Analyzing Common Student Errors in SQL Query Formulation to Enhance Learning Support

Davide Ponzini[1,2], Abdolhamid Livani[1], Giovanna Guerrini[1], Barbara Catania[1] and Mauro Coccoli[1]

[1]University of Genoa, Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, Genoa, Italy
[2]University of Genoa, Dipartimento di Lingue e Culture Moderne, Genoa, Italy

Abstract

Learning SQL is challenging for many undergraduate students, leading to recurring errors during query formulation. This paper systematically analyzes common SQL errors made by undergraduate students enrolled in Database courses at the University of Genoa during the 2023-24 academic year. Employing a comprehensive taxonomy, we evaluated 561 student queries collected from written exams, laboratory assignments, and unsupervised contexts. Results reveal that syntax errors are the most prevalent, especially in exam settings where queries cannot be executed. Logical errors and complications were also frequently identified, underscoring issues not only with syntax but also with logical reasoning. These insights highlight the need for pedagogical strategies that balance syntactic mastery and logical problem-solving skills. The paper concludes by proposing directions for enhancing learning support through automated error detection and personalized, AI-driven tutoring tools.

Keywords

Data Education, learning SQL, SQL misconceptions, educational data analysis, computer-assisted learning.

1. Introduction

In tertiary education, relational databases are an integral part of many undergraduate and postgraduate degree programmes in computer science, computer engineering, data science, business informatics and related subjects. The practical use of the Structured Query Language (SQL) is part of introductory relational database courses in such programmes, as SQL is the standard language for manipulating relational databases. Thus, students are asked to produce simple queries and, in some cases, to develop more significant projects manipulating data in SQL.

Despite its importance, many students find SQL difficult, and the queries they produce contain repeated errors that reflect common misunderstandings. These difficulties include grasping the ideas of relational databases, understanding difficult query structures, and applying theoretical ideas to real-world problems. One of the main barriers to learning SQL is the transfer of knowledge acquired in other contexts, such as mathematics and programming [1]. Students tend to apply familiar thinking patterns and language structures to SQL, and errors can also result from previous experience with other programming languages. In addition to syntax errors, errors can also occur at the semantic level, revealing an incomplete mental model of SQL [1]. In some cases, students see the database management system as a *black box* without understanding its underlying principles. Typical concepts that are prone to error are `JOIN` clauses, `GROUP BY`, sub-queries and self-joins [1, 2, 3, 4].

Since the development of targeted teaching tactics relies on a deep analysis and understanding of common mistakes, some approaches have been proposed in the literature aiming on one side at categorizing common errors in taxonomies [5] and on the other at identifying the most common misconceptions [1].

In this paper, we analyze the mistakes that our students make when learning SQL and categorize them according to the most comprehensive established taxonomy [5]. The reference target consists of second

SEBD 2025: 33rd Symposium on Advanced Database Systems, June 16–19, 2025, Ischia, Italy
davide.ponzini@edu.unige.it (D. Ponzini); s5437973@studenti.unige.it (A. Livani); giovanna.guerrini@unige.it (G. Guerrini); barbara.catania@unige.it (B. Catania); mauro.coccoli@unige.it (M. Coccoli)
0009-0006-4282-9652 (D. Ponzini); 0000-0001-9125-9867 (G. Guerrini); 0000-0002-6443-169X0 (B. Catania); 0000-0001-5802-138X (M. Coccoli)

year Bachelor students enrolled in database courses in the Computer Science and Computer Engineering Bachelor Degrees at the University of Genova in a.y. 2023-24. The research seeks trends in errors made by students, to gain insights about the areas that can be boosted to improve students learning. The classification involved 561 queries collected from 27 different information requests (i.e., request specifications in natural language), formulated in different settings by a total of 81 students/teams. Analysis of errors across different contexts and student groups highlights common difficulties in query construction. Syntax errors are the most common, accounting for almost half of all errors, in settings in which the students cannot execute the queries, highlighting the need to reinforce syntax rules and debugging skills. Logical errors are also common, suggesting that students struggle not only with syntax, but also with query logic, optimization, and interpreting the constraints in the request.

Although the numbers of the study are significant and allow for meaningful initial insights, automated support is needed to support high numbers of students throughout the courses and to avoid the risk of errors inherent in manual activities. The paper also discusses how the classification can be exploited and incorporated in other tools currently under development for assisting and enhancing student learning.

The remainder of the paper is organized as follows. After discussing related work in Section 2, the methodology adopted for the study is presented in Section 3. Results are presented and discussed in Sections 4 and 5, respectively. The limitations and ongoing directions to enhance learning support are discussed in Section 6 while Section 7 concludes the paper.

2. Related Work

In this section, we first review the most relevant approaches related to the analysis of typical errors and misconceptions and then briefly survey tools providing automatic feedback or proposing personalized exercises and learning paths.

2.1. Analysis of SQL Query Errors

Learning SQL poses challenges for students, involving both syntax and semantics[6]. To effectively address these issues, researchers emphasize the need for structured error categorization. Frameworks like those proposed by Ahadi et al. [7] and Taipalus et al. [5] provide taxonomies that help identify, analyze, and correct common student errors. This categorization aids in designing targeted instructional materials and feedback strategies.

Understanding the root causes of SQL errors is equally important [3]. Errors often stem from misconceptions, lack of foundational knowledge, or cognitive mismatches due to SQL's declarative nature. Students with experience in procedural programming or mathematics may struggle to adapt to SQL's logic and structure [8]. Misconceptions also arise from linguistic transfer, such as misusing keywords that resemble natural language or programming syntax [9].

Research [2, 1, 4] highlights frequent errors in subqueries, GROUP BY, and JOIN clauses, as well as improper use of primary keys and misunderstanding of DISTINCT. Additional issues include using incorrect operators (e.g., == instead of =), failing to apply self-joins, or confusing keyword syntax. Think-aloud studies [1] and expert analyses [10] further underline the importance of addressing these conceptual gaps to support effective SQL learning.

2.2. Automatic Feedbacks to SQL Queries and Personalization

Several approaches and prototypes have been designed to provide automatic feedback and correction for SQL teaching and learning. The tools aim to enhance learning outcomes by identifying errors, offering constructive feedback, and supporting self-guided improvement [11]. Many approaches target automatic correction and grading of SQL queries. The support ranges from the integration of CodeRunner in a Learning Management System like Moodle for providing immediate feedback by executing the query formulated by the student [12], to automatic grading systems [13, 14] and automatic correction based on Large Language Models [15].

The full potential of such tools in enhancing learning can only be leveraged if they extend beyond grading efficiency by also providing tutoring capabilities to the students. Hint generation is, for instance, the focus of [16, 17]. In [18] fine-tuned GPT models are used to detect and provide feedback on semantic errors in SQL queries. Another possible use of generative AI explored in [19] is for the automatic exercise generation.

3. Methodology

In this study, a systematic analysis of SQL query errors produced by students is carried out. The student-generated SQL queries were collected from the following sources:

- Written exams from the Databases courses in Computer Science and Computer Engineering[1]. In this context, students were not allowed to execute any queries or use any tools to provide support.
- Laboratory assignments specifically designed for the Database course in Computer Science [20]. In this setting, students had the opportunity to execute queries, observe the corresponding results, and compare them with the expected answers, thus allowing for iterative refinement.
- The queries described in Section 4 of [21] (referred to as the Miedema dataset in the following), which took place in an unsupervised environment wherein students autonomously decide whether or not to execute their queries. However, in this setting, no reference results were provided to students for comparison. We included this dataset since we aimed at comparing the errors detected by us with the ones presented in the thesis.

Throughout the paper, we refer to the laboratory assignments and the Miedema dataset collectively as interactive assignments. In both cases, students had the possibility to execute their queries and observe the corresponding outputs in real time. This interactive nature distinguishes these environments from written exams, where students are required to write queries without any execution or feedback. Despite this similarity, it is worth noting that the level of guidance differs: laboratory sessions are supervised and provide students with expected results for comparison, while the Miedema dataset was collected in an unsupervised context without reference solutions.

Each collected query was subjected to a manual review process, which entailed testing the query execution against the corresponding database to ascertain its correctness and identify any errors. Queries were evaluated based on their ability to execute without raising errors, to produce the expected result on a reference dataset, and their alignment with the requests outlined in the query specification. The manual analysis entailed evaluating each query for specific errors by comparing the student queries against the expected correct solutions.

Errors were identified and categorized following the established and comprehensive framework by Taipalus [5], distinguishing four main categories:

- **Syntax errors:** queries that are not executable due to incorrect SQL grammar.
- **Semantic errors:** queries that can be executed but are incorrect, irrespective of the specific query request.
- **Logical errors:** queries that can be executed but are not aligned with the specific query request.
- **Complications:** queries that accurately align with the specified query request, yet exhibit unnecessary complexity. Note that this does not concern optimization, but rather avoidable design choices, such as using unnecessary tables or overly elaborate constructs.

Additionally, queries were categorized based on the complexity of the request: **simple** queries involve basic selection, filtering, and limited joins; **medium complexity** queries introduce aggregation, sorting, or conditional logic; **hard** queries involve advanced SQL features such as complex joins, nested conditions, and multi-table aggregations.

[1]Note that the assignments of the written exams are different, as the corresponding syllabi are. Thus, the aim of this study is by no means to compare the two groups of students.

	Computer Engineering Exam	Computer Science Exam	Computer Science Laboratories	Miedema dataset [21]
Student Group (Degree)	Computer Engineering	Computer Science	Computer Science	Computer Science
Number of Requests	3	8	9	7
Number of Subjects (Students/Teams)	12	25	23	21
Language	Italian	Italian	Italian	English
Data Collection Method	Written Exam	Written Exam	Online Submission	Online Submission
Answer Time	Limited	Limited	Not Limited	Not Limited
Sample Data Provided	No	No	Yes	Yes
Work in Teams	No	No	Yes	No*
Possibility of executing the Queries	No	No	Yes	Yes
Expected Results Available	No	No	Yes	No

* The queries were formulated in an unsupervised context, precluding the possibility of determining whether students worked alone.

Table 1
Query collection sources and modalities.

		Errors detected				
Source	**Queries**	**Syntax**	**Semantic**	**Logical**	**Complications**	**Correct**
Computer Engineering Exam	29	45	8	19	10	0
Computer Science Exam	183	132	6	14	7	37
Computer Science Laboratories	209	17	4	83	74	85
Miedema dataset	140	30	35	25	27	66
Total	561	224	53	141	118	188

Table 2
Number of queries and errors detected, for each category, in each source

This structured approach allowed us to systematically record, categorize, and analyze errors, facilitating detailed statistical analysis and deeper insights into the error patterns and common misconceptions encountered by students. For a more thorough exposition on the modalities of data collection, refer to Table 1. Further details are presented in [22].

4. Results

We assigned 724 labels across 561 queries, identifying a total of 536 SQL errors, as well as 188 correct queries. A detailed breakdown of the error labels across the four datasets is presented in Table 2.

The analysis revealed that 66% of the queries contained at least one error or complication. Specifically, 37% of all queries exhibited at least one syntax error, 9% exhibited at least one semantic error, 22% exhibited at least one logical error, and 18% exhibited at least one complication, as illustrated in the *Overall* category of Figure 2. It is important to note that these categories are not mutually exclusive; a single query may exhibit multiple types of errors.

Figure 1 shows the error distributions overall and by each context, comparing the different modalities and student groups. Overall, syntax errors are the most prevalent type of error, followed by logical errors and complications. Semantic errors are the least prevalent type of error. The prevalent types of errors across different contexts are shown in Table 3. The most prevalent errors and complications exhibited by both student groups are analogous to the overall errors previously delineated. A finer-grained analysis of the most common errors and misconceptions detected in these queries is presented in [22].

As illustrated in Figure 2, the error type distribution for each query is compared across different

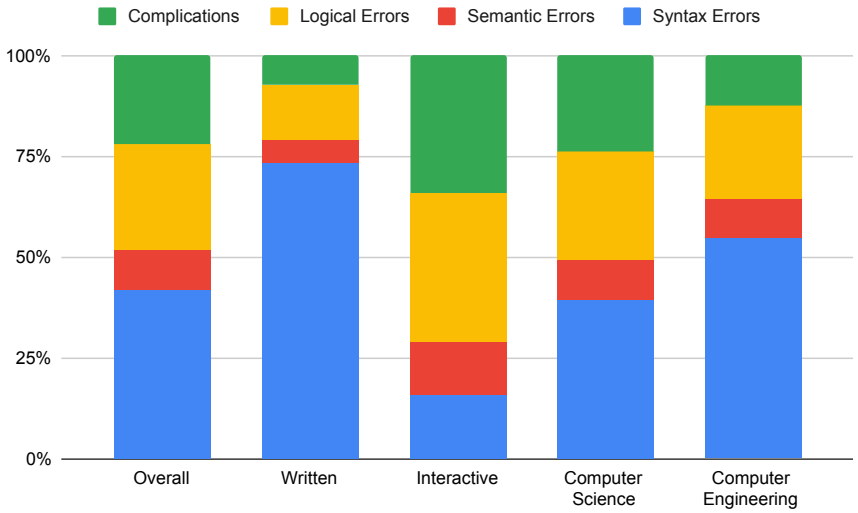

Figure 1: Overall error distribution and distributions by context, comparing different modalities and student groups. Only queries containing at least one error or misconceptions are considered.

Context	Most Common Errors	Most Common Complications
Overall	SYN: referencing a non-existing schema (34) LOG: superfluous columns in SELECT (33) LOG: omission of requested columns (30) SYN: referencing a non-existing column (29) SEM: returning many duplicate rows (27) SYN: use of non-standard keywords (21) SYN: use of non-standard operators (20) SEM: tautological or inconsistent expression (19) LOG: missing logical expression (19) SYN: invoking a non-existing function (14)	Join with unnecessary tables (48) Unnecessary DISTINCT (16)
Written	SYN: referencing a non-existing schema (34) SYN: referencing a non-existing column (26) SYN: use of non-standard keywords (21) SYN: use of non-standard operators (20) LOG: omission of requested columns (19)	Join with unnecessary tables (7)
Interactive	SEM: returning too many duplicate rows (27) LOG: superfluous columns in SELECT (26) LOG: missing logical expression (18) LOG: extraneous logical expression (12) SYN: confusing the logic of keywords (12)	Join with unnecessary tables (41) Unnecessary DISTINCT (14)

Table 3
Prevalent types of errors across different contexts. The numbers in parentheses represent the number of instances of each error.

modalities and student groups. It is important to note that a single query can exhibit multiple error types. In instances where a query exhibits multiple errors of the same type, it is considered as a single occurrence. Correct queries are defined as those that do not present any error or complication.

Overall, the majority of queries exhibit at least one syntax error. After excluding these errors, the majority of the remaining queries demonstrate no errors or complications. Logical errors are also frequent, followed by complications. The least prevalent error type is semantic errors. During written exams most queries contain syntax errors, with a mean value of 0.83 syntax errors per query. At the same time they present few semantic or logical errors. Complications are uncommon, with a mean value of 0.08 complications per query. These findings are supported by the data presented in Table 2.

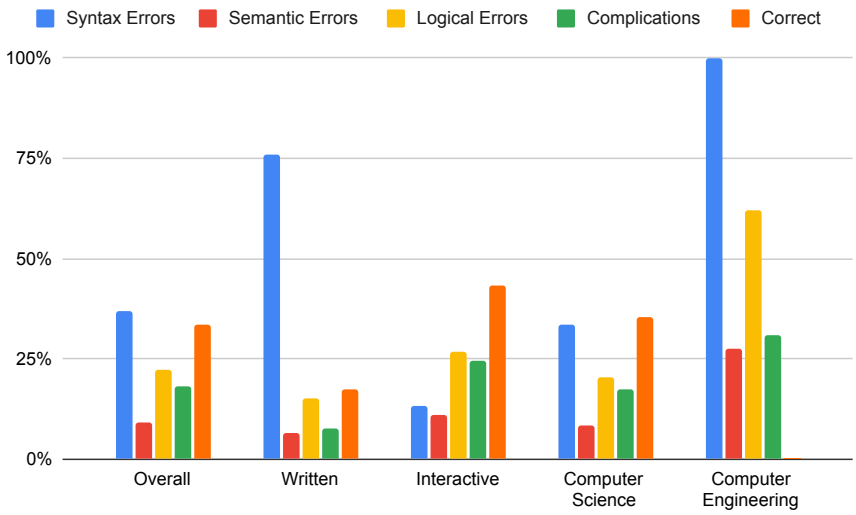

Figure 2: Error Type Distribution per Query. Each bar shows the percentage of queries with at least one error of that type. Multiple error types can co-occur in the same query, but repeated instances of the same type are counted only once. Correct queries contain no errors or complications and are mutually exclusive from the others. Note that the error percentages can add up to more than 100% due to overlapping error types.

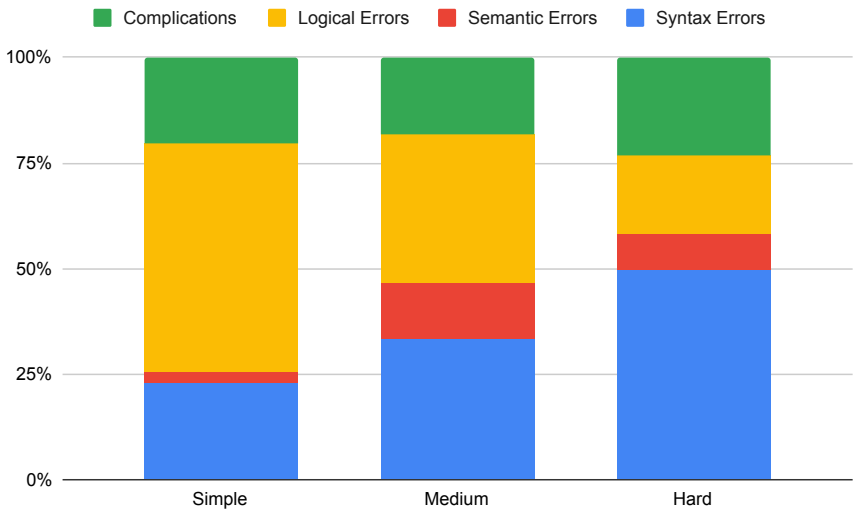

Figure 3: Error distributions by difficulty.

Interactive assignments present a different distribution: most of them are completely correct, followed by logical errors and complications. A small amount of queries presents syntax errors. The least common error type is semantic errors. Syntax, logical and semantic errors are more common in Computer Engineering students when compared to Computer Science students. Both student groups tend instead to write complications with a similar frequency. Interestingly, in the data we analyzed from Computer Engineering students, we were unable to find a query that did not present any error or complication. It is important to note that the queries and experimental settings differ between the two student groups. Furthermore, the Computer Engineering dataset is considerably smaller than the Computer Science one. Consequently, it is not feasible to draw any direct conclusions regarding their comparative performance.

Figure 3 analyzes error distribution by query difficulty. A close examination of the data reveals that syntax errors are more prevalent in complex queries, while logical errors are more prevalent in simpler queries. Semantic errors appear to be more frequent in medium-complexity queries. Complications are

present in all queries, regardless of their complexity.

5. Discussion

The analysis of SQL errors across various educational contexts and student groups provides significant insights into common issues students encounter when constructing database queries. The prevalence of errors identified across contexts underscores the challenges students face, particularly with syntax and logical reasoning in query construction.

Overall, syntax errors emerged as the most common type of error, representing nearly half of all errors. This highlights a fundamental issue students experience in mastering SQL syntax, suggesting a need for reinforcing syntax rules and debugging skills in teaching strategies. Logical errors and complications were also frequent, emphasizing that students not only struggle with basic syntax but also with query logic, optimization, and adherence to problem constraints.

When comparing written and interactive assignment modalities, a distinct pattern emerges. Written tests predominantly exhibited syntax errors, indicative of difficulties in recalling precise SQL syntax without interactive feedback. The high occurrence of referencing non-existent database objects suggests that students might benefit from additional practice in accurately memorizing and understanding database schemas. Conversely, interactive assignments showed fewer syntax errors but more logical errors and complications, suggesting that while interactive environments reduce syntactical mistakes, students might over-complicate solutions or misunderstand query requirements when immediate feedback is available. Therefore, teaching methodologies could integrate both written and interactive elements to balance syntax learning and logical problem-solving.

Analysis by query difficulty reveals additional trends. Complex queries correlated strongly with increased syntax errors, pointing to students' struggles with managing intricate query structures. Conversely, simpler queries saw increased logical errors, indicating complacency or misunderstandings about fundamental database query logic. Semantic errors were notably higher in queries of medium complexity, perhaps because intermediate queries involve nuanced schema understanding and data relationships without the explicit complexity of more challenging problems. Educational approaches could thus differentiate between levels of query difficulty, tailoring instructions and practice exercises to student needs at each complexity level.

The presence of complications across all difficulty levels suggests a pervasive inclination toward unnecessarily complex query formulations. This underscores a pedagogical opportunity to reinforce principles of efficient and simplified query design, potentially through examples contrasting complicated and straightforward solutions.

With regard to Miedema's queries, we found only a partial correspondence with the problems highlighted in [1, 21], perhaps also due to the different settings in which the queries were formulated.

In conclusion, our findings advocate for pedagogical adjustments, including enhanced syntax training, targeted logical reasoning exercises, and focused interventions based on student backgrounds and assignment modalities. Such targeted educational strategies promise to effectively address the identified challenges, thereby improving overall proficiency in SQL query formulation among students.

6. Limitations & Ongoing Research

This study presents some limitations that must be acknowledged. The primary limitation is the restricted amount of data, particularly from Computer Engineering students, which impacts the generalizability of our findings. The dataset from Computer Engineering students was notably smaller than that from Computer Science students, potentially skewing the analysis toward issues predominant within a single group. Additionally, the necessity of manually examining each query introduces potential for human error or bias in error categorization. Manual evaluation, while thorough, is time-intensive and not scalable to larger datasets without introducing considerable resource constraints.

Based on the findings of this study, future research focuses on the development and assessment of targeted educational interventions using innovative technologies. Currently, we are developing three distinct tools (further details can be found in [23]):

- **Automatic Error Identification and Categorization Tool:** This tool automatically analyzes SQL queries to detect and classify errors according to established taxonomies. This tool can be used to analyze more in depth students' weaknesses and to allow educators to rapidly pinpoint and address students' specific areas of weakness.
- **Generative AI-based Assignment Creator:** Leveraging generative AI, this tool generates tailored SQL assignments reflecting students' personal interests and targeting common error patterns. This personalized approach aims to improve student engagement and targeted skill development.
- **Interactive AI-based SQL Tutor:** An AI-driven interactive tutor designed to provide personalized guidance and immediate feedback on SQL queries. The tutor interacts dynamically with students, addressing misconceptions and facilitating a deeper understanding of SQL concepts.

These tools are anticipated to significantly enhance the precision of educational interventions, enabling educators to more effectively address specific SQL learning difficulties and misconceptions.

As per the analysis is concerned, it can be expanded to other degrees offering a database course, as well as to other universities and high schools, ideally on a common set of reference queries. Further refinements relate to the investigation of the impact of previous knowledge, both in terms of previous knowledge of SQL (e.g., students who have studied SQL during high school versus those who have not) and of experience with other programming languages, which can be different among students of different degrees.

7. Conclusions

This study analyses common SQL errors made by undergraduate students, revealing significant challenges with syntax and logic. Syntax errors were most common in exams, while interactive assignments highlighted logical errors. Computer Engineering students had higher error rates, probably due to less exposure to SQL. These findings highlight the need for tailored teaching approaches. To address these issues, we are developing AI-driven tools for error detection, personalized practice and real-time feedback to improve SQL learning and student proficiency.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT and DeepL in order to: Abstract drafting, Drafting content, Paraphrase and reword, Improve writing style, Grammar and spelling check. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] D. Miedema, E. Aivaloglou, G. Fletcher, Identifying SQL misconceptions of novices: Findings from a think-aloud study, ACM Inroads 13 (2022) 52–65.

[2] S. Brass, C. Goldberg, Semantic errors in SQL queries: A quite complete list, Journal of Systems and Software 79 (2006) 630–644.

[3] T. Taipalus, Explaining causes behind SQL query formulation errors, in: 2020 IEEE Frontiers in Education Conference (FIE), IEEE, 2020, pp. 1–9.

[4] K. Presler-Marshall, S. Heckman, K. Stolee, SQLRepair: Identifying and repairing mistakes in student-authored SQL queries, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), IEEE, 2021, pp. 199–210.

[5] T. Taipalus, M. Siponen, T. Vartiainen, Errors and complications in SQL query formulation, ACM Transactions on Computing Education (TOCE) 18 (2018) 1–29.
[6] P. Garner, J. Mariani, Learning SQL in steps, Learning 12 (2015) 23.
[7] A. Ahadi, J. Prior, V. Behbood, R. Lister, Students' semantic mistakes in writing seven different types of SQL queries, in: Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education, 2016, pp. 272–277.
[8] H. Al-Shuaily, SQL pattern design, development & evaluation of its efficacy, Ph.D. thesis, University of Glasgow, 2013.
[9] D. Traversaro, Insegnare SQL a chi non ha mai programmato: Analisi delle misconcenzioni, in: ITADINFO Secondo Convegno Italiano sulla Didattica dell'Informatica (In Italian), 2024.
[10] D. Miedema, G. Fletcher, E. Aivaloglou, Expert Perspectives on Student Errors in SQL, ACM Trans. Comput. Educ. 23 (2022).
[11] C. Kenny, C. Pahl, Automated tutoring for a database skills training environment, in: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, 2005, pp. 58–62.
[12] A. Wójtowicz, M. Prill, Relational Database Courses with CodeRunner in Moodle: Extending SQL Programming Assignments to Client-Server Database Engines, in: Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1, 2025, pp. 1239–1245.
[13] K. Manikani, R. Chapaneri, D. Shetty, D. Shah, SQL Autograder: Web-based LLM-powered Autograder for Assessment of SQL Queries, International Journal of Artificial Intelligence in Education (2025) 1–31.
[14] B. Chandra, B. Chawda, B. Kar, K. M. Reddy, S. Shah, S. Sudarshan, Data generation for testing and grading SQL queries, The VLDB Journal 24 (2015) 731–755.
[15] Z. Chen, S. Chen, M. White, R. Mooney, A. Payani, J. Srinivasa, Y. Su, H. Sun, Text-to-SQL error correction with language models of code, arXiv preprint arXiv:2305.13073 (2023).
[16] C. Kleiner, F. Heine, Enhancing Feedback Generation for Autograded SQL Statements to Improve Student Learning, in: Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1, ITiCSE 2024, Association for Computing Machinery, New York, NY, USA, 2024, p. 248–254.
[17] Y. Hu, A. Gilad, K. Stephens-Martinez, S. Roy, J. Yang, Qr-hint: Actionable hints towards correcting wrong sql queries, Proceedings of the ACM on Management of Data 2 (2024) 1–27.
[18] A. AlRabah, S. Yang, A. Alawini, Optimizing Database Query Learning: A Generative AI Approach for Semantic Error Feedback, in: ASEE Annual Conference and Exposition, Conference Proceedings, American Society for Engineering Education, 2024.
[19] W. Aerts, G. Fletcher, D. Miedema, A Feasibility Study on Automated SQL Exercise Generation with ChatGPT-3.5, in: Proceedings of the 3rd International Workshop on Data Systems Education: Bridging education practice with education research, 2024, pp. 13–19.
[20] B. Catania, G. Guerrini, D. Traversaro, Collaborative learning in an introductory database course: A study with think-pair-share and team peer review, in: Proceedings of the 1st International Workshop on Data Systems Education, DataEd '22, 2022, p. 60–66.
[21] D. E. Miedema, On learning SQL: Disentangling concepts in data systems education, 2024. URL: https://pure.tue.nl/ws/portalfiles/portal/314131184/20240112_Miedema_hf.pdf.
[22] A. Livani, Do the errors produced by generative AI in formulating queries reflect students' misconceptions in learning SQL?, Master's thesis, MSc in Computer Science, University of Genova, 2024. URL: https://unire.unige.it/bitstream/handle/123456789/10623/tesi31530643.pdf?sequence=1.
[23] D. Ponzini, B. Catania, G. Guerrini, Leveraging Frequent Errors, Generative AI, and Peer Collaboration to Enhance SQL Learning, Submitted for publication, 2025.