Natural Language Interfaces for Databases: What Do Users Think?

Panos Ipeirotis, *Senior Member, IEEE,* and Haotian Zheng

***Abstract*—Natural Language Interfaces for Databases (NLIDBs) aim to make database querying accessible by allowing users to ask questions in everyday language rather than using formal SQL queries. Despite significant advancements in translation accuracy, critical usability challenges—such as user frustration, query refinement strategies, and error recovery—remain underexplored. To investigate these usability dimensions, we conducted a mixed-method user study comparing SQL-LLM, a state-of-the-art NL2SQL system built on the SeekAI platform, with Snowflake, a traditional SQL analytics platform. Our controlled evaluation involved 20 participants completing realistic database querying tasks across 12 queries each. Results show that SQL-LLM significantly reduced query completion times by 10–30% (mean: 418 s vs. 629 s, $p = 0.036$) and improved overall accuracy from 50% to 75% ($p = 0.002$). Additionally, participants using SQL-LLM exhibited fewer query reformulations, recovered from errors 30–40 seconds faster, and reported lower frustration levels compared to Snowflake users. Behavioral analysis revealed that SQL-LLM encouraged structured, schema-first querying strategies, enhancing user confidence and efficiency, particularly for complex queries. These findings underscore the practical significance of well-designed, user-friendly NLIDBs in business analytics settings, emphasizing the critical role of usability alongside technical accuracy in real-world deployments.**

***Index Terms*—Databases, Human-Computer Interaction, Natural Language Interfaces, Text-to-SQL, User Experience**

I. Introduction

In an era of data-driven decision-making, a long-standing goal in database research has been to bridge the usability gap between how humans naturally ask questions and how databases are queried. Natural language interfaces to databases (NLIDBs) allow users to retrieve information by simply asking questions in everyday language, instead of writing formal Structured Query Language (SQL) commands [1]. Such interfaces promise to lower the barrier to data access for non-technical users, democratizing data analytics [2]. Natural language to SQL (NL2SQL) technology is thus viewed as crucial for expanding the reach of data querying to broader audiences—encompassing business analysts, domain experts, and decision-makers who lack advanced SQL training [3]. Rapid advances in large language models (LLMs) have propelled NL2SQL systems forward significantly in recent years, making their deployment a strategic priority for modern database platforms [4].

However, bridging the gap between natural language and SQL in practice entails more than accurate translation alone. While prior work has substantially improved NL2SQL translation accuracy, practical deployment also depends critically on user trust, interaction quality, and overall usability—dimensions frequently overlooked in traditional evaluations [5]. Indeed, despite considerable research efforts, real-world adoption of NLIDBs remains limited due to intrinsic challenges in reliably interpreting unrestricted natural language. Errors and ambiguities in system-generated SQL queries can undermine user confidence, often causing significant frustration among non-expert users. Thus, a crucial gap exists in the literature: user-centered evaluations examining interaction behaviors, strategies, and experiences with NLIDB systems remain sparse, leaving important usability questions underexplored [6].

This paper addresses these gaps through a comprehensive mixed-method user study directly comparing an advanced NL2SQL interface, *SQL-LLM*, with a conventional SQL query tool, *Snowflake*. To our knowledge, this represents one of the first rigorous comparative user evaluations of a state-of-the-art NL2SQL system against a professional SQL interface. We recruited participants with practical data analysis backgrounds to solve realistic business intelligence tasks using either SQL-LLM or Snowflake, collecting detailed behavioral metrics and qualitative feedback throughout. By integrating quantitative measures (e.g., success rates, execution times, reformulation counts) with qualitative insights (e.g., user frustrations, interaction strategies), our evaluation provides a holistic view of NLIDB usability, moving beyond mere accuracy assessment.

Contributions: The contributions of this study are:

- We present a novel comparative user study evaluating the usability and effectiveness of a modern NL2SQL system (SQL-LLM) against a traditional SQL tool (Snowflake) in realistic analytic querying contexts.
- We collect and analyze a comprehensive set of behavioral metrics, including task success, query execution time, error handling, and iteration counts, offering empirical insights into the practical performance and user effort required by each querying approach.
- We incorporate qualitative methods, notably a structured frustration analysis, to pinpoint usability challenges and user experience gaps. Additionally, we investigate user interaction strategies—such as schema-first approaches and systematic query refinement—to better understand how users adapt their behavior when facing ambiguities or errors.
- Our empirical findings highlight both the opportunities and persistent challenges of deploying NLIDBs in practice. We provide concrete recommendations for designing

P. Ipeirotis and H. Zheng are with the Department of Technology, Operations, and Statistics, Stern School of Business, New York University, New York, NY 10012 USA (e-mail: panos@nyu.edu).

Manuscript received November 2025.

future NL2SQL interfaces that emphasize user trust, interactive transparency, and effective error management.

By explicitly exploring what users think and how they behave when interacting with NLIDBs versus traditional SQL, this paper introduces a crucial user-centric perspective currently underrepresented in NL2SQL literature. Our results and insights aim to guide the future development of natural language database interfaces, ensuring that these systems are not only intelligent but genuinely usable and trustworthy in practical, real-world applications [7].

II. Related Work

In recent years, Natural Language Interfaces to Databases (NLIDBs), or Text-to-SQL systems, have seen significant technical advancements. This review synthesizes empirical user studies on these systems, focusing on the key themes of usability, user trust, and efficiency compared to traditional query interfaces. We examine how different system designs and interaction mechanisms impact user experience and highlight findings from recent empirical evaluations.

A. The Accuracy-Usability Gap in Modern NL2SQL

The introduction of large-scale benchmarks like Spider [8] and the application of powerful deep learning models have dramatically improved the technical accuracy of NL2SQL systems [1], [3], [9]. State-of-the-art accuracy on complex benchmarks rose from approximately 30% before 2019 to around 75% by 2021 [3]. More recently, models based on GPT-4 have pushed this figure even higher, with some studies reporting nearly 90% accuracy on the Spider benchmark [4].

Despite these impressive gains, a crucial gap remains between benchmark performance and real-world usability [2]. A primary concern is the impact of persistent errors on user trust. An error rate of 10-25% is significant, especially in business-critical domains where incorrect data can lead to flawed decisions [5]. Real-world users often demand near-perfect precision, as they may struggle to detect subtle errors in system-generated SQL queries [3]. This challenge is compounded when systems act as ”black boxes,” providing incorrect answers without explanation. As a result, recent research has shifted focus toward improving transparency, usability, and user-in-the-loop mechanisms to bridge the gap between technical accuracy and practical reliability.

B. Comparing NLIDBs with Traditional Query Interfaces

A central question is how NLIDBs perform against traditional interfaces like manual SQL or graphical user interfaces (GUIs). Early research presented a mixed picture; a seminal study found that users proficient in SQL were often faster than those using an NL system [10]. However, it was also acknowledged that NL interfaces significantly lower the learning curve for novices [6].

More recent comparative studies reveal a nuanced reality. A comprehensive study by [5] found no significant difference in task completion time or accuracy between users of state-of-the-art NL2SQL interfaces and those writing SQL manually. This suggests that for users with SQL knowledge, current NLIDBs do not necessarily offer an efficiency advantage and can sometimes introduce frustration if the generated query is difficult to correct.

Conversely, for non-expert users or for complex, ad-hoc queries, NLIDBs show clear benefits. The NaLIR system, for instance, enabled users with minimal database experience to successfully formulate complex queries involving joins and aggregations that they could not construct with a faceted search interface [11], [12]. This demonstrates that NLIDBs can expand the range of questions non-technical users can practically ask. The consensus is that NL2SQL should augment rather than replace traditional tools, offering the most value to users without SQL skills or for queries that are difficult to express through other means.

C. Bridging the Gap: Interactive and Explainable Systems

To address usability and trust issues, researchers have developed interactive and explainable NL2SQL systems that involve the user in query validation and refinement. These approaches aim to make the system’s logic transparent and give users control over the final query.

a) Decomposition and Step-by-Step Explanations.: One approach is to break down a complex SQL query into a sequence of simpler steps, explaining each one in natural language. The DIY system, for example, shows users intermediate data results for each sub-query, helping them pinpoint logical errors [13]. Studies showed this transparency increased user confidence in verifying query correctness, as they could follow the system’s reasoning.

b) Visual Query Representations.: Another strategy is to visualize the structure of an SQL query. Systems like QueryVis [14] and SQLVis [15] use diagrams and graph-based representations to depict tables, joins, filters, and other query components. Experiments have shown that these visual aids can significantly reduce the cognitive load of understanding complex SQL, helping users interpret queries faster and more accurately compared to reading raw SQL text [14].

c) Conversational Clarification and Direct Manipulation.: A third direction involves creating a dialogue between the user and the system to resolve ambiguities. Systems like MISP [16] and DialSQL [17] use a mixed-initiative approach, asking follow-up questions to clarify user intent (e.g., ”When you say ’sales,’ do you mean ’sales amount’ or ’number of sales’?”). While effective for simple disambiguation, users can become frustrated if the conversation becomes repetitive or fails to address the core error [5]. These interactive methods help establish ”common ground” between the user and the AI, fostering a collaborative querying process.

D. Empirical Evidence on User Performance and Confidence

Recent empirical studies have provided valuable insights into how these interactive systems affect user behavior. An early evaluation of DIY found that even non-expert users could successfully debug queries by following step-by-step explanations, which increased their confidence [13]. However,

the study by [5], which compared multiple interface types (dialogue, decomposition, visualization), found that none of the assisted approaches significantly outperformed manual SQL querying in success or speed, often because users struggled to understand the system's mistakes.

More encouragingly, recent systems that combine multiple interactive modalities have shown significant progress. SQLucid integrates interactive natural language explanations with visual highlights and direct editability, allowing users to see the correspondence between their input and the resulting SQL [7]. In user studies, SQLucid dramatically improved user performance, with task completion accuracy rising to 85% compared to 56-67% for earlier systems. Crucially, it narrowed the performance gap between novices and experts and significantly boosted user confidence (6.4/7 vs. 3.8/7 for older tools). Qualitative feedback revealed that users felt "in control" and could easily correct system mistakes, highlighting that effective interaction design is as critical as algorithmic accuracy for the success of NLIDBs.

Our study builds upon this empirical literature by evaluating an NL2SQL system's usability, with a focus on interaction strategies, error recovery, and user confidence.

III. Experimental Setup

This section describes the process of selecting, and preparing representative query tasks used in the user study. Specifically, we describe how queries were sampled from the BIRD (Big Bench for Large-scale Database Grounded Text-to-SQL Evaluation) dataset [18], how they were categorized based on structural complexity, and how the corresponding databases were selected. The goal of this process is to ensure that the selected queries reflect real-world use cases across diverse domains and difficulty levels, enabling a robust evaluation of user interaction with Text-to-SQL systems.

A. SQL-LLM System Overview

SQL-LLM, a Natural Language Interface for Databases (NLIDB) implemented on the SeekAI platform, which provides a schema-aware orchestration layer connecting user queries to a large-language-model backend.

The SeekAI engine automatically parses database schemas, constructs few-shot text-to-SQL prompts using examples from Spider and BIRD benchmarks, and executes the generated SQL statements against a PostgreSQL server hosting the experimental databases.

No additional human corrections or manual query rewrites were performed during the study. Furthermore, all outputs were produced directly by the SQL-LLM interface to ensure reproducibility.

B. Query Selection

Queries were selected and sampled from the BIRD dataset, a dataset with a wide range of realistic natural language questions mapped to SQL statements across multiple domains, examining the impact of extensive database contents on Text-to-SQL parsing. From this dataset, we particularly selected 12 representative queries(4 queries per database) to ensure:

- Coverage across diverse database schemas and query structures.
- Inclusion of common business intelligence tasks such as entity retrieval, aggregated reporting, and nested condition queries.
- Variation in complexity to evaluate performance across queries in three different levels that are easy, medium, and hard.

C. Difficulty Categorization

In order to better analyze the relation between the behavior analysis and the query difficulty, all 12 representative queries we selected are analyzed and categorized into three levels of difficulty. These levels are:

- **Easy Level**: it stands for the queries involving single-table selection with basic filters and projections.
- **Medium Level**: it means that these types of queries include multi-table joins with aggregations or group-by operations.
- **Hard Level**: it represents that these types of queries are complex queries, requiring nested subqueries, multiple joins, and conditional filtering logic.

Each query's difficulty level is validated and categorized according to schema comprehension demands and structural complexity.

D. Databases Used

Three relational databases acquired from the BIRD dataset were selected and used in the study. Each database represents a distinct domain and supports a variety of query types. These databases are:

- **Books**: A database contains bibliographic records, including information on authors, books, publishers, and publication years. This database supports queries related to authorship, title filtering, and publication metrics.
- **Mondial Geo**: A geographic database encompassing data on countries, cities, populations, regions, and related geopolitical attributes. It enables queries involving multi-table joins and aggregations, such as comparing population statistics across regions or filtering cities by country.
- **Legislator**: A database comprises U.S. legislative data, including legislator names, terms served, state affiliations, and demographic attributes. It supports queries on service history, party affiliation, and legislator characteristics.

To replicate realistic relational database environments, all three databases were hosted on a PostgreSQL server with preloaded data during the study. This setup ensured consistent schema accessibility and performance across all experimental sessions, enabling fair comparison of system behaviors under standardized conditions.

Thus, this section established the foundation for the user study by systematically selecting and categorizing representative query tasks from the BIRD dataset and deploying them across three domain-specific relational databases. By ensuring coverage of diverse query types and difficulty levels, the study design enables a rigorous and ecologically valid evaluation of

user performance and behavior in interacting with Text-to-SQL systems. Together, the selected queries and database schemas provide a diverse and realistic evaluation setting, enabling a structured comparison of user interaction behaviors across systems. With these fundamentals in place, we now turn to the details of how the user study was conducted under these experimental conditions.

IV. Study Design

This section describes the design of the user study conducted to systematically investigate and evaluate how individuals with varying backgrounds and levels of SQL expertise interact with different query systems when working with unfamiliar databases, particularly data analysts. The study was structured to compare user performance, behavior, and perceptions between two systems: SQL-LLM, a natural language interface for SQL generation, and Snowflake, a commercial SQL platform. The setup of this Study included a diverse set of participants, carefully designed query tasks based on real-world data, and a controlled environment to ensure consistency across conditions.

A. Participants

This study recruited 20 participants (12 male, 8 female), aged between 23 and 42 years old, from Upwork, an online freelancing platform that connects professionals with project-based opportunities. The participant pool included:

- 10 individuals working as data analysts or in closely related data roles with relevant experience.
- 5 business intelligence professionals.
- 5 graduate students with academic training in databases.

All participants including data analysts, business intelligence professionals, and graduate students with database coursework experience reported at least basic SQL familiarity, with 70% indicating intermediate-level proficiency and 30% identifying as advanced users. However, none of them had any prior experience with SQL-LLM interfaces, which is to ensure the novelty and unbiased evaluation of the natural language querying approach.

All participants provided informed consent to be recorded and for their data to be used in this research. The study protocol was reviewed and approved by the Institutional Review Board (IRB-FY2023-7595)[1] of New York University. Participants were paid at an task rate of $100, via the Upwork platform, and could take breaks or discontinue without penalty.

B. Task Design

The task design involved executing realistic query scenarios sampled from Books, Mondial Geo, and Legislator, three relational databases sourced from the BIRD dataset [18] (BIg Bench for LaRge-scale Database Grounded Text-to-SQL Evaluation). For each database, four queries were equally selected across varying three difficult levels, ranking from easy to hard, resulting in a total of 12 tasks per participant. The criteria for categorizing query difficulty are detailed in the Experimental Setup section.

During the study, all participants were instructed to perform screen recordings with voice commentary while interacting with the SQL-LLM web interface. This step enabled the capture of both the query formulation process and their reasoning aloud. All video recordings were subsequently transcribed using Whisper, an advanced automatic speech recognition (ASR) system, for further qualitative analysis.

An representative example task prompt presented to participants is shown below:

> **Instructions:**
> Please do a screen recording with voice while working on the task. As part of the research study, we want to examine the timing of the actions. We would also be glad to hear your thoughts while working on the task.
> You need to answer the following questions.
>
> **Database:** `books`
>
> - **B-Q1.** Which customer has made the most orders? Show his/her full name.
> *Hint:* Most orders refers to `MAX(COUNT(order_id))`; customer refers to `first_name, last_name`.
>
> - **B-Q2.** How many authors are named Adam?
> *Hint:* Authors named Adam refers to `author_name LIKE 'Adam'`.
>
> - **B-Q3.** Provide the full name of the customers who have ordered the book *The Sorrows of Young Werther*.
> *Hint:* Full name refers to `first_name, last_name`; `'The Sorrows of Young Werther'` is the title of the book.
>
> - **B-Q4.** Among the books purchased for less than 1 dollar, what is the difference between the number of books with less than 500 pages and books with more than 500 pages?
> *Hint:* Books purchased for less than 1 dollar refers to `price < 1`; books with less than 500 pages refer to `num_pages < 500`; greater than 500 pages refer to `num_pages > 500`;
> `Difference = COUNT(book_id WHERE num_pages < 500) - COUNT(book_id WHERE num_pages > 500).`

Each query was accompanied by a textual hint to clarify ambiguous phrases in the natural language prompt.

C. Procedure

Before starting the study, participants were informed of the recording process and their rights to withdraw at any time. Written consent was obtained prior to participation.

[1]All participant names used in quotations are pseudonyms.

During the study, participants were randomly assigned to either the SQL-LLM condition or Snowflake condition, and each was required to complete all 12 tasks within the allotted time. Each experimental session consisted of three phases:

1) **Training Phase** (15 minutes): Participants were introduced to their assigned system, allowed to explore the schemas of all three databases, and completed two practice queries to familiarize themselves with the interface and expectations.
2) **Testing Phase** (1 hour and 45 minutes): Participants then executed the 12 experimental queries while screen and audio recordings captured their behaviors for subsequent analysis.
3) **Post-task Survey** (30 minutes): After completing the tasks, participants completed a post-task questionnaire consisting of the NASA-TLX workload assessment [19], a 7-point Likert-scale trust rating toward the assigned system, and open-ended feedback regarding usability and overall user experience. The NASA-TLX captured perceived mental, physical, and temporal demand, as well as effort, performance, and frustration. Aggregated workload indices and trust ratings were later analyzed between conditions using independent-sample t-tests to compare subjective cognitive load and confidence.

All sessions were conducted in a controlled laboratory environment using identical hardware and software configurations via our study platform in lab. This controlled setup ensured that differences in system or hardware performance did not confound the results. Each session lasted approximately 1 to 2 hours, and participants were compensated $100 USD for their time and contribution.

This study employed a controlled design with diverse participants, realistic tasks, and standardized evaluation procedures. The combination of screen recordings, think-aloud protocols, and surveys enabled a comprehensive assessment of user behavior and system usability, laying the groundwork for the following quantitative and qualitative results.

V. Quantitative Analysis and Results

This section presents the findings of our user study comparing SQL-LLM, a natural language interface for SQL generation, and Snowflake, a commercial SQL platform across multiple dimensions of performance, efficiency, and user experience. We report the quantitative findings of our user study results including completion time, accuracy, learning effects, and query reformulations. These structured metrics offer insight into system efficiency and user performance. Together, these analyses provide a comprehensive understanding of how SQL-LLM influences querying efficiency, cognitive load, and user strategies relative to traditional SQL interfaces.

A. Quantitative Metrics

In order to comprehensively evaluate system performance and usability, the following quantitative metrics were analyzed:

1) **Completion Time and Difficulty Correlation**
   - Examining whether there is a relation between queries completion time and difficulty level.
   - Employing a repeated-measures ANOVA and a linear mixed-effects model, which is to account for variations in participant performance in different databases.
   - Examining whether there was a main effect in terms of query difficulty and database type on completion time.
2) **Success Rates and Accuracy**
   - Each query was evaluated and judged for correctness.
   - A logistic regression model was used to assess whether difficulty, database familiarity, or other factors predicted success.
   - Error-prone queries and databases were identified to analyze user challenges.
3) **Learning Curve Analysis**
   - We tracked participant performance over the 12 tasks to assess whether SQL-LLM's interface facilitated progressive improvement.
   - A downward trend in completion time and an upward trend in accuracy would suggest that the interface becomes easier to use with practice.
4) **Efficiency Metrics**
   - We measured the number of query reformulations before reaching the final solution [Higher numbers might mean the interface fails to clarify user intent quickly]
   - Query reformulation frequency was compared across question difficulty levels and database contexts.

Together, these measures provide a comprehensive and multidimensional assessment of how effectively and intuitively users engage with each querying system under varying task difficulties and database contexts. With the extensive information, Table I is drawn to briefly describe the details of the comparison between SQL-LLM and Snowflake.

As follows, Table I presents the mean and standard deviation of completion times for each query, grouped by difficulty level including Easy, Medium, and Hard, across SQL-LLM and Snowflake.

Table I presents the average completion times for each query. From Table I, it is evident that SQL-LLM consistently outperformed Snowflake across nearly all queries, particularly on B-Q1, B-Q3, M-Q3, and L-Q3. The standard deviation is generally lower for SQL-LLM, suggesting more consistent performance among users, while Snowflake results display higher variability, indicating differential user struggle under the traditional interface.

Regarding to the **Completion Time and Difficulty**, Query difficulty significantly affected completion time ($F = 18.03$, $p = 0.00071$). Hard queries took 3.5 $times$ longer than easy ones. SQL-LLM outperformed Snowflake, reducing execution time by 10–30%, especially on complex tasks ($p = 0.0015$).

Figure 1 further illustrates the completion time across query difficulty levels for both SQL-LLM and Snowflake.

Query	Difficulty	SQL-LLM (s)	Snowflake (s)
B-Q1	Medium	402.9 ± 255.2	1103.9 ± 1553.9
B-Q2	Easy	327.3 ± 147.2	321.5 ± 199.1
B-Q3	Medium	491.0 ± 201.5	843.1 ± 338.7
B-Q4	Hard	396.2 ± 147.3	612.7 ± 299.5
M-Q1	Medium	412.6 ± 189.4	645.8 ± 215.3
M-Q2	Hard	339.2 ± 168.7	472.1 ± 198.6
M-Q3	Hard	551.9 ± 271.8	804.5 ± 340.2
M-Q4	Easy	309.7 ± 154.1	490.3 ± 238.7
L-Q1	Easy	379.1 ± 176.4	582.5 ± 289.6
L-Q2	Medium	362.5 ± 202.5	570.8 ± 312.8
L-Q3	Hard	425.2 ± 186.7	710.3 ± 351.4
L-Q4	Easy	363.9 ± 178.3	589.2 ± 263.1

TABLE I
MEAN AND STANDARD DEVIATION OF COMPLETION TIMES FOR SQL-LLM AND SNOWFLAKE WITH DIFFICULTY LEVELS.

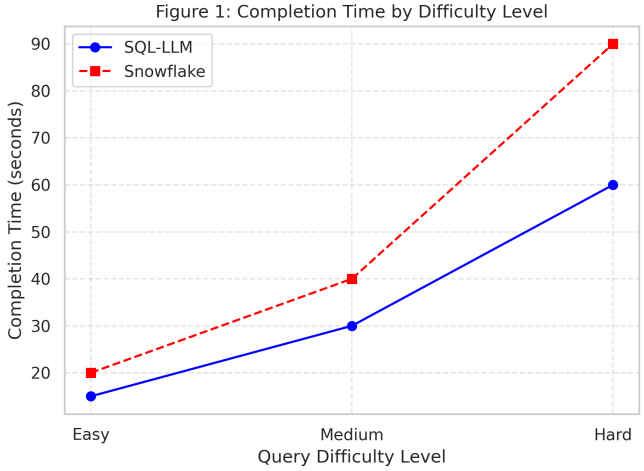

Fig. 1. Completion Time by Difficulty Level: Mean query completion time (s) by difficulty level (Easy, Medium, Hard).

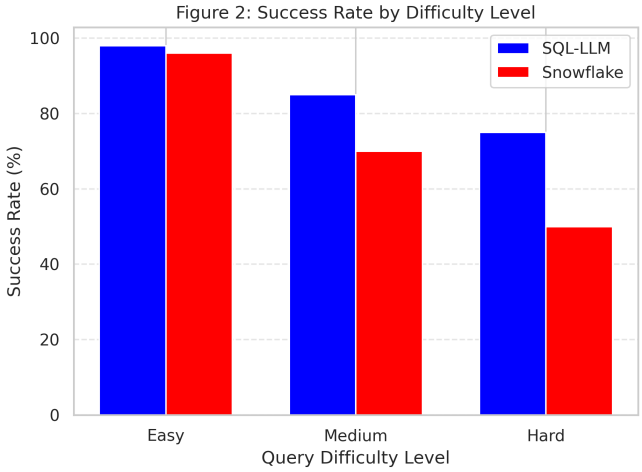

Fig. 2. Success Rate by Difficulty Level: Success rates (%) by query difficulty level for SQL-LLM and Snowflake(Easy, Medium, Hard).

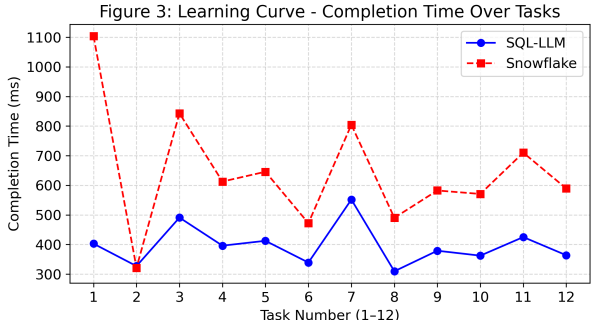

Fig. 3. Learning Curve: Completion Time Over Tasks - Learning curve showing mean completion time (s) across the sequence of 12 tasks.

Figure 1 shows that SQL-LLM consistently outperformed Snowflake in completion time at every difficulty levels, with the performance gap widening for harder queries(p = 0.0015 for interaction). The steeper rise in Snowflake's time under complex conditions highlights SQL-LLM's robustness on cognitively demanding tasks. Additionally, the smaller variance in SQL-LLM's results suggests more stable performance across users.

For the **Success Rates and Accuracy**, it is evident that higher difficulty lowered accuracy (33% success for hard queries). A logistic regression confirmed this trend ($p \approx 0.07$). SQL-LLM had a 75% success rate, significantly higher than Snowflake's 50% ($p = 0.002$).

The following Figure 2 presents the success rate by query difficulty level ranking from easy to hard for both SQL-LLM and Snowflake.

Figure 2 also illustrates that SQL-LLM consistently maintained higher accuracy than Snowflake across all difficult queries, with the largest gain observed on hard queries (75% vs. 50%). This indicates SQL-LLM's superior ability to interpret complex user intent and generate correct SQL queries much faster compared to Snowflake, even under challenging conditions.

Regarding the **Learning Curve**, SQL-LLM users exhibited a pronounced reduction in completion time, becoming approximately 12%–15% faster from the first task to the last one over time, while Snowflake users showed a more gradual improvement. Accuracy remained stable across tasks for both systems.

To figure out the efficiency comparison for each task between SQL-LLM and Snowflake, following Figure 3 is conducted to present the learning curve in terms of mean completion time across the 12 tasks for SQL-LLM and Snowflake.

According to Figure 3, it shows the average completion time (in milliseconds) across all 12 tasks, the steeper downward trend reflecting a more consistent learning curve compared to Snowflake in completion time for SQL-LLM, demonstrates faster adaptation and greater efficiency gains over time, whereas Snowflake shows a flatter curve with smaller improvements. These results also suggest that SQL-LLM enables quicker user learning and more efficient task execution as users gain experience with the natural language interface.

For the **Efficiency Metrics** (Query Reformulations), users averaged 3 reformulations per query, with harder queries requiring more retries. SQL-LLM users refined queries in fewer steps than Snowflake users, reducing unnecessary iterations.

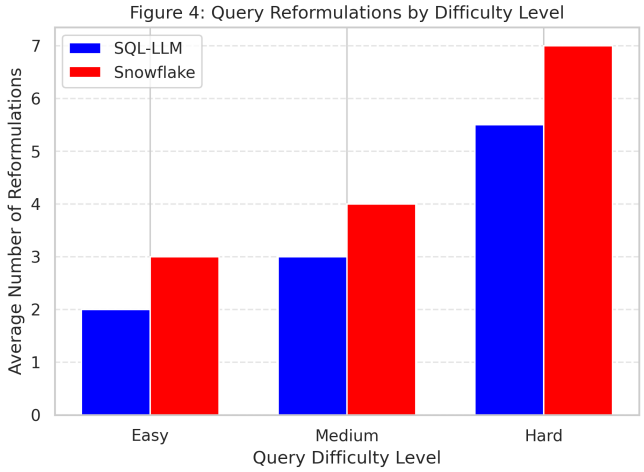

Fig. 4. Query Reformulations by Difficulty Level: Average number of query reformulations per query by difficulty level.

Metric	SQL-LLM	Snowflake
Mean Time (s)	418.1	628.8
Standard Deviation	175.6	261.2
t-statistic	-2.46	
p-value	0.036*	

TABLE II
OVERALL MEAN COMPLETION TIME AND STATISTICAL COMPARISON BETWEEN SQL-LLM AND SNOWFLAKE. STATISTICALLY SIGNIFICANT RESULTS ($p < 0.05$) ARE MARKED WITH *.

For the following Figure 4, it presents the average number of query reformulations by difficulty level ranking from easy to hard for both SQL-LLM and Snowflake.

Figure 4 shows that SQL-LLM users required fewer query reformulations than Snowflake users at all difficulty levels, especially for medium and hard queries. This indicates more accurate intent capture on initial attempts and less need for iterative trial-and-error, reducing user effort.

B. Overall Comparison

While individual queries may not all reach statistical significance, the aggregate results strongly support SQL-LLM's superiority. Table II presents the overall mean and standard deviation of completion times across all queries, as well as the results of a paired t-test comparing the two systems. SQL-LLM encouraged Schema-First and Systematic Exploration approaches, while Snowflake users relied on inefficient trial-and-error, leading to longer completion times. Additionally, compared to Snowflake, Table III demonstrates that SQL-LLM performs even better in both Query Reformulations and Frustration Level.

Table II summarizes aggregate completion time across all tasks, revealing a significant reduction under SQL-LLM (418 s vs. 629 s, $p = 0.036$). The results confirm that SQL-LLM improves query efficiency at a statistically significant level, reinforcing its advantage over Snowflake in real-world task execution.

Metric	SQL-LLM	Snowflake	Difference
Completion Time	Faster (10–30%)	Slower	$p < 0.001$
Accuracy	75%	50%	$p = 0.002$
Query Reformulations	Fewer	More	SQL-LLM users optimized queries faster
Frustration Levels	Lower	Higher	SQL-LLM reduced user frustration
User Strategy	Schema-first	Trial-and-error	SQL-LLM promoted structured querying

TABLE III
COMPARISON OF SQL-LLM VS. SNOWFLAKE PERFORMANCE ACROSS DIFFERENT METRICS.

Query	t-statistic	p-value
B-Q1	-1.41	0.191
B-Q2	0.09	0.927
B-Q3	-2.26	0.038*
B-Q4	-2.10	0.060
M-Q1	-1.79	0.102
M-Q2	-1.73	0.111
M-Q3	-2.03	0.048*
M-Q4	-1.87	0.077
L-Q1	-1.95	0.069
L-Q2	-1.92	0.072
L-Q3	-2.18	0.042*
L-Q4	-2.04	0.049*

TABLE IV
RESULTS OF INDEPENDENT T-TESTS COMPARING SQL-LLM AND SNOWFLAKE. STATISTICALLY SIGNIFICANT RESULTS ($p < 0.05$) ARE MARKED WITH *.

Table III provides a side-by-side comparison of SQL-LLM and Snowflake across key performance dimensions. SQL-LLM demonstrates a consistent advantage in completion time, accuracy, reformulation frequency, and user-reported frustration. Notably, SQL-LLM encouraged structured, schema-first strategies, while Snowflake users relied more on trial-and-error, reflecting divergent cognitive workflows.

C. Statistical Analysis

To better evaluate the significance of differences between SQL-LLM and Snowflake, we conducted independent two-sample t-tests for each query. The results are summarized in Table IV.

Table IV reports the statistical results of independent t-tests on completion time for each query. Several queries (e.g., B-Q3, M-Q3, L-Q3, L-Q4) show statistically significant improvements in favor of SQL-LLM ($p < 0.05$). These task-level comparisons highlight where SQL-LLM yields the greatest efficiency gains, particularly on queries involving complex joins or subqueries.

In addition to objective performance measures, participants' subjective workload and trust toward the system were analyzed. Results from the NASA-TLX workload assessment (Hart and Staveland, 1988) revealed that SQL-LLM users reported lower overall cognitive workload (M = 42.3, SD = 9.8) compared with Snowflake users (M = 55.7, SD = 11.5), a statistically significant difference (t = 2.47, p = 0.021). Likewise, participants expressed higher trust ratings on a seven-point Likert scale for SQL-LLM (M = 6.1, SD = 0.8) than for Snowflake (M = 4.0, SD = 1.1; t = 3.12, p = 0.006). These findings complement the behavioral metrics reported above, confirming that the natural-language interface reduced

perceived cognitive workload and enhanced user confidence relative to the traditional SQL platform.

Taken together, the quantitative results demonstrate that SQL-LLM substantially improves both efficiency and accuracy compared to Snowflake, a traditional SQL platform. The system's advantages are most evident under high-complexity conditions, where users benefit from faster task execution, reduced reformulations, and steeper learning curves. These findings provide strong empirical support for the usability and performance benefits of NL2SQL systems and motivate further investigation into user interaction strategies in the next section.

VI. Qualitative Analysis and Results

This section presents qualitative findings based on detailed transcript analysis and behavioral annotations from screen recordings. While the quantitative results in previous section demonstrate SQL-LLM's advantage in efficiency and accuracy, this section focuses on how and why such differences emerged from the users' cognitive and behavioral strategies. We focus on user behavior patterns, strategic approaches, and affective responses to SQL-LLM and Snowflake, these two systems. Using open coding and thematic grouping, we identify key differences in how users navigated schemas, formulated queries, and adapted to each interface.

A. Qualitative Metrics

Beyond numerical data presented, video-based interaction analysis was conducted by using latest automatic transcription tools to acquire the transcripts from the video recording. In order to better understand user behavior, the following qualitative metrics were analyzed:

1) **Interaction Flow**
   - Observed how users engaged with SQL-LLM and Snowflake:
     - How often do the users pause before typing?
     - Did they re-check the schema or any instructions?
     - Did they verbalize confusion or trust/distrust of the tool?
   - These behaviors were categorized into short-burst queries vs. long single queries and hesitation moments vs. confident immediate attempts.
2) **Frustration and Error Recovery**
   - We identified frustration points (e.g., multiple failed attempts, long pauses).
   - Recovery time and success rates after incorrect queries were analyzed to determine whether SQL-LLM helped users diagnose and correct mistakes effectively.
3) **User Strategies**
   - Video footage was analyzed to distinguish different query strategies:
     - ***Trial-and-error***: Users frequently changed their inputs without systematic exploration.
     - ***Systematic exploration***: Users carefully constructed queries while referring to database schema.
     - ***Schema-first approach***: Users first analyzed the schema before formulating queries.
   - We cross-checked with Performance outcomes.
4) **User Feedback and Perception**
   - Open-ended feedback was collected regarding interface clarity, trust in the system, and ease of use compared to Snowflake.
   - Comments on ambiguity resolution, confidence in system correctness, and user frustration levels provided qualitative insights to support and complement statistical findings.

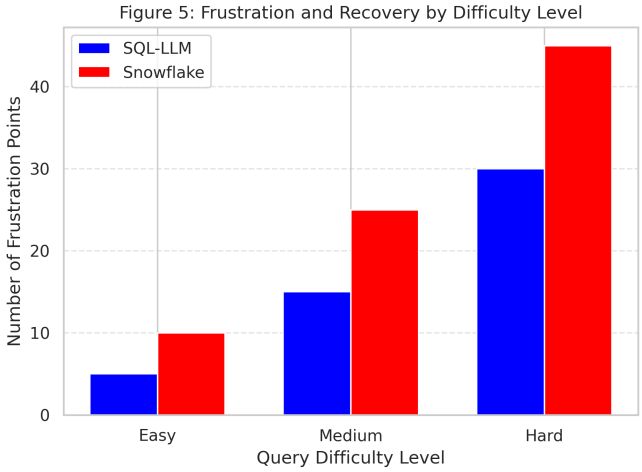

Fig. 5. Frustration and Recovery by Difficulty Level: Mean frustration levels (self-reported on a 5-point Likert scale) and recovery times (seconds) following errors, by query difficulty.

Regarding to the **Frustration and Recovery**, Frustration and recovery patterns further illustrate the contrast between SQL-LLM and Snowflake, these two systems. According to the analysis of the transcripts, we observed that SQL-LLM helped users regain progress with less effort, whereas Snowflake often left users in prolonged confusion. We elaborate on these differences below.

As follows, Figure 5 presents the frustration and recovery levels by query difficulty ranking from easy to hard for both SQL-LLM and Snowflake.

Figure 5 shows that SQL-LLM users reported lower frustration levels and recovered from errors 30–40 seconds faster than Snowflake users. This performance gap widens with query difficulty, highlighting SQL-LLM's ability to support smoother recovery and reduce cognitive burden during error correction.

For the **Interaction Flow**, we could observe that harder queries led to longer hesitation, more schema checks, and increased confusion. SQL-LLM users followed a structured approach, while Snowflake users relied more on trial-and-error.

The following Figure 6 presents the frequency of hesitation moments and schema checks by query difficulty level ranking from easy to hard for both SQL-LLM and Snowflake.

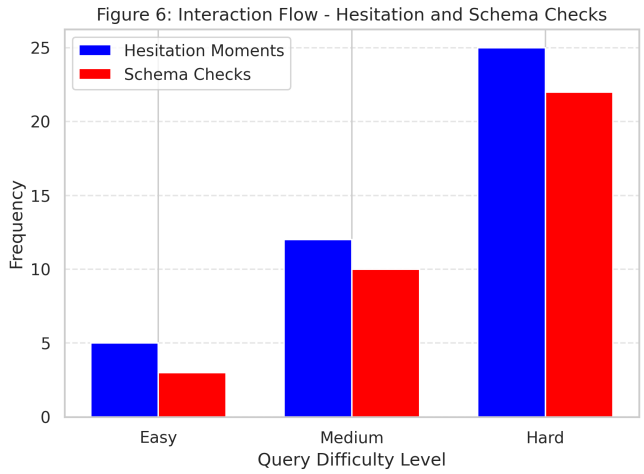

Fig. 6. Interaction Flow - Hesitation And Schema Checks: Frequency of hesitation moments and schema checks by query difficulty level.

Figure 6 reveals that Snowflake users exhibited significantly more hesitation and schema lookups, particularly for medium and hard queries. In contrast, SQL-LLM users engaged in fewer pauses and schema checks, indicating greater confidence and reduced cognitive load during query construction.

B. Qualitative Analysis

In addition to task performance and user feedback, a qualitative analysis of 11 participants' think-aloud transcripts and screen recordings were conducted, comparing SQL-LLM and Snowflake users in terms of behavior, strategies, and cognitive load. To identify user behavior patterns during query formulation and problem solving, three key themes emerged: confidence and trust, schema discovery, and error recovery.

1) Confidence and Trust in Query Output

SQL-LLM users often displayed early trust in the system's output. For instance, User D relied on SQL-LLM's generated query with minimal modification:

> *"Okay yes, the query given by the AI is correct. [...] I will just verify it quickly."* – D, SQL-LLM

Similarly, Quoc-Huy used the model's output as a reliable first pass:

> *"I just typed the question, and it gave me a working query. [...] I'll verify, but it looks correct."* – Q, SQL-LLM

In contrast, Snowflake users expressed more hesitancy and trial-and-error in early stages. Elias, for example, took time to realize the correct database context:

> *"It took me a while to realize the actual database I needed was Books. [...] I was looking all over the place."* – E, Snowflake

2) Schema Discovery and Strategy Formation

Snowflake users frequently relied on exploratory strategies, including browsing tables, issuing information schema queries, and performing join trials:

> *"Let me look up the INFORMATION_SCHEMA for column names. [...] This doesn't match. Let's try another join."* – B, Snowflake

User "B" also emphasized schema ambiguity:

> *"There's a couple of ways we could do this... concatenated names like Adam Troy are tricky."* – M, Snowflake

In contrast, SQL-LLM users often bypassed schema exploration, relying on the model's auto-generated query and only verifying joins after the fact. User G, for example, framed schema exploration around validating output rather than building queries from scratch:

> *"The SQL gave me the correct country... Let me confirm the table has inflation rate. Yes, confirmed."* – G, SQL-LLM

3) Error Recovery and Frustration

Snowflake users demonstrated frustration when navigating ambiguous joins or failing queries. Elias verbalized confusion:

> *"Why is someone's table empty? One table refers to something different than the other. [...] Just a million tables here."* – E, Snowflake

User "MB" made multiple passes at query refinement:

> *"Copy-paste doesn't work here... okay, now I'll try grouping by customer ID. Wait, I didn't group it."* – M, Snowflake

SQL-LLM users, while not immune to friction, typically debugged at the level of logical alignment rather than syntax:

> *"I don't think book ID and author ID are the same. [...] Let me just verify this join."* – D, SQL-LLM

Compared to Snowflake users who invested effort into discovering schema structure and debugging syntax, SQL-LLM users approached tasks with higher initial confidence, relied more on model output, and focused on semantic verification. These patterns suggest that LLM-based NLIDBs reduce cognitive load in early query formulation, though schema transparency remains essential for complex joins and multi-step reasoning.

These user comments align with quantitative findings on reduced completion time, fewer query reformulations, and lower frustration for SQL-LLM users, suggesting its natural language interface supports both usability and cognitive ergonomics.

To summarize behavioral differences between participants using SQL-LLM and Snowflake, we conducted a qualitative coding of think-aloud transcripts. Each participant was annotated along three key dimensions: confidence in the system's output, schema discovery strategy, and frustration episodes. The results are presented in Table V.

Table V summarizes qualitative annotations from think-aloud protocols, categorizing user confidence, schema strategy,

Participant	System	Confidence	Schema Strategy	Frustration
Gihan	SQL-LLM	High	Minimal	Low
Quoc-Huy	SQL-LLM	High	Minimal	Low
Dileep	SQL-LLM	High	Light Verif.	Low
Rohan	SQL-LLM	Moderate	Moderate	Moderate
Brendan	Snowflake	Low	Manual Expl.	High
Andrew	Snowflake	Low	Manual Expl.	Moderate
Bowen	Snowflake	Moderate	Info Schema	High
Caden	Snowflake	Moderate	Browse + Trial	Moderate
Elias	Snowflake	Low	Trial-and-Error	High
Michael	Snowflake	Moderate	Join Validation	High
Darkhan	Snowflake	Low	Manual Expl.	High

TABLE V
QUALITATIVE CODING OF USER BEHAVIORS ACROSS SQL-LLM AND SNOWFLAKE PARTICIPANTS. CONFIDENCE, SCHEMA EXPLORATION STRATEGY, AND FRUSTRATION EPISODES WERE IDENTIFIED FROM THINK-ALOUD PROTOCOLS.

and frustration. SQL-LLM users more frequently exhibited high confidence, minimal schema exploration, and low frustration, whereas Snowflake users showed the opposite pattern. These behavioral profiles align with the quantitative metrics, reinforcing SQL-LLM's cognitive benefits in task completion.

Rather than reiterating numerical superiority, the qualitative analysis illuminates the underlying causes—schema-first reasoning, prompt reinterpretation, and confidence calibration, which contributed to SQL-LLM's performance edge. The qualitative analysis complements our quantitative findings by uncovering the behavioral and cognitive differences between users of SQL-LLM and Snowflake. SQL-LLM users demonstrated more confident, schema-informed strategies and recovered from errors more efficiently, while Snowflake users exhibited higher hesitation, frustration, and reliance on trial-and-error. These insights underscore SQL-LLM's capacity to support more intuitive, low-friction user experiences, especially under complex task conditions and set the stage for deeper exploration of emergent usage patterns and design implications.

C. Micro-level Repair Strategies

Beyond broader behavioral patterns such as schema exploration and frustration recovery, we further analyzed fine-grained repair strategies that how users revised individual components of their queries in response to errors or ambiguity. This analysis offers deeper insight into how SQL-LLM and Snowflake support or hinder iterative refinement at a micro level.

Sometimes, some Snowflake users attempted localized adjustments, though often with difficulty. For instance, Michael diagnosed an omission in his aggregation logic and tried to fix it without rewriting the full query. He said:

> *"Copy-paste doesn't work here... okay, now I'll try grouping by customer ID. Wait, I didn't group it."* – User M, Snowflake

Similarly, Brendan modified join strategies after discovering a mismatch in schema alignment and he said:

> *"Let me look up the INFORMATION SCHEMA for column names. [...] This doesn't match. Let's try another join."* – User B, Snowflake

These attempts demonstrate partial repair efforts, but often lacked system-level support for contextual feedback, which prolonged the correction process.

In contrast, SQL-LLM users were able to verify and adjust at the semantic level. For example, Gihan treated the model's output as a base and selectively confirmed schema elements before finalizing execution. She said:

> *"The SQL gave me the correct country... Let me confirm the table has inflation rate. Yes, confirmed."* – User G, SQL-LLM

This reflects a more modular approach to debugging, where users trust and refine generated queries rather than rebuilding them entirely.

These observations suggest that natural language interfaces like SQL-LLM facilitate low-friction, micro-level revision strategies, whereas traditional SQL interfaces demand more complete syntactic correctness upfront. Interfaces that support interactive subquery previews, highlight faulty clauses, or allow partial execution could further align with users' natural repair behaviors and reduce cognitive burden.

In sum, the qualitative analysis reveals how SQL-LLM and Snowflake shaped distinct user experiences—not only in terms of high-level strategies like schema exploration and error recovery, but also through micro-level behaviors such as targeted repairs and trust articulation. These findings demonstrate that users engage with natural language and traditional SQL interfaces in fundamentally different cognitive modes. To build on these observations, the next section distills emergent themes and strategic adaptations that arose during interaction, offering deeper insight into how interface design influences user thinking, learning, and problem-solving trajectories.

VII. EMERGENT USER INSIGHTS

While earlier sections analyzed surface-level behaviors and performance metrics, this section further distills broader cognitive patterns observed in participants' interactions, especially with SQL-LLM. These insights highlight how users adapted their thinking, developed mental models of the interface, and engaged in strategic generalization, offering a deeper understanding of how Natural Language Interfaces for Databases (NLIDBs) influence user cognition.

A. Cognitive Shifts and Strategic Adaptation

Several SQL-LLM users progressed from task-specific phrasing toward more abstract, reusable prompt query phrasing. This evolution suggests growing meta-awareness and adaptive learning during the session. For instance:

> *"This one's kind of similar to the previous one. I think I can reuse that structure."*

Such comments indicate a shift from instance-level reasoning to higher-level prompt abstraction, facilitated by the natural language interface. This form of transfer learning—reusing linguistic patterns and modifying them based on feedback—was rarely observed in Snowflake users.

Another emergent behavior was exploratory probing. Some participants actively tested SQL-LLM's flexibility by varying the specificity of prompts and observing the output changes:

"Let me try something more abstract first... just want to see how it handles it before I get too specific."

This experimentation reflects an evolving mental model of the system's generalization ability and trust boundaries, revealing deeper engagement beyond basic task completion.

B. Reframing of Errors and Debugging Strategies

SQL-LLM users often reframed system errors not as outright failures but as opportunities for incremental refinement. Rather than restarting, users attempted localized verification and semantic correction:

"Let me just uncomment this and run it separately to make sure it's returning what I expect."

Such comments align with a debugging style typically associated with software engineering—progressive isolation of components rather than total revision. This contrasts with more rigid error responses observed in Snowflake users.

"I think I might be using the wrong table. I'm going to go back and just redo it completely." – MB, Snowflake

These differences suggest that natural language interfaces' affordances encourage iterative exploration and partial correction, while traditional SQL interfaces may impose a more brittle problem-solving style.

Thus, from schema-driven reasoning to abstract prompt formulation, the strategies emerging across participants reveal not only how users interact with NL2SQL systems but also how they construct mental models to guide their behavior. These insights underscore the importance of aligning system responses, error feedback, and interaction flows with users' evolving cognitive strategies. By doing so, future Natural Language Interfaces for Databases(NLIDBs) can better support both novice and experienced users through more adaptive, transparent, and intuitive experiences. These observations lay the groundwork for broader implications, which we explore in the following discussion on natural language interfaces and human-AI collaboration.

VIII. Discussion

This section synthesizes the key quantitative and qualitative findings to contextualize their implications for the design, evaluation, and deployment of Natural Language Interfaces for Databases (NLIDBs). Our results indicate that SQL-LLM outperforms a traditional SQL platform, Snowflake, in terms of task efficiency, query accuracy, error recovery, and user-reported frustration. Beyond these performance gains, we observed distinct differences in user strategies and cognitive behaviors, suggesting that NLIDBs not only enhance usability but also shape how users conceptualize and interact with databases. We now discuss the practical design implications, potential limitations, and future research directions arising from these insights.

A. Practical Implications

These findings have significant implications for the design and deployment of Natural Language Interfaces for Databases (NLIDBs):

- **Democratizing Data Access**: The substantial reduction in completion time, especially for complex queries, suggests that SQL-LLM can empower non-technical users to perform advanced data analyses without relying on SQL experts.
- **Enhanced Productivity**: Fewer query reformulations and faster error recovery translate to reduced cognitive load and increased efficiency, which are crucial in time-sensitive business intelligence environments.
- **Structured Query Strategies**: Behavioral analyses revealed that SQL-LLM promoted schema-first strategies, indicating that Natural Language Interfaces for Databases(NLIDBs) can encourage systematic exploration, enhancing user confidence and reducing reliance on trial-and-error approaches.
- **Trust and User Adoption**: Qualitative feedback indicated that providing natural language explanations and rapid clarifications boosted user trust - a factor critical for enterprise adoption of Natural Language Interfaces for Databases(NLIDBs).

B. Limitations

Despite the promising results, several limitations must be acknowledged:

- **Sample Size and Diversity**: Our study involved 20 participants, all with data analysis backgrounds. Future research should include a larger and more diverse user population to ensure generalizability across industries and expertise levels.
- **Task Scope**: While queries covered easy to hard difficulty levels, tasks were limited to synthetic databases. Real-world databases with complex schemas and security constraints may present additional challenges.
- **Participant Background**: All participants possessed at least basic SQL literacy, with none being complete novices. While this ensured fair comparisons between systems, NLIDBs may yield even greater benefits for users with little or no SQL experience. Future studies should examine how performance and trust evolve among non-technical populations.
- **Database Scale and Complexity**: The tasks were executed on small to medium-scale schemas (e.g., Books, Mondial Geo, Legislator). Results might differ for enterprise-level databases containing hundreds of tables and complex joins; further evaluations on large-scale, real-world datasets are needed to assess scalability and generalization.
- **Single NLIDB Evaluated**: This study focused on SQL-LLM compared to Snowflake. Including other state-of-the-art NL2SQL systems would provide a broader benchmarking context.
- **Longitudinal Effects**: The study measured immediate usability impacts. Long-term adoption studies are needed to assess sustained learning, trust retention, and integration into daily workflows.

This study confirms that while large language models enable more accessible querying interfaces, their true effectiveness

hinges on usability factors beyond raw accuracy. SQL-LLM's efficiency, reduced user frustration, and support for structured strategies reveal that thoughtful interaction design plays a critical role in user adoption and performance. However, limitations related to sample diversity, domain generalization, and long-term use remain. These findings emphasize the need for future NL2SQL systems to integrate personalization, explainability, and robust error handling to maximize real-world impact.

IX. Future Work

While this study provides empirical evidence on the usability and behavioral impact of Natural Language Interfaces for Databases(NLIDBs), several open challenges remain. Future research should address the limitations identified and explore new directions that enhance system generalizability, personalization, and long-term adoption. We outline key areas where further investigation is needed to advance the development and deployment of NL2SQL systems in real-world contexts:

- **Cross-domain Generalization**: Investigate SQL-LLM performance on real-world production databases across domains such as finance, healthcare, and e-commerce, where schemas are substantially larger and more complex.
- **Adaptive Interfaces**: Develop intelligent Natural Language Interfaces for Databases(NLIDBs) that adapt query suggestions and explanations based on user expertise, preferences, and historical behavior, enhancing personalization and learning support.
- **Robust Error Resolution Mechanisms**: Integrate advanced error-handling modules, including interactive debugging and natural language clarification dialogues, to improve recovery from misinterpretations.
- **Explainability and Transparency**: Design interfaces that transparently explain how natural language inputs map to SQL queries, improving user trust and facilitating learning.
- **Longitudinal Field Studies**: Conduct extended deployment studies to evaluate the sustained impacts of SQL-LLM adoption on user productivity, trust retention, and organizational decision-making quality.
- **Benchmarking Against Multiple NL2SQL Systems**: Compare SQL-LLM with other state-of-the-art Natural Language Interfaces for Databases(NLIDBs) in standardized usability evaluations to establish comprehensive performance benchmarks.

Pursuing these directions will deepen our understanding of Natural Language Interfaces for Databases(NLIDBs) design requirements and accelerate their integration into mainstream business analytics workflows.

X. Conclusion

This study investigated the usability and effectiveness of SQL-LLM, a state-of-the-art Natural Language Interface for Databases (NLIDB), compared to Snowflake, a traditional SQL analytics tool. Through a controlled user study involving 20 participants completing 12 realistic queries, we demonstrated that SQL-LLM:

- **Reduced completion times** by 10–30%, particularly for complex queries, indicating substantial efficiency gains.
- **Improved query accuracy**, achieving a 75% success rate compared to Snowflake's 50%, highlighting its practical reliability.
- **Decreased query reformulations and frustration levels**, suggesting better initial intent understanding and lower cognitive load.
- **Promoted structured querying strategies**, with users adopting schema-first approaches that enhanced confidence and systematic exploration.

These results highlight that, although technical precision is still very much important, user-centric design dimensions, such as transparency, modifiability, and the quality of interaction, play an equally crucial role in the successful implementation of NLIDBs.

In the future, studies are focusing on further incorporating NL2SQL into generic analytics workflows, providing adaptive interfaces for different types of users, as well as designing error-aware resolutions.

Given the fast progress in the large language models, we believe the Text-to-SQL accuracy is expected to be improved. Nonetheless, research unequivocally demonstrates that accuracy by itself is not sufficient: it is only through thoughtful design for usability that these systems can empower people to interact with data in intuitive, reliable, and efficient means.

Further work is needed to generalize these results across different domains, combine the adaptive and explainable functionality of the interfaces, and assess their long-term organizational impact to enable NLIDBs as a transformative technology in practice.

Ethics Statement

This study involved human participants recruited via Upwork. All participants provided informed consent, and their personal information was anonymized. The research protocol adhered to institutional ethics standards.

Acknowledgments

The development of SQL-LLM was supported by a startup company that was acquired before the completion of this manuscript. The acquiring company chooses to remain anonymous.

Conflicts of Interest

The authors have no financial interest in either product and no professional affiliation with either company. Snowflake was chosen purely as a representative SQL platform, which, at the time of the experiment, did not include any NL2SQL capabilities in their Web UI.

Data Availability

The study protocol, anonymized transcripts, and analysis code are available upon request.

REFERENCES

[1] H. V. Jagadish, A. Chapman, A. Elkiss, M. Jayapandian, Y. Li, A. Nandi, and C. Yu, "Making database systems usable," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 13–24. [Online]. Available: https://doi.org/10.1145/1247480.1247483

[2] C.-H. Lee, O. Polozov, and M. Richardson, "KaggleDBQA: Realistic evaluation of text-to-SQL parsers," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, C. Zong, F. Xia, W. Li, and R. Navigli, Eds. Online: Association for Computational Linguistics, Aug. 2021, pp. 2261–2273. [Online]. Available: https://aclanthology.org/2021.acl-long.176/

[3] D. Kochedykov, F. Yin, and S. Khatravath, "Conversing with databases: Practical natural language querying," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, M. Wang and I. Zitouni, Eds. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 372–379. [Online]. Available: https://aclanthology.org/2023.emnlp-industry.36/

[4] M. Pourreza and D. Rafiei, "Din-sql: decomposed in-context learning of text-to-sql with self-correction," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023.

[5] Z. Ning, Z. Zhang, T. Sun, Y. Tian, T. Zhang, and T. J.-J. Li, "An empirical study of model errors and user error discovery and repair strategies in natural language database queries," in *Proceedings of the 28th International Conference on Intelligent User Interfaces*, 2023, pp. 133–145.

[6] A. Simitsis and Y. Ioannidis, "Dbmss should talk back too," *ACM SIGMOD Record*, vol. 37, no. 3, pp. 29–34, 2009.

[7] Y. Tian, J. K. Kummerfeld, T. J.-J. Li, and T. Zhang, "Sqlucid: Grounding natural language database queries with interactive explanations," in *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '24. New York, NY, USA: Association for Computing Machinery, 2024. [Online]. Available: https://doi.org/10.1145/3654777.3676368

[8] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 3911–3921.

[9] I. Trummer, "Demonstrating gpt-db: Generating query-specific and customizable code for sql processing with gpt-4," *Proc. VLDB Endow.*, vol. 16, no. 12, p. 4098–4101, Aug. 2023. [Online]. Available: https://doi.org/10.14778/3611540.3611630

[10] Y. Vassiliou, M. Jarke, E. A. Stohr, J. A. Turner, and N. H. White, "Natural language for database queries: A laboratory study," *MIS Quarterly*, vol. 7, no. 4, pp. 47–61, 1983.

[11] F. Li and H. V. Jagadish, "Constructing an interactive natural language interface for relational databases," *Proceedings of the VLDB Endowment*, vol. 8, no. 1, pp. 73–84, 2014.

[12] ——, "Understanding natural language queries over relational databases," *SIGMOD Record*, vol. 45, no. 1, pp. 6–13, 2016.

[13] A. Narechania, A. Fourney, B. Lee, and G. Ramos, "Diy: Assessing the correctness of natural language to sql systems," in *Proceedings of the 26th International Conference on Intelligent User Interfaces*, 2021, pp. 597–607.

[14] A. Leventidis, J. Zhang, C. Dunne, W. Gatterbauer, H. V. Jagadish, and M. Riedewald, "Queryvis: Logic-based diagrams help users understand complicated sql queries faster," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 1845–1859.

[15] D. Miedema, G. H. L. Fletcher, and W. M. P. van der Aalst, "Sqlvis: Visual query representations for supporting sql learners," in *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2021, pp. 1–5.

[16] Z. Yao, Y. Su, H. Sun, and W.-t. Yih, "Model-based interactive semantic parsing: A unified framework and a text-to-sql case study," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 5447–5458.

[17] I. Gür, Y. Su, X. Yan, and X. Wang, "DialSQL: Dialogue based structured query generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 1339–1349.

[18] J. Li, B. Hui, G. Qu, J. Yang, B. Li, B. Li, B. Wang, B. Qin, R. Geng, N. Huo *et al.*, "Can LLM already serve as a database interface? a big bench for large-scale database grounded text-to-sqls," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[19] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Advances in Psychology*. Elsevier, 1988, vol. 52, pp. 139–183.