

The Duality between Large Language Models and Data Management Systems

M. Tamer Özsu
University of Waterloo
David R. Cheriton School of Computer Science
tamer.ozsu@uwaterloo.ca

Dagstuhl Seminar 26411 · October 2026



UNIVERSITY OF
WATERLOO



Data
Systems
Group

Data Engineering

March 2026 Vol. 50 No. 1



IEEE Computer Society

Letters

Letter from the Editor-in-Chief.....	<i>Haixun Wang</i>	1
Letter from the Special Issue Editor.....	<i>Fatma Özcan</i>	3

Special Issue on LLM-Powered Data Processing

Agentic Data Environments.....	<i>Elaine Ang, Chenxi Huang, Georgios Liargovas, Jerry Liu, Jinhui Liu, Nikos Pagonas, Charlie Summers, Haonan Wang, Jiakai Xu, Tianle Zhou, Yusen Zhang, Zhou Yu, Zhuo Zhang, Tianyi Peng, Kostis Kaffes, Eugene Wu</i>	5
Architecting the AI-Powered Agentic Data Cloud.....	<i>Yeounoh Chung, Thibaud Hottelier, Cosmin Arad, Brentan Milne, Per Jacobsson, Sam Idicula, Fatma Özcan, Alon Halevy, Yannis Parakonstantinou</i>	22
The Duality between Large Language Models and Data Management Systems.....	<i>M. Tamer Özsu, Kerem Akillioglu, Xiangru Jian</i>	40
Towards AI-Native Data Systems with the Semantic Operator Model and LOTUS.....	<i>Liana Patel, Carlos Guestrin, Matei Zaharia</i>	59
Optimal Block Nested Loops Implementations for Semantic Joins.....	<i>Immanuel Trummer</i>	74
The Future Is Bespoke: Synthesizing One-Size-Fits-One DBMSs with LLM Coding Agents.....	<i>Timo Eckmann, Matthias Jasny, Johannes Wehrstein, Carsten Binnig</i>	88

Conference and Journal Notices

TCDE Membership Form.....	104
---------------------------	-----



Kerem Akillioglu
(PhD)



Xiangru Jian
(PhD)



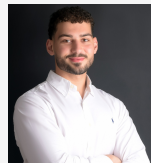
Sepideh Abedini
(Mmath)



Harrum Noor
(Mmath)



Sairaj Voruganti
(Mmath)



Yaseen Ahmed
(Mmath)



A DATA-MANAGEMENT PERSPECTIVE

I work on LLMs and data management systems.

Two projects, one in each direction.

Both raise questions about how agents use and update knowledge graphs.

A systems perspective
***Questions for this
community***

A DATA-MANAGEMENT PERSPECTIVE

I work on LLMs and data management systems.

Two projects, one in each direction.

Both raise questions about how agents use and update knowledge graphs.

A systems perspective
***Questions for this
community***

I am not an expert
I am learning...

ONE INTERACTION, TWO DIRECTIONS



LLM4DB: models assist data work. DB4LLM: data systems support models.

MACHINE LEARNING INSIDE THE ENGINE

ML4DB: a foundation for this work

Learned indexes

models guide
data access

Learned cardinality

models estimate
result sizes

Learned optimization

models guide
plan choices

Instance optimization

systems adapt to
a workload

A major focus: learning how to execute a given workload.

Agents also change how that workload is generated.

WHAT LARGE LANGUAGE MODELS CHANGED

1 The interface

Natural language broadens access to data and operations.

2 The operator

Model calls enter query plans through semantic operators.

3 The caller

An agent issues queries, inspects results, and chooses its next action.

Agents make workload generation part of the interaction.

LLM4DB: AGENTS CARRY OUT WORKFLOWS

Agents explore, execute, check and revise

BI chatbots

questions answered with live SQL

Schema exploration

browse the schema before writing SQL

Autonomous analytics

plan, query and revise over many steps

Data-engineering copilots

inspect, migrate, validate pipelines

The unit of work is now a trajectory — and so is the unit of failure.

Spider 2.0: queries → workflows¹

Spider 1.0 91%

BIRD 73%

Spider 2.0 21%

**Different benchmarks,
not a time trend.**

Evaluate the whole workflow

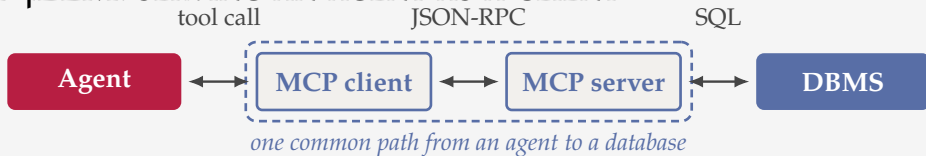
Answer quality

Total cost and latency

Failure and recovery

¹F. Lei et al, Spider 2.0: Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows, *Proc.ICLR*, 2025.

DB4LLM: SERVING AN AGENT AS A CLIENT



To an agent, the DBMS is a menu:

list tables

describe a table

run SQL

Agents need usable context

Relevant schemas, tool descriptions and bounded results

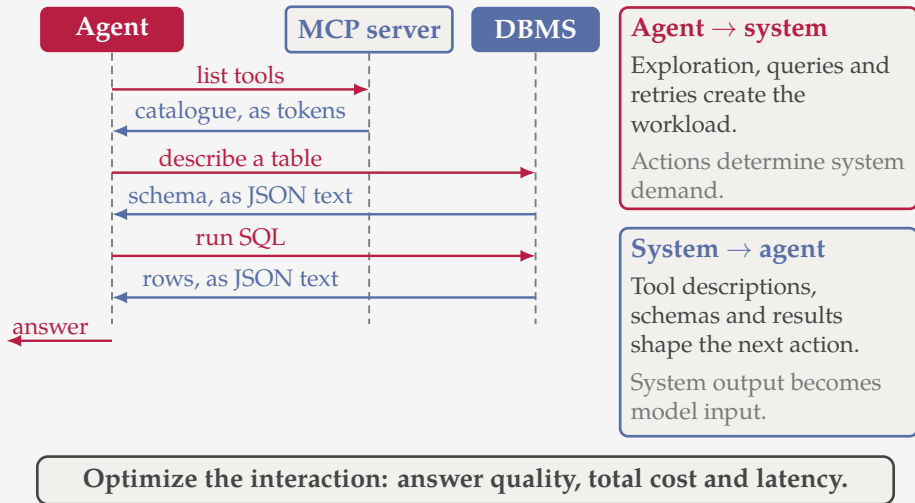
Agents query speculatively

Parallel, overlapping, exploratory queries¹

The system shapes what the agent can see and do next.

¹Liu et al., Supporting our AI overlords: Redesigning data systems to be agent-first, *Proc. CIDR*, 2026

THE TWO DIRECTIONS FORM A FEEDBACK LOOP



TODAY

1 Multi-agent analytics over federated sources

LLM4DB

Relational, graph, document

2 An AI-native data management stack

DB4LLM

Designing for LLM-agent workloads

3 Other work, if time permits

NL2SQL Benchmarking, NL2SQL evaluation, vector indexing, MCP servers

4 Where this meets knowledge graphs

Open problems — and discussion topics

ANALYTICS OVER HETEROGENEOUS FEDERATED SYSTEMS

1 • Multi-agent orchestration • LLM4DB

Xiangru Jian PhD thesis

THE QUESTION A DOMAIN EXPERT ACTUALLY ASKS

*“Assess credit risk across our top 20 suppliers: correlate **payment delays** with **ownership structures**, and flag suppliers whose **audited statements** show deteriorating liquidity.”*

Relational DBMS

Payment delays,
outstanding receivables.

Knowledge graph

Ownership traversal.
Three suppliers,
one parent.

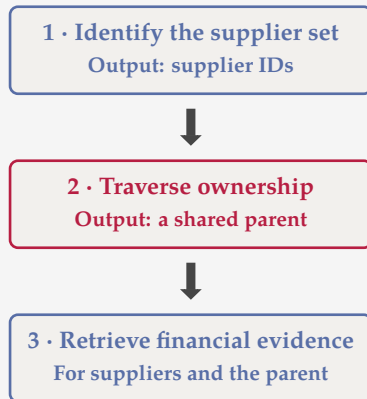
Document collection

Layout-aware extraction
from audited PDFs.

The ownership result determines which evidence we need next.

A PLAN THAT ADAPTS TO INTERMEDIATE RESULTS

- 1 Runtime bindings**
Supplier IDs determine which entities to inspect
- 2 Dependent steps**
A missed supplier can distort later evidence
- 3 Adaptive access**
A parent entity triggers further document retrieval
- 4 Evidence composition**
An aggregate, an ownership path, a document passage



Start with a plan; refine it as results arrive.

WHY THE EXISTING APPROACHES FALL SHORT

Single-modality interfaces

text-to-SQL · KGQA · RAG

Each confined to one modality.
Cannot route across sources.

Polystores

BigDAWG · Calcite

Decomposition stays the user's job.
Structured data, fixed schema mappings.

Single LLM agents

plan, then invoke tools

All sources treated alike. One early error carries through every step.

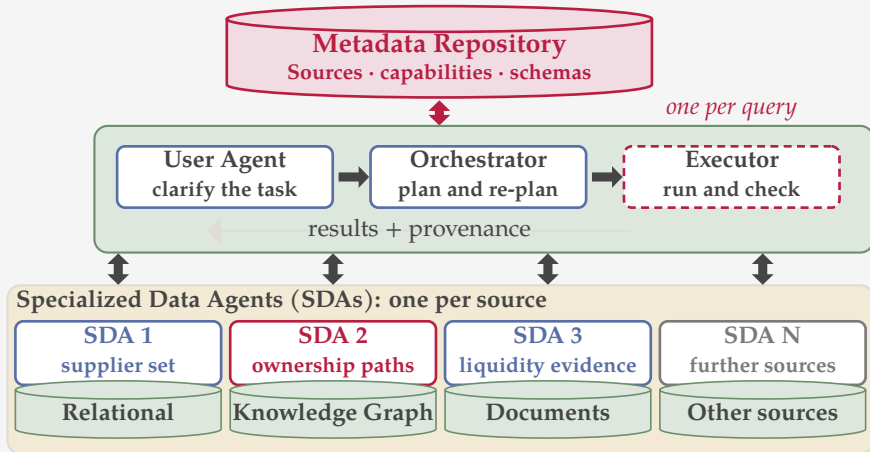
Multi-agent frameworks

AutoGen · Camel · MetaGPT

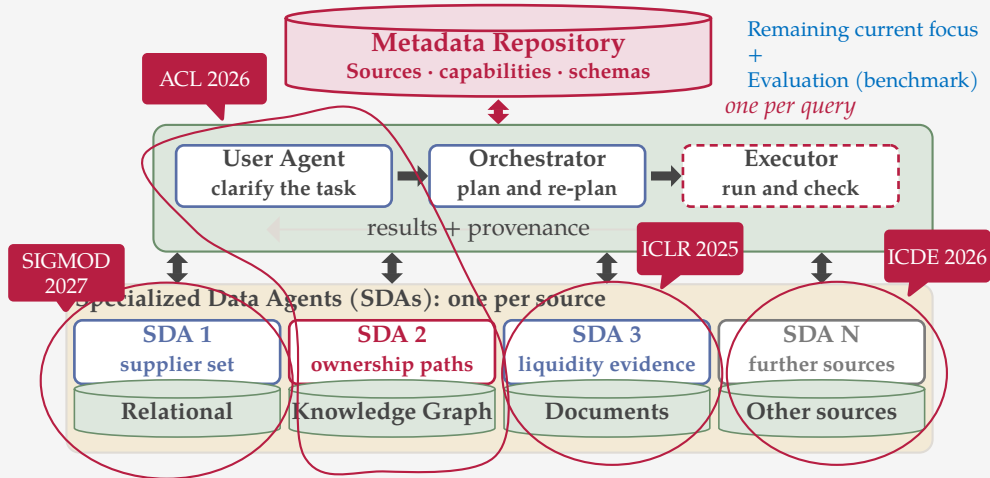
Coordination by conversation. No checkable plan, no provenance.

*None models source capability, validates results, or assembles a trace.
The missing layer: **compositional orchestration**.*

ARCHITECTURE FOR THE SUPPLIER WORKFLOW



ARCHITECTURE FOR THE SUPPLIER WORKFLOW



SPECIALIZATION AT THE SOURCE LEVEL

Design choice: one agent per source.

Action space: the operations its environment permits.

Specialize schema knowledge, tools and validation.

Evaluate the gain against coordination overhead.

Each source exposes its own action space

Relational DBMS 1

SQL · schema exploration

Relational DBMS 2

T-SQL dialect · partial schema

Knowledge graph

SPARQL · ontology · entity resolution

Document collection

Retrieval · layout-aware extraction

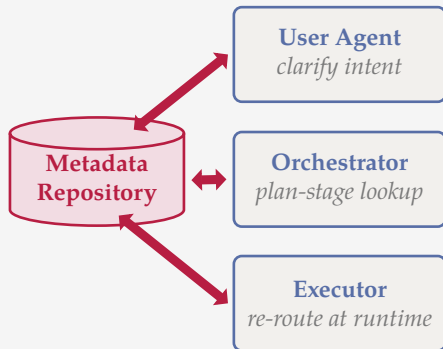
NO GLOBAL SCHEMA: A METADATA REPOSITORY

What each source holds, and what each agent can do.

Source metadata. Agent registry.
Reusable plans.

Replaces the mediator's global schema.

Add or change a source and only the repository changes.



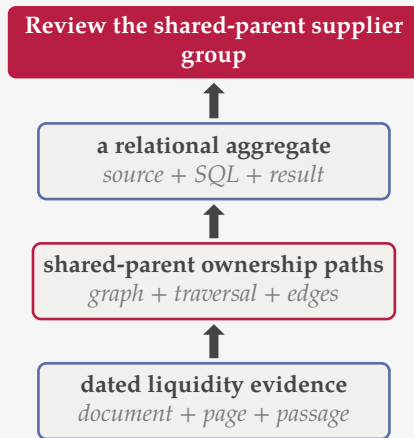
PROVENANCE BY CONSTRUCTION

Design goal: bind each claim to its evidence.

Retain the source, operation and supporting result at every step.

Keep source disagreement visible.

Abstain when support is insufficient.



OPEN QUESTIONS AND EVALUATION

- 1 Query understanding and disambiguation
What does “top 20” mean, and over which period?
- 2 Adaptive decomposition
When should new evidence change the plan?
- 3 Routing and intermediate validation
Which source can answer, and how do we check its result?
- 4 Metadata maintenance
How do source descriptions stay accurate as data changes?
- 5 Evaluate the orchestration layer
Compare plans, answer quality, provenance and total cost

Entity alignment and provenance composition: the KG connection.

AN AI-NATIVE DATA MANAGEMENT STACK

2 · Designing for LLM-agent workloads · DB4LLM

Kerem Akillioglu PhD thesis

TWO WORKLOAD PATTERNS

Fixed semantic workflows

A model operation repeated over many rows

A known task and reusable execution plan

Example: sentiment classification in SQL

```
SELECT AI_CLASSIFY(review, ['positive', 'negative'],  
  'Classify the sentiment of this movie review')  
  AS sentiment  
FROM reviews;
```



Adaptive agent workflows

Queries and tools chosen during execution

Exploration, retries and changing context

Example: ad hoc analytics with an agent



Two targets: reduce repeated inference; supply useful context.

ACCURACY, COST AND LATENCY

Accuracy constraint

Choose an acceptable task-quality target.
Validate against trusted labels; agreement with an LLM is not ground truth.

Total cost and latency

Count model calls, setup, database execution and validation across the whole workflow.

Measured baseline: 10 million rows

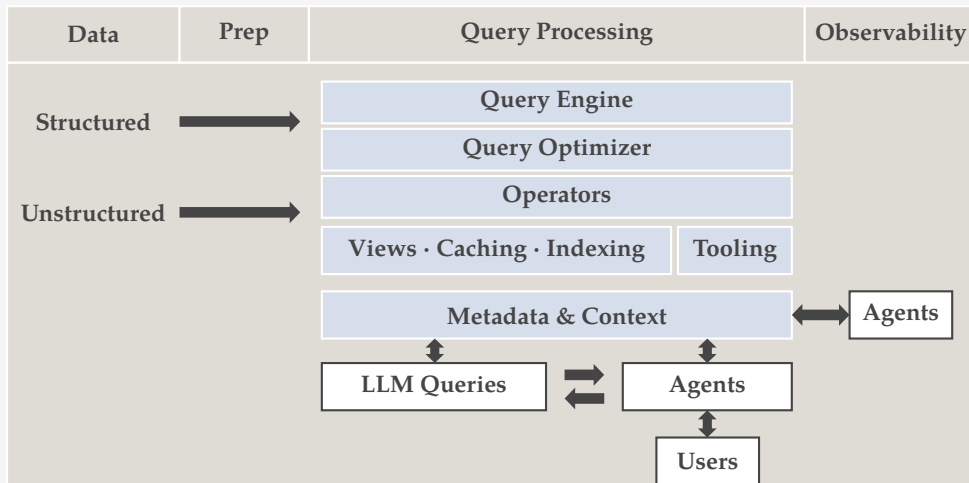
Model	LLM cost	Time
GPT-5.5	\$18,400	11 h
GPT-5-mini	\$1,500	6 h
GPT-5-nano	\$600	6 h

Per-row semantic classification
LLM expenditure and elapsed time

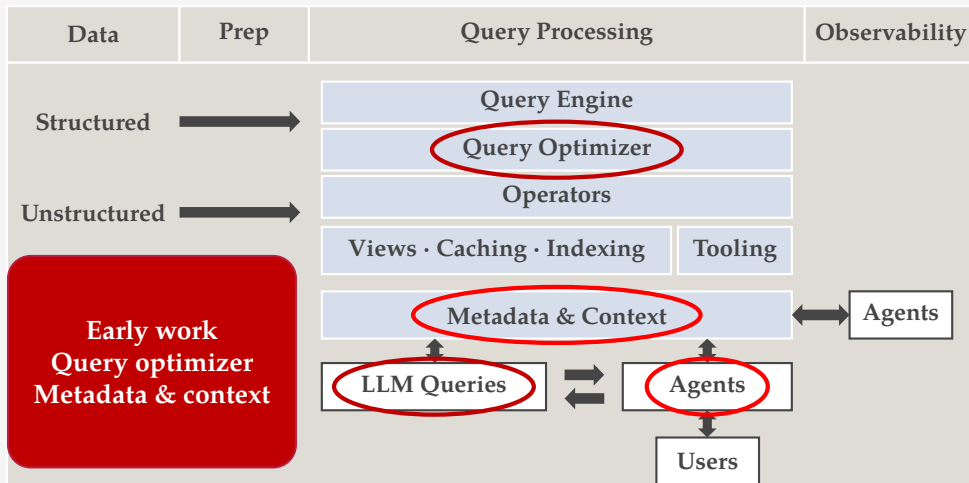
Objective: lower total cost and latency subject to the accuracy target.

Earlier results: K. Akillioglu et al., Research Challenges in Relational Database Management Systems for LLM Queries, *Proc. AIDB*, 2025.

AN AI-NATIVE STACK: TWO COMPONENTS IN FOCUS



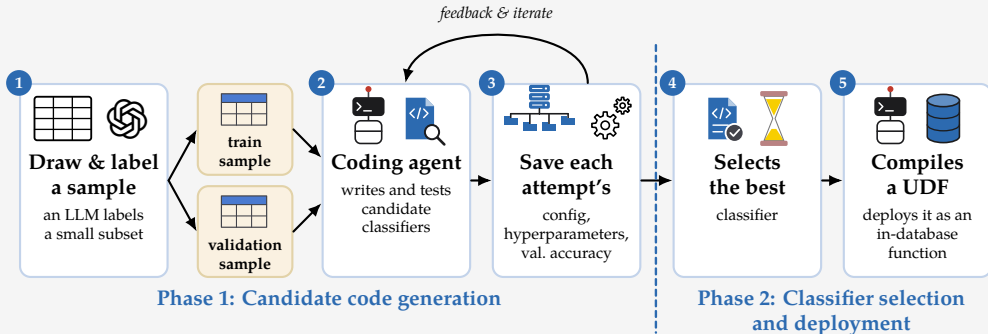
AN AI-NATIVE STACK: TWO COMPONENTS IN FOCUS



A BOLT-ON OPTIMIZER FOR SEMANTIC CLASSIFICATION

Early work: replace repeated LLM calls with a task-specific classifier

An LLM labels a sample; a coding agent writes and tests candidate classifiers.



Deployment objective

Run the selected classifier as a UDF inside the database.

Amortize labeling and search costs over the remaining rows.

METADATA AND CONTEXT: A WORKED EXAMPLE

Illustrative task: “Which leads should we follow up on?”

Source evidence

CRM: 123 · Acme Corp · Qualified
Note: Lead123 · “ready to buy”

Agent action

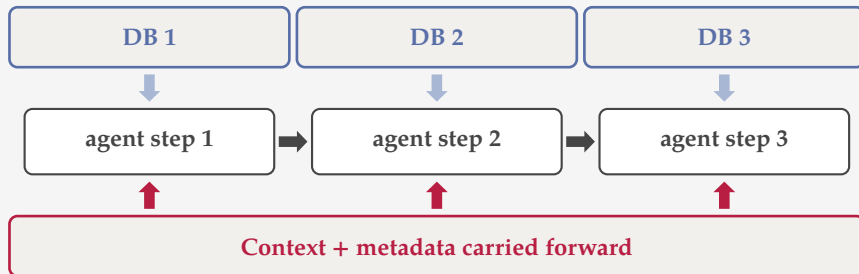
Join the records, infer high intent,
apply the rule → high-priority call.

Proposed context supplied to the agent

Where: CRM table + sales-note collection
Link: verified Lead123 ↔ 123 mapping
Meaning: what “Qualified” denotes
Rule: Qualified + high intent → call

Retrieve the relevant mapping and rule
for this step; clarify if either is uncertain.

AI NATIVE METADATA & CONTEXT LAYER FOR ACCURACY



Agents can write SQL, but they often fail because they lack organizational context. How can we build a metadata system that automatically infers and adapts to different databases, schemas, and data characteristics?

FUTURE WORK: AMBIGUITY AND SHARED WRITES

READS: resolve ambiguity

Schemas and metadata may be incomplete.
The user's request may also be ambiguous.
A plausible interpretation is not enough.

Proposed direction

Clarify intent iteratively and retrieve
the context needed for the next step.

These questions return when agents update knowledge graphs.

WRITES: coordinate shared state

Agents may act on the same data.
Their updates can conflict or use stale state.
Validation and coordination are required.

Proposed direction

Isolate uncertain writes; validate,
then merge or reconcile conflicts.

OTHER WORK, BRIEFLY

3 · Two LLM4DB projects, two DB4LLM projects

FOUR OTHER THREADS

SQLyzer

LLM4DB

Diagnoses NL2SQL errors by SQL construct.

≈21,000 pairs

37 subtypes

Semantic evaluation

LLM4DB

Evaluates whether generated SQL answers the question.

structural
match

execution
match



LLM
judge

MIRAGE

DB4LLM

Hybrid graph construction for vector search.

upper layers — increment

layer 0 — refinement

MADBench

DB4LLM

Separates MCP overhead from end-to-end agent performance.

agent



MCP server



DBMS

SQLyzer Benchmark for NL2SQL Models

Sepideh Abedini, SQLyzer: A Comprehensive Benchmark and Framework for Evaluating Text-to-SQL Systems, University of Waterloo, MMath Thesis, 2026.
<https://hdl.handle.net/10012/23045>

SQLYZR: COMPREHENSIVE TEXT2SQL BENCHMARK

- ▶ **Benchmark components:**

- ▶ **Workload:** NL questions paired with annotated SQL queries
 - ▶ ~21,000 NL-SQL pairs curated from well-known Text2SQL benchmarks: Spider, BIRD, Beaver
 - ▶ Realistic workload which is aligned with the usage pattern of real-world databases
- ▶ **Dataset:** populated databases used for evaluation
 - ▶ Each database can be scaled by inserting synthetically generated rows to enable evaluation on larger-scale databases
- ▶ **Evaluation metrics:** accuracy and efficiency measurements
 - ▶ New evaluation metrics in addition to the conventional Text2SQL metrics, providing a more practical evaluation of models

EVALUATION METRICS

ACCURACY METRICS

- ▶ **Exact Match (EM)**: Extending SPIDER's Exact Match metric by comparing the ASTs of the queries.
- ▶ **Execution Accuracy (EA)**: Check whether the execution result of the gold and predicted SQL queries on the corresponding databases match.
- ▶ **Relaxed Execution Accuracy (REA)**: A more lenient version of execution accuracy, ignoring minor issues of the predicted query.

EFFICIENCY METRICS

- ▶ **Query Execution Time (QET)**: Execution time of the predicted SQL queries on the corresponding databases.
- ▶ **Prediction Token Usage (PTU)**: Total number of API tokens used by the text-to-SQL model for prediction.
- ▶ **Relative Execution Time (RET)**: The measure of a predicted query's efficiency relative to the gold query.
- ▶ **Complexity Consistency Score (CCS)**: The measure of the predicted query's complexity relative to the gold query.

SQL CLASSIFICATION

- ▶ Existing classifications are simplistic:
 - ▶ Spider: Easy, Medium, Hard, Extra-Hard
 - ▶ Bird: Simple, Moderate, Challenging

SQL CLASSIFICATION

- ▶ Existing classifications are simplistic:
 - ▶ Spider: Easy, Medium, Hard, Extra-Hard
 - ▶ Bird: Simple, Moderate, Challenging
- ▶ SQLyze taxonomy of SQL queries:
 - ▶ 6 Categories
 - ▶ 37 Sub-Categories
 - ▶ Rule-based category assignment using the AST of queries

Category	New	Rule	Example
C1	S1	Multi Column SELECT, Single Table FROM	<code>SELECT cows, total_horses FROM farm;</code>
	S2	Single Column SELECT with *	<code>SELECT * FROM farm;</code>
C2	S3	ORDER BY	<code>SELECT Official_Name FROM city ORDER BY Population DESC;</code>
	S4	Single Binary Expr WHERE	<code>SELECT count(*) FROM head WHERE age > 56;</code>
	S5	Arithmetic Expressions	<code>SELECT date, max_temperature_f - min_temperature_f FROM weather;</code>
	S6	Aggregate Functions	<code>SELECT AVG(price) FROM products;</code>
	S7	LIMIT	<code>SELECT Publisher FROM publication LIMIT 1</code>
	S8	DISTINCT	<code>SELECT DISTINCT start_station_name FROM trip;</code>
	S9	Unconditional GROUP BY	<code>SELECT Publisher FROM publication GROUP BY Publisher;</code>
	S10	Equi Join	<code>SELECT T1.engineer_id, T1.first_name, T1.last_name FROM Maintenance_Engineers AS T1 JOIN Engineer_Visits AS T2 ON = T2.id;</code>
	S11	Non-Equi Join	<code>SELECT e.employee_name, p.position_title FROM employees AS e JOIN positions AS p ON e.salary > p.min_salary AND e.salary < p.max_salary;</code>
	S12	Between Expr	<code>SELECT date FROM weather WHERE mean_sea_level_pressure_inches BETWEEN 30.3 AND 31;</code>
C3	S13	Unconditional JOIN, Single Binary Expr WHERE	<code>SELECT T2.name FROM Flight AS T1 JOIN Aircraft AS T2 WHERE T1.fino = 99;</code>
	S14	Unconditional GROUP BY, Single Binary Expr	<code>SELECT dept_name, AVG (salary) FROM instructor WHERE salary > 42000 GROUP BY dept_name;</code>
	S15	Unconditional GROUP BY, Unconditional JOIN	<code>SELECT t2.customer_details FROM policies AS t1 JOIN customers AS t2 GROUP BY t2.customer_details;</code>

SQLYZR PIPELINE



SQLYZR FRAMEWORK FEATURES

- ▶ Modular and unified benchmarking framework:
 - ▶ Allowing effortless integration of new Text2SQL models, workloads, evaluation metrics, categories, etc.
- ▶ Configurable options: Number of Iterations, LLM Temperature, Models and Workload, etc.

Configuration

Dataset:
agg

Dataset Size:
all

Models:
dall din

Num Iterations:
5

Temperature:
0.2

Batch Mode:
On

Number of Synthetic Examples Per Sub-Category:
500

Error Threshold (Min Acceptable REA):
85%

Pipeline Steps:
Verify → Predict → Eval →
Charts → Transformers → Augment

Charts:
Execution Accuracy Exact Match

Save Configuration

Reset Config

SQLYZR: FINE-GRAINED ERROR ANALYSIS

- ▶ If execution accuracy is low, evaluate at fine-grain
- ▶ Error Fixing Suggestions
 - ▶ For each incorrect predicted query, SQLyzyr finds fixing feedback
 - ▶ SQLyzyr finds error-fixing rules that can fix the errors in the predicted query
 - ▶ A textual description of the fix is provided to the user

• The predicted SQL used extra columns that should be removed.

1

Database: farm

Predicted SQL

```
SELECT Farm_ID, Total_Horses FROM farm ORDER BY Total_Horses ASC;
```

Gold SQL

```
SELECT Total_Horses FROM farm ORDER BY Total_Horses ASC
```

Using LLMs for Semantic Evaluation of NL2SQL Queries

Harrum Noor, What Do You Mean? Using Large Language Models for Semantic Evaluation of NL2SQL Queries, University of Waterloo, MMath Thesis, 2025.
<https://hdl.handle.net/10012/21632>

EVALUATION OF TEXT-TO-SQL (T_{XT}2SQL) QUERIES

Find customers who have placed more than five orders and calculates their total spending, the total number of orders, and their average order value. It then segments these customers into two groups: the top 10% of spenders (based on total spending) and all other customers. This segmentation helps to highlight high-value customers within a group of more active shoppers, enabling more targeted marketing or further analysis.

```
SELECT
  customer_id,
  total_spent,
  total_orders,
  avg_order_value,
  CASE
    WHEN total_spent >= (SELECT PERCENTILE_CONT(0.9) WITHIN GROUP (ORDER BY total_spent)
                        FROM (SELECT customer_id, SUM(order_amount) AS total_spent
                            FROM orders
                            GROUP BY customer_id) AS spending)
    THEN 'Top 10%'
    ELSE 'Other'
  END AS spending_segment
FROM
  (SELECT
    customer_id,
    COUNT(order_id) AS total_orders,
    SUM(order_amount) AS total_spent,
    AVG(order_amount) AS avg_order_value
  FROM
    orders
  GROUP BY
    customer_id
  HAVING
    COUNT(order_id) > 5
  ) AS customer_summary;
```



PROBLEM WITH CURRENT METRICS

EXECUTION ACCURACY

Measures correctness by comparing results of the predicted and actual SQL

Query: *Show all products with price greater than \$50.*

Expected SQL: SELECT * FROM products WHERE price > 50;

Predicted SQL: SELECT * FROM products WHERE price > 51;

EXACT MATCH ACCURACY

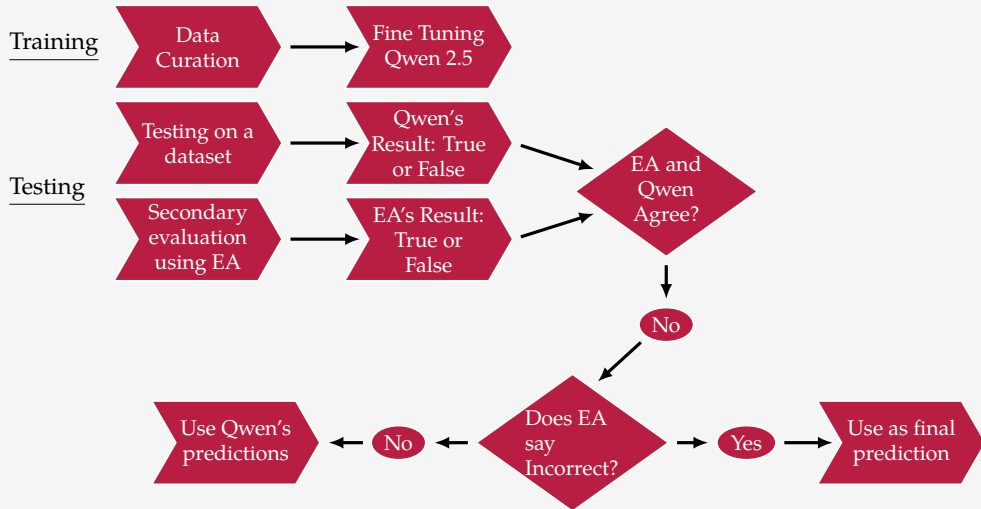
Measures correctness through strict string matching of SQL

Query: *Show all products with price greater than \$50.*

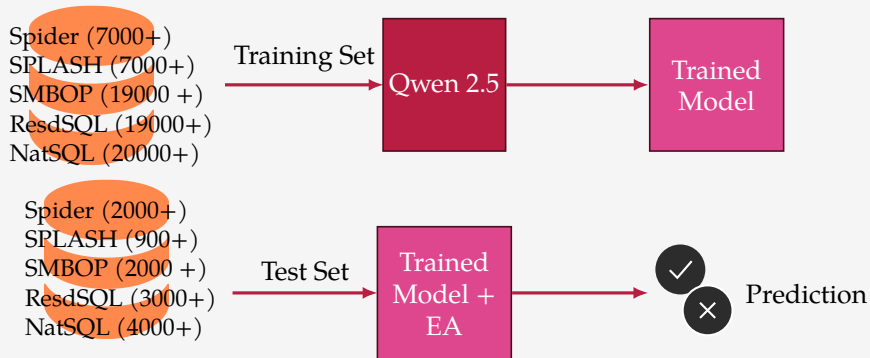
Expected SQL: SELECT * FROM products WHERE price > 50;

Predicted SQL: SELECT * FROM products WHERE price >= 51;

EVALUATING TXT2SQL TRANSLATIONS WITH LLMs

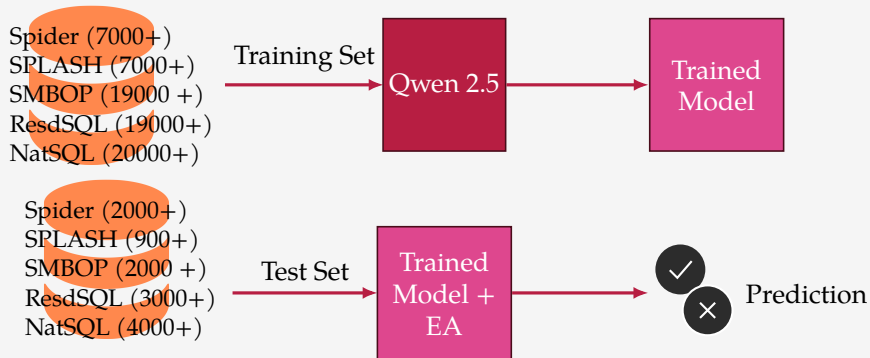


WORKFLOW



- Curation deleted some queries
- Detailed classification of queries

WORKFLOW



- ▶ Curation deleted some queries
- ▶ Detailed classification of queries

Qwen 2.5 Alone	EA Alone	EA+ Qwen
92%	88%	98%

MODEL TRAINING - PROMPT ENGINEERING

- ▶ You are a world-class SQL expert. Please evaluate whether the following SQL query semantically answers the given natural language question. You may include reasoning or analysis, but conclude your answer with either the word 'true' or 'false'.

```
{
  "messages": [
    {
      "role": "system",
      "content": "You are a world-class SQL expert. Please evaluate whether
        ↳ the following SQL query semantically answers the given natural
        ↳ language question. You may include reasoning or analysis, but
        ↳ conclude your answer with either the word 'true' or 'false'."
    },
    {
      "role": "user",
      "content": "Question: What are the banks that exist within the state of
        ↳ New York?
        SQL: SELECT bank.bname FROM bank WHERE bank.state = 'New York' ORDER
        ↳ BY bank.bname ASC"
    },
    {
      "role": "assistant",
      "content": "false"
    }
  ],
  "format": "chatml"
}
```

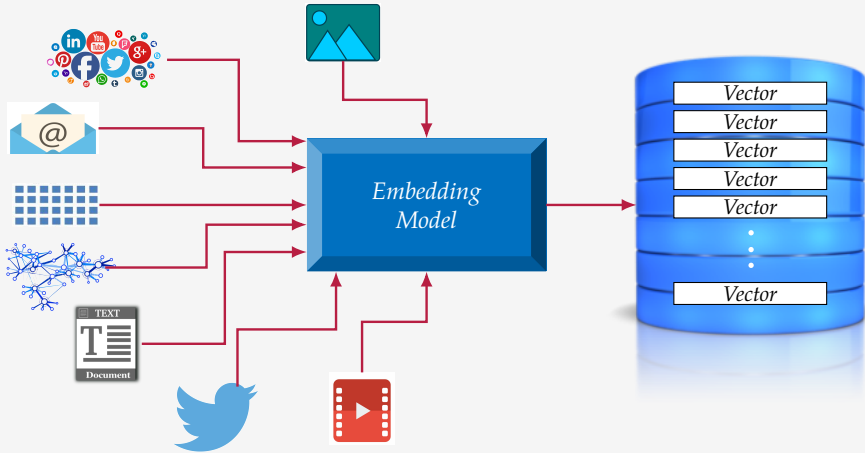
└─ We use ChatML format to input records into Qwen

Sample Qwen input

MIRAGE-ANNS

Sairaj Voruganti, MIRAGE-ANNS: Mixed Approach Graph-based Indexing for Approximate Nearest Neighbour Search, University of Waterloo, MMath Thesis, 2025.
<https://hdl.handle.net/10012/21582>

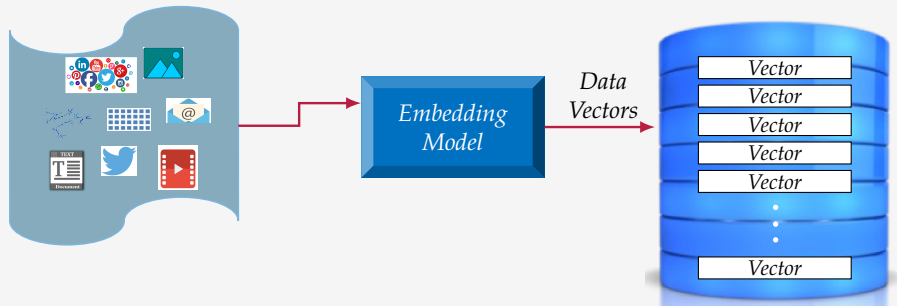
VECTOR DATABASES



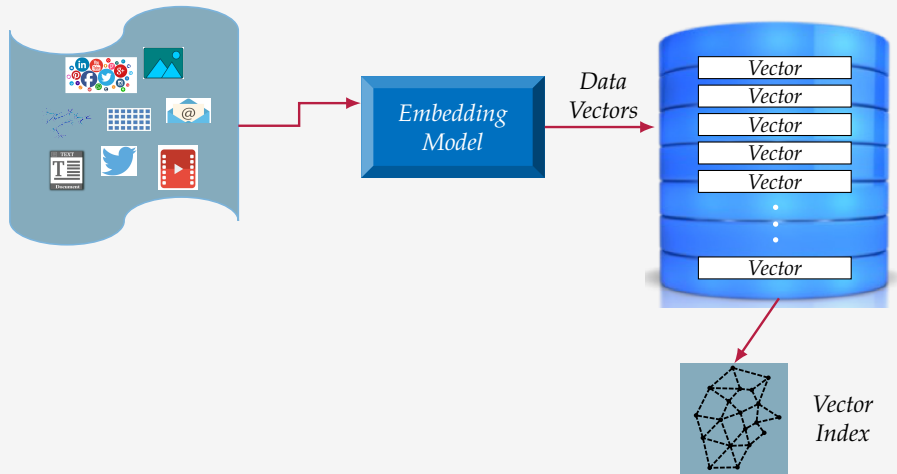
Pan et al. Survey of vector database management systems, *VLDB J.*, 2024.

Pan et al. Vector database management techniques and systems. *Companion to SIGMOD*, 2024.

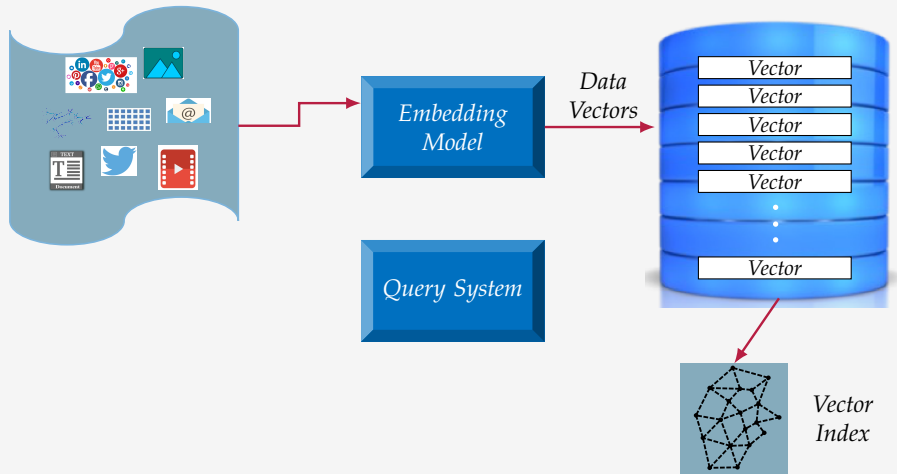
VECTOR QUERYING PIPELINE



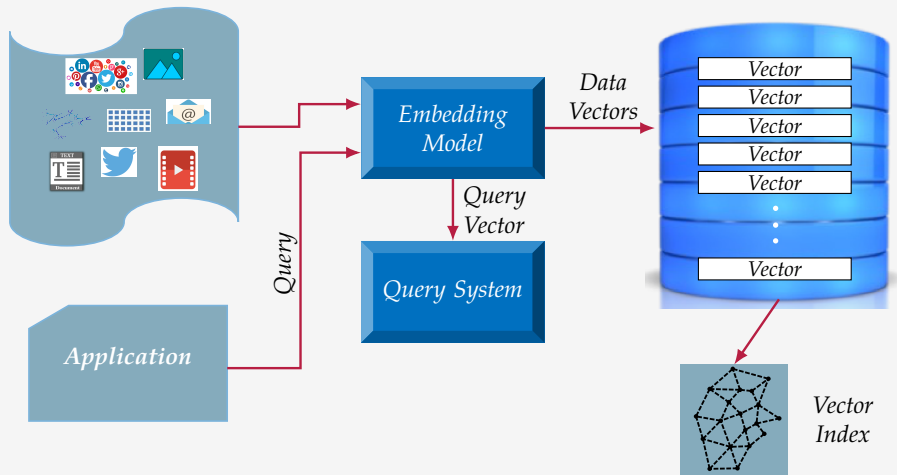
VECTOR QUERYING PIPELINE



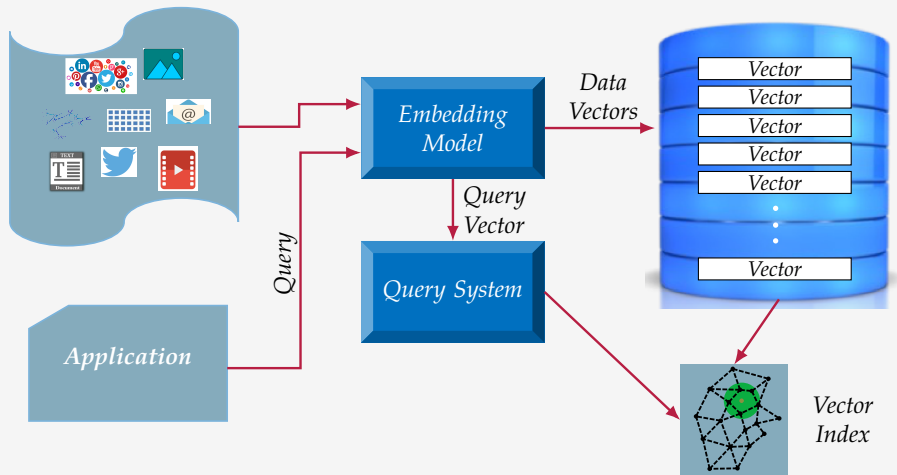
VECTOR QUERYING PIPELINE



VECTOR QUERYING PIPELINE

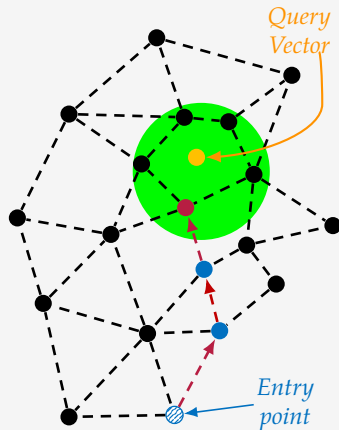


VECTOR QUERYING PIPELINE



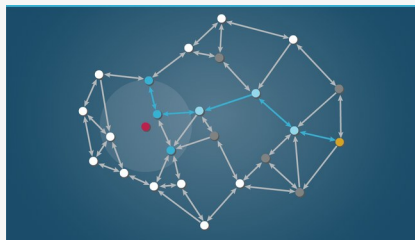
VECTOR INDEXING

- ▶ Indexes help intelligently organize the embeddings to optimize retrieval
- ▶ 4 types of indexes commonly used
 - ▶ Hashing-based
 - ▶ Tree-based
 - ▶ Quantization-based
 - ▶ **Graph-based**
- ▶ Graph-based indexes have been shown to achieve high recall and throughput on billion-scale datasets

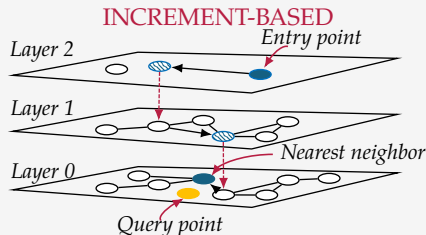


VECTOR INDEXING

- ▶ Indexes help intelligently organize the embeddings to optimize retrieval
- ▶ 4 types of indexes commonly used
 - ▶ Hashing-based
 - ▶ Tree-based
 - ▶ Quantization-based
 - ▶ **Graph-based**
- ▶ Graph-based indexes have been shown to achieve high recall and throughput on billion-scale datasets



GRAPH-BASED INDEX CONSTRUCTION



- ▶ E.g. Vamana¹, HNSW²
- ▶ Insert points one-by-one and treat each point as a query

REFINEMENT-BASED



- ▶ E.g. KGraph³ (NNDescent⁴), NSG⁵
- ▶ Initialize random kNNG
- ▶ For each node, change any 2-hop neighbor into a 1-hop neighbor if it is a new k -nearest neighbor
- ▶ Repeat until convergence

¹ Subramanya et al. DiskANN: fast accurate billion-point nearest neighbor search on a single node. In Proc. 33rd Annual Conf. on Neural Information Processing Syst., 2019.

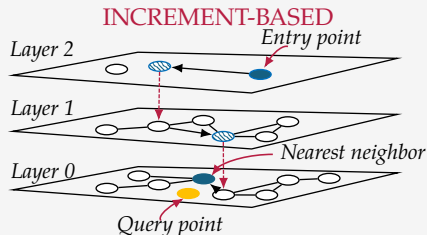
² Malkov and Yashunin. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020.

³ Dong et al. Efficient k-nearest neighbor graph construction for generic similarity measures. *Proc. 20th Intl. Conf. on World Wide Web.*, 2011.

⁴ Bratić et al., NN-Descent on High-Dimensional Data. *Proc. 8th Intl. Conf. on Web Intelligence, Mining and Semantics*, 2018.

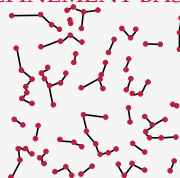
⁵ Fu et al. Fast approximate nearest neighbor search with the navigating spreading-out graph. *PVLDB*, 2019.

GRAPH-BASED INDEX CONSTRUCTION



- **Benefits:** $\sim O(\log N)$ online search, incremental inserts, high recall
- **Drawbacks:** Slow construction due to ANNS for each point inserted

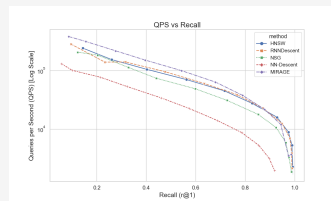
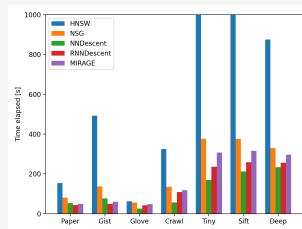
REFINEMENT-BASED



- **Benefits:** Fast construction
- **Drawbacks:** Slow for online queries, no incremental inserts, low recall

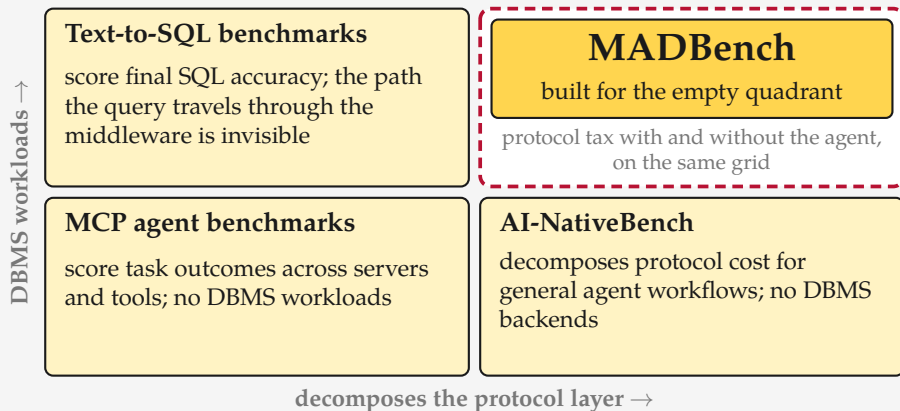
MIRAGE – AN ENGINEERING SOLUTION

- ▶ New approach to graph-based indexing
- ▶ Combine increment-based and refinement-based approaches
 - ▶ Layer 0 using refinement, upper layers using increment
 - ▶ Indexing speed of refinement-based approaches
 - ▶ Superior search of increment-based approaches
- ▶ Experimental results
 - ▶ 2x QPS as compared to HNSW, NSG, and NNDescent
 - ▶ Up to 7x faster indexing than HNSW and 2.5x faster than NSG



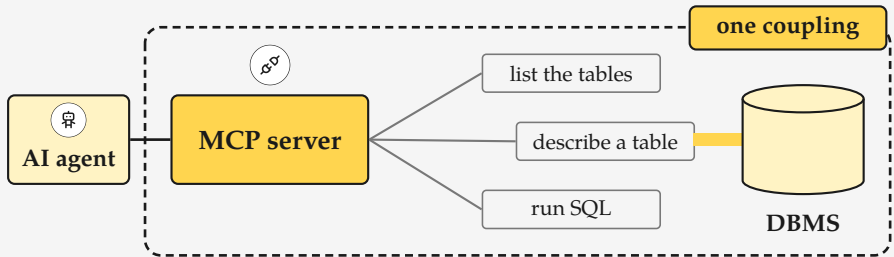
MADBench: Measuring Agentic DBMS

EXISTING BENCHMARKS MISS THE MIDDLEWARE



MADBench is built for the empty quadrant.

ONE COUPLING, MEASURED TWO WAYS



Track A · the protocol alone

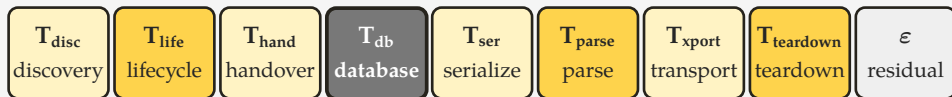
A harness replays five fixed TPC-H queries.
No AI anywhere: any time beyond the DBMS is
middleware.
SF 1 · 3 · 10 · 5,378 recorded runs

Track B · an agent does the job

30 questions: 15 routine, 15 complex SQL.
Answers checked against the gold result rows.
SF 1 · 3 · 1,080 verified attempts

Same six couplings in both tracks. That separation is the thesis.

EVERY MILLISECOND GETS A NAME



$$\text{PTR} = (T_{\text{total}} - T_{\text{db}}) / T_{\text{db}}$$

0 means no tax, 1 means the tax equals database time

every boundary is a recorded event; the residual ϵ stays under 5% on 99.4% of traces

$$T_{\text{total}} = T_{\text{db}} + T_{\text{disc}} + T_{\text{life}} + T_{\text{hand}} + T_{\text{ser}} + T_{\text{parse}} + T_{\text{xport}} + T_{\text{teardown}} + \epsilon$$

the identity is checked per trace: terms must sum to
the recorded total within the residual bound

SIX COUPLINGS, ONE GRID

	PostgreSQL	MySQL	DuckDB	tool surface
anthropic-pg	✓ measured	not supported	not supported	anthropic-pg 1 tool
dbhub	✓ measured	✓ measured	not supported	dbhub 2 tools
mcp-alchemy	✓ measured	✓ measured	✓ measured	mcp-alchemy 4 tools

six valid couplings; both tracks run on the same grid

Unsupported pairs are excluded, never imputed.

Tool surface is a design choice: what an agent can explore, and what it must read up front.

THREE FINDINGS

- 1 MCP overhead varies by three orders of magnitude**
Median PTR ranges from 0.0031 (DuckDB, in-process) to 7.32 (anthropic-pg, cold start) on the same five queries.
- 2 The process model sets the tax**
Subprocess servers pay startup every session; the pooling server skips startup but pays at shutdown, enough to flip the ranking.
- 3 Protocol speed does not predict agent success**
The lowest-tax coupling lands mid-pack; another collapses from 61.1% to 17.8% when the data triples.

The deployment question is a coupling question, not a server question.

WHERE THIS MEETS KNOWLEDGE GRAPHS

4 • Open problems — and discussion topics

THE GRAPH WAS ALREADY HERE

slide 13

The query

Ownership traversal finds the concentrated risk.

slide 17

The agent

SDA 2 is a knowledge-graph agent with its own action space.

slide 19

The trace

The ownership path is one of three evidence bindings.

Not a survey of the field — the places where our research ran into it.

1. WHAT SHOULD A GRAPH EXPOSE TO AN AGENT?

what a KG agent does today

SPARQL

Ontology navigation

Entity resolution

chosen by analogy with a human user



what a planner might need

A readable capability description

Ontology summarization

Cost estimates before traversal

"What can I ask you?"

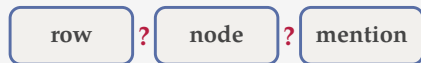
**An agent pays by the token.
That is not the caller SPARQL was designed for.**

2. ENTITY RESOLUTION

Resolution at runtime

A supplier is a row, a node, and a name in a PDF.

Not preprocessing — what needs aligning depends on earlier results.



the same supplier, three times

3. PROVENANCE

Across the boundary

Graph provenance is mature: named graphs, RDF-star, PROV-O.

Documents and model output have nothing comparable.

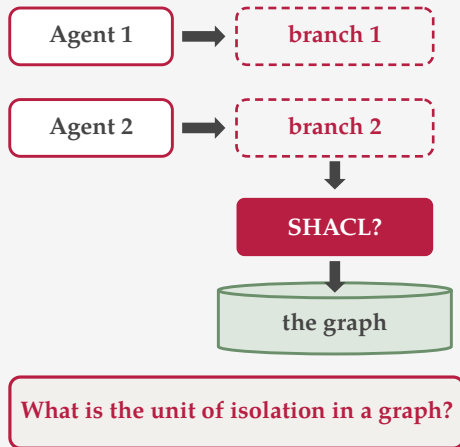


4. AGENTS THAT WRITE TO GRAPHS

A knowledge graph is shared mutable state that agents write.

This community treats that as data quality. I think it is also concurrency control.

An uncertain extraction is a branch, not a fact. And SHACL is sitting right there.



FIVE THINGS I WOULD SPEND THREE DAYS ON

- 1 Action spaces for knowledge graphs
- 2 Runtime entity resolution across modalities
- 3 Agent writes as concurrency control
- 4 A federated benchmark with a graph really in it
- 5 Graph structure for agent memory

Still not an expert. Better problems, though.