LinkedIn

Empowering self-serve data analytics with a text-to-SQL assistant at LinkedIn

Dec 11, 2024

Agenda

1. Introduction
2. Five strategies to deploy a practical text-to-SQL solution at scale
3. User adoption
4. Agentic future

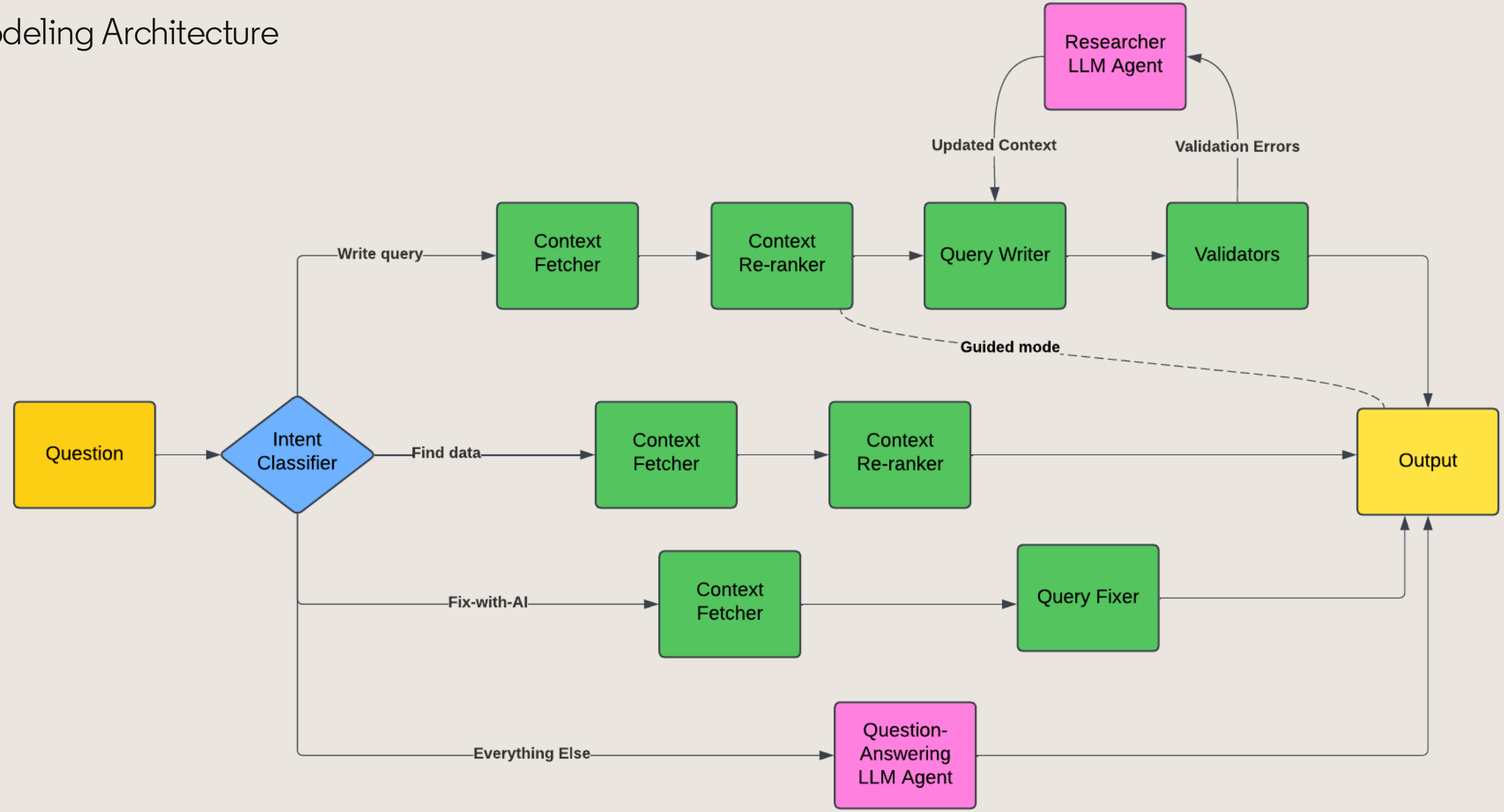
Text-to-SQL comes in many shapes and sizes

- Who is the end user?
- How many datasets?
- How much specialized knowledge?
- What is the target user experience?

Our use case

- All LinkedIn Employees from no SQL experience to SQL / Data experts
- Millions of datasets across product areas
- Data/query assistant for data analytics

Modeling Architecture



Five strategies to deploy a practical text-to-SQL solution at Scale

5 strategies:

1. Quality Metadata
2. Query Validation
3. Interactive Chat UX
4. Self-serve Customization
5. Ongoing Benchmarking

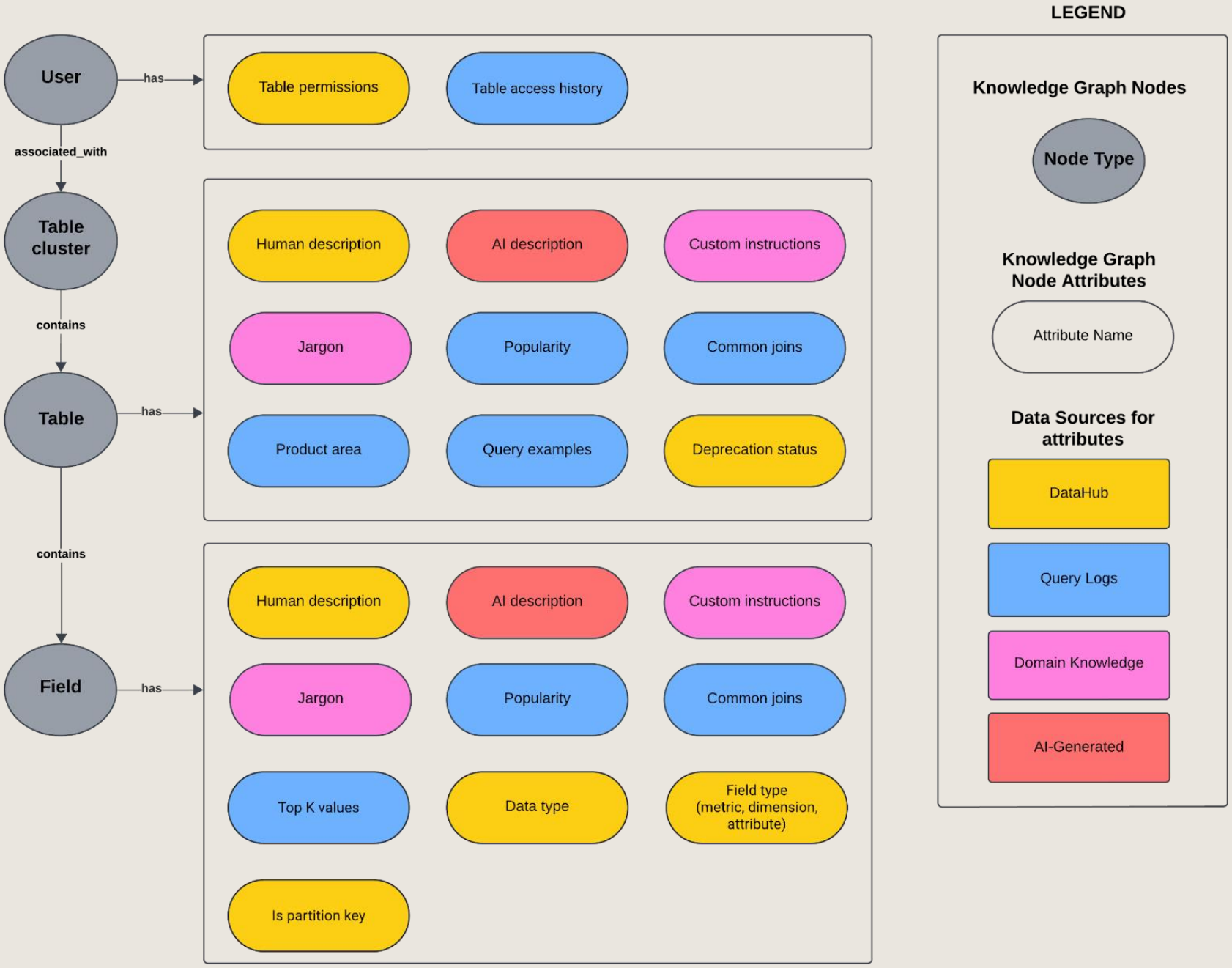
#1 Quality Context

Start with the basics!

Prioritize reliable sources with high information density

Improved descriptions and example queries ➜ big lift in accuracy

Knowledge graph



#2 Query Validation

Always check your work!

Fix errors before returning the query to the user:

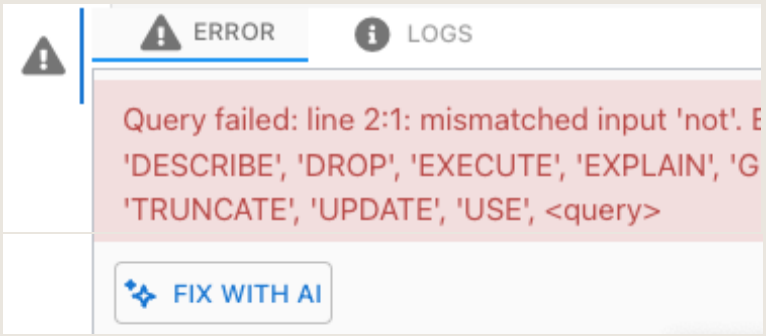
- Do all tables exist?
- Do all columns exist?
- Will the query run?
- Does the user have access to the tables?

We use a self-correction agent to fix errors

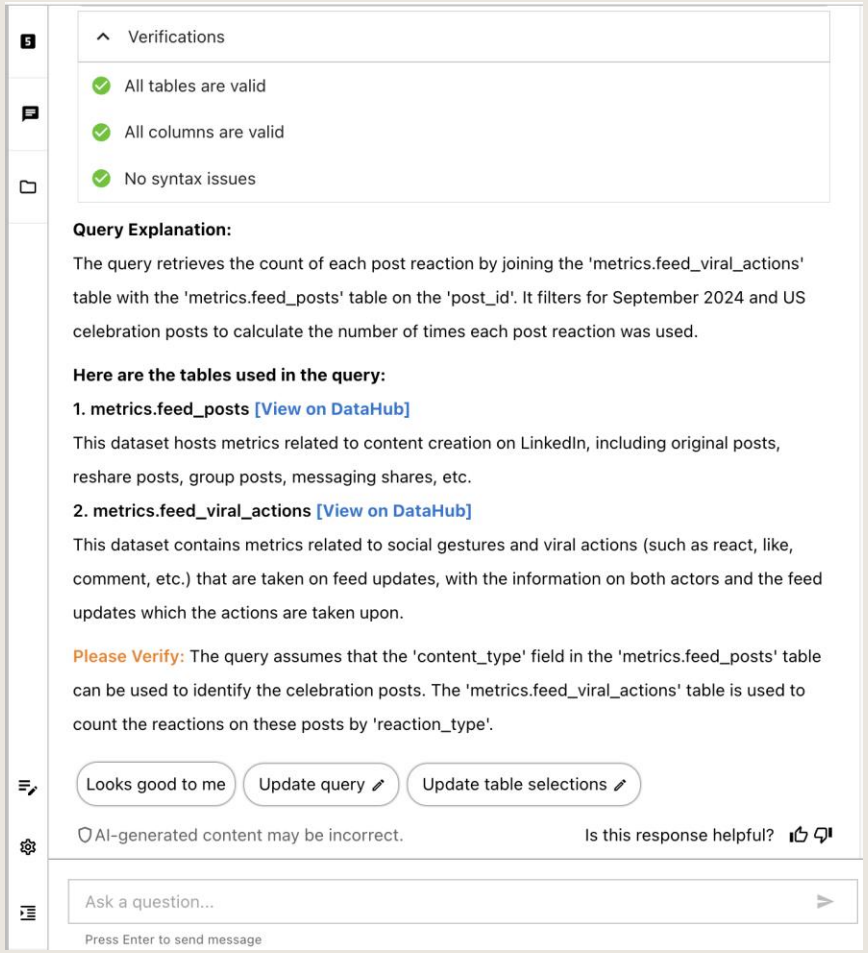
#3 Interactive Chat UX

It's about more than the models!

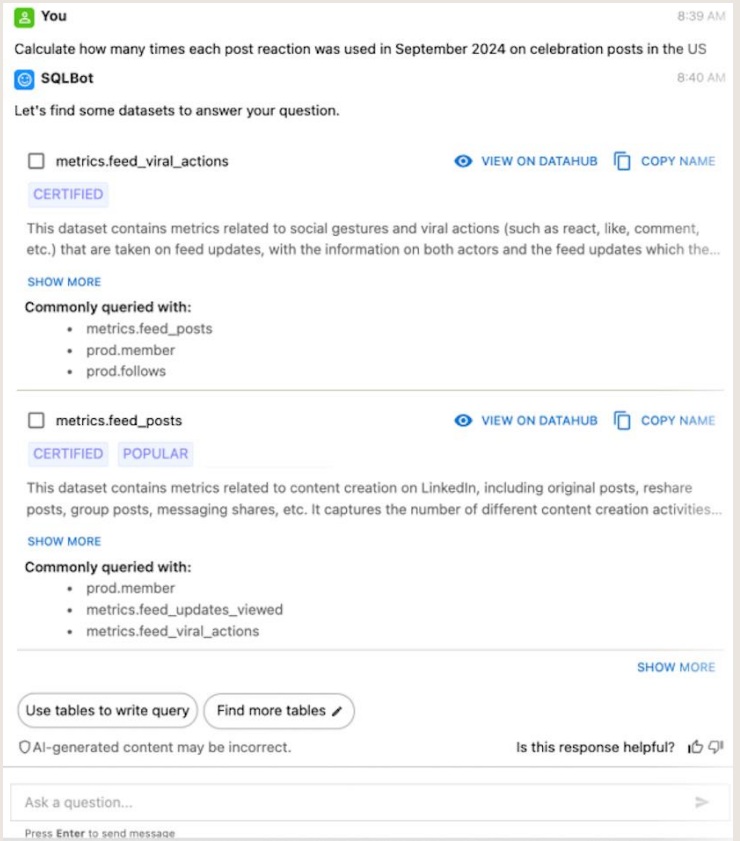
- Timely assistance



- Human-in-the-loop



- Rich chat elements

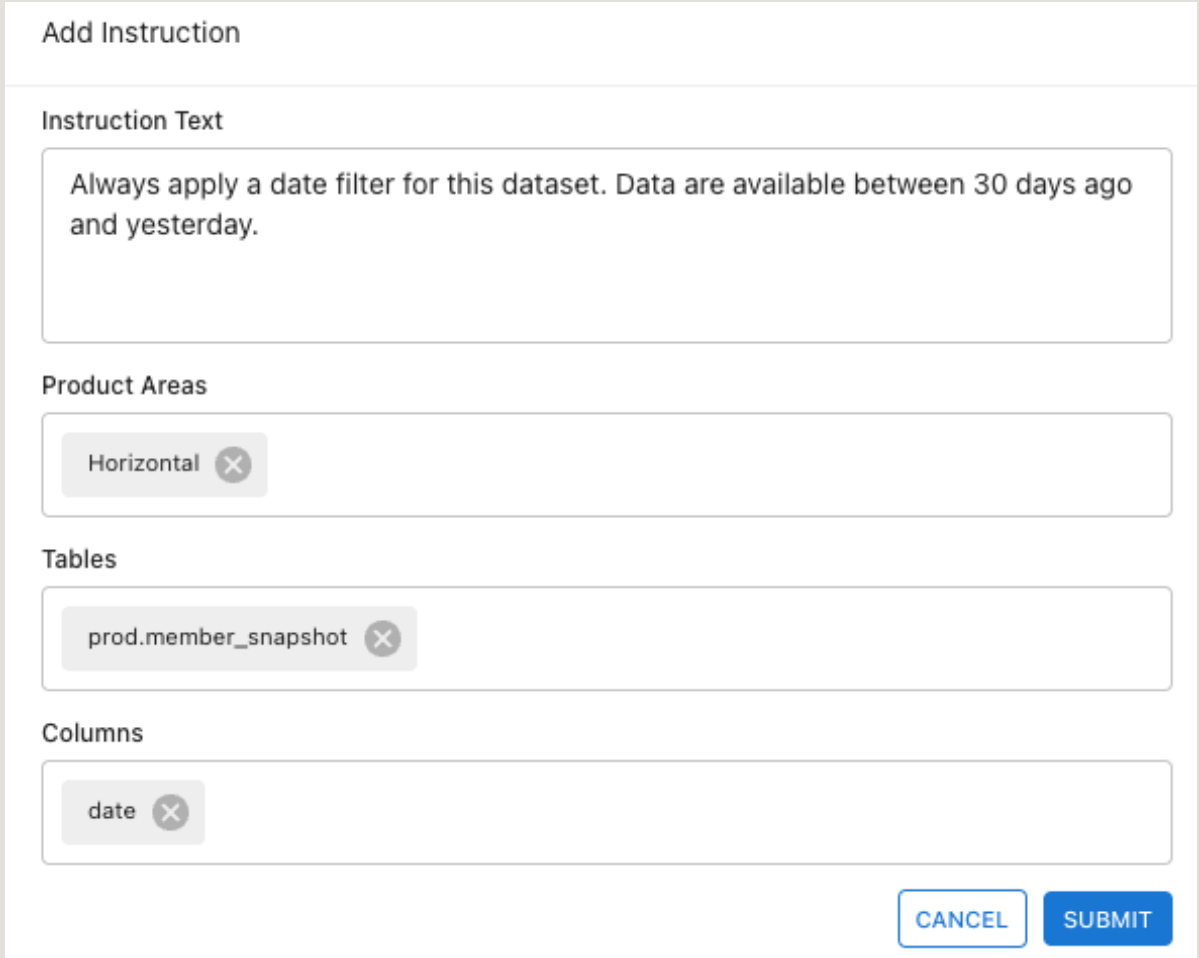


#4 Self-Serve Customization

Empower your users!

Allow users to specify:

- Relevant datasets
- Custom instructions
- Example queries



#5 Ongoing Benchmarking

Speed up your development!

- Curate your own benchmark set
- Define metrics for each part of the stack: retrieval, query writing, fixing, general QA
- Use LLM-as-a-judge to complement human eval
- Note! The same question can be answered with different tables/queries

Key Takeaways from our benchmark experiments

		Table Recall	Field Recall	Hallucination %
Impact of Knowledge Graph Context	Adding example queries	+16%	+14%	+1%
	Adding descriptions and human annotated data	+4%	+8%	-1%
Impact of Modeling Components	Adding re-rankers	+9%	+5%	-8%
	Adding self-correction agent	+2%	+1%	-14%

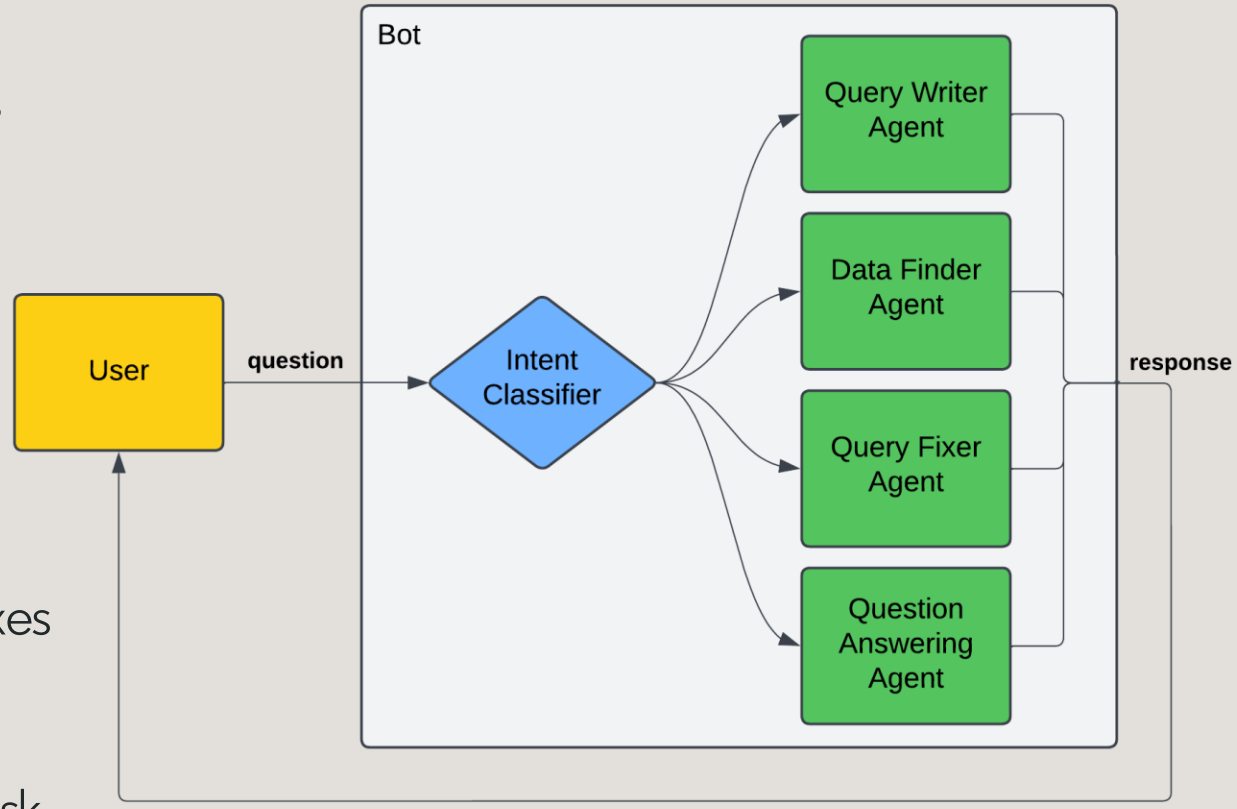
User adoption strategies

Our Audience : All LinkedIn Employees from no SQL experience to SQL / Data experts

- Meeting users where they are - Integration with existing Data Analytics Platforms
    - Drove 5-10X increase in user adoption
- Understanding the user's needs
    - Analyzing the logs to understand the needs based on the intent of the questions users ask
    - UX interviews for different user groups
- Building User confidence in the output
    - "Step-by-step" mode for building user confidence
    - by providing rich chat elements, detailed explanations and certified code snippets
- Making the bot interactions frictionless through quick replies

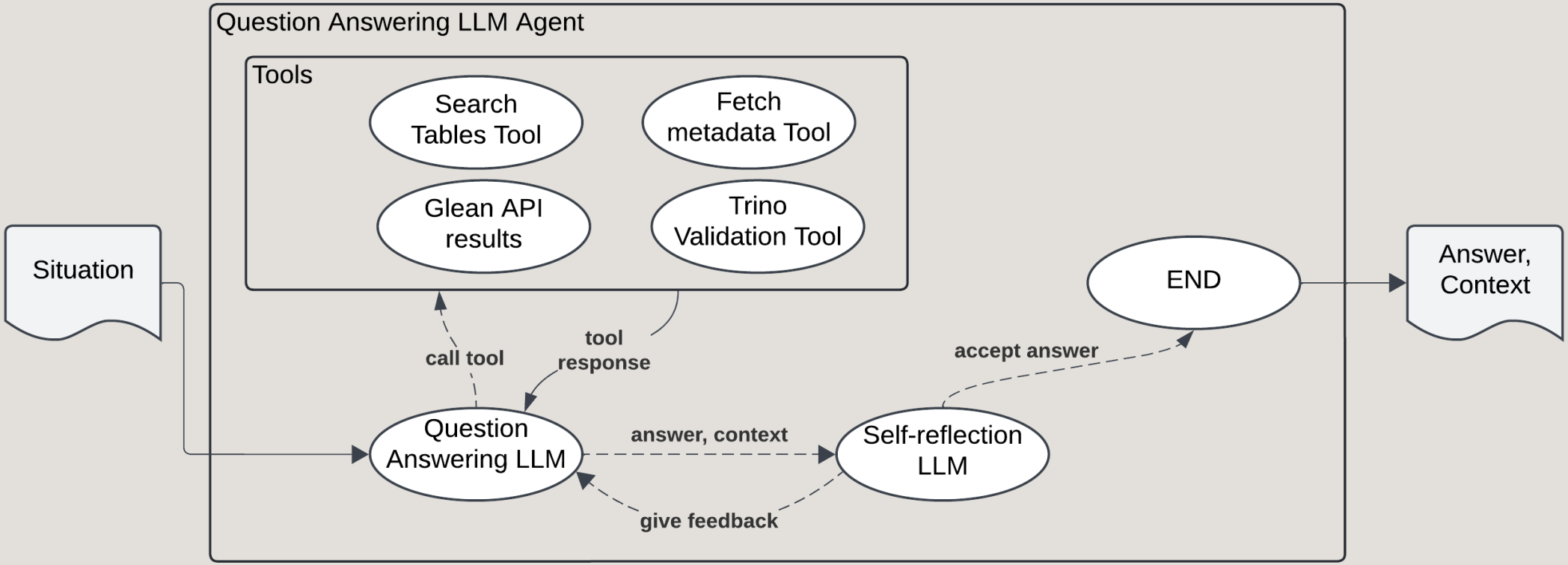
Unleashing potential through LLM Agents

- LLM Agents have an intrinsic versatility to themselves with their access to different tools for updating their knowledge
- Self-Correction Agent
  - Use "Trino Explain" for detecting errors and hallucinations
  - Fetches additional metadata and updates context
  - Searches table/schema info and suggests fixes
- Question Answering Agent
  - Expanding the types of question you could ask this bot!
  - Making the responses more well-rounded



Deeper Dive into the Question Answering LLM Agent

- For "meeting the user where they are" - QA LLM Agent was the solution
- Helps answer questions about the metadata stored across the vector stores and Knowledge Graph
- This helps support questions ranging from optimizing code to updating existing code snippets to answering any general question
- Also improves the interactivity with the user for follow-ups



Questions?

Contributors and Acknowledgements!

- Core team: Albert Chen, Manas Bundele, Gaurav Ahlawat, Patrick Stetz, Zhitao Wang, Qiang Fei, Don Jung, Audrey Chu, Bharadwaj J, Ayushi Panth, Yatin Arora, Sourav Jain, Renjith Varma, Alex Ilin, Iuliia Melnychuk, Chelsea Chueh, Joyan Sil, Xiaofeng Wang

- Thanks to all contributors to the project across Engineering / Product, including Michael Cheng, Clarisse Rahbar, Sparsh Agarwal, Manohar Mohan Rao, Paul Lee, Vishal Chandawarkar.

- Thanks to Trino experts for brainstorming ideas, writing a query plan parser: Erik Krogen, Slim Bouguerra.

- Thanks to Data Science experts for curating the benchmark dataset and evaluating query accuracy: Kavi Tan, Andrew Jabara, Noora Wu, Ruoyun Guo, Steve Na, Ashish Tripathy, Michael Kosk, Lingjun Chen, Cole Silva, Feiran Ji, Janet Luo, Franklin Marsh, Mengyao Yang, Tao Lin, Huanqi Zhu, Paul Matsiras, Andrew Kirk.

- Thanks to Engineering partners for providing APIs for knowledge graph construction: ShaileshJannu, Na Zhang, Leo Sun, Alex Bachuk, Steve Calvert.

- Thanks to Leadership for supporting this project: Ya Xu, Zheng Li, Jia Ding, Kuo-Ning Huang, Harikumar Velayutham, Shishir Sathe, Justin Dyer.