Natural Language Meets Database: System Transparency and User Understanding in NL-to-SQL Translation

Bergische Universität Wuppertal

PARSSA JASHNIEH, THIVYAN SIVANANTHAN, and JACOB ORTENBERG

Online shopping platforms often struggle to meet diverse user search needs due to the rigidity of traditional SQL templates and filtering mechanisms, leading to suboptimal user experiences. This paper explores an AI-driven natural language interface (NLI) that dynamically generates SQL queries from user inputs, aiming to enhance search intuitiveness and reduce dependency on manual filters. We developed a prototype e-commerce interface focused on a dog webshop, comparing AI-generated queries (via the Claude API) against a static template approach in a user study with 13 participants. Results show high user satisfaction with the AI system (mean 1.69, SD 0.48) compared to the conventional system (mean 3.62, SD 0.51), driven by its ability to interpret intent and handle synonyms (e.g., "Wiener Dog" as "Dachshund"). While 69.2% of users could forgo filters, a minority favored their retention for visual clarity, highlighting a nuanced role for traditional mechanisms. Technical feasibility was demonstrated, though security vulnerabilities (e.g., unintended query execution) necessitate safeguards like blacklists. User-suggested correction strategies, such as dynamic filter adjustments and confidence indicators, to further enhance transparency and trust. Despite limitations—including a small, tech-savvy sample and a basic static baseline—our findings suggest NL-to-SQL systems can transform e-commerce search, with broader implications for human-computer interaction, provided usability and security challenges are addressed.

CCS Concepts: • **Human-centered computing** → *Web-based interaction*; Interaction design theory, concepts and paradigms.

Additional Key Words and Phrases: AI, Seq2Seq, SQL, UI, UX, Filters, Search, Transfer Learning, NLP

ACM Reference Format:
Parssa Jashnieh, Thivyan Sivananthan, and Jacob Ortenberg. 2025. Natural Language Meets Database: System Transparency and User Understanding in NL-to-SQL Translation. In *Final Project for the HIC Module at the Bergische Universität Wuppertal(Human-Computer Interaction & Artificial Intelligence), October 7–February 28, 2025, Wuppertal, Germany*. ACM, Wuppertal, North Rhine Westphalia, Germany, 14 pages.

1 Introduction

Online shopping platforms like Amazon and eBay have become the go-to sources for a vast range of products. However, users often face difficulties when searching for specific items, especially products with more complex or technical specifications, such as electronic devices, computer components, or specialized tools. These challenges can lead to frustration and even cause users to switch platforms in search of a better experience. This frustation stems from individual differences in search behaviour, which varies widely among users. Some rely on generic search terms, such as "laptop," while others prefer highly specific queries, like "Dell XPS 15 9530 with Intel Core i7 and 32GB RAM." In a preliminary analysis conducted as part of this research, we observed that search behavior splits roughly evenly between these two approaches, a finding derived from user interactions with our prototype interface described later in this paper, making it essential for e-commerce platforms to optimize their search functionalities to accommodate diverse

Authors' Contact Information: Parssa Jashnieh; Thivyan Sivananthan; Jacob OrtenbergBergische Universität Wuppertal.

Manuscript submitted to ACM

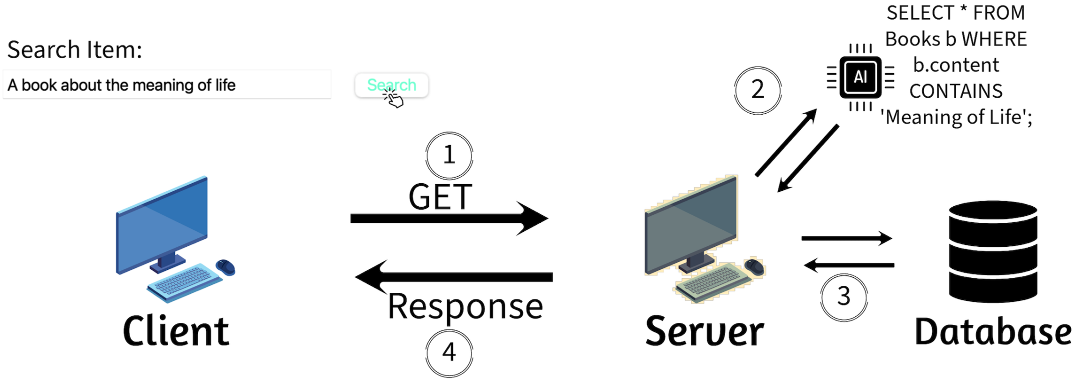

Search Item:
A book about the meaning of life
Search
1
GET
Response
4
2
AI
SELECT * FROM Books b WHERE b.content CONTAINS 'Meaning of Life';
3
Client
Server
Database

Fig. 1. Highlighting the interaction paradigm of the user with an AI. Instead of traditionally passing the users input to a static SQL-Template, it is passed to an AI-System that interpretes the users search request and dynamically generates a SQL-Query . The resulting query is then executed and displayed for the user in the Interface.

user needs. At the same time, this presents a challenge, as creating a platform that satisfies all users equally is difficult. While our investigation focuses on e-commerce due to its widespread use and direct impact on user satisfaction and business success, the proposed approach could also benefit other domains, such as medical databases or academic research platforms, where complex queries and diverse user needs are similarly prevalent.

In addition to search queries, filtering options play a crucial role in the online shopping experience. Filters help users refine their results based on criteria such as price, size, brand, technical specifications, or customer ratings. However, ineffective or overly complex filtering systems can hinder rather than enhance the search process. If filters fail to match user expectations, prove difficult to navigate, or do not efficiently narrow down results, users may abandon their search altogether. Therefore, improving search functionalities is essential to enhance user experience, increase customer satisfaction and ensure platform loyalty.

Building upon these challenges, we propose an alternative approach to traditional search mechanisms. Instead of relying on predefined SQL templates, we suggest dynamically generating SQL queries using machine learning techniques. By leveraging AI to generate queries in real time, the system can automatically interpret the user's intent based on their input, providing more precise and relevant search results initially. If the AI achieves high accuary, this approach could reduce or even eliminate the need for manual filtering by embedding user preferences directly into the generated queries. This shift could significantly enhance human-computer interaction, making the search process more intuitive, efficient, and user-centric. To determine whether such a scenario is truly possible and to understand how users would interact with this approach, we conducted a study. For this purpose, we developed a prototype interface that accepts natural language search queries, allowing us to assess user behaviour and compare the effectiveness of dynamic query generation against traditional static templates.

1.1 The conventional search mechanism

The Open Worldwide Application Security Project (OWASP) regularly updates its top 10 list of web application security risks. SQL Injection attacks have consistently remained in this list across multiple iterations, including the 2017 and 2021 versions, largely due to the widespread use of custom SQL query templates in web applications[11]. These templates follow predefined schemas where user input, such as search queries, is inserted into predetermined placeholders. This approach presents multiple significant limitations. These templates are vulnerable to SQL Injection attacks (SQLIA) if not properly secured. For example, a poorly sanitized input like "laptop; DROP TABLE products;" could compromise the entire database. Additionally, their search capabilities are inherently restricted, as the searchable attributes or columns must be explicitly defined by developers based on specific requirements. While these templates excel at simple keyword matching, they struggle with natural language queries due to their rigid, predefined structure. Although this keyword-based approach can efficiently process basic search terms, users typically need to rely on additional filtering mechanisms to narrow down their results to find exactly what they are looking for. However, these filtering systems come with their own set of limitations.

1.2 Limitations of traditional filtering mechanisms

Traditional e-commerce platforms typically implement filtering systems through a combination of predefined dropdown menus, checkboxes, and range selectors. While these mechanisms provide basic functionality, they present several significant challenges. First, the static nature of these filters requires regular maintenance and updates to accommodate new product categories and attributes. Second, the presentation of filter options often follows a one-size-fits-all approach, disregarding individual user preferences and shopping patterns. For instance, technical users searching for computer components might prefer detailed specification filters, while casual shoppers might find such granularity overwhelming. Furthermore, traditional filtering systems often struggle with semantic relationships between different product attributes. A user searching for a "gaming laptop" might need to individually select filters for high RAM (e.g., 16GB) and a fast process (e.g., Intel Core i7), despite these being inherently tied to gaming performance. This disconnect between user intent and filtering capabilities becomes particularly problematic when dealing with complex product categories that have numerous interdependent specifications, such as electronic devices or specialized equipment.

1.3 Moving beyond traditional search limitations

These limitations of conventional search templates and filtering mechanisms highlight the need for more sophisticated approaches that can adapt to diverse user needs. Modern developments in database querying, machine learning, and natural language processing offer promising directions for adressing these challenges through more flexible and intelligent search solutions.

Based on the identified challenges in e-commerce search mechanisms, we formulate three main research questions that guide our investigation.

1. We examine how natural language interfaces affect user search behavior and satisfaction compared to traditional keyword-based approaches.
2. We investigate the technical feasibility of translating diverse natural language queries into SQL statements.
3. We analyze user preferences and requirements regarding correction mechanisms in AI-driven search systems. This investigation focuses on understanding which types of refinement options users would find most helpful when initial search results don’t meet their expectations.

2 Background

The gap between human communication and database interaction has been a persistent challenge in computer science. Users of information systems naturally express their information need in everyday language, yet databases require precise, syntactically correct and structured queries - a mismatch that has fueled decades of research into natural language interfaces to databases [5]. Traditional keyword-based search systems often fail to capture the nuanced intent behind user queries, particularly in e-commerce environments where user frustation with suboptimal search results can directly impact business success [10]. This fundamental challenge has led to various approaches in bridging the gap between natural language and database queries over the past decades.

2.1 Evolution of Natural Language Interfaces

The journey of Natural Language Interfaces (NLIs) dates back to the 1960s with pioneering systems like *BASEBALL*, which translated natural language questions about baseball statistics into database queries [8]. The development of *LUNAR* [15] followed, helping NASA geologists query chemical analyses of lunar rocks. Both operated in narrow domains, relying heavily on manually crafted rules and domain-specific knowledge [4].

By the early 2000s, systems like *PRECISE* advanced the field by treating translation as a graph matching problem, achieving high precision for semantically tractable queries within a supported domain. However, such systems still required extensive manual engineering and struggled with ambuigity.

Current generation systems, powered by foundation models such as ChatGPT, have largely overcome these historical limitations, demonstrating broad language understanding with minimal engineering effort [12]. In the context of e-commerce, these models can interpret complex queries like "cheap gaming laptop with high-quality webcam" by leveraging vast training data to infer intent and map it to database fields/columns. For example, such systems could dynamically link "gaming" to attributes like high RAM or GPU performance, adressing user needs more intuitively than static templates. Nonetheless, new challenges emerge, such as the lack of transparency in the output of these foundatin models and the need to ensure that generated queries are both semantically accurate and syntactically correct. This is critical, as erroneous SQL queries cannot be executed, potentially resulting in a degraded user experience, where users encounter errors or fail to retrieve search results. These emerging issues highlight the need for research into transparent and reliable NL-to-SQL systems, particularly in domains like e-commerce where user satisfaction is paramount.

2.2 Current Approaches to NL-to-SQL

The rise of deep learning has transformed NL-to-SQL translation from rule-based and statistical methods to neural network based approaches. Early systems like Seq2SQL [16] used sequence-to-sequence models to generate SQL from natural language, performing well on simple queries but struggling with complex ones involving multiple conditions or joins. Modern transformer-based models, such as those build on BERT and GPT-architecture, improve accuaracy by incorporating schema linking and intermediate representation, as as seen in systems like RAT-SQL [14]. Tools like SQLAI.ai [1] exemplify this trend, leveraging large language models to generate SQL queries from natural language inputs, with minimal user effort. However, SQLAI.ai introduces trade-offs: 1) it requires payment, 2) lacks transparency in its query generation process and 3) functions primarily as a developer tool, prioritizing flexibility over safety or interpretability. Nevertheless, these advances enable robust query generation across diverse schemas, yet challenges remain in ensuring transparency and handling ambiguous input-issues, which is critical for practical applications. As these techniques mature, their potential to enhance domain-specific systems, such as those in e-commerce, has sparked growing interest, bridging the gap between technical innovation and real-world usability.

2.3 Usability in NL-to-SQL Systems

Usability is a cornerstone of effective human-computer interaction, especially in e-commerce, where intuitive and efficient search experiences can determine whether users complete a purchase or abandon their search altogether. Unlike traditional interfaces that force users to adapt to rigid keywords and filters, NL-to-SQL systems aim to reverse this dynamic by interpreting natural language. The success of such systems relies heavily on aligning with user expectations. An important factor is interpreting diverse queries, from vague inputs like "good laptop" to precise ones like "16Gb Ram Laptop under 600€". Studies show that flexibility in handling such variaty is vital, as users rarely follow a single search query pattern [10]. Such system have to map these input to relevant database field using the context of the domain or available metadata to ensure relevance. Another key aspect is feedback: when results miss the mark (for example, let the search-query be "gaming laptop"), clear feedback like "Did you mean high GPU performance?" - improve relevance and build trust by revealing the systems logic [4], which is crucial in e-comerce settings to reduce frustation and to retain users.

Effective feedback naturally leads to the broader challenge of designing an interface that balances accessibility with functionality. Rather than burdening users with technical complexity, the system should deliver results in a way that feels intuitive for casual shoppers—e.g., prioritizing top matches based on intent—while allowing more experienced users to refine searches through additional prompts or guided options. This approach ensures a seamless experience across diverse audiences, minimizing search friction in a competitive e-commerce landscape. By prioritizing such usability principles, NL-to-SQL systems can pave the way for practical implementations, such as the interface we developed to explore these dynamics in a controlled user study, as detailed in the following section.

3 Methods

To investigate the potential of our proposed approach, we designed a study centered around a webshop interface that enables users to directly compare dynamically generated SQL queries with traditional static templates [9]. This dual-display setup allows participants to experience both methods in real time and assess their effectiveness in retrieving relevant search results. To ensure a focused and controlled analysis, we limited the webshop to a single product category. We chose dogs as the focal category due to their popularity and the diverse range of attributes—such as breed, age, gender, color, size, and over 50 unique features—that lend themselves well to complex, semantic search queries. The underlying dataset comprises 147 AI-generated dog records, synthesized using ChatGPT and Claude, and enriched with real-world data from www.edogs.de [6]. A structured relational database was developed to support efficient querying across these attributes.

The interface presents search results in two side-by-side sections, as illustrated in Fig. 6. The left section displays results from AI-driven, dynamically generated queries, while the right section shows results from a static query template enhanced with semantic matching. Users submit their queries via a shared search bar, with filtering options applied consistently across both sections through a unified sidebar. Although the AI-driven approach could theoretically incorporate filter-like constraints within natural language inputs, we retained traditional filters to maintain a consistent user experience and facilitate direct comparison. This design empowers participants to evaluate both methods simultaneously, offering insights into how each aligns with their search preferences and behaviors.

3.1 Implementation of the Interface

Initially, we aimed to leverage a local large language model for dynamic SQL query generation, drawn by the advantages of cost-free access, robust documentation, and adaptability. However, hardware constraints quickly surfaced. Testing models like Meta Llama-3.3-70B-Instruct [2] and Microsoft Phi-4 [3] from Hugging Face on a system with 8GB RAM proved unfeasible due to insufficient memory and suboptimal performance, even with quantized versions (e.g., GGUF models). These limitations prompted a shift to an API-based solution. We selected the Claude API, which has demonstrated superior performance in complex tasks like programming and translation compared to alternatives like ChatGPT [13].

For the AI-driven approach, we crafted a prompt that includes the database schema, a comprehensive list of available features (e.g., "Hündin" or "Rüde" for gender), and a base query to guide the generation process. This base query, shown in Fig. 2, provides a standardized structure that ensures consistent table relationships and attribute retrieval, which the AI refines based on user input. This setup enables the system to interpret both vague and detailed queries accurately, mapping them to appropriate SQL statements. The static template-based method employs a streamlined design, tokenizing user input and applying an OR-condition across dog breeds and descriptions (Fig. 3). Our static template was designed to provide a fair basis for comparison by reflecting typical keyword-matching techniques used in many e-commerce systems.

Base-Query

```
SELECT h.*, GROUP_CONCAT(F.Name)
FROM Hund h
INNER JOIN
main.Feature_Hund FH on h.id = FH.Hund_id
INNER JOIN main.Feature F on F.id = FH.Feature_id
```

Fig. 2. Base query utilized in our study, dynamically extended by the Claude API. The GROUP_CONCAT function aggregates all associated features into a single entry per dog, while the predefined structure ensures a consistent result order, such as the sequence of attributes, essential for our search implementation.

Static Template Implementation

```
SELECT h.*, GROUP_CONCAT(F.Name)
FROM Hund h
INNER JOIN main.Feature_Hund FH on h.id = FH.Hund_id
INNER JOIN main.Feature F on F.id = FH.Feature_id
WHERE (h.description LIKE '%Labrador%' AND h.breed LIKE '%Labrador%')
   OR (h.description LIKE '%freundlich%' AND h.breed LIKE '%freundlich%')
```

Fig. 3. Static query template implementation exemplified with the input 'Labrador freundlich'. For each token ('Labrador' and 'freundlich', meaning 'friendly' in German), the template generates LIKE conditions to match both the h.breed and h.description attributes, linked by OR, extending the base query for keyword-based search.

3.2 User Study

Our study involved 13 participants—12 male and one female—predominantly computer science students pursuing Bachelors or Masters degrees, many with prior exposure to artificial intelligence through coursework or personal interest. This convenience sample, drawn from our academic network, is small and relatively homogeneous but sufficient for an exploratory investigation. Participants were briefed on the study's goal: to compare the AI-driven search system against a conventional template-based approach and assess its potential to enhance product search accuracy and user satisfaction. We hypothesized that the AI-generated queries would outperform the static method, particularly for vague, descriptive, or multi-criteria searches, while the static approach would remain competitive for simple, attribute-based queries.

The study unfolded in three phases. First, participants answered general questions about their past search experiences, focusing on challenges encountered and their impact on shopping behavior. In the second phase, they interacted with the interface, completing six predefined search tasks designed to explore varying levels of complexity:

- Find the Labrador Retriever
- Find the most expensive Labrador Retriever
- Find the most expensive Labrador Retriever (without price filter)
- Find all dogs except Australian Shepherd
- Find Chihuahuas that are eager to work
- Find all Rottweilers that are senior-friendly or German Shepherds younger than 12 months

These tasks progressed from basic attribute searches to complex queries unachievable with filters alone, such as those requiring logical operators like "OR" across distinct attributes. Participants could freely use natural language or static filters (unless specified), with the option to refine queries if results fell short. Following this guided exploration, they were encouraged to devise their own complex searches. The final phase involved a detailed questionnaire with a 5-point Likert scale assessing accuracy, ease of use, and efficiency, complemented by open-ended questions for qualitative insights (please refer to Appendix A for an overview). This mixed-methods approach illuminated user preferences, system strengths, and areas for refinement, while also probing whether AI-driven search could supplant traditional filtering mechanisms.

4 Result

This section presents the findings of our user study, evaluating the performance of an AI-driven natural language interface (NLI) against a conventional search system in an e-commerce context. The results address user satisfaction, the role of filters, technical feasibility, and possible correction mechanisms for query optimization, providing insights into the effectiveness of dynamic SQL query generation compared to static templates.

4.1 User Satisfaction and Search Behavior(RSQ1)

Figure 4 illustrates that all participants rated the AI-driven system's results highly on a 5-point Likert scale, with 69% (9/13) selecting 'very satisfied' (1/5) and 31% (4/13) 'satisfied' (2/5), resulting in a mean of 1.69 (SD 0.48, Bessel-corrected). This low variability reflects consistent user approval, attributed to the system's accurate intent interpretation, including its handling of synonyms (e.g., 'Wiener Dog' as 'Dachshund,' Fig. 6). This feature was absent in the conventional system. In contrast, satisfaction with the conventional system was markedly lower, averaging 3.62(SD = 0.51) on the same scale, with only 38% (5 out of 13) rating it as neutral (3/5), and 62% (7 out of 13) as dissatisfied (4/5), with no participants rating it as very high,high or very dissatisfied (see Figure 5). Users found its performance lacking, particularly when compared to the AI's intuitive query handling. An interesting observation emerged: as participants grew accustomed to the AI's ability to understand their intent, their reliance on manual filters decreased. This shift likely contributed

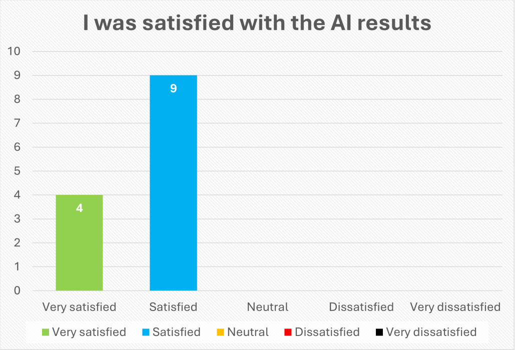


Fig. 4. The users' rating regarding the satisfaction of the results displayed by the AI. As can be seen in the picture, the participants were very satisfied with the results of the AI.

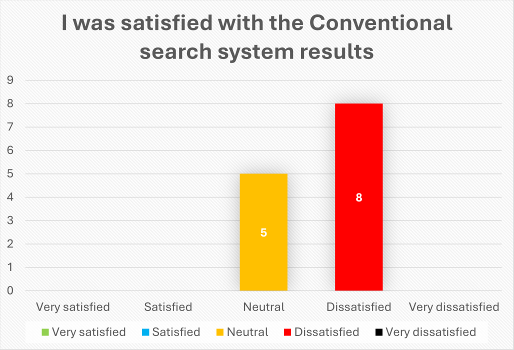
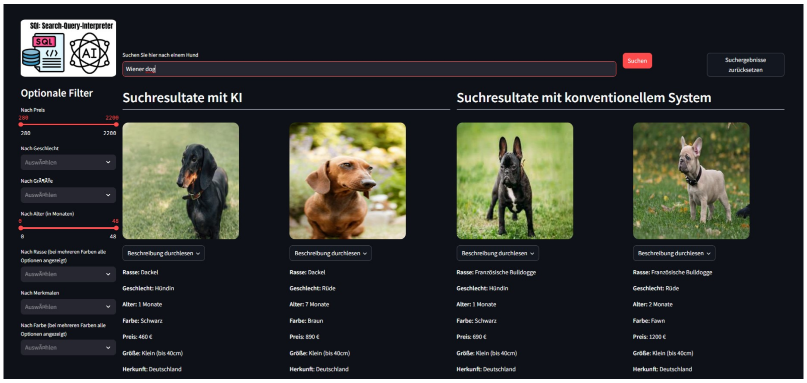


Fig. 5. The evaluation of the users regarding the satisfaction with the result displayed by the conventional system. The picture shows that they were not satisfied with the results displayed by the conventional system.

to the conventional system's perceived inferiority, as it depended heavily on filter usage to refine results. This rapid adaptation suggests a potential change in search behavior, though further analysis is needed to confirm its extent. It's worth noting that the results for the conventional system should be considered in light of the relatively simple static template we used. This basic design may have contributed to its poorer performance, and a more sophisticated static template could potentially have yielded better outcomes.



Suchen Sie hier nach einem Hund: Wiener dog

Suchresultate mit KI

Rasse: Dackel
Geschlecht: Hündin
Alter: 1 Monate
Farbe: Schwarz
Preis: 460 €
Größe: Klein (bis 40cm)
Herkunft: Deutschland

Rasse: Dackel
Geschlecht: Rüde
Alter: 7 Monate
Farbe: Braun
Preis: 890 €
Größe: Klein (bis 40cm)
Herkunft: Deutschland

Suchresultate mit konventionellem System

Rasse: Französische Bulldogge
Geschlecht: Hündin
Alter: 1 Monate
Farbe: Schwarz
Preis: 690 €
Größe: Klein (bis 40cm)
Herkunft: Deutschland

Rasse: Französische Bulldogge
Geschlecht: Rüde
Alter: 2 Monate
Farbe: Fawn
Preis: 1200 €
Größe: Klein (bis 40cm)
Herkunft: Deutschland

Fig. 6. In this picture we see the website that was developed for the project. The website was divided so that the user can quickly see which results belong to AI and which to the conventional system. To demonstrate the performance of the AI, Wiener dog was entered in the input field. As you can see, the AI can handle this, but the conventional system cannot.

4.2 The Role of Filters in Enhancing User Experience

While the AI demonstrates potential to supplant traditional filters, our study underscores their enduring value. Figure 7 details preferences from a 5-point Likert scale (1='strongly agree', 5='strongly disagree'): 69.2% (9/13) agreed or strongly agreed they could forgo filters (scores 1-2), 23.1% (3/13) disagreed, favoring retention (score 4), and 7.7% (1/13) were neutral (score 3). Open-ended responses further suggest filters complement the AI for specific tasks (e.g., price sorting).

Of the participants who preferred filters, three stated in open-ended comments that they appreciated the visual clarity, such as when sorting by price or selecting ranges (e.g., ‘price between 50-100€’), compared to text-based inputs. For instance, participants noted that setting a price range via a slider was faster and more intuitive than specifying it textually. These findings, however, are based on a small, homogeneous sample of 13 participants, primarily computer science students, which may limit generalizability.

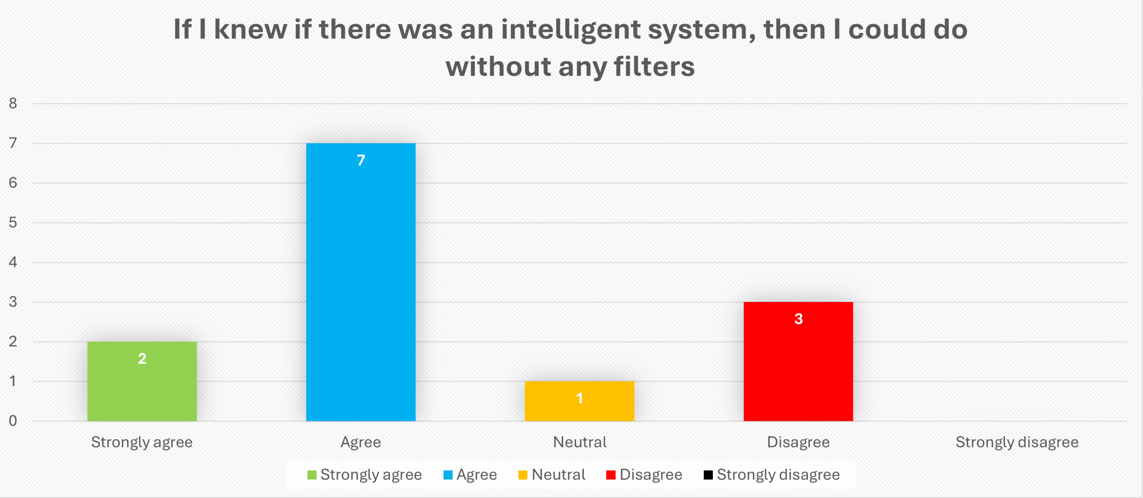


Fig. 7. After the users have used the system, they should state whether they would manage without the filter.The results show that the majority of users can manage without a filter without any problems. Three times “disagree” was indicated, while no “strongly disagree” was recorded.

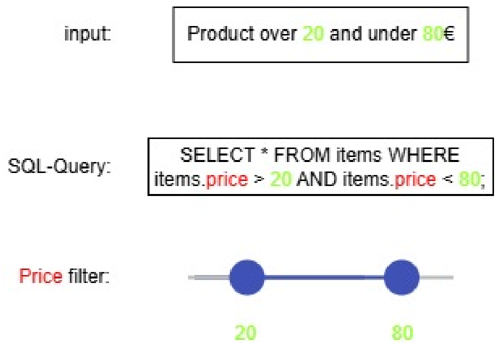
4.3 Technical Feasibility and Security Considerations (RSQ2)

Building on user experience insights, this subsection examines the technical implications of the AI-driven system. During the study, some participants tested the system’s boundaries, revealing critical security concerns. For example, one user entered a command to ’delete all tables,’ which the AI translated into an executable SQL query—an unintended capability that exposed a vulnerability. Ideally, the system should restrict operations to read-only searches, preventing modifications to the database or its schema. While our simple dog webshop schema excluded sensitive data like user information, real-world applications would demand robust safeguards. To mitigate this vulnerability, we propose implementing a blacklist configuration to restrict the AI system’s SQL query generation. This file (e.g. in csv-format) would specify tables or columns deemed off-limits, ensuring they remain inaccessible within the generated queries. Additionally, given that search results constitute a read-only operation, we recommend incorporating a list of prohibited keywords—such as ’DELETE,’ ’UPDATE,’ or ’DROP’—into the lock-file. Prior to execution, each AI-generated SQL query would be validated against these restricted elements (tables, columns, and keywords), for example through the use of regular expressions, to guarantee compliance with security and regulatory standards.

4.4 Optimizing Query Correction Strategies (RSQ3)

To address usability challenges and enhance trust, we investigated user preferences for refining inaccurate search results. Participants suggested several strategies, detailed below, to improve the system’s transparency and responsiveness.

4.4.1 Dynamic Filter Adjustment. A participant has proposed an adjustment of the filters, based on the input provided. As illustrated in Figure 8, the idea is to highlight implicitely used filters visually, i.e. dynamically adjusting filters on the interface. Therefore, the user is able to ascertain which filters the AI utilizes and, consequently, identify the potential origin of an error. Therefore, the user has the option of either utilize the filter to resolve the issue or adjusting the initial search query.



```
SELECT * FROM items WHERE
items.price > 20 AND items.price < 80;
```

Fig. 8. Here the user can see how the filter is adapted to the user's input. The filter adapts to user input, revealing AI misinterpretations (implicitely).

4.4.2 User Notification for No Results. A further potential avenue for enhancing the efficancy of the user's outcomes is the implementation of an artificial intelligence system that can evaluate the combinability of diverse features during the user's input phase. In such a case, it is essential that the user be alerted to this possibility. The current state of the AI, reflecting the original user input, is shown in Figure 9. The subsequent version has been enhanced to alert the user to the absence of search results for a given combination of features. In this instance, the "under 2 months" feature is distinctly emphasized, as it does not result in any search results. Therefore, the user has the capacity to modify the preliminary search query and discern the elements that are not compatible.

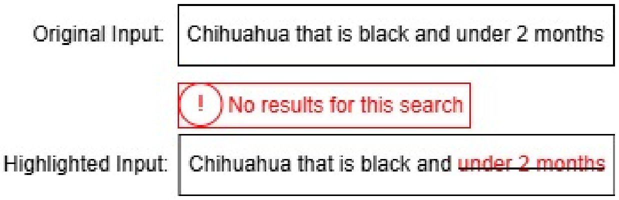

Original Input:
Chihuahua that is black and under 2 months
! No results for this search
Highlighted Input:
Chihuahua that is black and ~~under 2 months~~

Fig. 9. Here the user is informed by the AI if the features are not combinable. It is made clear that 'under 2 months' cannot be combined with the other features, as this feature is crossed out.

4.4.3 Display of Similar Results. A further point to be considered is the potential for the AI to present analogous products in the event that the search yields few results. A relevant example would be a search for dogs that are of medium size. In the event that the available results are limited, the artificial intelligence could be programmed to display dogs of smaller stature.

4.4.5 Query Confidence Indicator. Another suggestion from users is that the AI should more clearly indicate its uncertainty when interpreting a query, for example, by providing an uncertainty score/confidence ranging from 0 to 1. This score would reflect how confident the AI is in its understanding of the request. This would allow users to better assess whether the provided results might be inaccurate or incorrect, enabling them to make adjustments if needed. An example from the current AI system illustrates this. When a user searches for a "cheap dog," the Claude API simply returns the least expensive dog available. However, this may not align with what the user intended; perhaps they were

4.4.4 Real-time AI-driven Suggestions. Another idea for improving the system is to offer the user search suggestions as they type. This not only enables the AI to deliver more relevant results, but also to better understand what the user is looking for. An example of this is shown in Figure 10. The user enters 'Puppy that is black and...' and the AI suggests narrowing the search - for example to 'Puppy that is black and under three months'.

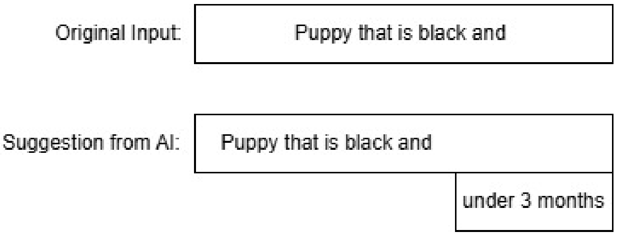


Fig. 10. The AI provides real-time suggestions for narrowing searches, such as additional features for 'Chihuahuas'.

looking for a cost-effective yet suitable dog for specific needs. An uncertainty score could indicate how reliable the results are, signaling to the user that the interpretation of their query might not be entirely accurate. This would give the user the opportunity to refine their request for more appropriate responses.

5 Discussion

Our study set out to explore the potential of an AI-driven natural language interface (NLI) for enhancing e-commerce search experiences by dynamically generating SQL queries, as opposed to relying on traditional static templates and filtering mechanisms. The results, detailed in Section 4, provide compelling evidence that users find the AI-driven system more intuitive and satisfying (mean satisfaction score of 1.69, SD 0.48) compared to the conventional system (mean 3.62, SD 0.51). However, several limitations and biases in our methodology warrant a critical discussion to contextualize these findings and guide future improvements.

5.1 User Satisfaction and Behavioral Shifts

The high satisfaction with the AI-driven system, as reported in Section 4.1, aligns with our hypothesis from Section 1 that dynamically generated SQL queries could better capture user intent, reducing reliance on manual filters. Participants appreciated the AIs ability to handle synonyms (e.g., "Wiener Dog" interpreted as "Dachshund") and typographical errors, a capability absent in the static template approach. This finding echoes Section 2.2 discussion of modern NL-to-SQL systems leveraging large language models to interpret complex queries intuitively. However, this positive reception may be inflated by the participant pools composition—exclusively computer science students, with only one female participant among the 13. As noted in Section 3, this homogeneous group likely possesses above-average technical literacy, potentially skewing their comfort with an AI-driven interface over traditional filters. A more diverse sample, including non-technical users or older adults, might reveal different preferences, particularly given the adaptation period observed in Section 4.1, where users initially favored filters before shifting reliance to the AI.

5.2 Role of Filters and Implementation Bias

Section 4.2 revealed a nuanced picture: while 69.2% of participants could manage without filters in the AI-driven system, a notable minority (23.1%) preferred their retention for visual clarity (e.g., price sliders). This suggests that while the AI holds promise for reducing filter dependency—as hypothesized in Section 1.3—the transition may not be seamless for all users. A significant confounder here is the naive implementation of the static template, described in Section 3.1 as a basic tokenization and OR-condition matching across breeds and descriptions. This rudimentary design, acknowledged in Section 4.1, likely underperformed compared to industry-standard static templates (e.g., those on Amazon or eBay), potentially exaggerating the AIs perceived superiority. A more sophisticated static baseline, incorporating weighted attributes or semantic matching, could have provided a fairer comparison. This limitation underscores a methodological bias that future iterations must address to validate the AIs advantages more robustly.

5.3 Technical Feasibility and Prompt Design

The technical feasibility of NL-to-SQL translation, a central research question from Section 1.3, was confirmed by the Claude API's ability to generate accurate SQL queries from diverse user inputs, as detailed in Section 3.1. However, a significant security vulnerability emerged in Section 4.3: the AI translated malicious inputs, such as „delete all tables," into executable SQL queries. Unlike the static template, which avoids such risks through its predefined structure, this flaw underscores a critical gap in our implementation and echoes the SQL Injection concerns raised in Section 1.1. The system prompt for the Claude API, outlined in Section 3.1, was rudimentary—limited to the database schema and a base query—lacking explicit constraints (e.g., restricting operations to SELECT statements). This oversight clashes with Section 2.2's emphasis on transparency and safety as prerequisites for modern NL-to-SQL systems. A more robust solution would have integrated our proposed blacklist configuration (Section 4.3), filtering out hazardous commands like DELETE or DROP. Additionally, instructing users to test the system with adversarial inputs akin to „delete all tables" could have exposed further weaknesses, but time constraints prevented this enhancement. These refinements would have strengthened both security and practical applicability, aligning the system more closely with real-world requirements.

5.4 Correction Strategies and Missing Metrics

Section 4.4 outlined user-suggested strategies like dynamic filter adjustment and query confidence indicators, addressing the third research question from Section 1.3 about refining inaccurate results. These suggestions align with Section 2.3 emphasis on usability and feedback in NL-to-SQL systems, enhancing transparency and trust. However, our study lacked quantitative metrics to assess filter interaction rigorously, such as a counter, tracking filter usage frequency or duration. This omission, stemming from the study design in Section 3, limits our ability to statistically correlate filter reliance with satisfaction or adaptation, weakening claims about behavioral shifts (Section 4.1). Future work should integrate such metrics to provide a data-driven basis for evaluating the AIs potential to supplant filters, as proposed in Section 5.4.

5.5 Additional Limitations

Beyond the aforementioned biases and limitations, additional weaknesses emerged. The small sample size (n=13), acknowledged in Section 4.1, precludes statistical significance testing, rendering our findings exploratory rather than conclusive. The choice of dogs as a product category (Section 3), while practical, may not generalize to more complex domains like electronics, where interdependent attributes (Section 1.2) could challenge the AIs query generation accuracy. Furthermore, the side-by-side interface design (Section 3.1) may have introduced a contrast effect, where the AIs results appeared more favorable simply due to the static systems visible shortcomings, a potential bias not addressed in our analysis.

6 Conclusion

This study underscores the transformative potential of AI-driven NL-to-SQL systems in e-commerce search, with users favoring their flexibility—such as synonym recognition (Section 4)—and reporting higher satisfaction (mean 1.69, SD

0.48) than with static templates (mean 3.62, SD 0.51). Practical deployment is achievable through our proposed blacklist configuration (Section 4.3), which could effectively mitigate security vulnerabilities.

Yet, limitations persist. The participant pool, dominated by tech-savvy computer science students with only one female (Section 4.1), calls for broader demographic validation. Likewise, the basic static template (Section 3.1) and rudimentary Claude API prompt (Section 4.3) reflect time constraints that hindered refinement; additional time could have yielded a more advanced prompt to boost performance further [7]. Incorporating filter-interaction metrics and testing across complex domains like electronics (Section 1.2), as suggested in Section 5.4, would strengthen these findings. Overcoming these technical, methodological, and usability challenges is essential to fulfilling the vision outlined in Section 1: a seamless, intuitive search experience tailored to diverse user needs, aimed to reshape human-computer interaction in e-commerce and potentially beyond.

7 Acknowledgement

We wish to express our sincere gratitude towards our lecturer, Prof. Hendrik Heuer, at the Bergische Universität Wuppertal, for his insightful and valuable feedback provided following our lectures and during the final presentation of our research topic.

A Appendix

Prior to conducting the user study, we administered a set of preliminary questions and Likert-scaled statements to participants, as referenced in Section 3.2, to assess their prior experiences and perceptions. These were divided into pre- and post-study phases as follows:

Pre-Study Questions and Statements

- Have you encountered challenges with search queries on e-commerce platforms such as Amazon when seeking specific products? If so, what were the consequences of these difficulties?
- Were there instances where you identified missing filter options on a website that could have improved your search experience? If yes, which specific filters did you desire, and how did their absence impact your overall interaction with the platform?
- 'I frequently utilize the filter options provided on websites.' (Likert scale: 1 = very frequently, 2 = frequently, 3 = neutral, 4 = infrequently, 5 = very infrequently)
- When searching for a product, do you tend to formulate generic or specific search queries? Do you employ natural language phrasing, or do you prefer concise keyword-based inputs?

Post-Study Questions and Statements

- Having interacted with our demo, do you consider the provided filter mechanisms redundant, or should they be retained as a complementary feature?
- Did you find the filter functionality adequate, or did you frequently modify your search query, aware of the underlying AI system?
- Did you find it more intuitive to refine your search query for improved results, or did you still prefer relying on filter options?
- 'The search results generated by the AI system met my expectations.' (Likert scale: 1 = strongly agree, 2 = agree, 3 = neutral, 4 = disagree, 5 = strongly disagree)

- ’The search results produced by the conventional method met my expectations.’ (Likert scale, as above)
- ’I perceived the AI system as capable of accurately interpreting complex search queries.’ (Likert scale, as above)
- Do you trust the AI-generated search results? If not, please elaborate on the reasons for your skepticism.
- ’I could envision forgoing certain filter options on a website if an AI system capable of processing natural language were implemented.’ (Likert scale, as above)
- ’If my initial search query yielded unsatisfactory results, I found it straightforward to revise or refine it.’ (Likert scale, as above)
- ’It was easy for me to assess whether the retrieved search results aligned with my intended query.’ (Likert scale, as above)
- Would you favor the inclusion of a mechanism to adjust or correct the AI-suggested search results? If so, could you provide a specific example of such a feature?
- ’The search capabilities offered by the AI system were transparent and comprehensible to me.’ (Likert scale, as above)
- Would you prefer this AI-driven search approach on a live website, and would you recommend it to friends or colleagues?

References

[1] 2023. Generate SQL Queries in Seconds for Free - SQLAI.ai. https://www.sqlai.ai

[2] 2024. meta-llama/Llama-3.3-70B-Instruct · Hugging Face. https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

[3] 2025. microsoft/phi-4 · Hugging Face. https://huggingface.co/microsoft/phi-4

[4] Henry Kautz Ana-Maria Popescu, Oren Etzioni. 2003. *Towards a Theory of Natural Language Interfaces to Databases.* https://dl.acm.org/doi/10.1145/604045.604070

[5] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch. 1995. Natural Language Interfaces to Databases - An Introduction. https://doi.org/10.48550/arXiv.cmp-lg/9503016 arXiv:cmp-lg/9503016.

[6] edogs. 2017. Hundemarkt: Hunde online finden - edogs.de. https://www.edogs.de/

[7] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation. https://doi.org/10.48550/arXiv.2308.15363 arXiv:2308.15363 [cs].

[8] Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. 1961. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference on - IRE-AIEE-ACM ’61 (Western)*. ACM Press, Los Angeles, California, 219. https://doi.org/10.1145/1460690.1460714

[9] Parssa Jashnieh. 2025. ParssaJ/HCI-AI. https://github.com/ParssaJ/HCI-AI original-date: 2024-10-09T15:55:13Z.

[10] Saurav Manchanda, Mohit Sharma, and George Karypis. 2019. Intent term selection and refinement in e-commerce queries. https://doi.org/10.48550/arXiv.1908.08564 arXiv:1908.08564 [cs].

[11] OWASP Foundation. 2024. *OWASP Top Ten | OWASP Foundation.* https://owasp.org/www-project-top-ten/

[12] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. https://doi.org/10.48550/arXiv.1910.10683 arXiv:1910.10683 [cs].

[13] Andrei Sobo, Awes Mubarak, Almas Baimagambetov, and Nikolaos Polatidis. 2025. Evaluating LLMs for Code Generation in HRI: A Comparative Study of ChatGPT, Gemini, and Claude. *Applied Artificial Intelligence* 39, 1 (Dec. 2025), 2439610. https://doi.org/10.1080/08839514.2024.2439610

[14] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2021. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. https://doi.org/10.48550/arXiv.1911.04942 arXiv:1911.04942 [cs].

[15] William A Woods. 1973. Progress in natural language understanding: an application to lunar geology. In *Proceedings of the June 4-8, 1973, national computer conference and exposition*. ACM, 441–450.

[16] Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. https://doi.org/10.48550/arXiv.1709.00103 arXiv:1709.00103 [cs].