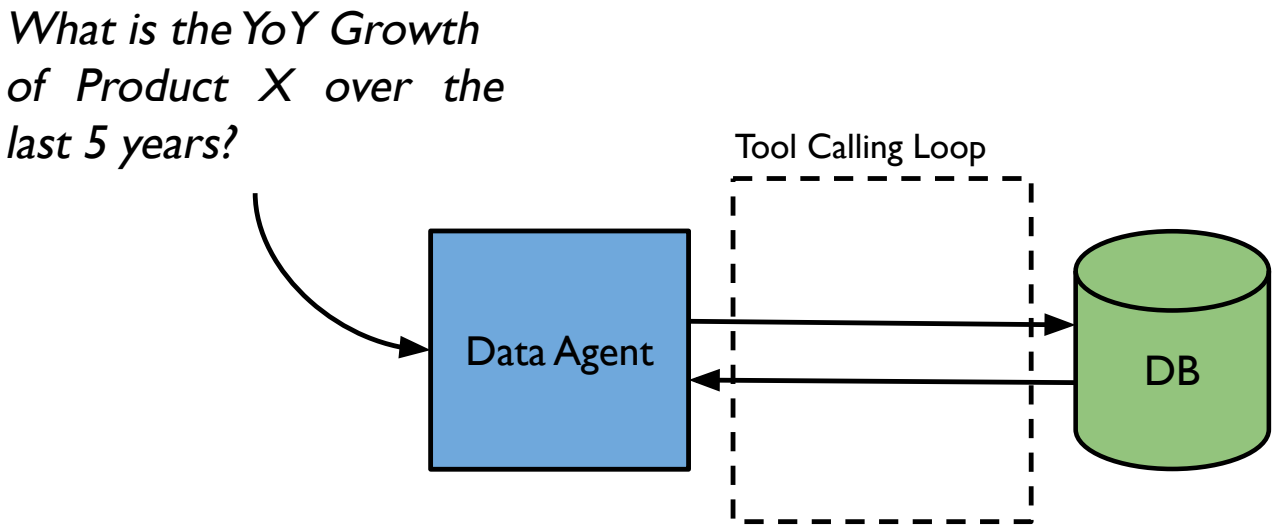
Sophrosyne: Agentic Exploration of Relational Data Systems Needs Moderation

Madhav Jivrajani, Ramnatthan Alagappan, Aishwarya Ganesan

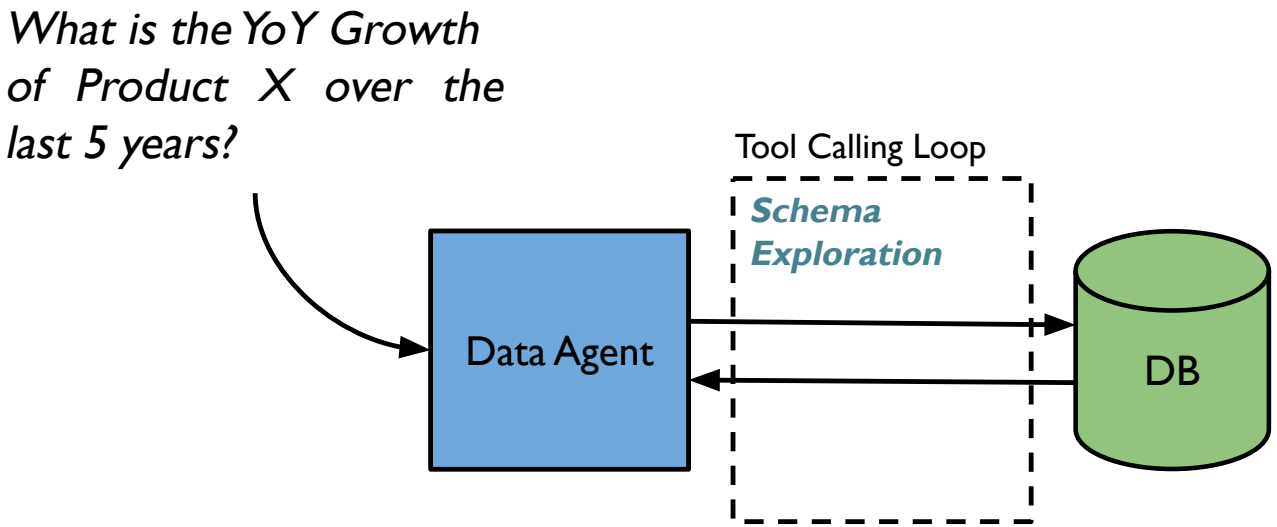
Agentic Text2SQL

AI agents interacting with data systems offer a promising approach to *text2SQL* tasks.



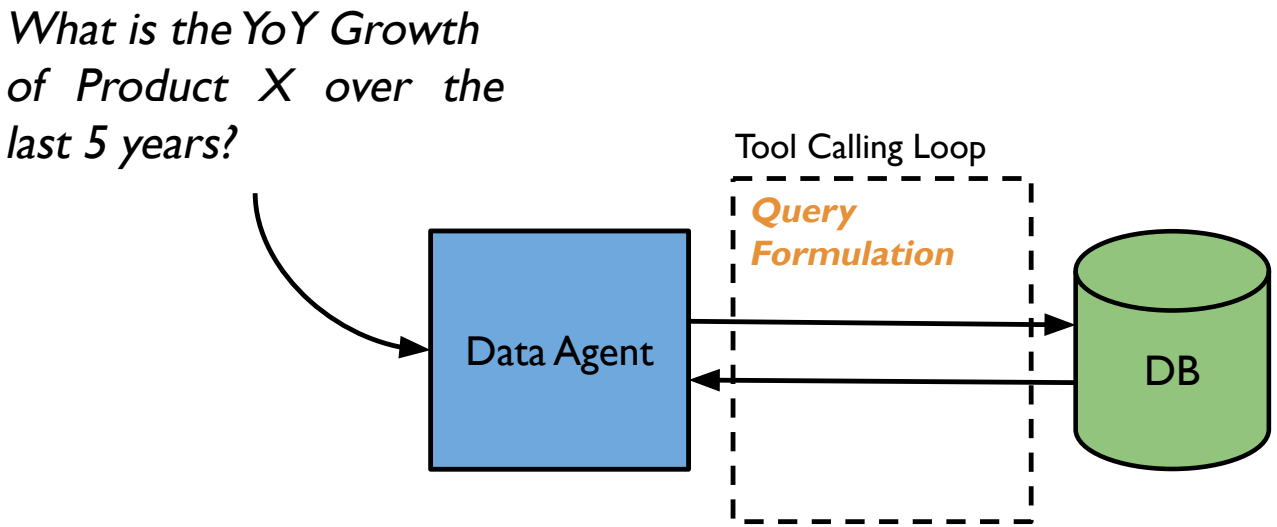
Agentic Text2SQL

Agents begin by ***exploring*** the data system schema to discern relevant elements.



Agentic Text2SQL

Based on what is explored, agents proceed to formulate queries.



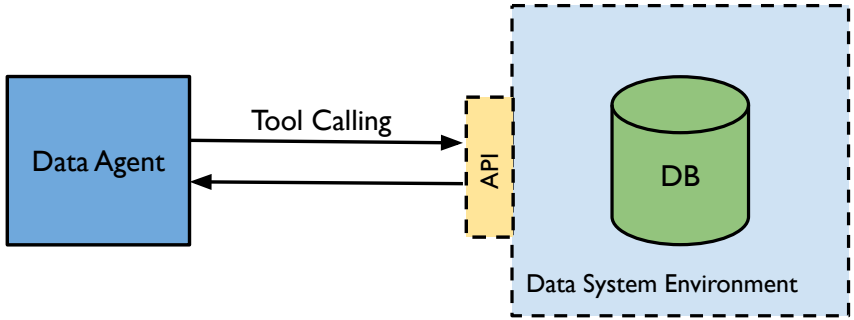
Data System Environments

Given that these agents are interacting with data systems, ensuring standardised, secure and scoped access to them is **crucial**.

Data System Environments

Given that these agents are interacting with data systems, ensuring standardised, secure and scoped access to them is **crucial**.

As a result, these agents are exposed to *environments* with *explicit API surfaces* using the Model Context Protocol (MCP).



Data System Environments

Exposing data systems this way is becoming the norm.




Data System Environments

Given this is the norm of exposing data systems to agents, we ask the question:

Data System Environments

Given this is the norm of exposing data systems to agents, we ask the question:

How does the exposed API surface effect the agent's ability to perform text2SQL tasks?

API Surface Categorization

We conduct the first study of API surfaces exposed to agents and find that 2 categories emerge: ***fine-grained*** and ***coarse-grained***, with a majority (**83%**) exposing fine-grained surfaces.

Coarse-grained	Fine-grained
Spanner, PlanetScale	BigQuery, Azure SQL, DSQL, MySQL, Postgres, Redshift, Neon, Supabase, DuckDB, Turso, ClickHouse

API Surface Categorization

We conduct the first study of API surfaces exposed to agents and find that 2 categories emerge: ***fine-grained*** and ***coarse-grained***, with a majority (**83%**) exposing fine-grained surfaces.

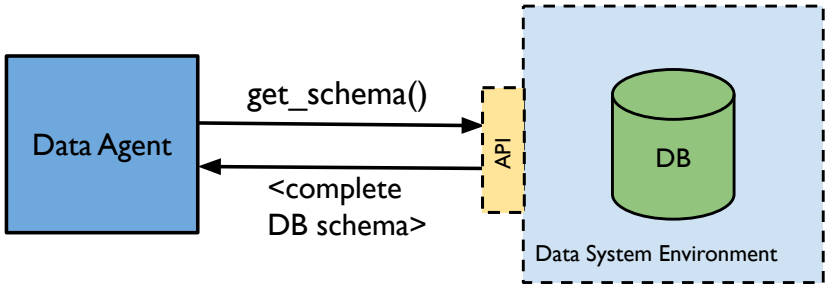
Coarse-grained	Fine-grained
Spanner, PlanetScale	BigQuery, Azure SQL, DSQL, MySQL, Postgres, Redshift, Neon, Supabase, DuckDB, Turso, ClickHouse

These API surfaces differ in how they enable *exploration.*

API Surface Categorization

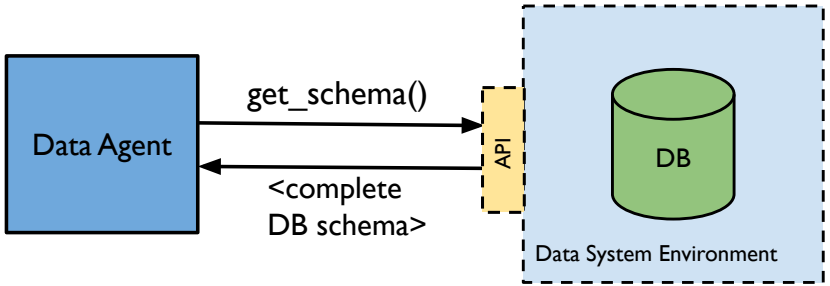
Coarse-grained Exploration	Fine-grained Exploration

API Surface Categorization

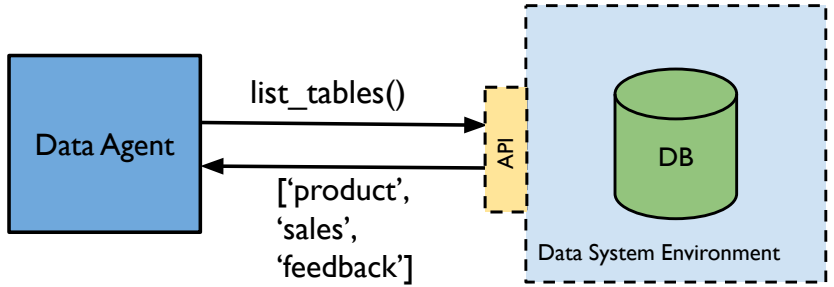
Coarse-grained Exploration	Fine-grained Exploration
	

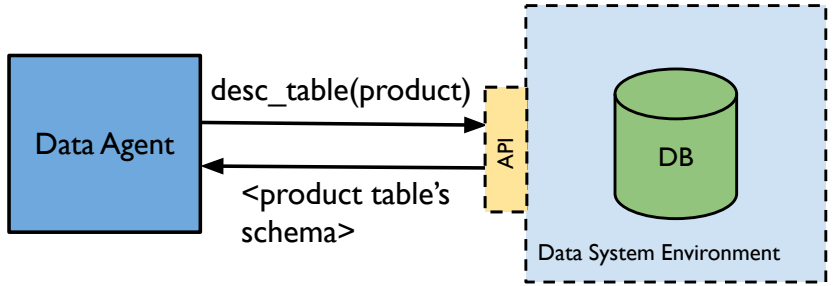
API Surface Categorization

Coarse-grained Exploration



Fine-grained Exploration





Dichotomy In Exploration APIs

In the BIRD benchmark:

Coarse-grained exploration consumes up to **5.7x** more tokens than needed per database.

Dichotomy In Exploration APIs

In the BIRD benchmark:

Coarse-grained exploration consumes up to **5.7x** more tokens than needed per database.

While fine-grained exploration is more token-efficient (up to **2x** more tokens per database):

Agents suffer from ***over-exploring*** the schema (explore up to **65%** more than what is needed) and incorporate irrelevant schema elements in query formulation, leading to **inaccurate** results.

Dichotomy In Exploration APIs

In the BIRD benchmark:

Coarse-grained exploration consumes up to **5.7x** more tokens than needed per database.

While fine-grained exploration is more token-efficient (up to **2x** more tokens per database):

Agents suffer from ***over-exploring*** the schema (explore up to **65%** more than what is needed) and incorporate irrelevant schema elements in query formulation, leading to **inaccurate** results.

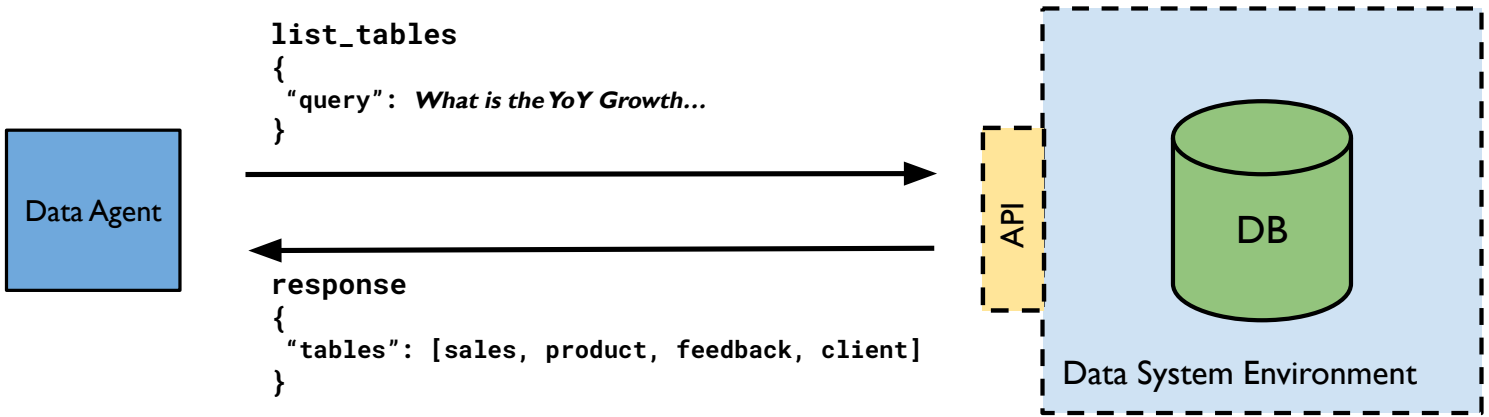
Given that most data systems expose fine-grained APIs,
any agent interacting with them is prone to over-exploration.

Dichotomy In Exploration APIs

Key insight: curbing over-exploration is key for effective use of fine-grained APIs and letting the environment return ***directives*** to guide the agent's exploration process helps mitigate this.

Sophrosyne

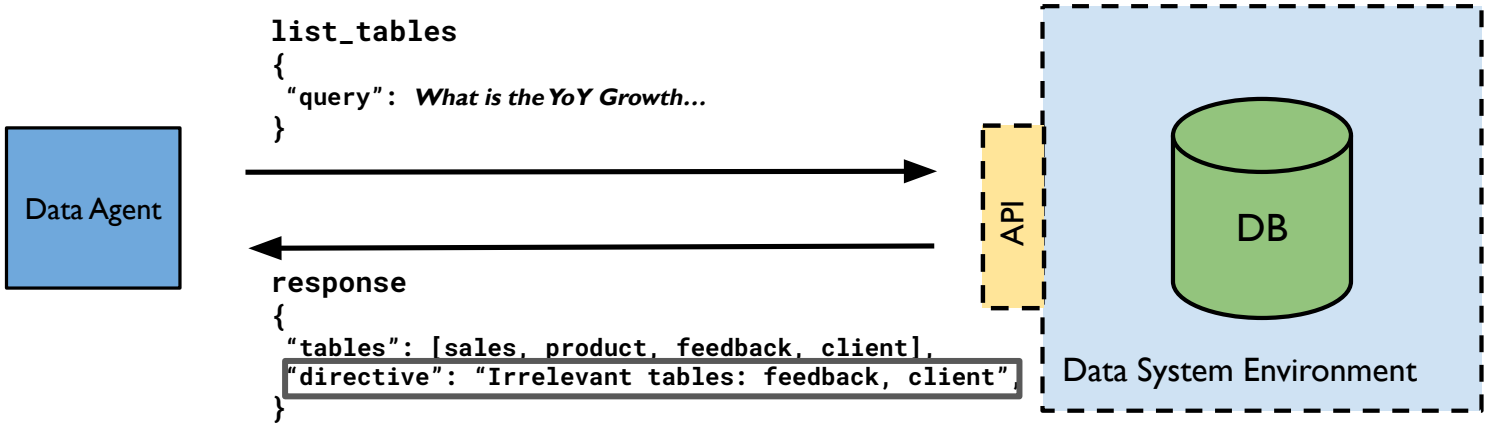
Over-exploration stems from exploring more tables than necessary.



Sophrosyne

Over-exploration stems from exploring more tables than necessary.

To curb it, we approximate tables that are *potentially irrelevant* for the given query.



Evaluation

Benchmark: BIRD LiveSQLBench

Agent: OpenCode agent with GPT-5.4-mini, GPT-5.4 and Sonnet 4.5

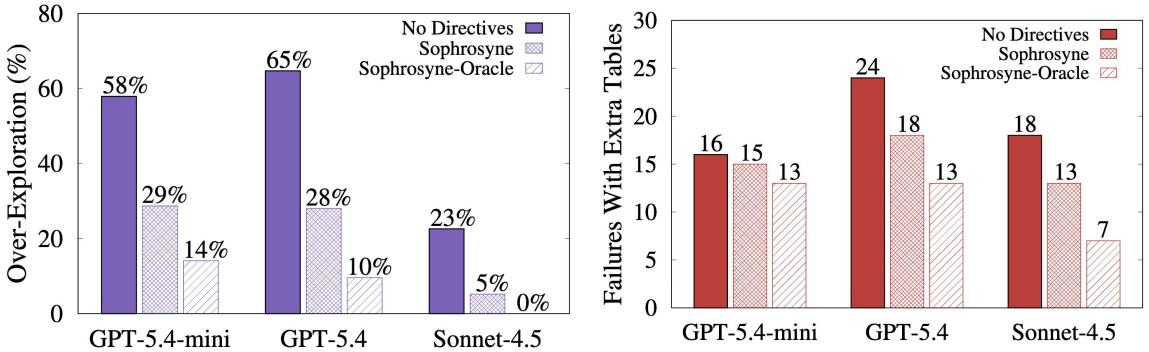
Systems: Fine-grained API surface with *No Directives*, with directives (*Sophrosyne*) and perfect directives (*Sophrosyne-Orcale*).

Evaluation

Benchmark: BIRD LiveSQLBench

Agent: OpenCode agent with GPT-5.4-mini, GPT-5.4 and Sonnet 4.5

Systems: Fine-grained API surface with *No Directives*, with directives (*Sophrosyne*) and perfect directives (*Sophrosyne-Orcale*).



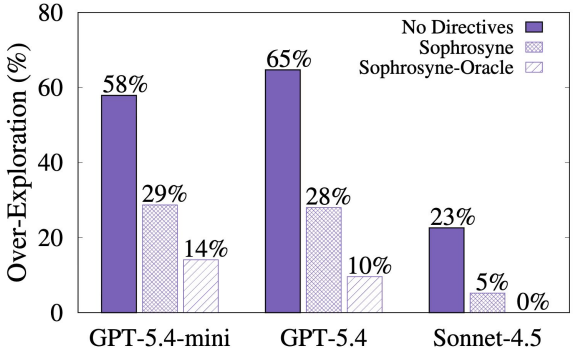
Directives consistently reduce over-exploration and reduce the number of failed queries with extra tables thereby increasing accuracy.

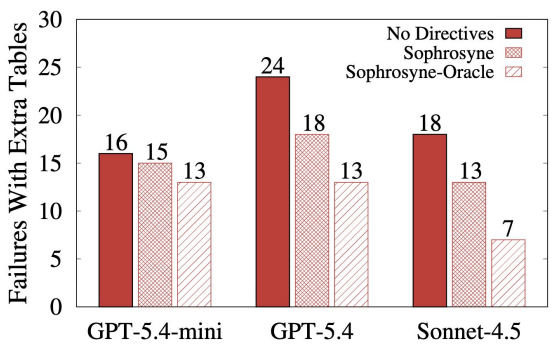
Evaluation

Benchmark: BIRD LiveSQLBench

Agent: OpenCode agent with GPT-5.4-mini, GPT-5.4 and Sonnet 4.5

Systems: Fine-grained API surface with *No Directives*, with directives (*Sophrosyne*) and perfect directives (*Sophrosyne-Orcale*).





Directives consistently reduce over-exploration and reduce the number of failed queries with extra tables thereby increasing accuracy.

More evaluations in the paper:

The directive computation mechanism achieves an accuracy of **82.5%**.

Sophrosyne incurs an overhead of **8 cents** for the directive computation.

Sophrosyne reduces the total $ cost by up to **~8%** with the exception of GPT-5.4-mini where it incurs a **0.5% overhead** in cost.

Conclusion

We observe *fine-grained* and *coarse-grained* as 2 broad categories of API surfaces exposed to agents by data systems.

Conclusion

We observe *fine-grained* and *coarse-grained* as 2 broad categories of API surfaces exposed to agents by data systems.

We identify **over-exploration** as a symptom of fine-grained APIs.

Conclusion

We observe *fine-grained* and *coarse-grained* as 2 broad categories of API surfaces exposed to agents by data systems.

We identify **over-exploration** as a symptom of fine-grained APIs.

Effective use of fine-grained APIs is enabled through curbing over-exploration and ***directives*** help achieve this.

We present *Sophrosyne,* a practical implementation of a data system environment that computes and returns directives to mitigate the problem of over-exploration.

Thank you!

Please see our paper for more details.