DashQueries Final Capstone Report

Advisor: Aditya Parameswaran

Team: Anshita Arya, Anukriti Goyal, Molly Zhang, Melissa Licari, Richa Verma (MIMS' 23)

Introduction

The motivation behind the creation of DashQueries stems from the shared experience of our team members who have often found themselves solving ad-hoc business requests for non-technical users. These requests can typically be addressed with a simple SQL query, but they create a significant workload for data teams who must handle a constant stream of data requests that are not served by BI dashboards or internal reports. After speaking with numerous business data users, we discovered that this issue is widespread, as business users frequently require ad-hoc data queries that aren't addressed by existing reporting tools. Therefore, we were inspired to create a bot that can answer these simple business questions. DashQueries is an intelligent system that enables business users to ask queries and obtain answers from their data without having to interact with a data analyst. By addressing this pain point, we hope to free up valuable time for data teams to work on more strategic projects.

Literature Review

In order to build a system that is able to convert a users query that is written in everyday language into a result, we researched Natural Language Queries (NLQ), how other companies have attempted to use NLQs, and how Large Language Models (LLMs) are becoming a standard for converting natural language into code.

Natural Language Queries translate questions asked using everyday language, without special syntax using Natural Language Processing (NLP) into data query languages like SQL. It doesn't require the user to know the technical language or understand the databases they want to query from to find the information they need to make business decisions. Several companies, notably Tableau and Thoughtspot have attempted to build NLQ capabilities within their current product offerings; however, they have come up short in their efforts.

Tableau released a feature, "Ask Data", in an effort to help users visualize and analyze their data easily. Users start by typing a question in plain language which Tableau parses and transforms into a query and then displays the result as a Tableau dashboard. While users are first prompted to type in a phrase, Tableau then prompts the user with a drop down list of its own interpretation of the query that it will ultimately execute. This may require a user to have background knowledge about the data and its schema to fully interpret the options given. Based on how the user types in the query, Tableau's interpretation of the query may or may not be correct. Additionally, when a user is asking a question pertaining to a column in the dataset, the question has to be phrased in a way that matches the column name exactly in order for Tableau to interpret it and to list the options of queries it can execute. Hence, concepts are not automatically recognized and the users have to intervene to select what they believe is the most appropriate query or data attribute.

Some other limitations with Tableau are that users are only permitted to ask questions for the data that they have access to in Tableau, potentially discounting other data sources a business has access to that may be integral to their analysis. Its current NLP capabilities struggle with questions around comparison, negations and percentages. The software itself has a learning curve that non-technical users have to learn before they can start asking natural language questions since their NLQ component was not built to be a standalone feature to Tableau's current functionality of building dashboards.

ThoughtSpot's platform enables self-service data requests by using NLQ and subsequently allows users to create interactive dashboards, reports, and visualizations to gain insights from their data. It has been designed to be a natural language search engineer with a search-based BI interface. Similar to Tableau, a user types in a query using natural language after which they are prompted to select a query to execute from a list presented to them.

Some limitations with ThoughtSpot are that there is no feedback loop where a user can choose optimized queries to tune the system's performance. Additionally, only simple or constrained NLQs are currently supported. The interactive visualizations that are provided in response to the queries are minimal, hence, the question a user asks must be the type where there is an exact answer rather than having a range.

Tableau and ThoughSpot both provide natural language suggestions to help the user select the query they would like to execute. However, they only focus on existing natural language queries. Users also need to specify their data queries for every exploration step. The suggestions presented are a result of the systems attempt to understand the analytic intent behind the user's input as well as the current visualization state and recent actions that have been taken by the user. The underlying NLP pipeline in both systems uses attributes from the data source and analytics tasks to determine user intent for generating the options presented to the user (Wang and Cheng).

The effectiveness of leveraging LLMs like GPT-3 and ChatGPT as alternatives to NLP for converting natural language to code has also been evaluated. Paula Maddigan and Teo Susnjak created Chat2Vis that generated data visualizations using code generated from LLMs. They found that using LLMs to interpret user queries increased the accuracy of understanding user requests and decreased the costs associated with the parsing and converting this information when compared to previous NLP and deep learning approaches to do the same. Implementations of current Natural Language Interactions have been challenging since user queries written in natural language can be ambiguous or poorly written that make it difficult for the system to understand user intent. LLMs also preserved data privacy which meant that it could be generalized to be used and applied to all types of datasets, irrespective of confidentiality concerns (Maddigan and Susnjak). This study also highlighted the importance of prompt engineering to provide the LLMs with clear natural language requests but outlined the challenges with prompting the LLM to produce graphs with additional aesthetics such as grid lines and colors. Another study verified ChatGPT's robustness with converting user input text to SQL statements that can be executed on databases, especially when using common and simple prompts (Liu and Hu).

What we Evaluated/Hypothesis

We are trying to understand the following two questions:

1. Will business users' actually trust a natural language query based system to make decisions?

   While such systems have the potential to provide a more accessible means of accessing and exploring data, the question of trust is crucial. We hypothesize that by providing a good user interface through the chatbot and by providing accurate answers, users will be able to develop a higher level of trust in the tool and its output.

2. What are the querying/exploration limitations of such a tool?

   We acknowledge that bots can be an assistive tool to solving business questions, but we also are aware that there are limitations to using AI. Our hypothesis is that by providing a balance between intuitive natural language querying and the ability to handle complex queries, our system will enable business users to effectively explore and analyze their data.

Through rigorous evaluation of these hypotheses, we aim to gain a better understanding of the potential benefits and limitations of natural language query-based tools for business users. This will inform our learnings and provide insights into how we can better improve such a tool in the future.

Methodology

We employed a Praxis approach to gain insights about the effectiveness and limitations of our natural language query-based tool, "DashQueries." We followed the following process.

1. We created "DashQueries," a tool that enables users to ask data queries through Slack, a widely used communication platform. This tool was designed to provide an intuitive and accessible means of accessing and exploring data, particularly for business users who lack technical expertise in data analysis. To create this chatbot, we used Patterns app, a platform that assists with creating AI tools. The primary technology used is GPT-3 to build our model.
2. We analyzed the queries run through our tool to gain insights into the accuracy of the system's outputs and to identify potential limitations in its ability to handle complex queries, through several test datasets that contained sales and product data. This

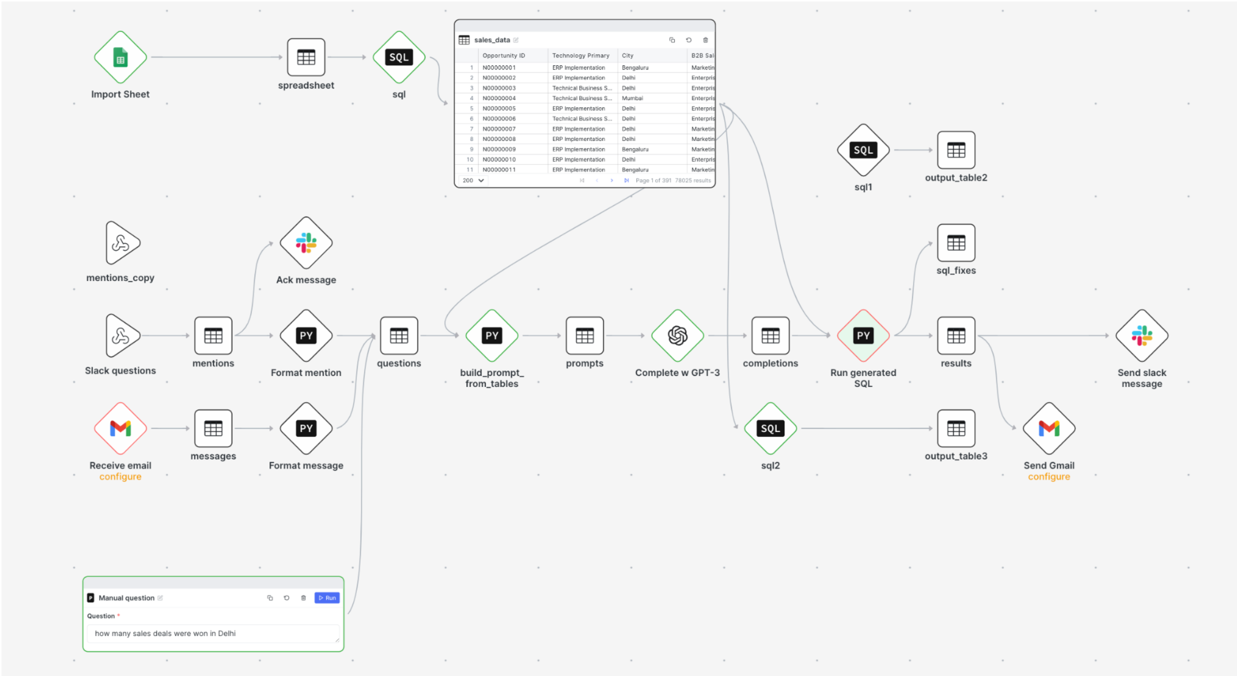
analysis involved a thorough examination of the queries processed by DashQueries, including their syntax, semantics, and context.
3. To gain a deeper understanding of users' experiences and expectations of our tool, we conducted informal interviews with both business and data users. These interviews involved open-ended questions designed to elicit feedback on the usability, effectiveness, and limitations of the system. Through the user interviews, we were better able to understand what questions our bot could answer well and what questions were out of scope.

Phase 1 Learnings:

Application Status at the start of Phase 1
Our initial prototype was a simple version that parsed the user's question and displayed the corresponding query and response. The overall impression that we got was that the users loved the concept, they were surprised but also delighted by the capabilities, and that they would definitely use a bot like this with improvements made.

First Prototype details:

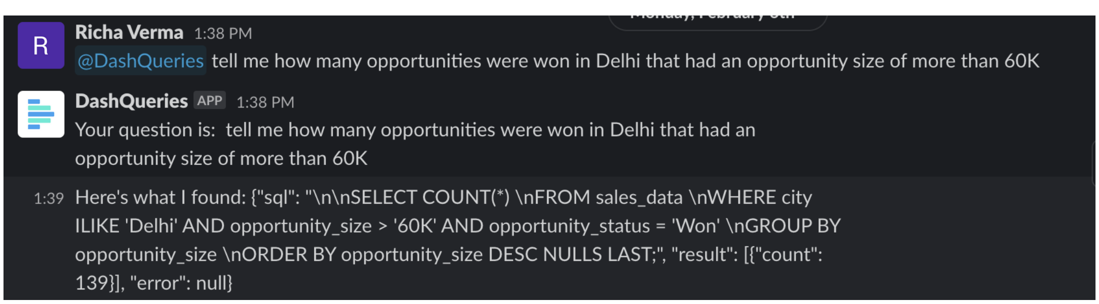
- Architecture - We built a prototype using patterns, to deploy our end to end AI workflow. See image below for the architecture of our phase 1 tool.
- Slackbot - We decided to integrate this product with slack so that it is easy to use by users with minimal setup barrier.
- Data Integration - For the prototype test we integrated with a google sheet static data, but we have done POC to connect to multiple data sources.
- Generative AI - We used the GPT-3 model to generate SQLs
- Prompt Engineering - Based on the user questions and the business user persona, we engineered the prompt to feed into the model.

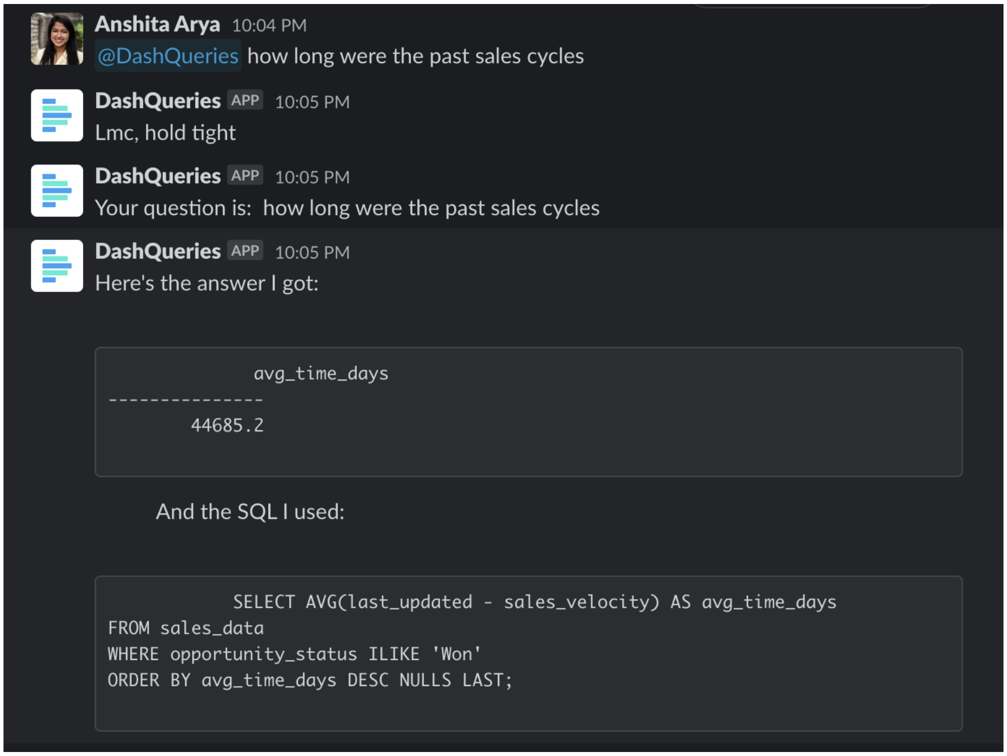


How did we test?

For this phase we tested the bot with over 200+ queries and coded errors/query responses for 70 queries. This was largely done by individuals on the team, by coming up with different types of questions and verifying the responses.

To gather further qualitative responses, we interviewed different User Personas (Technical Engineers/Scientists/Analysts, Sales Engineers/Managers, Product Manager/Owners) to see how users from different backgrounds are interacting with the bot.



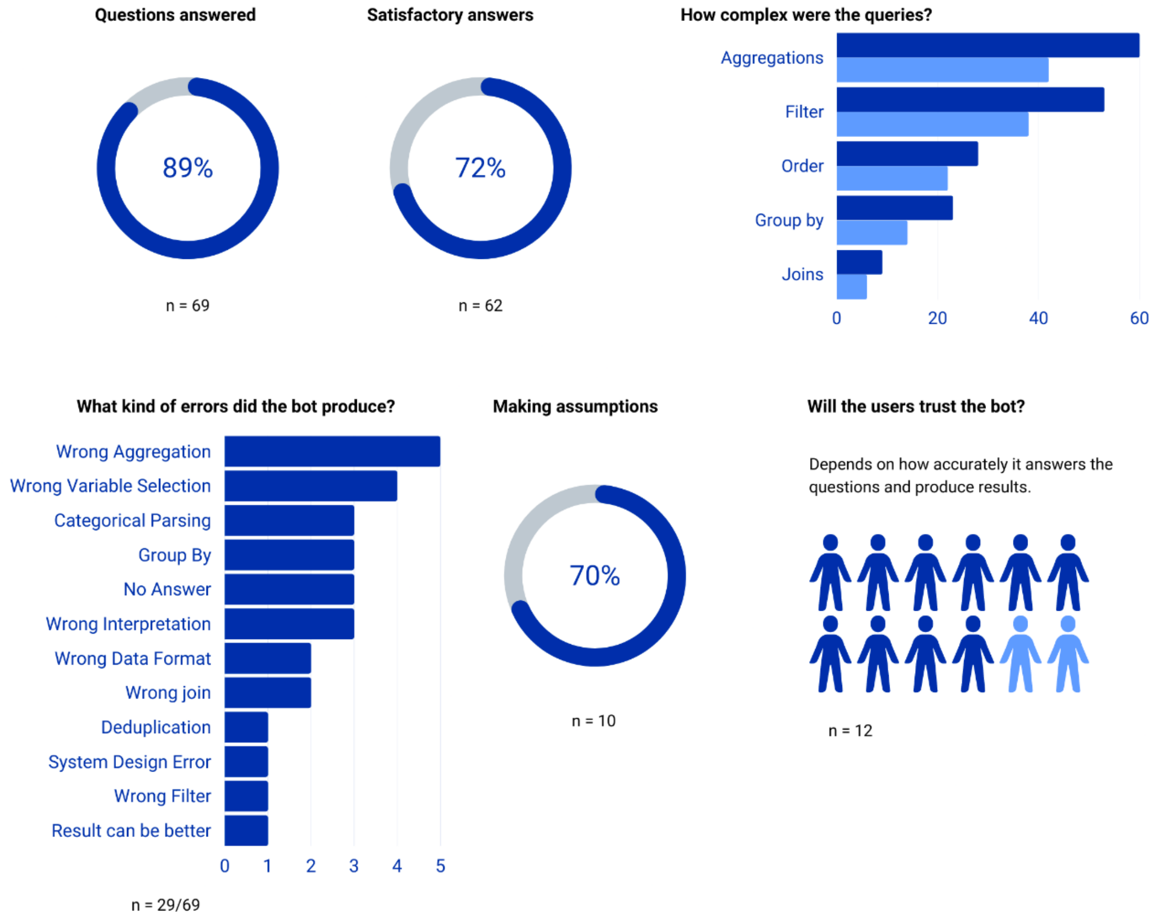


See appendix for our User Script.

Qualitative Analysis of User Interviews:

- SQL query: 3 out of 5 users noted that they really like that they could see the SQL query and it helped build trust. *"It's good that it shows the sql also, so if I want to check it I can run it directly"*
- First response: For 2 out of 5 users, the first response was unsatisfactory which made them more critical. For 1 out of the 2, the second response was accurate.
- Simple queries: at the end of the study 4 out of 5 participants were confident that they could use this bot for simple queries, for more complex queries they would still involve a person, but that might change if the product is improved. *"For simple questions ("directional") and as a where-to-start point. After finding a point of information, use something else for deeper analysis."*
- Visualizations: All users preferred some visualizations like graphs and charts.
- Feedback: 3 out of 5 participants wanted to be able to give feedback to the bot, to correct it if they got an unsatisfactory response or even to ask followup questions.
- Complex queries and Semantics: Most users had concerns around it being able to answer complex questions, which involved multiple parameters or analysis. The users would also try to ask semantic questions.

Error Analysis of Phase 1



We analyzed 67 queries that we ran on the phase 1 bot and found that the bot was able to answer ~90% of the questions (which could be super vague) or can be super directed, and around 70% times of the questions answered, users found that to be satisfactory. Aggregations and Filters were the most used SQL statements, and had the lowest satisfaction rate. And, the most common errors were selecting the wrong aggregation and selecting the wrong variable, which suggests that overall semantics needs to improve. There were major errors in categorical parsing, getting a response, and group bys. Most interestingly, of all the queries where the bot made an assumption, it got 70% right which means NL interpretation was largely right.

Here are some sample queries from phase 1:

Can you actually tell me the number of opportunities that have sales velocity more than 70 and client revenue more than 1M?
Tell me how many opportunities were won in Delhi that had an opportunity size of more than 60K

Why are deals won in Delhi more than Mumbai?
Where are the largest opportunity sizes
How many clients have done repeated sales with

Users were not very specific with variable names, or even the kind of calculation they expect, and are using qualifying words such as “largest”/“deals”/“sales”/“opportunities” interchangeably.

Key Takeaways:

- Trust:

  In general, the trust of users in the bot increased with its usage. Some things which contributed to trust were:

  - Transparency - Transparency such as being able to see the SQL and what is happening in the backend, contributed to higher trust levels
  - First impressions - First impressions also played a critical role in building trust. If the response to the first query was unsatisfactory, trust dropped quickly and was difficult to recover. Conversely, if the first response was satisfactory, trust increased even if in subsequent questions there were unsatisfactory responses.
  - User background - User background was found to be another important factor in trust levels. In general, technical users like data scientists or business analysts had lower trust in the bot than non-technical users.

  Overall, building and maintaining trust is a crucial aspect of our product. Transparency, positive first impressions, and user background consideration all play important roles in establishing trust.

- User Feedback:
  - Technical error - We were giving only one response to the user based on which query might be optimal. There were situations in which one or two parameters were missed and needed to be handled. In the current state, the user would have to re-ask the question.
  - Unsatisfactory response - Often, the first response which was given by the product, although there were no technical errors, was not what the user was looking for. Thus it was deemed unsatisfactory. In the current state, the user would need to rephrase their question.
  - No feedback loop i.e. no way for the user to tell us whether we did a good job or not as well as what they are actually trying to do with it.

Snapshot after improvements during Phase 1:

Fri

Richa Verma 9:34 AM
@DashQueries tell me which city has the most opportunities won?

DashQueries APP 9:34 AM
Runnin' the numbers, gimme a sec

I am trying my best to answer your question: tell me which city has the most opportunities won?

How I work? 🤔

```
    It's simple, I take your question and interpret (1) table you are curious
about, and (2) write an SQL query for it.
    Once the results are ready, I get them back to you.

    Kind of questions I can answer:
    1. Tell me whether sales have increased in a <region>?
    2. Which opportunities were won/lost?
    3. Tell me the average sales for the last quarter/month?
    4. Tell me which product has been selling the most?

    Questions I can't answer:
    1. Why sales changed?
    2. Predict the sales in future..
    3. What's the purpose of life?
```

In the meantime, go back to your work, I'll ping you in this channel once the answer is ready.

Here's the answer I got:

```
         city      opportunitieswon
------  ------------------
Mumbai                5333
```

And the SQL I used:

```
SELECT city, COUNT(*) AS OpportunitiesWon
FROM sales_data
WHERE opportunity_status ILIKE 'won'
GROUP BY city
ORDER BY OpportunitiesWon DESC NULLS LAST
LIMIT 1;
```

Which basically means this in plain English:

```
-- This query returns the city with the most Opportunities Won.
```

Are you satisfied with this answer? (Yes or No)

Which brought us to the question: Can LLMs replace the Business Analyst?

Phase 2 Learnings

Application Status at the start of Phase 2

After our phase 1 research, we thought of fixing the bot further and made some improvements in the results quality as well as the user experience of the bot.

Molly 12:58 PM
@DashQueries what are my biggest revenue streams?

DashQueries APP 12:58 PM
Runnin' the numbers, gimme a sec

Finding the right data for your question: what are my biggest revenue streams?

In the meantime, go back to your work, I'll ping you in this channel once the answer is ready.

Here's the answer I got to your question: what are my biggest revenue streams?

```
            client_revenue_sizing           sum
-----------------------  ----------
100K or less             1763516989
More than 1M              197752656
500K to 1M                154573140
250K to 500K              127992201
100K to 250K              103599025
```

And the SQL I used:

```
SELECT client_revenue_sizing, SUM(opportunity_size_usd)
FROM sales_data
WHERE opportunity_size_usd IS NOT NULL
GROUP BY client_revenue_sizing
ORDER BY SUM(opportunity_size_usd) DESC NULLS LAST;
```

Which basically means this in plain English:

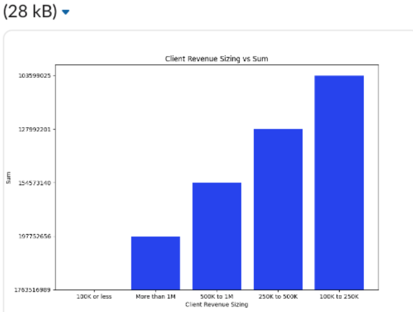
```
            --This query uses the opportunity_size_usd column from the
sales_data table to determine the biggest revenue streams by summing up the
total revenue for each client_revenue_sizing.
```

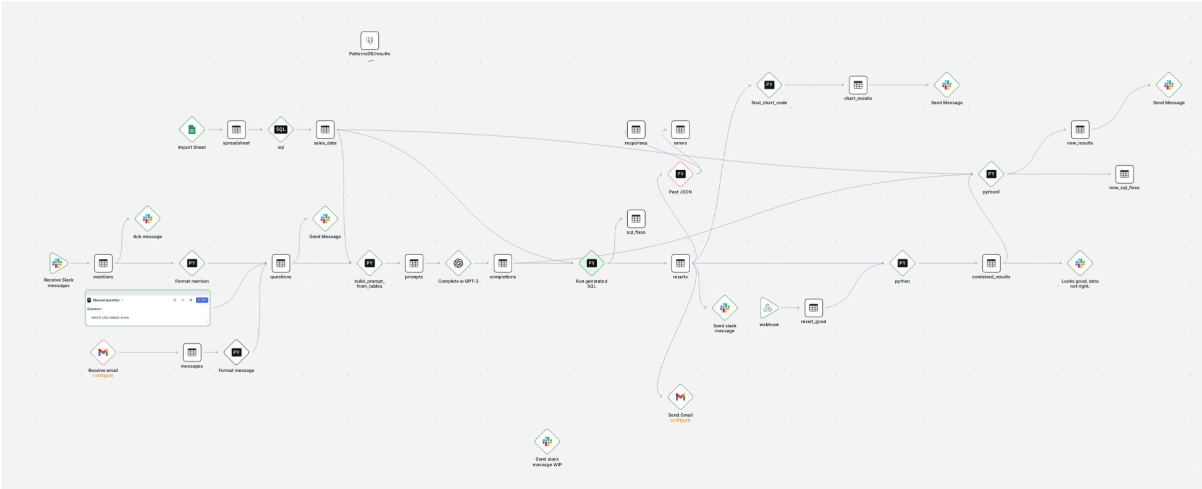
Are you satisfied with this answer?

Looks good | Data seems wrong | Query seems wrong

Here's the chart I got for you: https://dashqueries.s3.us-west-2.amazonaws.com/chart-1683143968.png

(28 kB)





Improvement summary:

1. UX improvements:
    a. Improved the number and frequency of slack messages that the bot was sending.
    b. Improved the workflow by letting the user report whether the answer was correct or not ("Looks good", "Query seems wrong", "Data seems wrong"), and included options to run a second query if needed.
    c. Included a GPT-3 generated chart response as well to give a visual aid for the data.
    d. Improved the length of the resulted table in the slack message.
2. Result quality improvements
    a. We improved the results that the bot was showing, particularly improved "categorical parsing" capabilities of the bot so that all options are considered and the data type is not mishandled.
    b. We improved recognition of previous vs. current query, even though we couldn't test this thoroughly with our users.
    c. We improved the logic for creating charts so that an appropriate chart is generated while creating the charts.
    d. We improved the workflow of getting the second query in case the first query is not correct as we generate 3 queries overall.
    e. Did some system error fixes to ensure that the data pipeline is working properly, the slack responses have a higher limit to showcase as much data, etc.

Who we tested?

For this phase we tested the bot with over 100+ queries and coded errors/query responses for X queries.

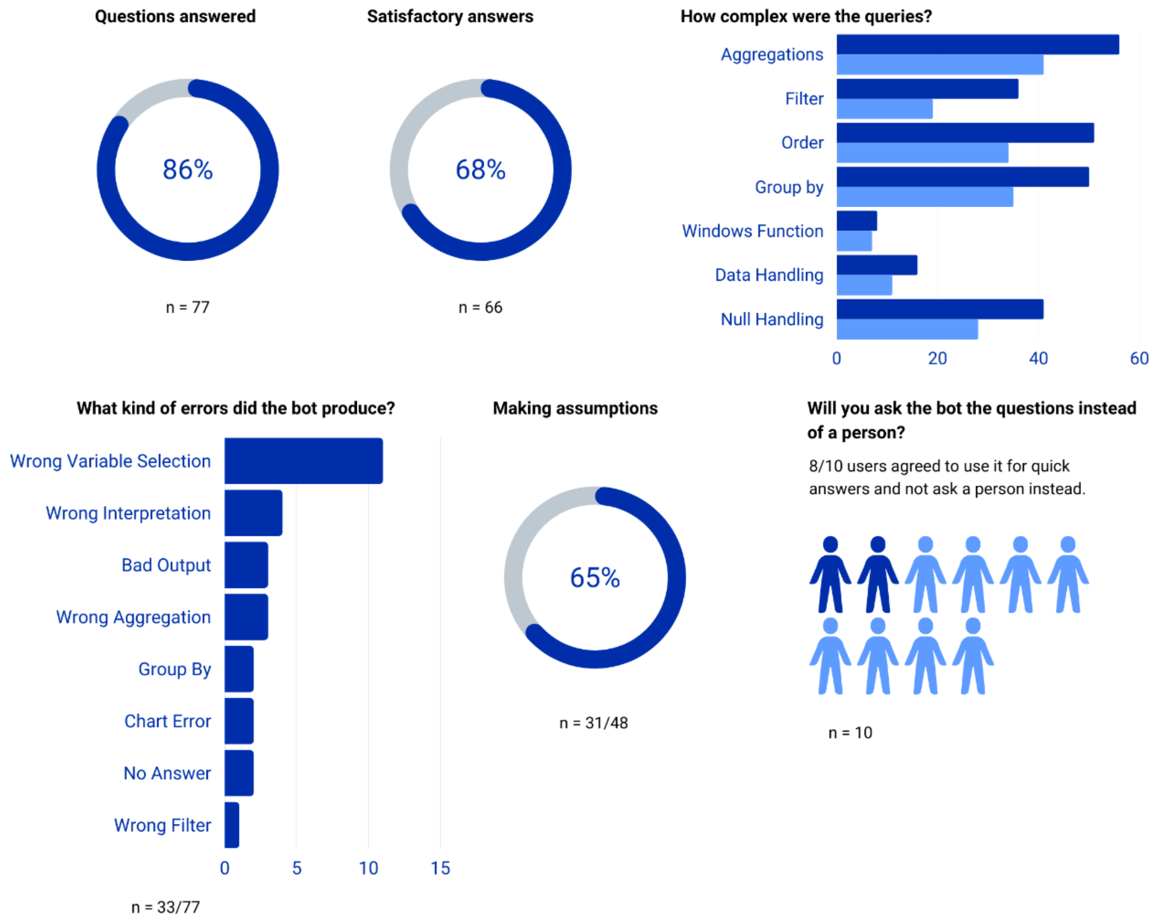
To gather further qualitative responses, we interviewed 5 Data Personas (Engineers, Scientists, Analysts), and 5 Business Personas (Managers, Partnerships, Consultants) to see how everyday business users are pursuing the responses.

Error Log Analysis

We looked at 77 queries that we asked the bot after phase 2 improvements to see how the bot performance changes, and we didn't notice a significant change in the performance. Here are some sample queries from phase 2:

Please give me a line graph of the client revenue sizing compared to the client employee sizing
What are my biggest revenue streams?
Tell me about year to year growth of opportunities?
How much business was done last year in all cities?
In which technology do we have the highest potential for revenue increase

As we look at the data below, we can see that we eliminated "Categorical Parsing" as an error type, as well as reduced "Wrong Aggregation" as a major error type. However, semantics remain the biggest challenge for this bot as "variable selection" and "wrong interpretation" can be eliminated with better semantics.



Further, in terms of the complexity of the queries, “filtering” remains a problem, and the bot performs quite well in writing windows functions when needed. The bot also does a great job of handling nulls and data types (note that not all unsatisfactory queries are because of the complexity type). Interestingly, the bot handles spelling errors quite well.

And, finally, what’s heartening is that 8/10 people will prefer using a bot like this to answer simple business questions than asking a person (as depicted by the last isotopic chart).

Learnings looking at queries:

- The better described the query, higher the chances of getting a good answer. If the user gets very specific about the variable name, the better response the user gets.
- Usually errors are towards recognizing the wrong variables, so we strongly think that a semantics layer would help a lot.
- Advanced functions like windows functions, casting variables, handling nulls, or generating chart/visualization code is not a worry.

- Our system wasn't the most resilient system, so some of the lack of trust also comes with a higher character (upwards of 3500) not coming to the surface. Sometimes, our system parses the message wrong and doesn't show the "plain english" explanation.
- Time to respond is slower than what a few users would expect, that's partially because of the compute resources we dedicated to it. A few users were happy with the time it took to respond, because it was faster than a person.

Qualitative Analysis of User Interviews:

- Semantic Analysis: Most participants wanted to ask predictive or semantic questions, like *"Which city should I target next"* or "*What is the projected growth for xyz."*
- Supplementary analysis: 4 participants agreed that they would use this bot to complement their work, either by doing the first iteration of analysis and then reaching out to a data analyst to verify or dig deeper. *"Helps me to get better at my job, iterate with it. Assess if the response is apt or not."*
- Simple vs Complex: Most participants were confident in asking simple queries and wanted to use the bot. They were, however, skeptical in making their questions complex and multi-dimensional.

Qualitative Insights

1. How much do users trust the bot?

Average Trust at Beginning: **5.5** /10
Average Trust at the End: **5.2**/10

Here's each data point from 10 user interviews:

Trust Beginning	Trust End
7	5
3	3
0	0
5	5.5
6	3
8	7
5	7
8	7
5	6.5
8	8

As we can note, that trust increased in 3, decreased in 4, and in 3 remained the same. So, we cannot tell how much the users trust the current bot at the end of the interviews.

Why the change?
In the individuals whose trust decreased, it seemed to be due to receiving inaccurate answers and because of the “tech debt" that they noticed. Conversely, trust among individuals increased if the bot was able to interpret their query well.

Conclusion and Further Work

In conclusion, the creation of DashQueries is aimed at addressing the common pain point that many business data users as well as data engineers/analysts face when trying to extract/answer ad-hoc data queries. Through our research on Natural Language Queries (NLQ), we have found that NLQ has significant potential in addressing this pain point. However, the current offerings from Tableau and ThoughtSpot have limitations in terms of their NLP capabilities, as they require users to have prior knowledge of the data and their schemas to interpret queries accurately and they are unable to handle complex or nuanced queries.

Large Language Models (LLMs) like GPT-3 do have potential to convert natural language into code, and the results have been promising in terms of improving the accuracy of understanding user requests and coming up with more accurate SQL queries. However, poor written SQL queries are still possible if the user’s intent or the data’s schema is not very clear.

As for further work, we believe that there is significant potential in expanding DashQueries' capabilities to handle more complex queries and increase the accuracy of the system in interpreting natural language, as well as improving reliability by using more production-ready systems. Additionally, there is a huge potential in researching “semantics” of data that works very well with LLMs to generate the right code. Finally, we believe that there is significant potential for DashQueries to become an essential tool for non-technical users to extract ad-hoc data queries and for data teams to work on more strategic projects if we solve the semantics problem.

Circling back to our question: Can LLMs replace the Business Analyst?

We believe LLMs can contribute greatly to being a great supplement to a Business Analyst or any Business User. It still needs to advance more to be able to replace them.

References

Liu, Aiwei, and Xuming Hu. “[2303.13547] A comprehensive evaluation of ChatGPT's zero-shot Text-to-SQL capability.” *arXiv*, 12 March 2023, https://arxiv.org/abs/2303.13547. Accessed 2 May 2023.

Maddigan, Paula, and Teo Susnjak. “[2303.14292] Chat2VIS: Fine-Tuning Data Visualisations using Multilingual Natural Language Text and Pre-Trained Large Language Models.” *arXiv*, 24 March 2023, https://arxiv.org/abs/2303.14292. Accessed 2 May 2023.

Wang, Xingbo, and Furui Cheng. “[2201.04868] Interactive Data Analysis with Next-step Natural Language Query Recommendation.” *arXiv*, 13 January 2022, https://arxiv.org/abs/2201.04868. Accessed 2 May 2023.

Appendix - User Test Script

1. Welcome

Hi __.
Thanks for doing this interview. I am __, we are doing a capstone project to understand how sales people get their data questions answered. So, we would like to ask you some open-ended questions and then show you an interface and get your thoughts.

There are no benefits to you for participating, other than what may be an educational experience. We hope that the research will benefit society by providing more information. This research poses no risks to you other than those normally encountered in daily life.

We will not name you if and when we discuss your behavior in research publications. After the research is completed, we may save the notes for future use by ourselves or others. However, this same confidentiality guarantees given here will apply to future storage and use of the materials.

Your participation in this research is voluntary, and you are free to refuse to participate or quit the experiment at any time.

We are testing the prototype and not you. So feel free to speak your mind, there are no right or wrong answers here. So, don't hesitate to answer.

Can we record this interview so that we can analyze and revisit what you did as a team later?

2. Context Questions

Alright, so before we start, we would like to understand you better as a person. This helps us set context and run user interviews better.

- What's your day to day job look like?

- How do you interact with data? What kind of questions do you ask?

3. Introduce the prototype

We have created a slackbot that can help you ask a question related to your business data. For the purpose of this interview, we are using sales data that's mostly clean.

(Pause for them to look through the data.)

Let's say that you have this data that you work with or deal with, what kind of business questions would you ask.

- Can you ask a question based on this ? And, let's see if our bot can actually return a response.
- Do you trust the response? How would you rate from 0-10? 10 being the highest.

[Initial trust number]
- Ask another 4 data questions

- Now, how much do you trust this bot out of 10?

[Final trust number]

So, let's talk more about that experience..

- If you had a bot like this, would you ask this bot a business question? Or, would you rather ask a person?
- What kind of questions do you expect the bot to answer well?
  - 4-5 examples
- How else do you plan to interact with such a bot?
- What are your biggest concerns / worries if you had to rely on such a bot?
- Do you think people around you would actually use something? IF not, why?

Debrief
- What did you like? Not like?
- What are some things that can be improved? Why?
- How would you overall rate the experience over dashboards/reports today?
  - **Easier**
  - **Not so easier**
  - **Will always prefer a dashboard**