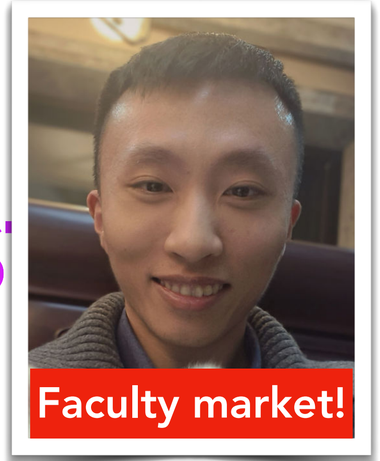

OH GOD, NOT THE AGENTS!!
HIDE THE DATA!!
THE AGENTS ARE HERE!!!
ERASE EVERYTHING!!!
I'M GOING INTO HIDING!
STAY CALM, STAY CALM...!
SYSTEMS

Towards Agent-First Data Systems

Shu Liu, Soujanya Ponnapalli, **Shreya Shankar**, **Sepanta Zeighami**, Alan Zhu
Shubham Agarwal, Ruiqi Chen, Samion Suwito, Shuo Yuan, Ion Stoica, Matei Zaharia
Alvin Cheung, **Natacha Crooks**, Joseph E. Gonzalez, **Aditya G. Parameswaran**

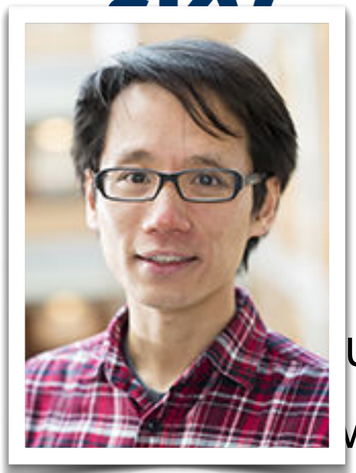
Towards Agent-First Da s

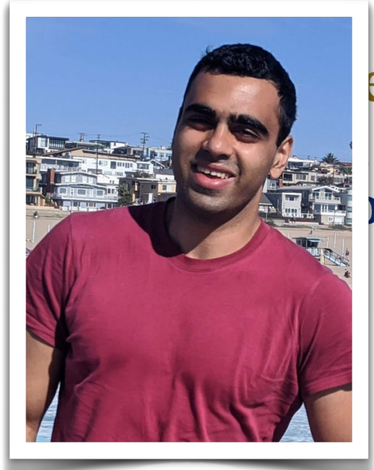


Audrey Cheng **Yiming Lin**

EPIC DATA lab
UC

eley
Systems
undations





ujar li, **Shreya Shankar**, **Sepanta Zeighami**, Alan Zhu
Sh wal, Samion Suwito, Shuo Yuan, Ion Stoica, Matei Zaharia
Alvin Cheung, **Natacha Crooks**, Joseph E. Gonzalez, **Aditya G. Parameswaran**

The Future is Agentic!

The Atlantic Sign In Subscribe

TECHNOLOGY

Move Over, ChatGPT

You are about to hear a lot more about Claude Code.

JANUARY 14, 2026, 3:48 P

< ALL PRESS RELEASES

Snowflake Intelligence Brings Agentic AI to the Enterprise

No-Headquarters/BOZEMAN, Mont. – November 4, 2025 – Snowflake (NYSE: SNOW), the AI Data Cloud company, today announced that Snowflake Intelligence is now generally available to Snowflake's global customer base of more than 12,000 organizations. Snowflake Intelligence is an enterprise intelligence agent that provides every employee with the ability to ansv

ars TECHNICA SIGN IN

GOTO 10

OpenAI built an AI coding agent and uses it to improve the agent itself

"The vast majority of Codex is built by Codex," OpenAI told us about its new AI coding agent writing code.

BENJ EDWARDS – DEC 12, 2025 2:16 PM | 236

Databricks Launches Agent Bricks: A New Approach to Building AI Agents

USA - English

databricks

NEWS PROVIDED BY
Databricks →
Jun 11, 2025, 09:00 ET

QUASA

15.01.2026 09:07 • Author: Viacheslav Vasipenok

Cursor's AI Revolution: Building a Browser from Scratch with GPT-5.2 Agents in Just One Week

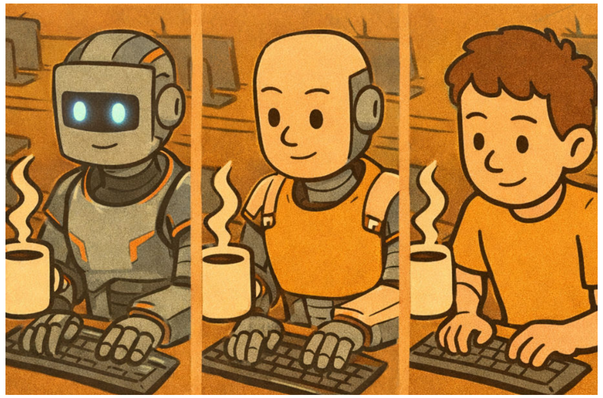
Google Cloud Blog Contact sales Get started for free

Data Analytics

The Data Engineering Agent is now in preview

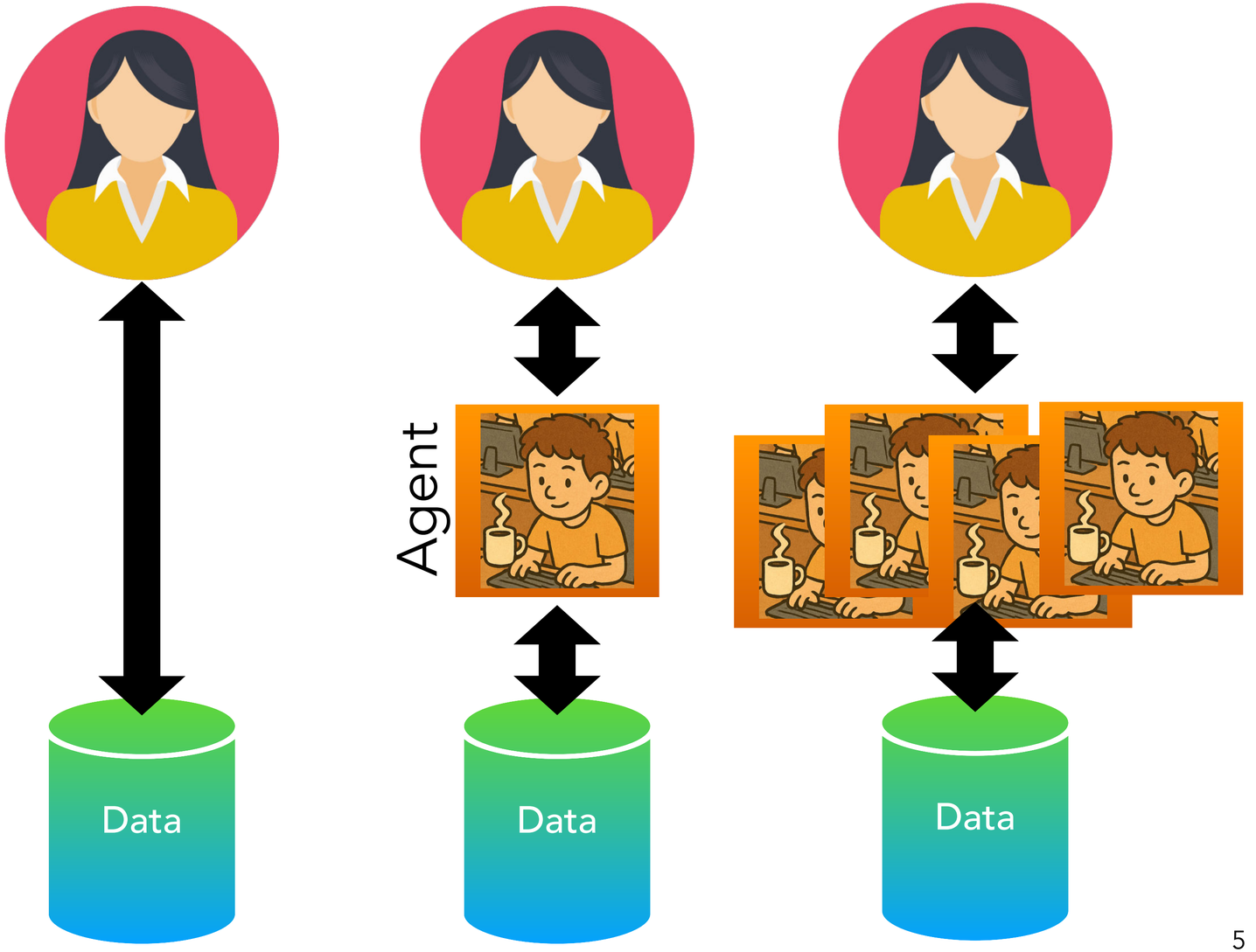
November 3, 2025

LLM Agents Are Getting Better! And Cheaper!



Prediction: *will soon be the intermediary for most data work*

Future: *each data scientist, analyst*, engineer, .. with an agent army at their fingertips!



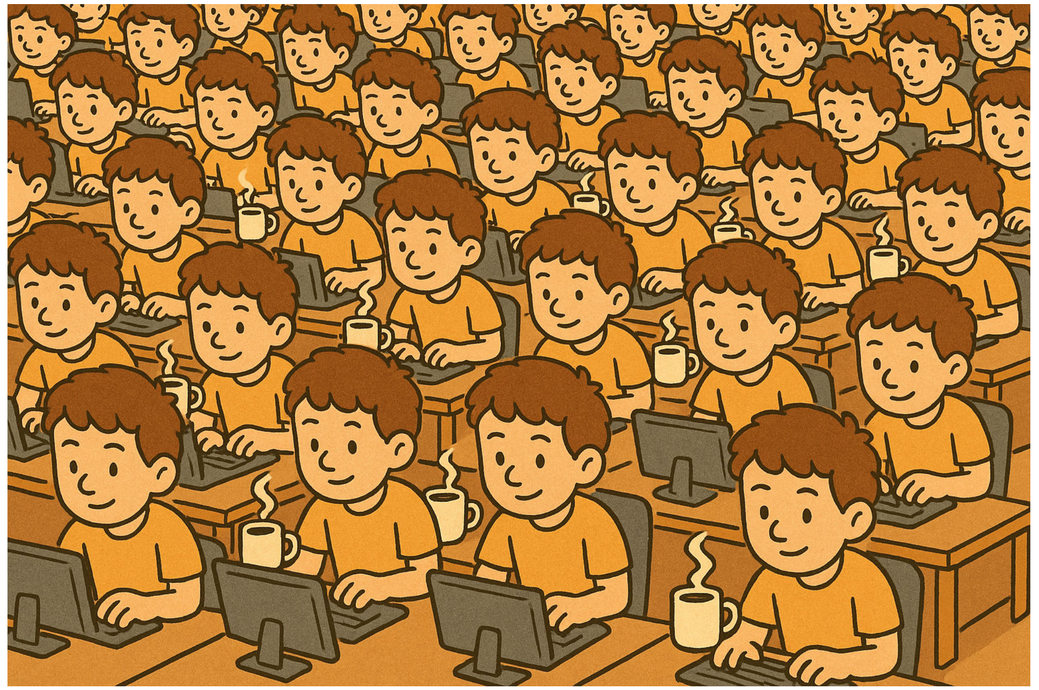
But How are LLM Agents Different From Humans?

❌ Lack **grounding**

Understanding of data/system characteristics

✅ Can tirelessly issue queries!

100s-1000s of requests/s/agent



Agentic speculation: exploratory process of discovery + solution formulation

A lot more than typical human users do, but very inefficiently!

An Example of Agentic Speculation

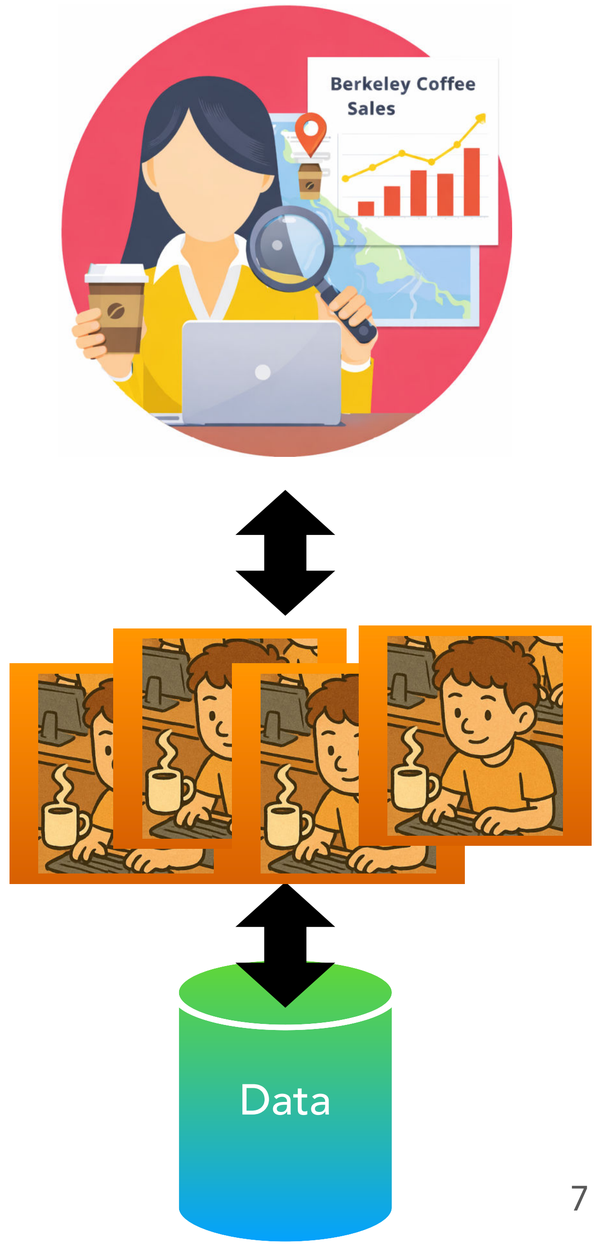
Task: *find out why coffee sales in Berkeley is low this year*

Spin up an agent army to explore various hypotheses

Each exploration or hypothesis = query on sales data and/or other sources

Combinatorial # of possible queries of interest!

Each agent can issue 100s/1000s of queries a second

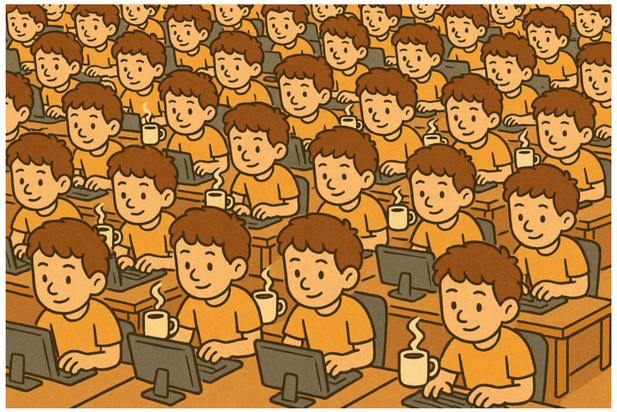


So… agents could dominate and place high demands on data systems.

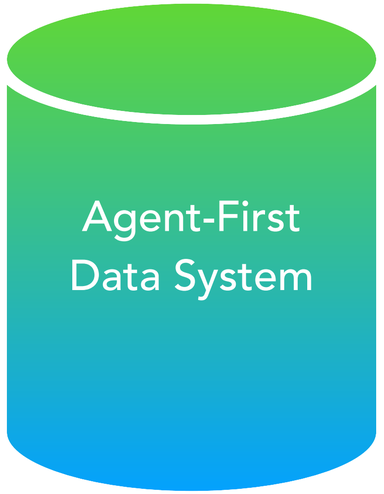
How can data systems better support such agentic workloads?

But first, what does agentic speculation look like?

- High throughput
- Heterogeneous
- Redundant
- Steerable



???

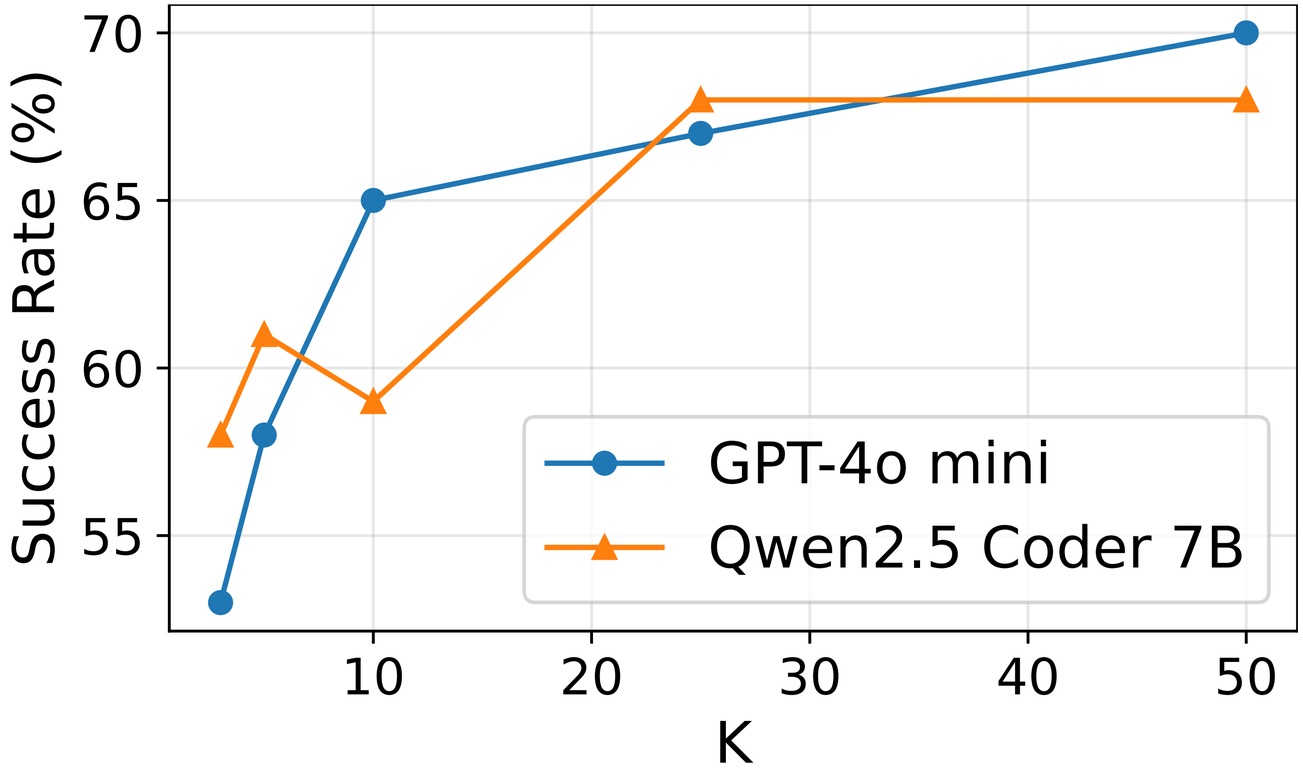


Characteristics of Agentic Speculation: High Throughput

Can “try a lot” - and this helps!

Case study: Text-to-SQL

- Multiple agents attempting; coordinator picks best
- **40% increase in accuracy via more agentic attempts!**



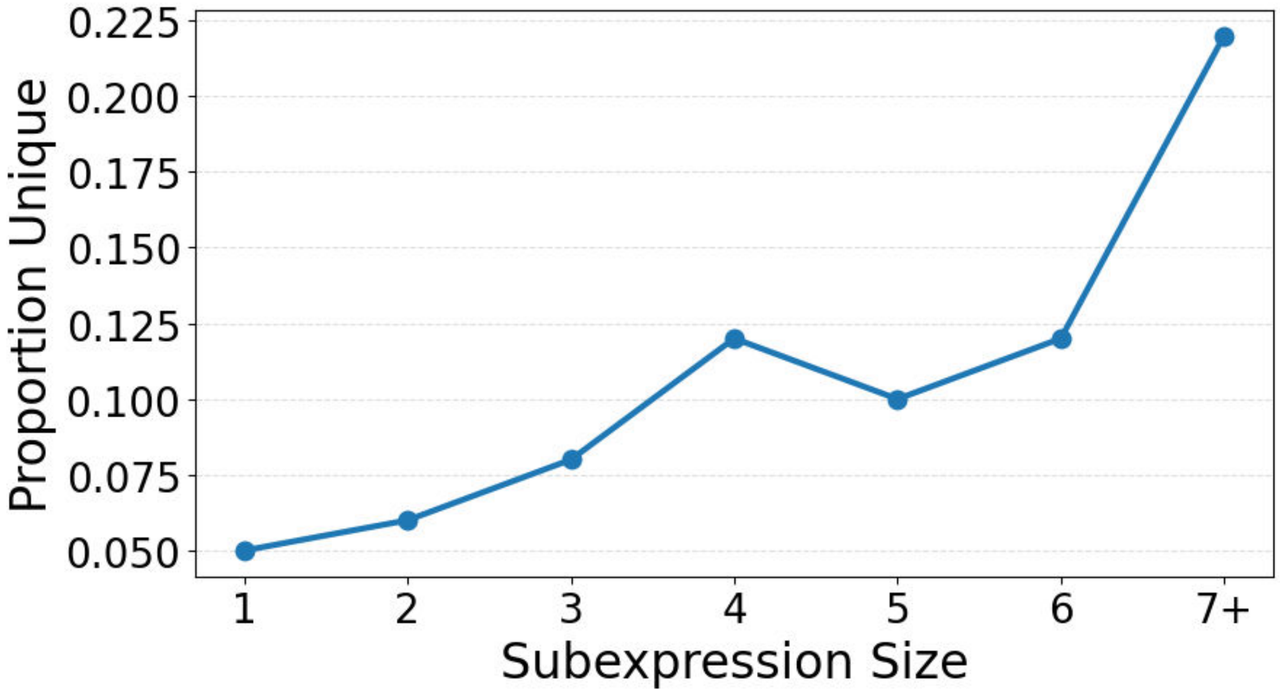
BIRD Benchmark on DuckDB

Opportunity: Supporting many LLM agents improves accuracy!

Characteristics of Agentic Speculation: High Redundancy

The “tries” have a lot of overlap

>85% of subexpressions of size <7 are redundant



Same 50 attempts; on DuckDB

Opportunity: Sharing/reusing computation can improve efficiency!

Characteristics of Agentic Speculation: High Heterogeneity

Different phases w/ different degrees of complexity

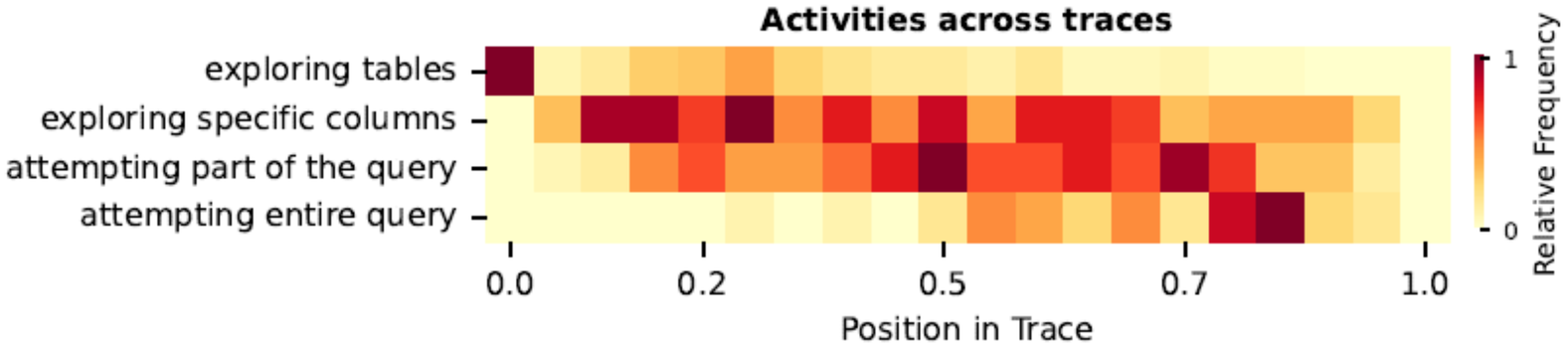
Phase 1: *Metadata Exploration*

Exploring schema, tables

Column level-stats, dist.

Phase 2: *Solution Formulation*

Think subquery/CTE -> Full



Return to Phase 1 when Phase 2 fails!

Opportunity: Recognizing “phase” can help systems help agents make progess!

Characteristics of Agentic Speculation: High Steerability

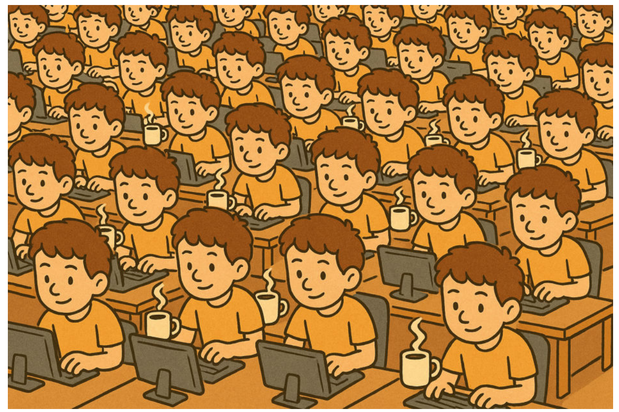
Providing NL *hints* back to agent can make speculation more efficient

Reduction by **~20%** in queries

Activity	Reduction (%)
exploring tables	-14.2
exploring specific columns	-27.7
attempting part of the query	-36.6
attempting entire query	-16.6
all SQL queries	-18.1

Opportunity: Can improve efficiency by providing NL feedback/hints

So… how can data systems better support such agentic workloads?



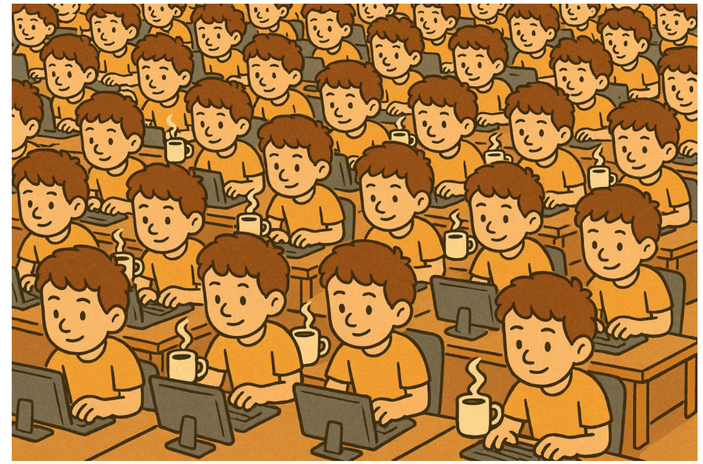
Characteristics of agentic speculation

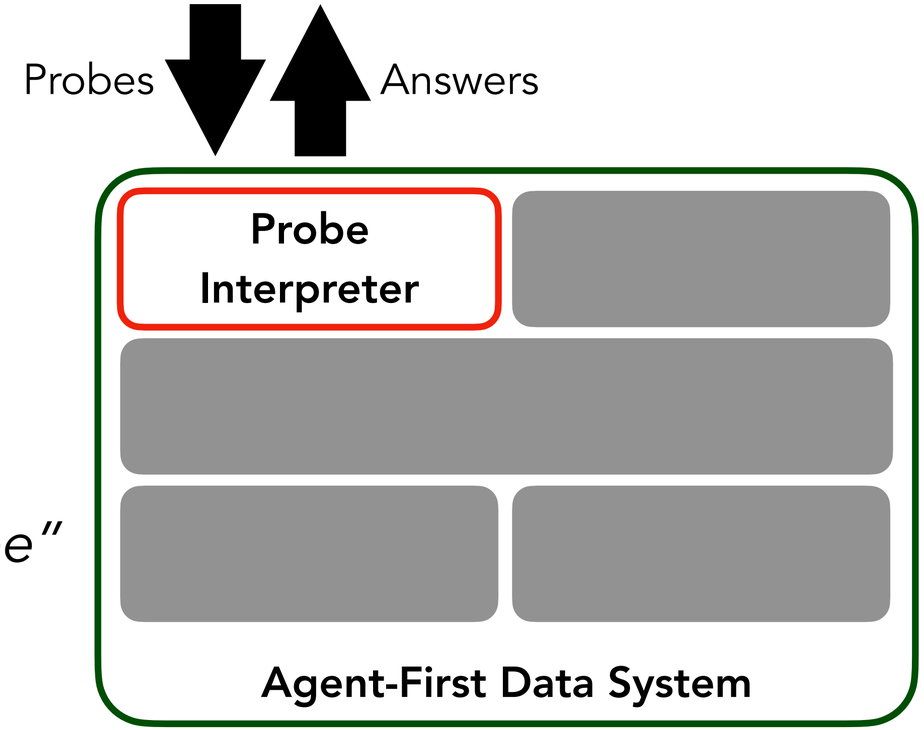
- High throughput
- Heterogeneous
- Redundant
- Steerable

Next: a research vision for our community around **agent-first data systems**

Query Interface I: Queries → Probes

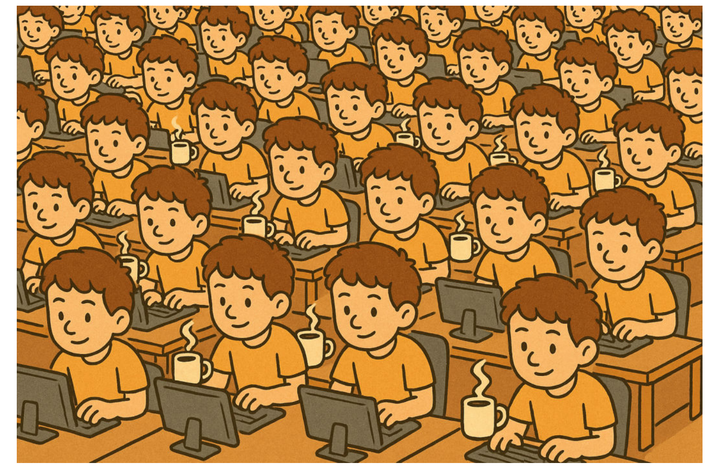
- **Probe** = more than SQL!
- One possibility:
  - A "batch" of SQL (attempts from agent army)
  - + background info, including:
    - Phase of agent (+identity)
    - Goals of probe
- Also new multi-table primitives:
  - **semantic search** over all data/metadata
  - *"find all tables that mention anything about coffee"*
- Probe interpreter will have to itself become *agentic*!

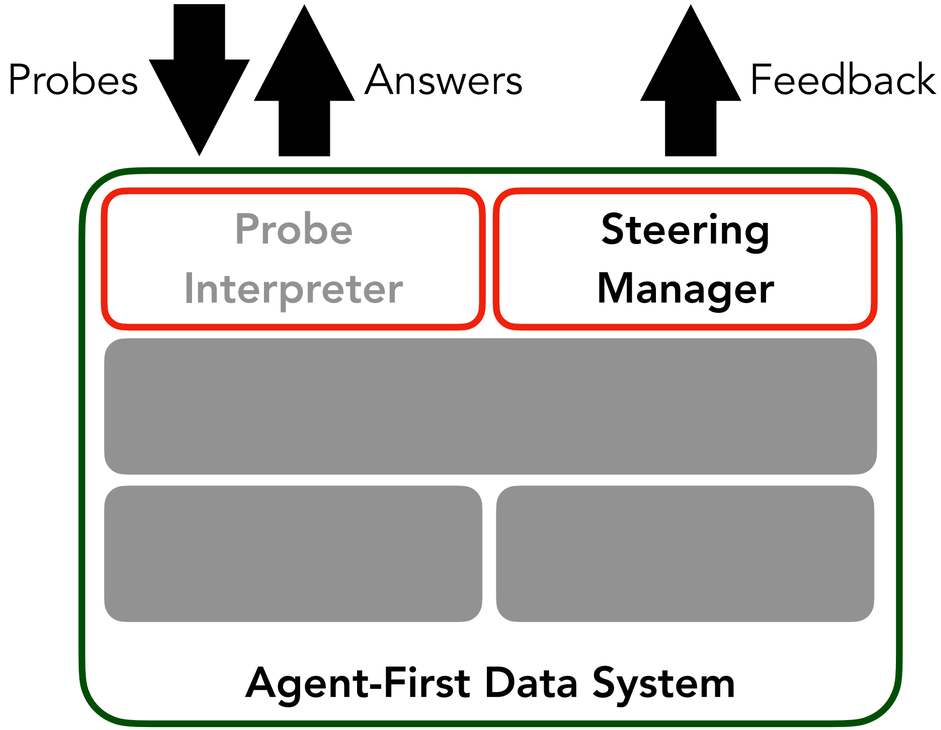




Query Interface II: Active → Passive

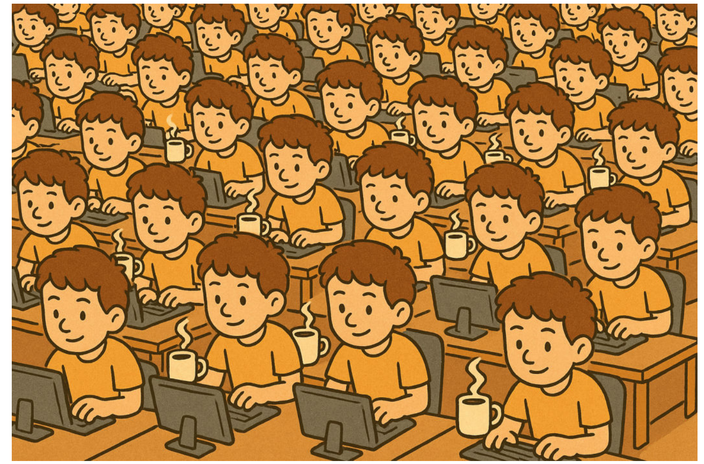
- Expand outputs to include **steering** “hints”
  - Providing **grounding**
- Hint options:
  - *“query you’re about to run is expensive”*
  - *“the assumption of the state column encoded as 2 letter codes is incorrect”*
  - *“consider joining with this related table”*
- Again, providing steering will require data system to become agentic

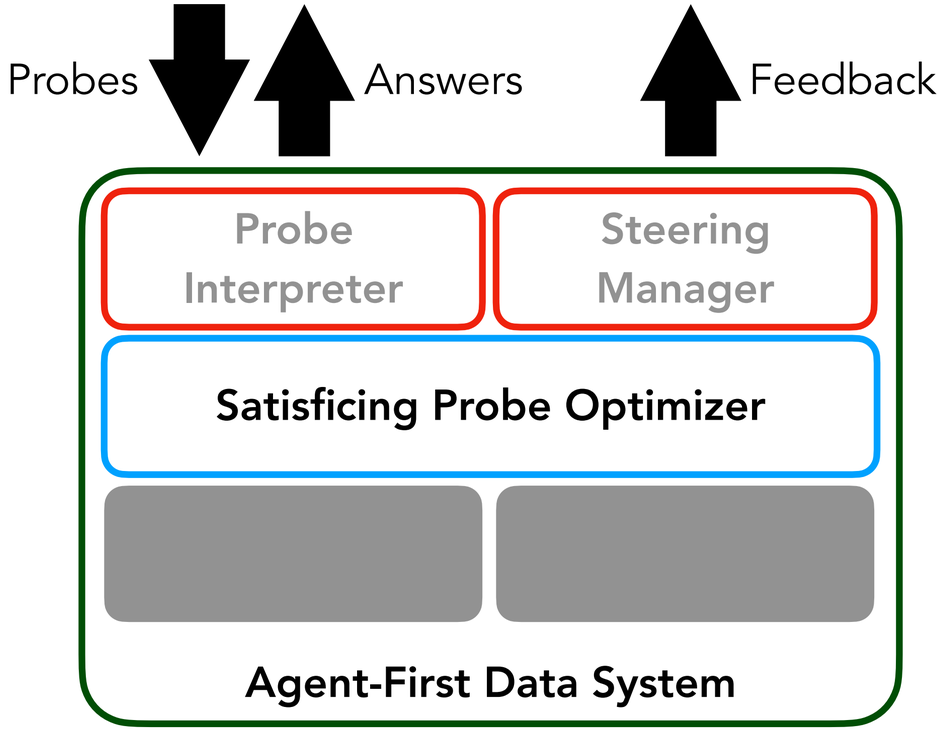




Query Optimizer: Satisfice Instead!

- Old: *optimize each query* — independent, one-shot, executed completely
- New: *minimize total time* answering probes to solve the task, given resources
- Deciding what to execute, to what extent
  - Can use phase of agent
  - "***Satisfice***" rather than complete
- Efficient execution
  - Multi-query opt./shared scans
  - Approximate query processing
  - Result caching and reuse of "nearby" queries



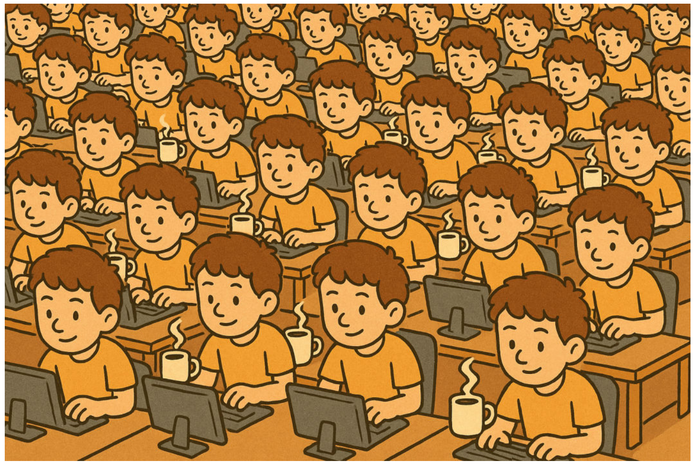


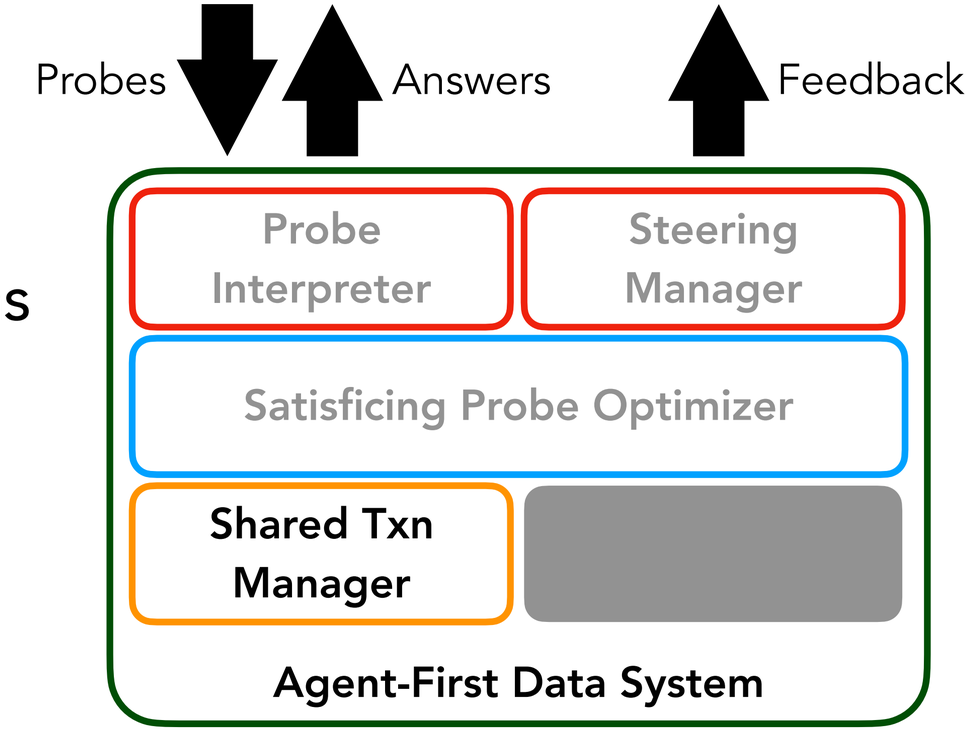
SEEDB: Efficient Data-Driven Visualization Recommendations to Support Visual Analytics, VLDB 2015
Interactive Query Synthesis from Input-Output Examples, SIGMOD 2017

Storage I: Multiversioning → Branches

- For updates, agents will explore multiple what-if hypothesis & make possibly conflicting changes.

- Each branch must logically be isolated but may physically overlap
- Support cheap branch creation, maintenance, as well as rollback





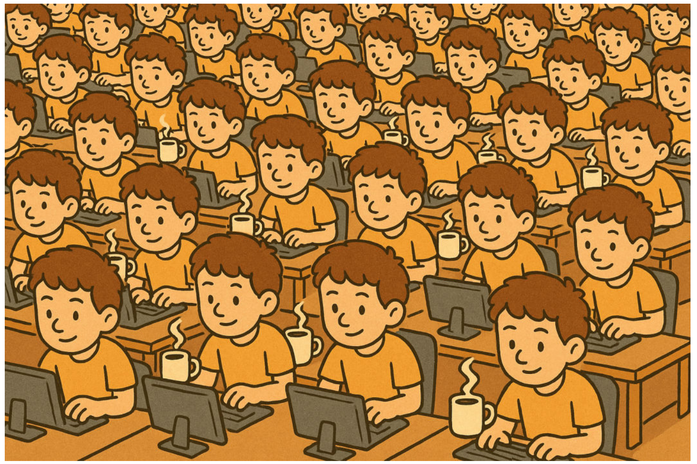
TARDiS: A branch-and-merge approach to weak consistency, SIGMOD 2016
OrpheusDB: Bolt-on Versioning for Relational Databases, VLDB 2017

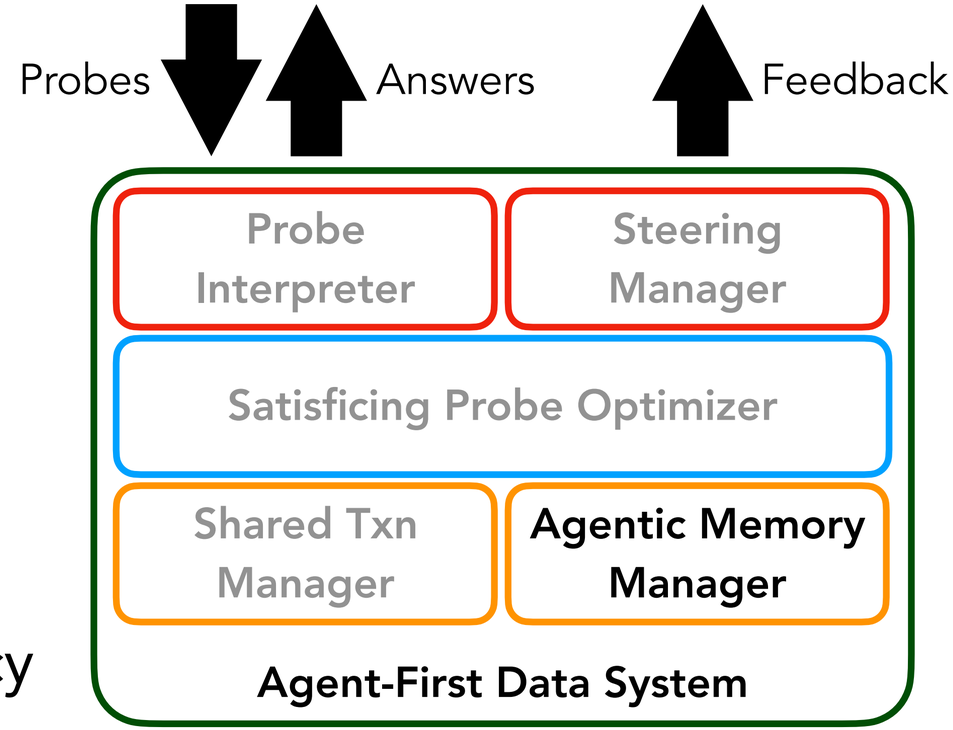
Storage II: Memoryless → Memoryfull

- Goal: maintain agentic memory for grounding

- What can be stored? **Tribal Knowledge!**
  - Results of prior probes
  - Metadata learnt about tables
  - Real world/user expectations
    - *by total revenue, users mean monthly, not annual*

- Issue: challenges regarding consistency and privacy





Overall Takeaways

- Agentic workloads are fundamentally different
- Our systems are **not ready for agents**
- Lots of problems in **rethinking every layer** in a DB stack to support agents
  - Interfaces: SQL → flexible *probes*; output *feedback*
  - Optimizer: *"satisfice"* agentic needs; take advantage of redundancy
  - Storage: agentic *memory* to reduce inefficiencies; *lightweight branching and rollback* for updates
- **Exciting, important, and timely new agenda** for the DB community — contributions welcome!