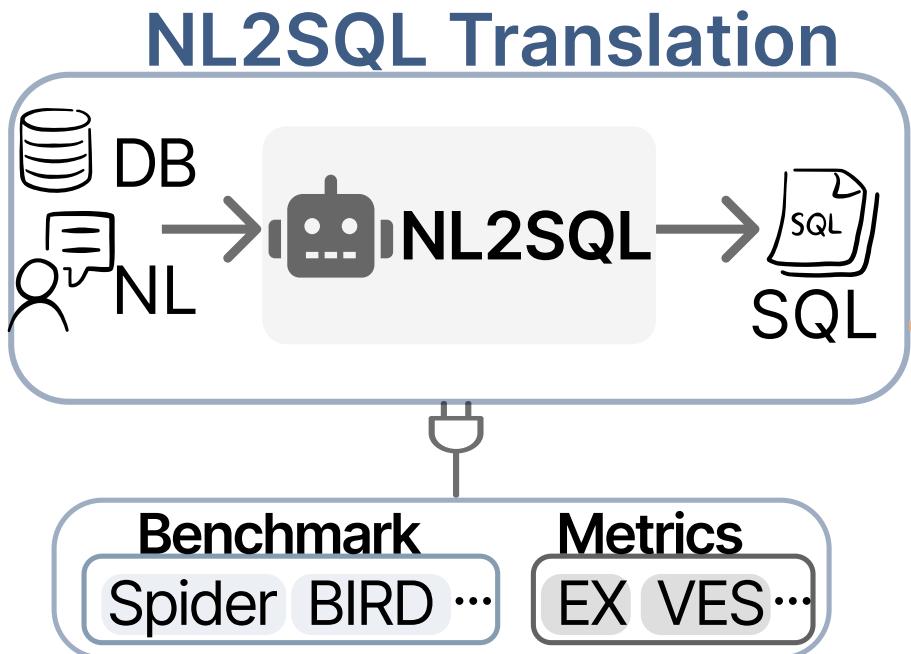
Predicted SQL by NL2SQL methods:

```
SELECT s.name, e.grade  
FROM student s  
INNER JOIN enrollment e  
ON s.id = e.id
```



This SQL is incorrect. The join type is mismatched, and the foreign key connection is incorrect.

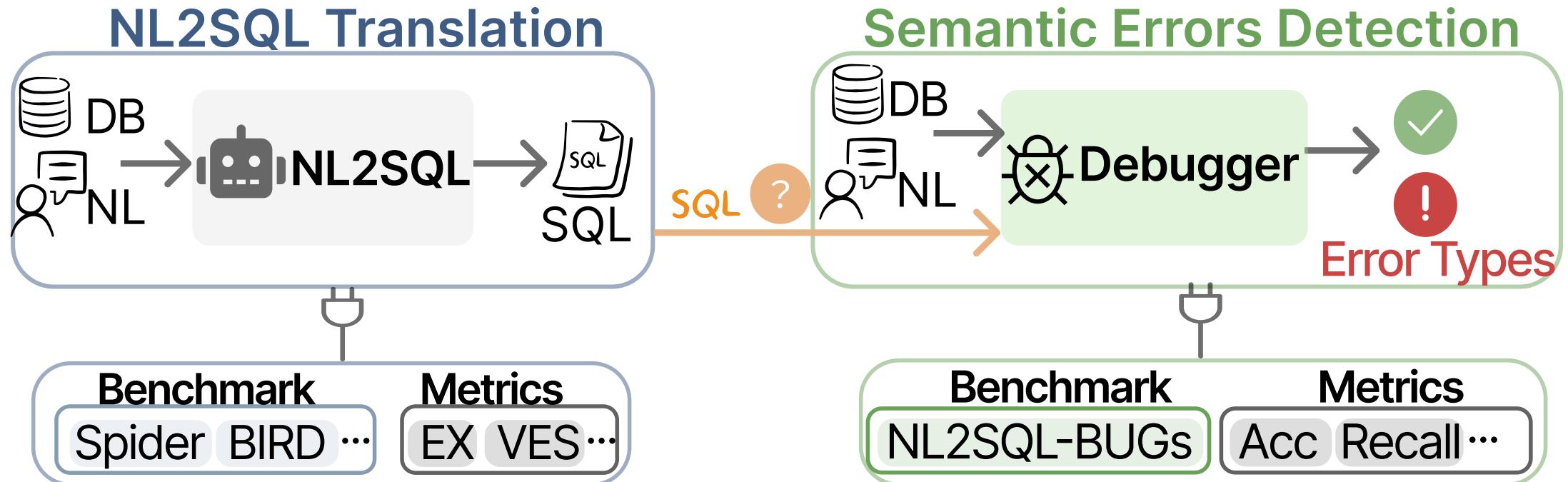
Research Gap: Lack of Robust Verifiers



Lack of Research



NL2SQL-BUGs Benchmark for Verifier



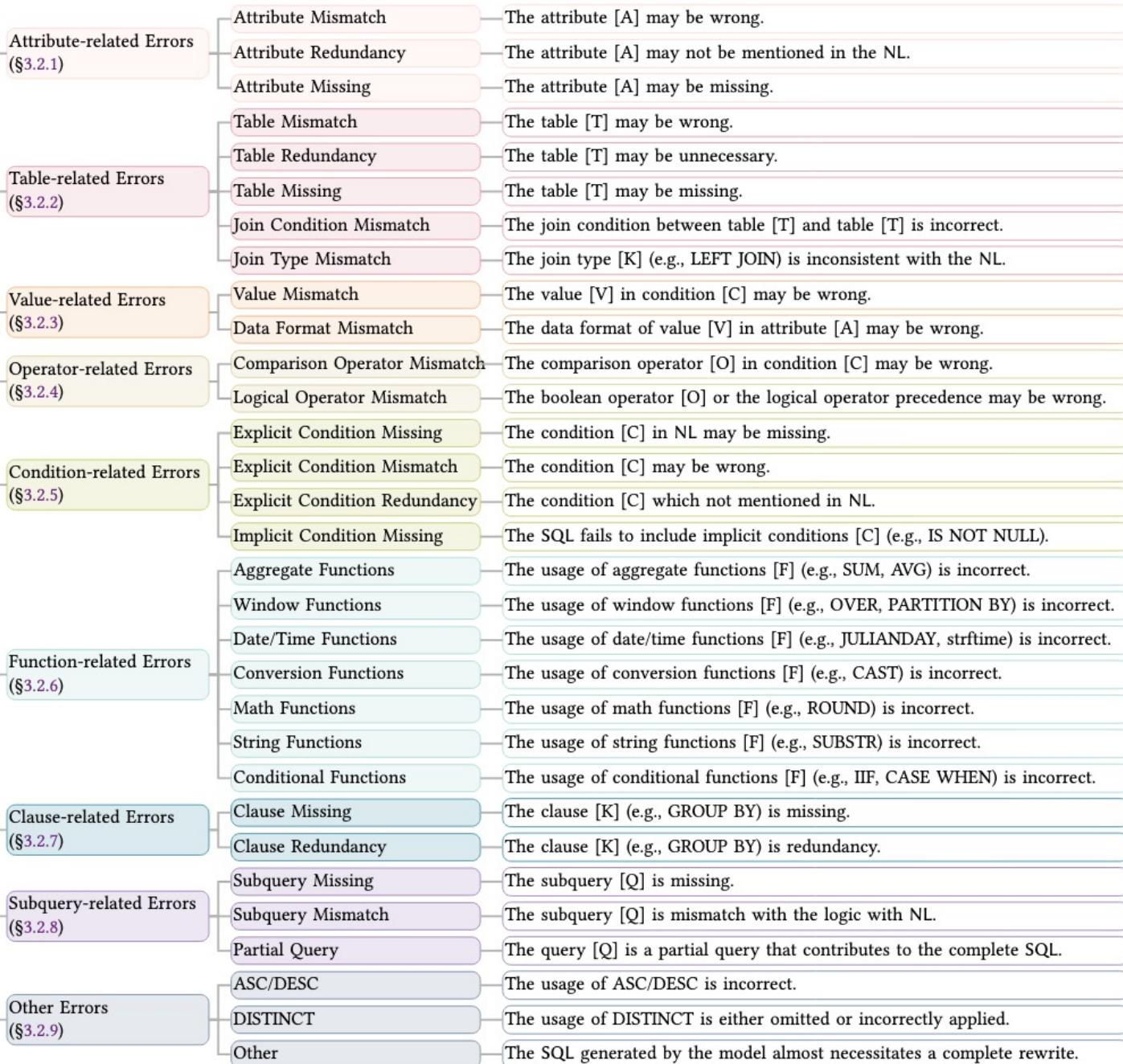
Error Taxonomy

To systematically analyze semantic errors, we propose a two-level taxonomy with

9 main type
31 subtype

to analysis semantic errors in NL2SQL translation.

The Taxonomy of
NL2SQL Translation
Semantic Errors





SQL

```
SELECT Fname, Sex
FROM Student
WHERE StuID IN (
    SELECT StuID
    FROM Has_Pet
    GROUP BY StuID
    HAVING count(PetID) > 1)
```

Matching case structure



SQL

```
SELECT DISTINCT
model_list.Model
FROM model_list
JOIN cars_data ON
model_list.ModelId =
cars_data.Id
WHERE cars_data.Year > 1980
```



NL
Find the first name and gender of student who have more than one pet.



pets_1
Label

True



NL
Which distinct car models are the produced after 1980?



car_1

Label

False

Error Type

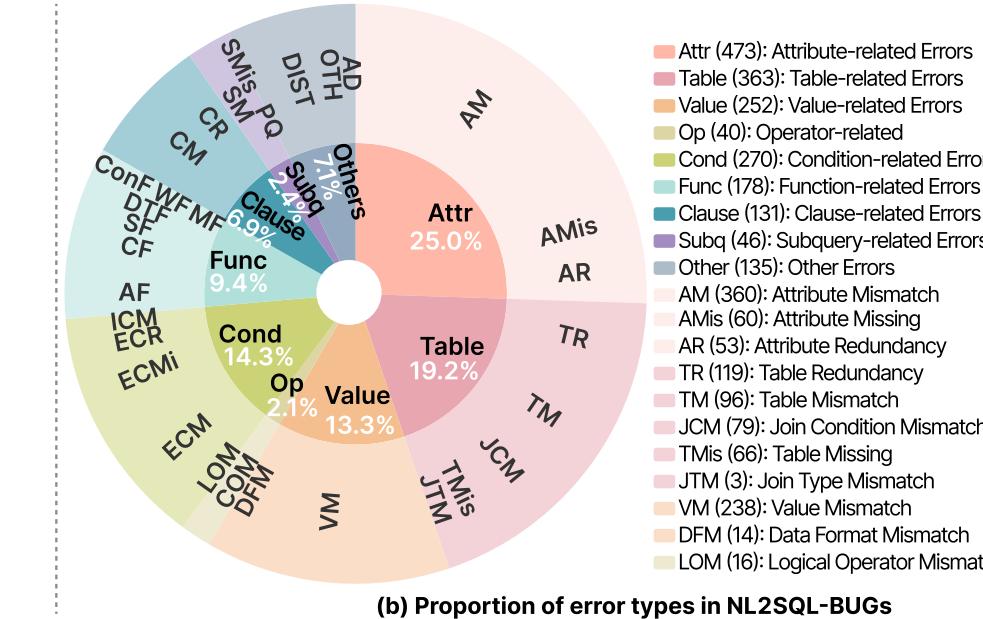
```
{ "MainType": "Table Error",
"SubType": "Condition Error"},  

{ "MainType": "Table Error",
"SubType": "Table Missing"}
```

Mismatched case structure

(a) Data Structure of NL2SQL-BUGs

NL2SQL-BUGs Benchmark



(b) Proportion of error types in NL2SQL-BUGs

2,018 expert-annotated examples, 1,019 correct examples, 999 incorrect examples

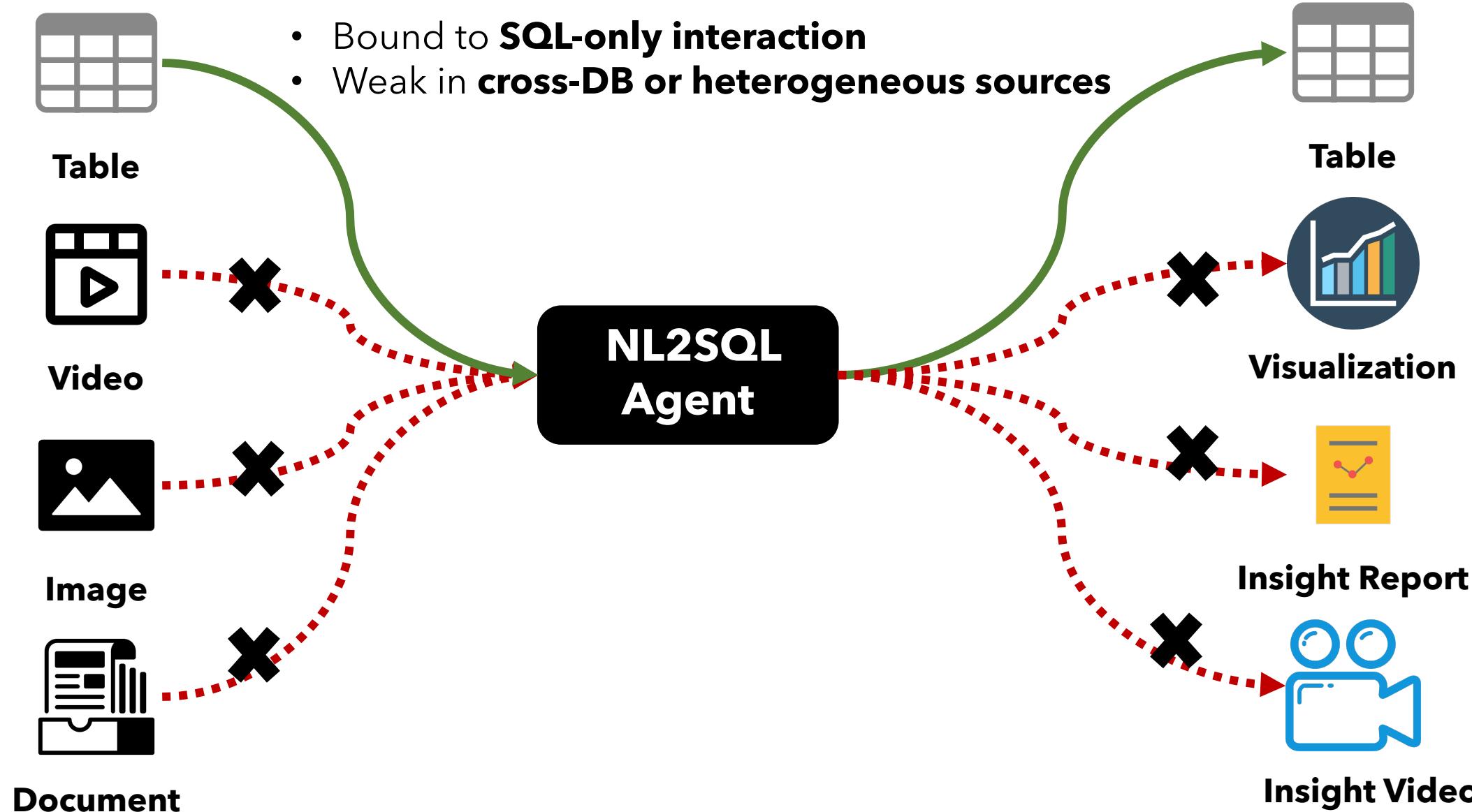
Xinyu Liu, Shuyu Shen, Boyan Li, Nan Tang, Yuyu Luo:
NL2SQL-BUGs: A Benchmark for Detecting Semantic Errors in NL2SQL Translation. SIGKDD 2025

- Attr (473): Attribute-related Errors
- Table (363): Table-related Errors
- Value (252): Value-related Errors
- Op (40): Operator-related
- Cond (270): Condition-related Errors
- Func (178): Function-related Errors
- Clause (131): Clause-related Errors
- Subq (46): Subquery-related Errors
- Other (135): Other Errors
- AM (360): Attribute Mismatch
- AMis (60): Attribute Missing
- AR (53): Attribute Redundancy
- TR (119): Table Redundancy
- TM (96): Table Mismatch
- JCM (79): Join Condition Mismatch
- TMis (66): Table Missing
- JTM (3): Join Type Mismatch
- VM (238): Value Mismatch
- DFM (14): Data Format Mismatch
- LOM (16): Logical Operator Mismatch
- DFN (14): Data Format Mismatch
- ECM (118): Explicit Condition Missing
- ECMi (95): Explicit Condition Mismatch
- ECR (29): Explicit Condition Redundancy
- ICM (28): Implicit Condition Missing
- AF (71): Aggregate Functions
- CF (43): Conditional Functions
- SF (19): String Functions
- DTF (21): Date/Time Functions
- ConF (11): Conversion Functions
- MF (10): Math Functions
- WF (3): Window Functions
- CM (94): Clause Missing
- CR (37): Clause Redundancy
- SM (33): Subquery Missing
- SMis (7): Subquery Mismatch
- PQ (6): Partial Query
- DIST (88): DISTINCT
- OTH (35): Other
- AD (12): ASC/DESC

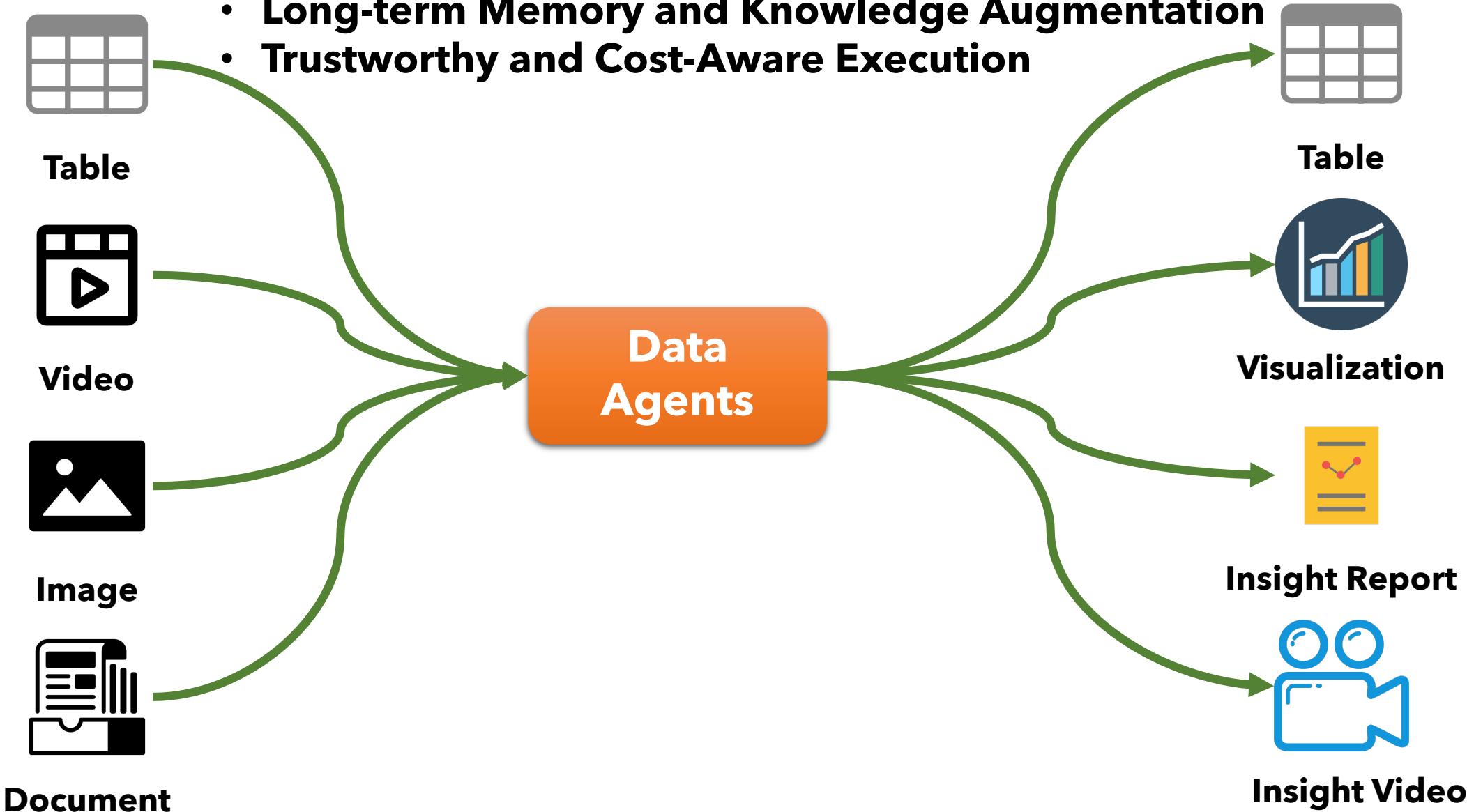
Opportunities: NL2SQL Agents

- **Human-as-an-Agent and Human-in-the-Reasoning-Loop**
 - How can we *dynamically integrate human experts into the reasoning loop* to address complex tasks beyond LLM agents' current capabilities and clarify the question ambiguities?
- **Explainable and Interpretable SQL Reasoning Agents**
 - Users typically require explanations for the reasoning steps and decisions underlying SQL generation (i.e., *knowing both "what" and "why"*).
 - How can we design reasoning agents that transparently communicate their thought processes, decisions, and final SQL statements to improve system transparency and foster user trust?
- **Metadata Management and Schema Interpretation**
 - Real-world databases commonly feature complex schemas, detailed metadata (e.g., column annotations, table descriptions, foreign key constraints, data types).
 - How can we enable *data agents* to effectively extract, manage, and utilize this metadata to generate more accurate semantic mappings, informed reasoning processes, and precise SQL generation?

Are NL2SQL Agents Enough?

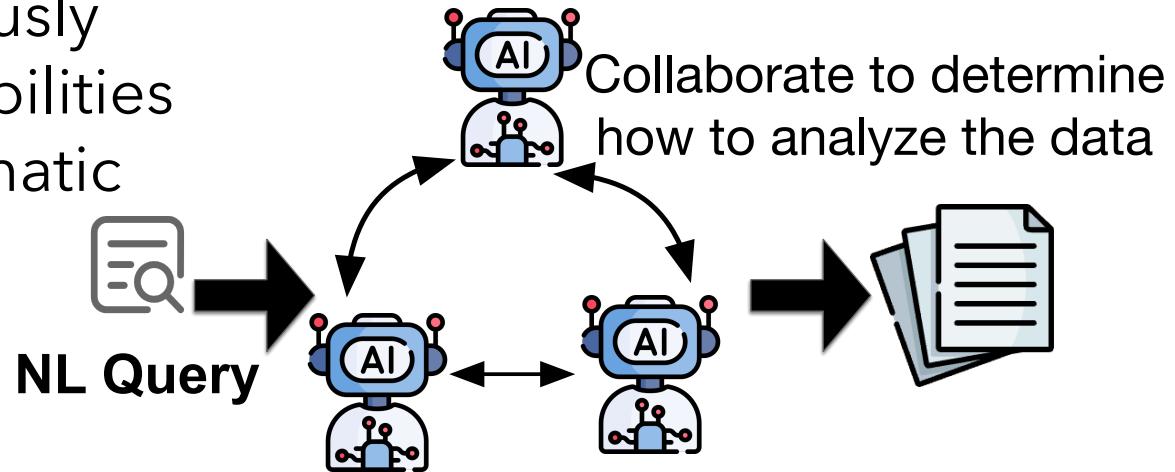


- **Unified Query Interface:** SQL → Semantic Operators
- **Multimodal Data Analysis**
- **Adaptive Reasoning and Orchestration**
- **Long-term Memory and Knowledge Augmentation**
- **Trustworthy and Cost-Aware Execution**



Data Agent

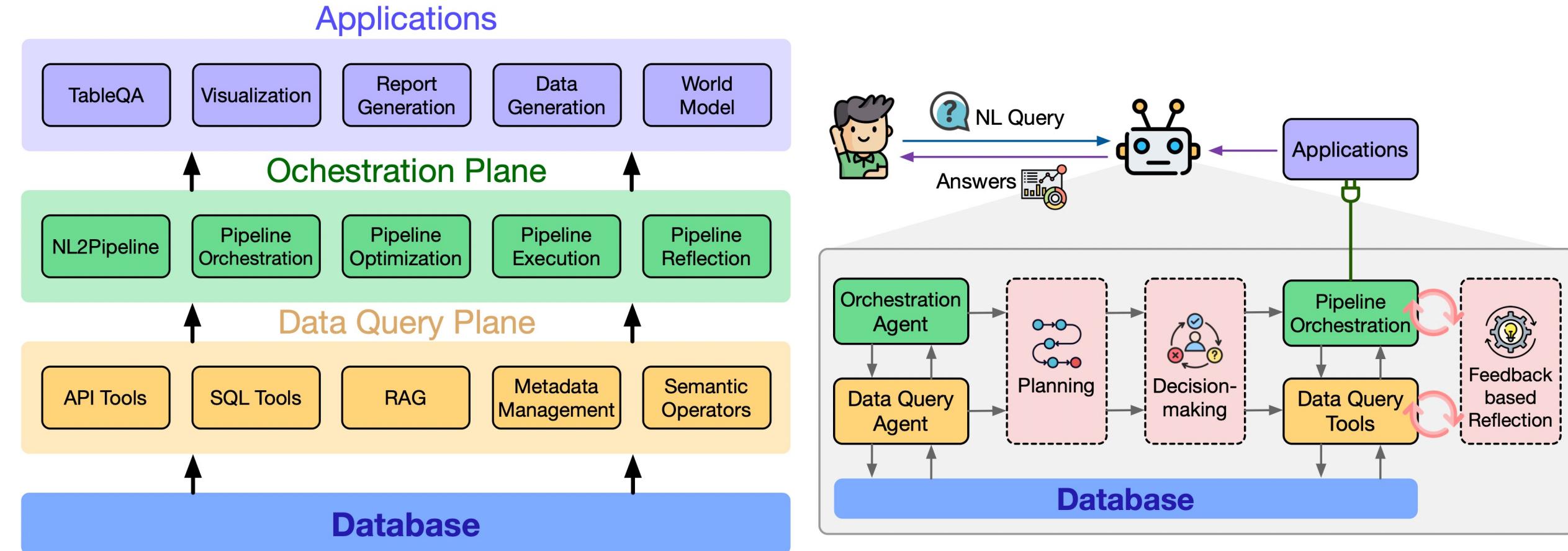
□ **Data Agent:** designed to autonomously carry out data-related tasks with capabilities for knowledge comprehension, automatic planning, and self-reflection of LLMs

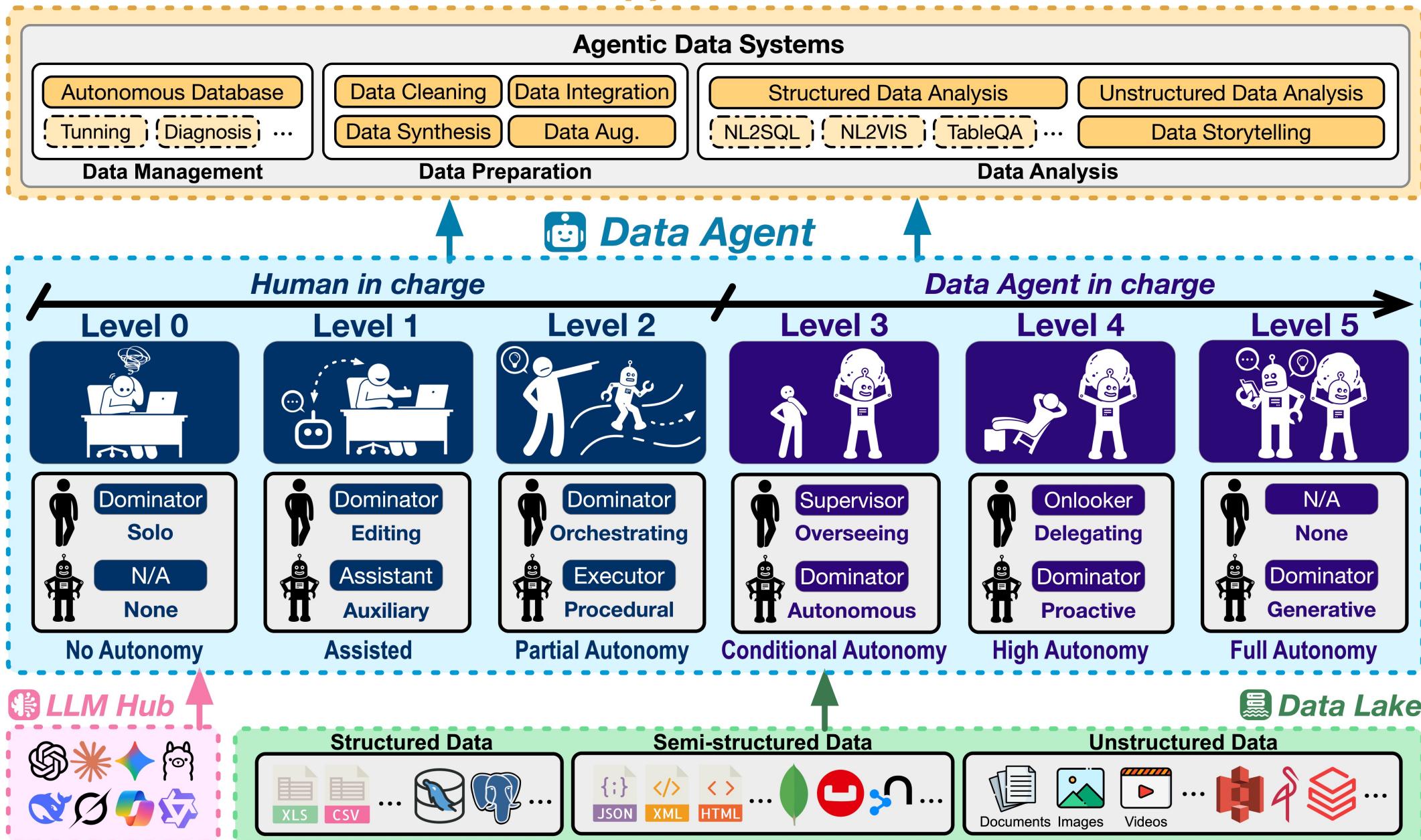


□ Challenges:

- How can data agents **understand** queries, data, other agents, and tools?
- How can data agents **orchestrate** effective and efficient pipelines to bridge the gaps between user requirements and underlying heterogeneous data?
- How to **schedule and coordinate** agents/tools to improve effectiveness?

Data Agent: A High-level View

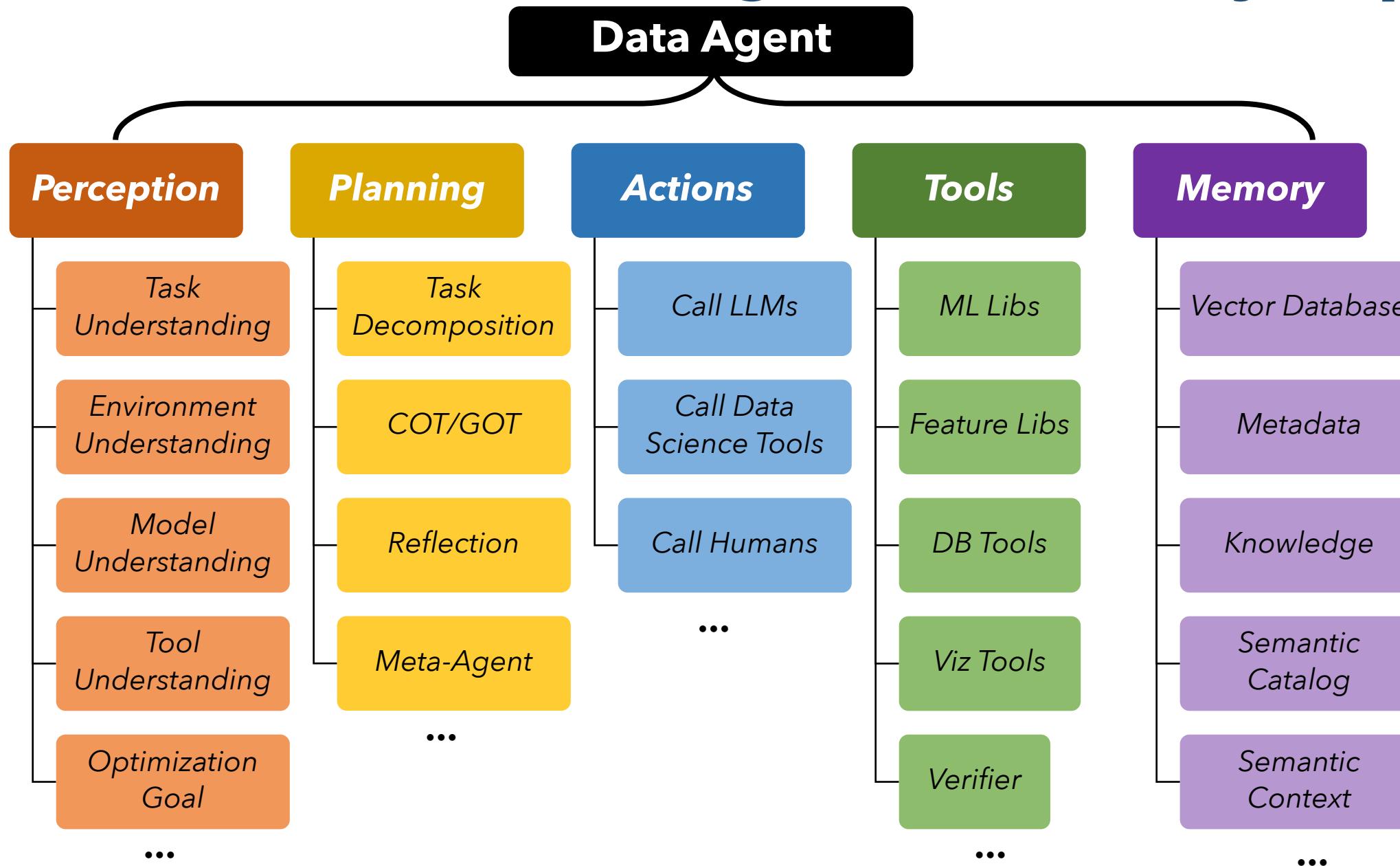




#	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	User_Count	Rating	Critic_Score	Critic_Count	User_Score
2	Wii Sports	Wii	2006	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	322	E	76	51	8
3	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	709	E	82	73	8.3
4	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	192	E	80	73	8
5	New Super Mario Bros.	DS	2006	Platform	Nintendo	11.28	9.14	6.5	2.88	29.8	431	E	89	65	8.5
6	Wii Play	Wii	2006	Misc	Nintendo	13.96	9.18	2.93	2.84	28.92	129	E	58	41	6.6
7	New Super Mario Bros. Wii	Wii	2009	Platform	Nintendo	14.44	6.94	4.7	2.24	28.32	594	E	87	80	8.4
8	Mario Kart DS	DS	2005	Racing	Nintendo	9.71	7.47	4.13	1.9	23.21	464	E	91	64	8.6
9	Wii Fit	Wii	2007	Sports	Nintendo	8.92	8.03	3.6	2.15	22.7	146	E	80	63	7.7
10	Kinect Adventures!	X360	2010	Misc	Microsoft Game Studios	15	4.89	0.24	1.69	21.81	106	E	61	45	6.3
11	Wii Fit Plus	Wii	2009	Sports	Nintendo	9.01	8.49	2.53	1.77	21.79	52	E	80	33	7.4
12	Grand Theft Auto V	PS3	2013	Action	Take-Two Interactive	7.02	9.09	0.98	3.96	21.04	3994	M	97	50	8.2
13	Grand Theft Auto: San Andreas	PS2	2004	Action	Take-Two Interactive	9.43	0.4	0.41	10.57	20.81	1588	M	95	80	9
14	Brain Age: Train Your Brain in Minutes a Day	DS	2005	Misc	Nintendo	4.74	9.2	4.16	2.04	20.15	50	E	77	58	7.9
15	Grand Theft Auto V	X360	2013	Action	Take-Two Interactive	9.66	5.14	0.06	1.41	16.27	3711	M	97	58	8.1
16	Grand Theft Auto: Vice City	PS2	2002	Action	Take-Two Interactive	8.41	5.49	0.47	1.78	16.15	730	M	95	62	8.7
17	Brain Age 2: More Training in Minutes a Day	DS	2005	Puzzle	Nintendo	3.43	5.35	5.32	1.18	15.29	19	E	77	37	7.1
18	Gran Turismo 3: A-Spec	PS2	2001	Racing	Sony Computer Entertainment	6.85	5.09	1.87	1.16	14.98	314	E	95	54	8.4
19	Call of Duty: Modern Warfare 3	X360	2011	Shooter	Activision	9.04	4.24	0.13	1.32	14.73	8713	M	88	81	3.4
20	Call of Duty: Black Ops	X360	2010	Shooter	Activision	9.7	3.68	0.11	1.13	14.61	1454	M	87	89	6.3
21	Call of Duty: Black Ops II	PS3	2012	Shooter	Activision	4.99	5.73	0.65	2.42	13.79	922	M	83	21	5.3
22	Call of Duty: Black Ops II	X360	2012	Shooter	Activision	8.25	4.24	0.07	1.12	13.67	2256	M	83	73	4.8
23	Call of Duty: Modern Warfare 2	X360	2009	Shooter	Activision	8.52	3.59	0.08	1.28	13.47	2698	M	94	100	6.3
24	Call of Duty: Modern Warfare 3	PS3	2011	Shooter	Activision	5.54	5.73	0.49	1.57	13.32	5234	M	88	39	3.2
25	Grand Theft Auto III	PS2	2001	Action	Take-Two Interactive	6.99	4.51	0.3	1.3	13.1	664	M	97	56	8.5
26	Super Smash Bros. Brawl	Wii	2008	Fighting	Nintendo	6.62	2.55	2.66	1.01	12.84	1662	T	93	81	8.9
27	Mario Kart 7	3DS	2011	Racing	Nintendo	5.03	4.02	2.69	0.91	12.66	632	E	85	73	8.2
28	Call of Duty: Black Ops	PS3	2010	Shooter	Activision	5.99	4.37	0.48	1.79	12.63	1094	M	88	58	6.4
29	Grand Theft Auto V	PS4	2014	Action	Take-Two Interactive	3.96	6.31	0.38	1.97	12.61	2899	M	97	66	8.3
30	Animal Crossing: Wild World	DS	2005	Simulation	Nintendo	2.5	3.45	5.33	0.86	12.13	242	E	86	57	8.7
31	Halo 3	X360	2007	Shooter	Microsoft Game Studios	7.97	2.81	0.13	1.21	12.12	4100	M	94	86	7.8
32	Gran Turismo 4	PS2	2004	Racing	Sony Computer Entertainment	3.01	0.01	1.1	7.53	11.66	272	E	89	74	8.5
33	Super Mario Galaxy	Wii	2007	Platform	Nintendo	6.06	3.35	1.2	0.74	11.35	2147	E	97	73	8.9
34	Grand Theft Auto IV	X360	2008	Action	Take-Two Interactive	6.6	5.07	1.4	1.01	11.01	2951	M	98	86	7.9
35	Gran Turismo	PS	1997	Racing	Sony Computer Entertainment	4.02	3.87	2.54	0.52	10.95	138	E	96	16	8.7
36	Super Mario 3D Land	3DS	2011	Platform	Nintendo	4.89	3	2.14	0.78	10.81	921	E	90	82	8.4

This is game sales data for market analysis

Opportunities for Data Agents: Five Key Aspects



From NL2SQL Agents to Data Agents

- **Cross-DB & heterogeneous orchestration**
 - Plan over multiple stores/APIs with join-path inference and result fusion; measure success beyond single-DB EM
- **Semantic operator layer**
 - Lift from raw SQL to semantic operators that unify tabular, text, image, and report generation tasks—support *table → viz → insight report/video* workflows
- **Meta-planning & reflection**
 - A **meta-agent** that decomposes tasks, schedules tools/agents, and reflects with feedback loops
- **Memory & Semantic Catalog**
 - Unified task-specific+ long-term memory;
 - Auto-induce units, constraints, keys, value normalizations, synonyms, KPI definitions, policies, lineage from DDL/docs/logs/queries;



NL2SQL:
Paper List & Slides

Thanks!



Data Agents:
Paper List

NL2SQL_Handbook Public

main 1 Branch 0 Tags Go to file Add file Code

Apricity-Ixy Update README.md 69732ff - last week 136 Commits

File	Description	Last Commit
assets	Update Readme.md	2 months ago
chapter	Update Pre_Processing.md	11 months ago
report	update Archer Benchmark	last year
slides	update	2 months ago
src/dataset_analyze	update Archer Benchmark	last year
.gitignore	update Archer Benchmark	last year
README.md	Update README.md	last week

README

Text-to-SQL Handbook

NL2SQL Handbook

From this repository, you can view the [latest advancements](#) in Text-to-SQL (a.k.a. NL2SQL). This handbook corresponds to our survey paper [TKDE'2025]: [A Survey of Text-to-SQL in the Era of LLMs: Where are we, and where are we going?](#). We also provide [tutorial slides](#) [Update soon for VLDB'2025 Tutorial] to summarize the key points of this survey. Based on language model trends, we've created a river diagram of Text-to-SQL methods to trace the field's evolution.

About [TKDE'25] This is a continuously updated handbook for readers to easily track the latest Text-to-SQL techniques in the literature and provide practical guidance for researchers and practitioners. Official repo for A Survey of Text-to-SQL in the Era of LLMs: Where are we, and where are we going? arxiv.org/abs/2408.05109 nlp awesome tutorial ai survey dataset db nlp-resources text-to-sql nl2sql text2sql ai4db text-to-code llms nl-to-code nl-to-sql awesome-agents awesome-text2sql awesome-text-to-sql awesome-nl2sql

Readme Activity Custom properties 912 stars 19 watching 57 forks Report repository

https://github.com/HKUSTDial/NL2SQL_Handbook

<https://github.com/HKUSTDial/awesome-data-agents>

<http://luoyuyu.vip>

yuyuluo@hkust-gz.edu.cn