The Journal of Systems & Software 181 (2021) 111034

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Error messages in relational database management systems: A comparison of effectiveness, usefulness, and user confidence☆

Toni Taipalus [a,*], Hilkka Grahn [a], Hadi Ghanbari [b,c]

[a] Faculty of Information Technology, University of Jyvaskyla, P.O. Box 35, FI-40014, Jyvaskyla, Finland
[b] Aalto University School of Business, Ekonominaukio 1, 02510, Espoo, Finland
[c] FinEst Twins Smart City Center of Excellence, TalTech, Estonia

ARTICLE INFO

Article history:
Received 23 March 2021
Received in revised form 23 June 2021
Accepted 29 June 2021
Available online 12 July 2021

Keywords:
Structured query language (SQL)
Compiler
Error message
Database management system

ABSTRACT

The database and the database management system (DBMS) are two of the main components of any information system. Structured Query Language (SQL) is the most popular query language for retrieving data from the database, as well as for many other data management tasks. During system development and maintenance, software developers use a considerable amount of time to interpret compiler error messages. The quality of these error messages has been demonstrated to affect software development effectiveness, and correctly formulating queries and fixing them when needed is an important task for many software developers. In this study, we set out to investigate how participants ($N = 152$) experienced the qualities of error messages of four popular DBMSs in terms of error message effectiveness, perceived usefulness for finding and fixing errors, and error recovery confidence. Our results show differences between the DBMSs by three of the four metrics, and indicate a discrepancy between objective effectiveness and subjective usefulness. The results suggest that although error messages have perceived differences in terms of usefulness for finding and fixing errors, these differences may not necessarily result in differences in query fixing success rates.

1. Introduction

Ease of use and accessibility have been growing research topics, as information technology is becoming more and more inclusive, and even systems intended for expert use are reaching non-expert user bases (e.g., Gutiérrez et al., 2019; Espinosa et al., 2015). One branch of systems intended for experts is database management systems (DMBS), typically tightly connected with Structured Query Language (SQL) for managing database data. Database management systems are annually a multi-billion scale industry. DBMSs are one of the major enabling factors behind almost all information systems, and most of the popular modern DBMSs use SQL as their query language. Possibly due to SQL's popularity in the industry, the language has received ample attention in research (Taipalus and Seppänen, 2020; Lawal et al., 2016), and remains a topic in almost all information technology curricula guidelines in higher education (Topi et al., 2010; Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society, 2013; The Joint Task Force on Computing Curricula, 2015).

Consequently, also human factor considerations in the context of query languages have been studied in several lines of research in the 1970s and 1980s (Welty and Stemple, 1981; Reisner, 1977, 1981; Reisner et al., 1975). Rather unfortunately, however, more recently these research topics have received less and less attention from scholars. More specifically, query language usability in terms of SQL compiler error messages has not received scientific attention in the language's modern implementations. The timeliness in this regard is an important concern because, since the 1980s, SQL has emerged as the most popular query language, yet SQL is not the same language as it was 40 years ago (Taipalus and Seppänen, 2020). In the context of programming languages, programmers use a considerable amount of time on reading and interpreting error messages (Barik et al., 2017). Especially for novice developers, error messages are an important usability concern (Lee and Ko, 2011). Therefore, how these error messages are presented and formatted may help the user considerably in translating intent into programming language constructs (Kölling, 1999). However, a recent study (Becker et al., 2016) summarizes programming language error messages as "terse, confusing, too numerous, misleading, and sometimes misleadingly wrong", and concludes that some studies have shown that enhancing error messages is beneficial for error recovery, even though contradictory evidence has also been presented. For these reasons, it seems

☆ Editor: Kelly Blincoe.

* Corresponding author.
E-mail addresses: toni.taipalus@jyu.fi (T. Taipalus), hilkka.grahn@jyu.fi (H. Grahn), hadi.ghanbari@aalto.fi (H. Ghanbari).

https://doi.org/10.1016/j.jss.2021.111034

justified to argue that error message research should be extended from programming languages to query languages.

To the best of our knowledge, this study is the first attempt to bring together the vivid research fields of human–computer interaction and database management systems with the topic of SQL compiler error messages. We compare the error messages of the four most popular relational database management systems (MySQL, Oracle Database, PostgreSQL, and SQL Server) in terms of error message effectiveness, perceived usefulness for finding and fixing errors, and error message effects on error recovery confidence. Our research setting is a between-subjects comparison of responses from a total of 152 participants, and the results show statistically significant differences between error messages of different DBMSs in error message effectiveness and perceived error message usefulness for finding and fixing SQL errors. The results indicate, at least in the scope of our study, that *(i)* although there are differences between error messages of different DBMSs regarding perceived usefulness for error finding and fixing, these differences are less clearly reflected in successful error fixing. Furthermore, the results imply that *(ii)* PostgreSQL error messages are generally the most effective and considered the most useful, yet all differences are not statistically significant. These findings also denote that *(iii)* the differences between the DBMS error messages are likely concretized in aspects other than error fixing success rates, e.g., in time required to fix errors. Finally, *(iv)* we observed no differences between error message effects on error recovery confidence, which is a possible indication of uniformly positive or neutral effects of error messages to error recovery.

The rest of this study is structured as follows. In the next section, we discuss the theoretical background and prior works. In Section 3, we describe our research questions, the scope of our study, how our data were collected, and threats to validity. In Section 4, we present our results for each research question, and in Section 5 discuss practical implications of our results for research, industry, and education, as well as identified limitations. Section 6 concludes the study.

2. Theoretical background

2.1. SQL errors

Errors in SQL query formulation have usually been divided into either syntax and semantic errors (Smelcer, 1995; Ahadi et al., 2016b,a), or syntax, semantic and logical errors, with the addition of complications (Brass and Goldberg, 2006; Taipalus et al., 2018; Taipalus and Perälä, 2019). According to both categorizations, only syntax errors are recognized by the DBMS, and the latter division describes logical errors as closely related to the current task, or *data demand*, i.e., to what information the query is intended to retrieve. If a query does not correspond to the current data demand, it is considered logically incorrect (Taipalus et al., 2018). Queries with semantic errors, in turn, are characterized as being *always incorrect* (Brass and Goldberg, 2006), as queries that exhibit semantic errors can be deemed incorrect even without the knowledge of the data demand. Finally, complications are unnecessary elements that could be omitted from the query without effects on the result table.

As syntax errors are the only errors that output an error message instead of a result table, syntax errors are a natural point of interest in database management system usability research, as opposed to other types of errors. Although the SQL standard (ISO/IEC, 2016a,b) defines and describes how SQL operates, there is no single stand to what is a syntax error, as the standard leaves room for interpretation. As a simple example, the SQL standard describes that character literals are enclosed in single quotes, and character strings in double quotes, and, e.g., MySQL tolerates expressions such as `column_1 = "a"`, yet PostgreSQL returns a syntax error caused by the double instead of single quotes. Consequently, different DBMSs tolerate certain SQL constructs like implicit type conversions to different degrees. Partially as a result of different implementations of syntax checks, different DBMSs have different types of syntax errors, and consequently output different error messages. For example, Microsoft SQL Server categorizes syntax errors differently than Oracle Database (Randolph, 2003). In addition to studying user errors in the context of a particular DBMS, at least one SQL standard based error categorization has been attempted (Taipalus et al., 2018).

Several studies discuss the wide variety of SQL syntax errors (Smelcer, 1995; Ahadi et al., 2016a) such as unmatched parentheses, typographic errors in SQL keywords and database object names, and ill-placed aggregate functions. Some studies have shown that syntax errors are the most frequent errors (Taipalus et al., 2018; Poulsen et al., 2020), yet they are usually fixed (Taipalus and Perälä, 2019), as opposed to logical errors which are both frequent and relatively difficult to spot and fix (Taipalus and Perälä, 2019). Ample scientific attention has also been given to the causes behind query formulation errors, such as cognitive factors (Smelcer, 1995; Taipalus, 2020b; Shin, 2020; Mills et al., 2020), the effects of data demand ambiguity (Borthick et al., 2001a; Casterella and Vijayasarathy, 2019), database normal form (Borthick et al., 2001b; Bowen et al., 2004), and database structure complexity (Taipalus, 2020a). These studies have focused on data retrieval, and there is a solid scientific footing to build upon, as opposed to, for example other Data Manipulation Language statements such as UPDATE or DELETE. Then again, data retrieval statements can be repurposed as updates or deletes with relative ease.

2.2. Error messages

The DBMS's SQL compiler generates the error message. From the perspective of an error message relevant to this study – as opposed to, e.g., DBMS errors resulting from something other than erroneous user-written queries – there are several steps that each have potential triggers for syntax errors. Although different DBMSs have different implementations, according to a general description (Hellerstein et al., 2007), the SQL compiler contains a parser that checks SQL keywords, fully qualifies table references in the query, and checks them along with attribute references against the system catalog. This uncovers, e.g., potential misspellings in database object names and SQL keywords. The parser also checks if the query follows implemented SQL logic and set configuration, such as full grouping. Next, the query rewriter simplifies and evaluates arithmetic, if applicable, and the query optimizer generates an execution plan if one is not found. Finally, the plan executor, working in tandem with the storage manager, fetches the data. In summary, the parser is the most relevant DBMS component in the SQL compiler in the scope of this study.

Despite the age of SQL and some DBMSs, SQL compilers have remained in the sidelines of research concerning error messages. Several SQL debuggers, some with demonstrated positive effects have been proposed, e.g., Habitat (Dietrich and Grust, 2015; Grust et al., 2011; Grust and Rittinger, 2013) and I-REX (Miao et al., 2020), yet it remains unclear how widely these, and other third party debuggers are used in industry and education. In fact, we could not find a single peer-reviewed scientific publication focusing specifically on query language compiler error messages, and therefore we discuss programming language error message research in this subsection. Although SQL and programming languages share similarities, and programming language error messages are a tangential line of research, it is worth

noting that query and programming languages have some fundamental differences, e.g., declarative versus imperative nature, the complexity of the units of execution, and the underlying purpose (Taipalus, 2019).

Programming language compiler error messages have received ample and increasing attention in scientific literature. A recent study found hundreds of scientific articles on the topic, with 107 proposals on enhancing error messages (Becker et al., 2019), e.g., providing context, reducing cognitive load, and showing programmers examples of similar errors. The consensus view appears to be that current error messages are confusing in different ways (Denny et al., 2011; Becker, 2016; Becker et al., 2018), terse programming language compiler error messages have been considered difficult for finding and fixing errors (Denny et al., 2011; Schorsch, 1995; McCall and Kolling, 2014), and the problem of confusing error messages also seems a wider problem (Shneiderman, 1982). On the other hand, while some techniques such as error message spacing, colors (Dong and Khandwala, 2019), and syntax highlighting (Hannebauer et al., 2018) have been shown to increase error message effectiveness, the evidence has sometimes been anecdotal (Becker et al., 2019), or even controversial (Sarkar, 2015; Denny et al., 2014; Pettit et al., 2017). Finally, the challenges in enhancing error messages are also complex, and although solutions are proposed, it may be unclear if these solutions can be or should be implemented. Some of the recognized problems are related to programming language performance, error mapping between original code and the version of the code being compiled, and the distance between the source of the error and the point in which the error is realized (Becker et al., 2019).

2.3. Error recovery

Error recovery refers to the process of three phases, namely *detecting*, *explaining*, and *correcting* an error (Zapf and Reason, 1994; van der Schaaf, 1995). The scientific field around error recovery is vivid, yet mainly studied outside the domain of information technology (e.g., Seifert and Hutchins, 1992; van der Schaaf, 1995). In terms of SQL compiler syntax error messages, the first phase is initiated (at the latest) by the compiler, if the query writer is unable to fix the error before sending the SQL statement to the compiler. The first phase refers solely to the notion of realization of an error, without knowledge of the error's location, nature, or cause. This first phase is the most studied aspect of error recovery, as it initiates the error recovery process (Kontogiannis, 1999). In the second phase, the error is located, either solely by the end-user, or by the end-user with the help of the DBMS error message, and the cause of the error is, at least implicitly, explained or speculated. Finally, in the third phase, an attempt is made to fix the error. From an educational point of view, confidence in completing tasks is an important concern, especially when a task is not mandatory. A recent study found that subjective confidence influences human decision making in choosing which tasks to pursue (Carlebach and Yeung, 2020).

From the above considerations, three intuitive metrics for error recovery may be drawn. *Error finding* refers to the process of locating the erroneous part or parts of the query. When the task's complexity increases, it is natural that finding the error becomes more and more difficult, although the difficulty of *error fixing* may remain the same. In the context of SQL syntax errors, the query may span several lines, making error finding a more difficult task, yet fixing the error may be a trivial task once the erroneous part is found. In addition to subjective indicators, error fixing can be measured objectively, i.e., with success rates for fixing errors. As discussed in the previous sections, however, many error messages are perceived confusing. A confusing error message may decrease *user confidence in error recovery* even though the user felt confident in error recovery after detecting the error, but before reading the error message. Increased confidence has been shown to be related to increased success (Carlebach and Yeung, 2020), and objective measures and subjective confidence have usually been shown to have a strong correlation (Martino et al., 2012; Fleming et al., 2010). However, some studies have argued for breaking the confound between confidence and success (Desender et al., 2018), and a discrepancy between confidence and success may be an indicator of a problem in the recovery process. Finally, studying the effects of error messages on error recovery may give indications on how different types of error messages may influence SQL education.

3. Research setting

3.1. Research questions

Our research questions measure both objective and subjective indicators of syntax error message qualities discussed in Section 2. First, we measure success rates for fixing erroneous queries based on participant skill and DBMS error messages. Second, we measure participants' perception of error message usefulness in terms of finding the erroneous part of the query, fixing the error, and participant confidence in error recovery.

RQ1: Which error messages are effective for fixing erroneous queries? To answer this question, we investigate success rates [0..1] for fixing erroneous SQL queries based on DBMS error messages. Answers are presented in Section 4.2.

RQ2: Which error messages are perceived useful for finding the error? To answer this question, we analyze participants' subjective experiences on how useful different DBMS error messages are in pinpointing the erroneous part in the SQL query using a five point Likert scale. Answers are presented in Section 4.3.

RQ3: Which error messages are perceived useful for fixing the error? To answer this question, we analyze subjective experiences on how useful different DBMS error messages are for fixing the error using a five point Likert scale. Answers are presented in Section 4.4.

RQ4: Which error messages are perceived to increase confidence in error recovery? To answer this question, we examine subjective experiences on how different DBMS error messages affect user confidence in error recovery using a five point Likert scale. As this study focuses on the two latter error recovery phases, and particularly on the output from different DBMSs, we deemed worth studying how the error messages are perceived in relation to user expectations and perceptions on the error. Answers are presented in Section 4.5.

3.2. Study scope

As DBMSs are numerous, and our pool of potential study participants was limited, we chose the four most popular relational database management systems for our study. As DBMS popularity is a rather ambiguous concept, we utilized *DB-Engines Ranking*[1] in the selection process. According to the website, the popularity ranking is based, among other metrics, on frequencies of technical discussions, mentions in social media platforms, and

[1] https://db-engines.com/en/ranking.

the number of listings on professional networks. The four most popular DBMSs chosen for this study were Oracle Database (19c Enterprise Edition 19.5.0.0.0), MySQL (8.0.12 with InnoDB storage engine), Microsoft SQL Server (2019 Developer), and PostgreSQL (12.1), respectively. At the time of testing, these versions were the most recent available, excluding beta versions and release candidates.

As discussed in Section 2.1, it is typical that only queries with syntax errors result in an error message. However, DBMSs categorize syntax errors differently, return at least slightly different error messages, and sometimes tolerate errors to different degrees (Taipalus et al., 2018). In contrast, other types of errors typically behave similarly in different DBMSs, i.e., they invoke no DBMS messages to the query writer. For these reasons, we chose to focus solely on syntax errors. Furthermore, although SQL contains several sublanguages, each with several types of SQL statements, we chose to study solely syntax errors in SELECT statements, as SELECT has received the most scientific attention in tangential research concerning SQL errors (Taipalus and Perälä, 2019; Taipalus and Seppänen, 2020).

Finally, even in the scope of SELECT statements, the number of different syntax errors are measured in dozens, and the number of different syntax errors is dictated by the DBMS used. Because studying all syntax errors in the four chosen DBMSs was not feasible, we chose to focus on sixteen most frequent syntax errors identified in a previous study (Taipalus et al., 2018). The study categorized different SQL errors in a DBMS independent fashion, rather than allowing a single DBMS categorize the errors. This approach allows the categorization to be utilized in different DBMSs. These sixteen most common syntax errors are summarized in Table 1, and form the basis for our sixteen tests described in Section 3.3 and Appendix A.3. The Appendix A.3 also contains concrete examples of each syntax error.

3.3. Data collection

We selected the study participants among second, third, and fourth year university students from software engineering, computer science, and information systems science fields. The participants took part in the study by answering an online form (cf. Appendix A). Prior to participation, the participants were given a formal training of approximately 30 h over the course of four weeks in relational theory and SQL. The potential participants were randomly divided into four different database management system groups (MySQL, Oracle Database, PostgreSQL, and SQL Server), and asked to answer the form. In other words, our study design is a between-subjects comparison of four groups. By answering the form, participants were given course points towards a better grade. Answering was not mandatory, and participating in the study was not mandatory even though a participant chose to answer the form.

In the form, the participants were first shown a data privacy statement, which was followed by a question whether they chose to participate in the study, or merely answer the form. Out of the 175 students who answered, 152 (87%) chose to participate. Next, participants were shown four control questions testing their skill in fixing SQL errors. These control questions were the same for every participant, regardless of the DBMS group they were assigned to. Next, a participant was shown the database schema, a data demand, a corresponding erroneous SQL query and the error message from the DBMS respective to the DBMS group they were assigned to, and the answer form consisting of a text box to write the fixed query, and Likert scale questions Appendix A. After the participant had answered one test testing a particular syntax error (cf. Table 1) and the associated error message, the next test was shown, until the participant had answered all sixteen tests. The order of the tests was randomized for each participant. The test could be paused and continued later or stopped altogether. In the latter case, we would have omitted the participant's answers from this study, yet none of the participants stopped answering. The participants could use any materials during the tests, yet the online form gave no feedback on the correctness of the answers regarding, e.g., fixing erroneous SQL statements. After data collection, the first author analyzed the queries written by the participants for errors. If a query contained at least one syntax error, the query was marked incorrect.

3.4. Threats to validity

3.4.1. Training prior to participation

As described in the previous section, the participants were given formal training before participating in the study. The formal training involved practical SQL exercises with a DBMS, and the error messages returned by the DBMS may accustom participants to some types of syntax errors. For this reason, we utilized SQLite in the training, and not one of the DBMSs studied in this research. It is still possible that the error messages of SQLite resemble some of the error messages returned by the four DBMSs. However, we deemed providing practical SQL training unfeasible without a DBMS.

3.4.2. Differences in participant skill

As described in Section 3.3, the study participants were randomly assigned to one of the four database management system groups. This design invites the potential threat of participants with higher (or lower) skill ending up in the same group, thus skewing the results with no regard to the error messages studied. To mitigate the effect of this control variable, the form included four control questions similar to the tests proper. These control questions were the same for all participants regardless of group. We ran a Kruskal–Wallis H test to determine if there were differences in control question score between the four groups of participants using different database management systems: MySQL ($n = 51$), Oracle Database ($n = 36$), PostgreSQL ($n = 25$), and SQL Server ($n = 40$). Distributions of control question scores were similar for all groups, as assessed by visual inspection of a boxplot. Median control question scores were not statistically significantly different between groups, $H(3) = 1.289$, $p = .732$. Kruskal–Wallis H test was chosen because the data were not normally distributed.

3.4.3. Skill improvement

As the participants were novices, the tests needed to be relatively simple in order to eliminate the potential of the results biasing towards low success rates (i.e., *floor effect*). On the other hand, as the form contained four control questions followed by sixteen tests, it is possible, even likely, that a novice participant's skill in error fixing improves during the study. Furthermore, as the study progresses, a participant is more likely to realize that the erroneous SQL queries contain only one error. These two considerations, combined with relatively simple tasks, have the potential of biasing the results of the latter tests towards high success rates (i.e., *ceiling effect*), making comparisons unfeasible. To mitigate this threat, we randomized the order in which the sixteen tests were displayed for each participant. Still, a nascent ceiling effect can be observed in some of the tests' success rates, but this is not due to the order in which the tests were shown to the participants.

Table 1
Sixteen most common syntax errors (Taipalus et al., 2018) and corresponding tests.

Test	Syntax error name	Test	Syntax error name
T01	ambiguous column	T09	failure to specify column name twice
T02	omitting quotes around character data	T10	using an aggregate function outside SELECT or HAVING
T03	IS where not applicable	T11	grouping error: extraneous grouping column
T04	confusing the syntax of keywords	T12	nonstandard operators
T05	confusing the logic of keywords	T13	using WHERE twice
T06	too many columns in subquery	T14	nonstandard keywords or standard keywords in wrong context
T07	undefined column	T15	synonyms
T08	misspellings	T16	curly, square or unmatched brackets

3.4.4. Unnatural environment

One threat to validity is the unnatural characteristics of the online form used, as opposed to a software developer writing and fixing SQL queries and testing their solutions against an SQL compiler and a database. Contrary to a natural environment, the online form did not provide feedback on success or failure in fixing errors. It has been shown that although some SQL errors are numerous, they are usually fixed by query writers, while others are more difficult to fix (Taipalus and Perälä, 2019). In this regard, the effectiveness of different error messages (*RQ1*) should be interpreted with caution. Although this presents a threat in a more general sense, the research setting was similar for all four DBMS groups and should have a minimal effect between the groups. Furthermore, building an environment where user confidence can be measured requires some unnatural elements — confidence cannot be measured in a similar fashion if participants engage in a feedback loop with the DBMS.

3.4.5. Novice participants

Some techniques are better suited for novices, and some for professionals (Feldt et al., 2018). As we attempt to study the effectiveness of different error messages, we deemed novices more appropriate participants, and therefore, we decided to recruit students for this study. Arguably, using professionals, who are more likely able to fix errors regardless of the error message, and who have professional experience in using one or more DBMS, our research setting would have introduced multiple problematic threats to validity. In the past, critique on using students as participants has been raised. However, current research does not appear to validate such a clear cut view (Falessi et al., 2017), but rather suggests choosing participants appropriate for each study, as both students and professionals induce different threats to validity (Feldt et al., 2018).

4. Results

4.1. Outlook of the results

To present an overview of our findings, we present the results first as sum variables comprised of the results of all sixteen tests. The test by test (T01 through T16) results are presented in the following sections (Sections 4.2 through 4.5), each dedicated to one research question. Data analyzed for *RQ2* (i.e., perceived usefulness for finding the error) contained no outliers, were normally distributed, and met requirements for homogeneity of variance. For this analysis, we used one-way ANOVA, which is a parametric test for determining whether there are statistically significant differences between the means of independent groups.

For the other research questions, we used Kruskal–Wallis H test, because data were not normally distributed. Kruskal–Wallis H test may be considered a non-parametric alternative for ANOVA when certain assumptions such as normal distribution are not met. Group sample sizes were unequal due to participant assignment and choices regarding study participation, but both ANOVA (Blanca et al., 2017) and Kruskal–Wallis H test (Lachenbruch and Clements, 1991) have been shown to be robust in this case. The significance level was set to $\alpha = .05$ for all the statistical tests, and Bonferroni correction was used in multiple pairwise comparisons. The null and alternative hypotheses are summarized in Table 2.

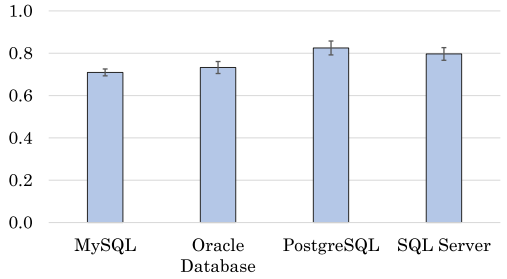
It is worth noting that MySQL tolerated the syntax errors in tests T05 and T09. For these tests for MySQL, we used a made-up error message and omitted the results from the analyses. This approach was chosen because we wanted each group to have the same number of tests. Furthermore, MySQL error messages concerning grouping were produced with `sql_mode=only_full_group_by`, which enables grouping related behavior without the optional feature T301 in the SQL standard (ISO/IEC, 2016b), and similar to that of PostgreSQL, Oracle Database, and SQL Server. This affects test T11.

For RQ1 (Fig. 1a), a Kruskal–Wallis H test was run to determine if there were differences in success rates between four groups of participants with different database management systems: MySQL ($n = 51$), Oracle Database ($n = 36$), PostgreSQL ($n = 25$), and SQL Server ($n = 40$). Success rate refers to the portion of participants in a DBMS group who were able to fix a query divided by the total number of participants in the respective DBMS group. Distributions of success rates were similar for all groups, as assessed by visual inspection of a boxplot. Median success rates were statistically significantly different between the database management system groups, $H(3) = 22.0$, $p < .001$, $\eta^2 = .048$ (a small effect size). Subsequently, pairwise comparisons were performed using Dunn's (1964) procedure with a Bonferroni correction for multiple comparisons. Adjusted p-values are presented. This post hoc analysis revealed statistically significant differences in success rates between MySQL ($Mdn = 0.75$) and SQL Server ($Mdn = 0.81$) ($p = .002$, $\eta^2 = .018$), and MySQL and PostgreSQL ($Mdn = 0.83$) ($p = .002$, $\eta^2 = .038$), but not between MySQL and Oracle Database ($Mdn = 0.75$) ($p = 1$), Oracle Database and PostgreSQL ($p = .098$), or Oracle Database and SQL Server ($p = .168$).

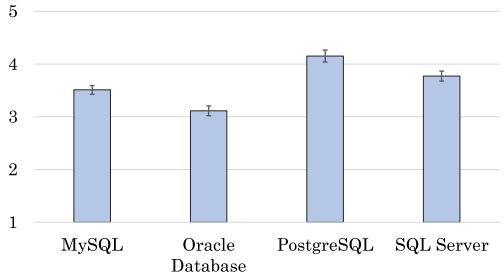
For RQ2 (Fig. 1b), a one-way ANOVA was conducted to determine if database management system error messages provided by different database management systems are perceived useful for finding the error. Participants were assigned into four groups: MySQL ($n = 51$), Oracle Database ($n = 36$), PostgreSQL ($n = 25$), and SQL Server ($n = 40$). There were no outliers, as assessed by boxplot; data were normally distributed for each group, as assessed by Shapiro–Wilk test ($p < .05$); and there was homogeneity of variances, as assessed by Levene's test of homogeneity of variances ($p = .994$). The results are presented as means with a 95% confidence interval. Perceived usefulness for finding the error was statistically significantly different between database management system groups, $F(3, 148) = 17.635$, $p < .001$, $\omega^2 = .25$ (a small effect size). Perceived usefulness for finding the error increased from Oracle Database ($M = 3.11$, $SD = 0.56$) to MySQL ($M = 3.51$, $SD = 0.59$) to SQL Server ($M = 3.77$, $SD = 0.59$) to PostgreSQL ($M = 4.15$, $SD = 0.57$), in that order. Tukey post hoc analysis revealed that the mean increase from Oracle Database to

Table 2
A summary of tested hypotheses (null, alternative) — the statistical tests showed statistically significant differences between the database management groups regarding success rates, error message usefulness for finding the error, and error message usefulness for fixing the error, but not regarding error recovery confidence.

RQ	Hypotheses
1	H_0: *the distributions of success rates for all database management groups are equal.* H_A: *the distributions of success rates for all database management groups are not equal.*
2	H_0: *the means of error message usefulness for finding the error for database management groups are equal.* H_A: *at least one group mean is different (i.e., they are not equal).*
3	H_0: *the distributions of error message usefulness for fixing the error for all database management groups are equal.* H_A: *the distributions of error message usefulness for fixing the error for all database management groups are not equal.*
4	H_0: *the distributions of error recovery confidence for all database management groups are equal.* H_A: *the distributions of error recovery confidence for all database management groups are not equal.*



(a) Error message effectiveness, measured in success rate



(b) Perceived usefulness for finding the error

Fig. 1. Results for RQ1 and RQ2 as sum variables (presented as mean with a 95% confidence interval) — for example, the bar labeled MySQL in Fig. 1a represents the mean of successfully fixed syntax errors for all sixteen tests for all participants in the MySQL database management system group.

MySQL (0.40, 95% CI [0.07, 0.72]) was statistically significant (p = .011) with a large effect size (d = .688), but no other immediate increases between groups were statistically significant.

For RQ3, (Fig. 2a), a Kruskal–Wallis H test was run to determine if error messages provided by different database management systems are perceived useful for fixing the error: MySQL (n = 51), Oracle Database (n = 36), PostgreSQL (n = 25), and SQL Server (n = 40). Distributions of perceived usefulness were similar for all groups, as assessed by visual inspection of a boxplot. Medians for perceived usefulness for fixing the error were statistically significantly different between the database management system groups, $H(3) = 24.626$, $p < .001$, $\eta^2 = .219$ (a large effect size). Subsequently, pairwise comparisons were performed using Dunn's (1964) procedure with a Bonferroni correction for multiple comparisons. Adjusted p-values are presented. This post hoc analysis revealed statistically significant differences in perceived usefulness for fixing the error between MySQL (Mdn = 3.21) and PostgreSQL (Mdn = 4.00) (p = .001, η^2 = .292), Oracle Database (Mdn = 3.03) and PostgreSQL ($p < .001$, η^2 = .263), and Oracle Database and SQL Server (Mdn = 3.38) (p = .018, η^2 = .127), but not between MySQL and Oracle Database (p = 1), SQL Server and PostgreSQL (p = .314), or MySQL and SQL Server (p = .254).

For RQ4 (Fig. 2b), a Kruskal–Wallis H test was run to determine if database management system error messages are perceived to affect error recovery confidence with different database management systems: MySQL (n = 51), Oracle Database (n = 36), PostgreSQL (n = 25), and SQL Server (n = 40). Distributions of perceived usefulness regarding confidence were similar for all groups, as assessed by visual inspection of a boxplot. Median error recovery confidence increased from MySQL (Mdn = 3.36) to Oracle Database (Mdn = 3.44) to SQL Server (Mdn = 3.72) to PostgreSQL (Mdn = 3.75), but the differences were not statistically significant between groups, $H(3) = 4.379$, p = .223.

4.2. Error message effectiveness

A Kruskal–Wallis H test was run to determine if there were differences in success rates between four groups of participants with different database management systems: MySQL (n = 51), Oracle Database (n = 36), PostgreSQL (n = 25), and SQL Server (n = 40). Regarding error message effectiveness test by test (Fig. 3), only test T01 showed statistically significant differences between groups. Distributions of success rates were similar for all groups, as assessed by visual inspection of a boxplot. Median success rates were statistically significantly different between the database management system groups, $H(3) = 12.327$, p = .006, η^2 = .082 (a medium effect size). Subsequently, pairwise comparisons were performed using Dunn's (1964) procedure with a Bonferroni correction for multiple comparisons. Adjusted p-values are presented. This post hoc analysis revealed statistically significant differences in success rates between Oracle Database (M = 0.67) and SQL Server (M = 0.90) (p = .028, η^2 = .082), and between Oracle Database and MySQL (M = 0.92) (p = .007, η^2 = .005). Other pairwise comparisons were not statistically significant.

4.3. Error message usefulness for finding the error

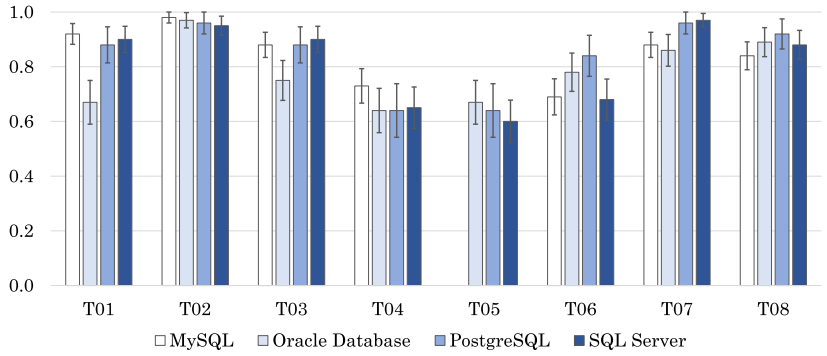
A Kruskal–Wallis H test was run to determine if database management system error messages are perceived useful for finding the error with different database management systems: MySQL (n = 51), Oracle Database (n = 36), PostgreSQL (n = 25), and SQL Server (n = 40). Regarding error message usefulness for finding the error test by test (Fig. 4), tests T02 and T15 showed no statistically significant differences between groups. Distributions of perceived usefulness for finding the error were similar for all groups, as assessed by visual inspection of a boxplot. Statistically significant differences between database management systems and pairwise comparisons are summarized in Table 3, test by test.

4.4. Error message usefulness for fixing the error

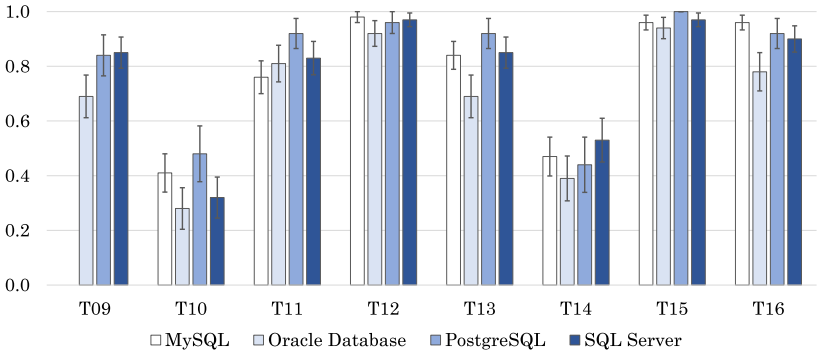
A Kruskal–Wallis H test was run to determine if database management system error messages are perceived useful for fixing the error with different database management systems: MySQL (n = 51), Oracle Database (n = 36), PostgreSQL (n = 25), and SQL

(a) Perceived usefulness for fixing the error (b) Perceived confidence in error recovery

Fig. 2. Results for RQ3 and RQ4 as sum variables (presented as mean with a 95% confidence interval) — for example, the bar labeled MySQL in Fig. 2a represents the mean of perceived usefulness for fixing the error for all sixteen tests for all participants in the MySQL database management system group.



(a) Error message effectiveness in tests T01 – T08

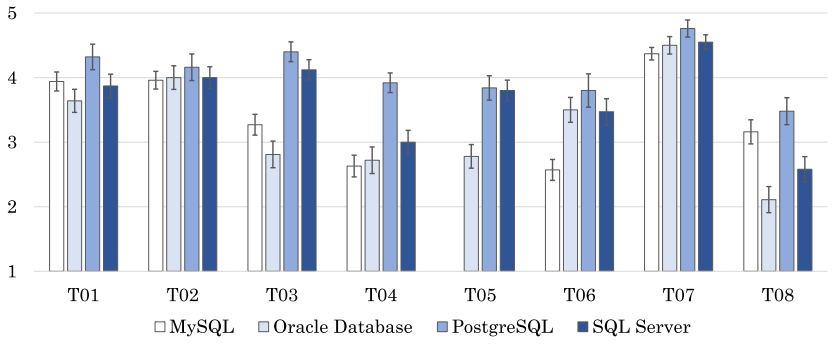


(b) Error message effectiveness in tests T09 – T16

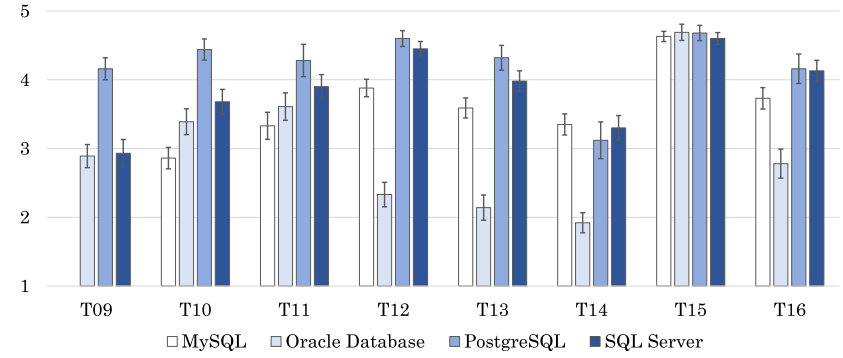
Fig. 3. Error message effectiveness test by test measured in success rate (presented as mean with a 95% confidence interval); results for tests T05 and T09 for MySQL are omitted.

Table 3
Statistically significant differences in error message usefulness for finding the error — only statistically significant differences are listed along with corresponding p-values and effect sizes (η^2); DBMSs are abbreviated as MySQL (MY), Oracle Database (OR), PostgreSQL (PG), and SQL Server (SS)

Test	$H(3)$	p (η^2)	MY – OR	MY – SS	MY – PG	OR – SS	OR – PG	SS – PG
T01	7.912	$p = .048$ (.052)					$p = .030$ (.121)	
T03	36.010	$p < .001$ (.238)			$p = .001$ (.211)	$p < .001$ (.259)	$p < .001$ (.364)	
T04	21.281	$p < .001$ (.141)			$p < .001$ (.252)		$p = .001$ (.241)	
T05	24.124	$p < .001$ (.271)	*(n/a)*	*(n/a)*	*(n/a)*	$p = .001$ (.193)	$p = .003$ (.208)	
T06	21.598	$p < .001$ (.143)			$p < .001$ (.204)			
T07	8.687	$p = .034$ (.058)			$p < .023$ (.113)			
T08	21.684	$p < .001$ (.144)	$p = .001$ (.144)				$p < .001$ (.268)	$p = .042$ (.129)
T09	23.890	$p < .001$ (.281)	*(n/a)*	*(n/a)*	*(n/a)*		$p < .001$ (.307)	
T10	32.973	$p < .001$ (.218)		$p = .008$ (.122)	$p < .001$ (.393)		$p = .002$ (.231)	
T11	11.423	$p = .010$ (.076)			$p = .007$ (.125)			
T12	67.607	$p < .001$ (.248)	$p < .001$ (.321)		$p = .028$ (.146)	$p < .001$ (.582)	$p < .001$ (.241)	
T13	54.064	$p < .001$ (.358)	$p < .001$ (.297)		$p = .029$ (.135)	$p < .001$ (.433)	$p < .001$ (.200)	
T14	34.615	$p < .001$ (.229)	$p < .001$ (.339)			$p < .001$ (.327)	$p = .002$ (.244)	
T16	27.554	$p < .001$ (.182)	$p = .007$ (.127)			$p < .001$ (.273)	$p < .001$ (.283)	



(a) Error message usefulness for finding the error in tests T01 – T08



(b) Error message usefulness for finding the error in tests T09 – T16

Fig. 4. Error message usefulness for finding the error test by test (presented as mean with a 95% confidence interval); results for tests T05 and T09 for MySQL are omitted.

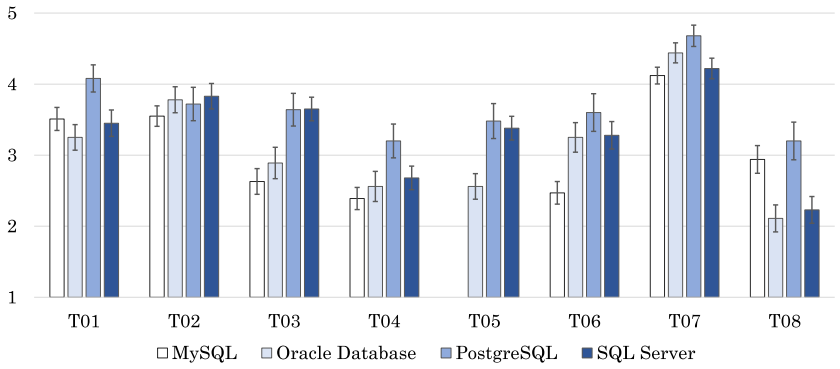
Table 4
Statistically significant differences in error message usefulness for fixing the error — only statistically significant differences are listed along with corresponding p-values and effect sizes (η^2); DBMSs are abbreviated as MySQL (MY), Oracle Database (OR), PostgreSQL (PG), and SQL Server (SS)

Test	$H(3)$	p (η^2)	MY – OR	MY – SS	MY – PG	OR – SS	OR – PG	SS – PG
T01	8.783	$p = .032$ (.058)					$p = .021$ (.143)	
T03	18.873	$p < .001$ (.125)		$p = .001$ (.153)	$p = .010$ (.126)			
T05	16.474	$p = .001$ (.508)	*(n/a)*	*(n/a)*	*(n/a)*	$p = .012$ (.113)	$p = .013$ (.131)	
T06	17.080	$p = .001$ (.113)	$p = .029$ (.099)	$p = .020$ (.104)	$p = .002$ (.168)			
T07	12.419	$p = .006$ (.082)			$p = .006$ (.147)			
T08	16.029	$p = .001$ (.106)	$p = .029$ (.089)				$p = .012$ (.161)	$p = .027$ (.132)
T09	13.121	$p = .004$ (.468)	*(n/a)*	*(n/a)*	*(n/a)*			$p = .008$ (.122)
T10	21.769	$p < .001$ (.144)		$p = .011$ (.108)	$p < .001$ (.234)			
T11	17.555	$p = .001$ (.116)		$p = .014$ (.105)	$p = .001$ (.167)			
T12	50.455	$p < .001$ (.334)	$p < .001$ (.244)			$p < .001$ (.441)	$p < .001$ (.520)	
T13	43.716	$p < .001$ (.290)	$p = .009$ (.153)	$p = .009$ (.130)	$p = .042$ (.116)	$p < .001$ (.402)	$p < .001$ (.400)	
T14	18.052	$p < .001$ (.120)	$p = .002$ (.154)			$p = .001$ (.194)		
T16	15.768	$p = .001$ (.104)				$p = .003$ (.154)	$p = .008$ (.177)	

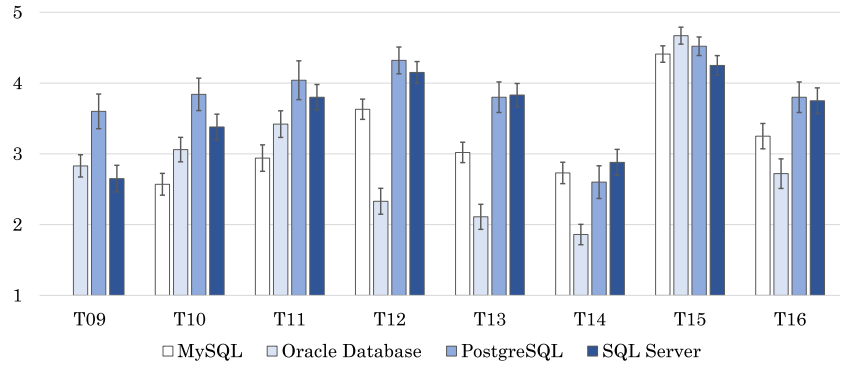
Server ($n = 40$). Regarding error message usefulness for fixing the error test by test (Fig. 5), tests T02, T04, and T15 showed no statistically significant differences between groups. Distributions of perceived usefulness for fixing the error were similar for all groups, as assessed by visual inspection of a boxplot. Statistically significant differences between database management systems and pairwise comparisons are summarized in Table 4, test by test.

4.5. Perceived confidence in error recovery

A Kruskal–Wallis H test was run to determine if database management system error messages are perceived to affect error recovery confidence with different database management systems: MySQL ($n = 51$), Oracle Database ($n = 36$), PostgreSQL ($n = 25$), and SQL Server ($n = 40$). Regarding user confidence in error recovery (Fig. 6), only test T03 showed statistically significant differences between groups. Distributions of perceived confidence were similar for all groups, as assessed by visual inspection of a boxplot. Medians of perceived confidence were statistically significantly different between the database management system groups, $H(3) = 11.471$, $p = .009$, $\eta^2 = .033$ (a small effect size). Subsequently, pairwise comparisons were performed using Dunn's (1964) procedure with a Bonferroni correction for multiple comparisons. Adjusted p-values are presented. This post hoc analysis revealed statistically significant differences in error recovery confidence between MySQL ($M = 3.00$) and SQL Server ($M = 3.62$) ($p = .046$, $\eta^2 = .028$), and MySQL and PostgreSQL ($M = 3.76$) ($p = .043$, $\eta^2 = .053$). Other pairwise comparisons were not statistically significant.



(a) Error message usefulness for fixing the error in tests T01 – T08



(b) Error message usefulness for fixing the error in tests T09 – T16

Fig. 5. Error message usefulness for fixing the error test by test (presented as mean with a 95% confidence interval); results for tests T05 and T09 for MySQL are omitted.

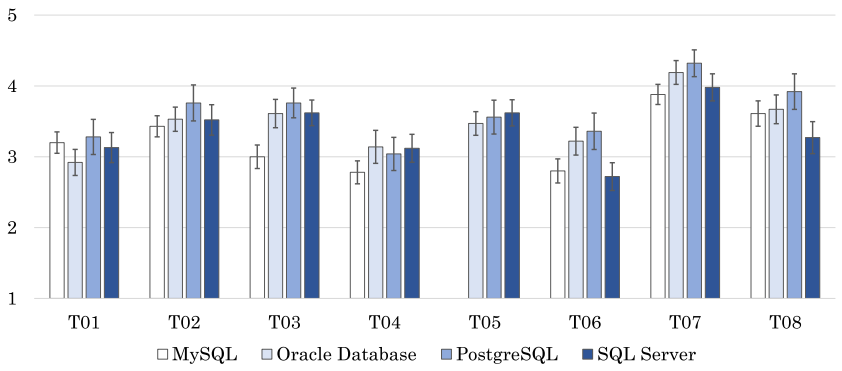
5. Discussion

5.1. Implications for research

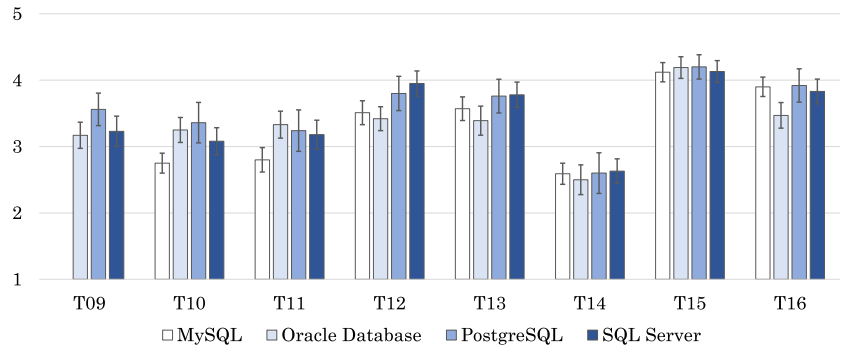
In summary, the analyses showed that there are differences between some DBMSs, but not all. For example, MySQL error message effectiveness was lower than PostgreSQL and SQL Server. In terms of error finding, Oracle Database error messages were perceived less useful than those of the other three DBMSs. In terms of error fixing, PostgreSQL error messages were perceived more useful than those of MySQL and Oracle Database. There were no differences between the DBMS error messages in terms of error recovery confidence.

Based on the evidence yielded by this study, we suggest that the objectively measured effectiveness in success rates between the four most popular relational DBMSs are on the same level, although the analyses indicated *some* statistically significant differences. Generally, MySQL error messages were less effective than those of PostgreSQL and SQL Server, with a statistically significant effect. It is worth noting, though, that there seem to be no prior studies to which these results can be compared. On the general level described in Figs. 1 and 2, PostgreSQL error messages were the most effective by all four metrics used in this study. However, the decrease from PostgreSQL's score to the next best performing DBMS was not statistically significant in any of the summary tests. However, compared to the worst-performing DBMSs in respective tests, PostgreSQL error messages were more effective, and were perceived more useful in error finding and error fixing, with statistically significant differences. This finding presents several implications for research. First, if enhanced error messages are implemented and evaluated, they should be compared to PostgreSQL error messages rather than any other DBMS, as data appear to suggest, at least in the scope of this study, that PostgreSQL error messages are the most effective by several metrics. Second, according to Figs. 1a and 2b, error message effectiveness seems to correlate with error recovery confidence with all the studied DBMSs, although this association was not statistically tested in this study. This further validates the results previously presented about objective measures and perceived confidence (Martino et al., 2012; Fleming et al., 2010). Third, in terms of perceived usefulness for error finding, the data seem to suggest that error finding is closely related to error fixing. This might be due to several factors. It is possible that the error messages that are perceived useful for finding errors are also truly perceived useful for fixing errors, suggesting that helpful error messages are typically helpful by both of these two metrics, not just one. However, it might also be possible that the participants were not able to separate error finding from error fixing, as the former typically precedes the latter (Zapf and Reason, 1994), and arguably an error which is easier to find is also easier to fix, if error fixing is assumed to subsume error finding.

Moving on to the test by test level of specificity, the level of difficulty of a test affects the overall scores of effectiveness, and the effectiveness should be interpreted in relation to the other DBMSs, e.g., success rates in T10 and T14 are lower than the other tests, but this is uniform for all the studied DBMSs. For this reason, we stress against interpreting the results as a general indication that DBMS error messages are effective or ineffective. A rather interesting observation arises from test by test comparison of error fixing (Fig. 5) and message effectiveness (Fig. 3). Although we observed statistically significant differences in perceived usefulness for fixing the error in, e.g., tests T06, T08,



(a) Error recovery confidence in tests T01 – T08



(b) Error recovery confidence in tests T09 – T16

Fig. 6. Error recovery confidence based on the error message test by test (presented as mean with a 95% confidence interval); results for tests T05 and T09 for MySQL are omitted.

T12, T13, the differences in success rates in these tests were not statistically significant. This might suggest that the importance of the qualities of error messages are not related to success rates, but some other metric. A likely candidate is *time*, and the relationship between the effectiveness of an error message, and time taken to fix an error has been demonstrated in the context of programming languages (Ahmed et al., 2019). However, evidence in the context of SQL error messages is not currently available and warrants the attention of future research.

5.2. Implications for industry

The results yielded by this study have implications for DBMS vendors. On a general level, and although the effectiveness of different DBMSs error messages had merely ostensible differences, the perceived, i.e., subjective qualities of error finding and fixing differed to a statistically significant degree. Accessibility and ease of use are important parts of technology adoption, and *perceived ease of use* is one of the two key variables affecting an individual's information system acceptance according to the Technology Acceptance Model (Davis, 1985), one of the most influential information systems science theories (e.g., Lee et al., 2003). Arguably, other qualities such as compatibility, cost, features, and performance are important metrics in choosing the most appropriate DBMS. However, it can be argued that part of the popularity of DBMSs such as MySQL and MongoDB might be explained by accessibility and community support. It seems reasonable to argue that for a DBMS novice, a part of this perceived ease of use is due to compiler error message qualities.

On a more specific level, an interesting aspect in terms of DBMS error message iteration is *how to improve DBMS error messages*. For perceived usefulness, tests with at least three statistically significant differences between groups were T03, T06, T08, T12, T13, T14, and T16. In T03, MySQL, PostgreSQL, and SQL Server error messages provide the position of the error (Fig. 7). However, the position provided by MySQL is not accurate and implies that the error is contained in the portion listed in the extracted part of the query, while the error actually appears before the part provided. Oracle Database identifies the erroneous part of the query, but provides no error position. Instead, the error message provided is contradictory to what the data demand requires — the problem behind the error is not missing NULL keyword, but rather a wrong operator. In T06, MySQL error message was considered worst in terms of error fixing, with statistically significant differences between MySQL and the other DBMSs. However, there is no clear indication of what is different in MySQL's error message, as all messages communicate the problem of too many columns in the subquery's SELECT clause. In T12 and T13, MySQL, PostgreSQL and SQL Server error messages provide the position of the error, and the error messages of these three DBMSs were considered statistically significantly more helpful for finding and fixing the error when compared to Oracle Database. In T12, Oracle Database interprets the use of a wrong operator as *missing expression*, which is probably why Oracle Database's error message was considered relatively unhelpful in both finding and fixing the error. In T13, MySQL, PostgreSQL and SQL Server provide the position of the error, while none of the DBMSs clearly articulate that one SQL query (were it a subquery or the top-level query) cannot include more than one WHERE clause. In T14 and T16, MySQL, PostgreSQL and SQL Server error messages were considered statistically significantly more helpful for finding the error when compared to Oracle Database. The errors in T14 and T16 are

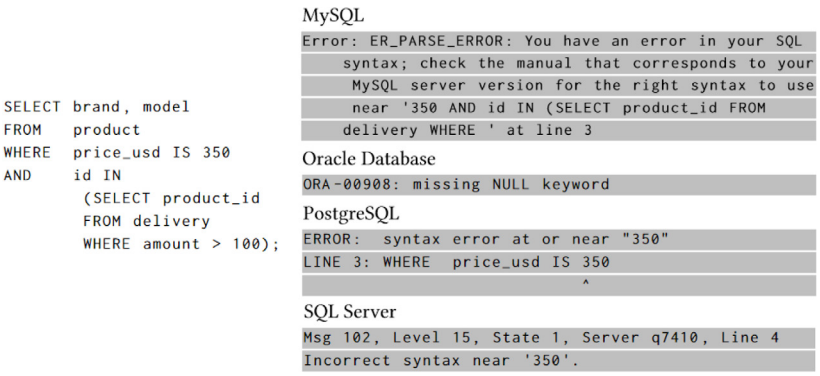


Fig. 7. Examples of error messages in test T03.

rather different from each other, yet Oracle Database identifies both as *missing expression*. The other DBMSs, again, provide the position of the error. All the differences highlighted above seem to suggest that providing the error position, and providing it accurately in an important characteristic for an error message. As the SQL queries in the tests were relatively simple, it would be reasonable to suggest that the differences are more highlighted as query complexity increases. These findings are in line with tangential programming language error message studies (Becker et al., 2019).

Finally, in this study, we did not consider runtime SQL errors (cf. e.g., Brass and Goldberg, 2006), and the number of syntax errors in production should be low, as the application programs do not operate using *ad hoc*, but tested queries which are embedded into the host language, executed via host language libraries, or as stored database procedures. However, the development environment could benefit from more informative error messages, and the performance considerations in development (as opposed to production) should play a less significant role. With the potential effects of enhanced error messages on DBMS performance in mind, a possible and intuitive solution for more effective error messages would be to implement different error messages for development and production, controlled by a DBMS level setting. As all professionals are novices to SQL, different SQL dialects, or different DBMSs at some point, ease of use should be among the considerations towards selecting the most appropriate DBMS. Furthermore, as future professionals often familiarize themselves with a DBMS through formal education, and the DBMS is typically chosen or recommended by a teacher who strives to utilize the most appropriate tools *for learning*, it seems reasonable to argue that the most accessible tools will prevail in education as well.

5.3. Implications for education

As discussed in Section 2.2, implementing enhanced error messages into a programming language or SQL compiler is not as straightforward as merely following proposed guidelines. However, some of these complexities may be mitigated through third party learning environments positioned between the end-user and the DBMS. If error tracing and error message generation are computationally tasking, some of this work may be delegated to the learning environment. Often this is the only feasible solution to enhancing error messages, as educators typically have no access to modify DBMS internals. Multiple online SQL learning environments exist (Prior, 2003; Brusilovsky et al., 2010, 2008), but the particulars behind what types of support in terms of enhanced error messages they provide are unclear. Nevertheless, and as touched in the previous section, it would be beneficial for learning if DBMSs could provide extended error messages on demand, similar to those of some programming language compilers. Alternatively, DBMSs could provide an extended error stack to be used by third party tools such as interactive learning environments.

Learning through errors has been shown to be beneficial in understanding SQL (Miao et al., 2019; Zilligen and Hidayat, 2008), especially in the *explaining* phase of error recovery. Arguably, it may be considered helpful if the error message supports the query writer's expectation of what failed in query formulation. In contrast, it seems fair to suggest that if the error message is contradictory to the viewpoint of the query writer, it may have negative effects on error recovery confidence. It is intuitive that when a novice encounters an error message which conflicts with their expectations, it negatively affects their confidence, as error messages may be seen as authoritarian facts rather than suggestions — if the compiler deems the query erroneous, it is not debatable. From an educational point of view, low confidence may result in low attempt rates, which are problematic to learning (Migler and Dekhtyar, 2020), as non-attempts do not constitute to learning, whereas failures do (Metcalfe, 2017). The relationship between error recovery confidence and error message usefulness for finding and fixing the error suggests that although there are differences in error message usefulness (Figs. 4 and 5) between the DBMSs, these differences are not necessarily reflected on error recovery confidence (Fig. 6). It has been shown that people tend to choose tasks which they feel highly confident about over tasks of low confidence (Carlebach and Yeung, 2020). Consequently, if perceived usefulness for finding and fixing errors is not proportional to error recovery confidence, it may be speculated that less useful error messages do not necessarily result in low attempt rates.

Finally, a rough comparison of error message effectiveness and error recovery confidence in general (Figs. 1a and 2b) and test by test (Figs. 3 and 6) reveal that effectiveness and confidence appear highly uniform. This might be due to one of two explanations. First, if error messages are considered to increase error recovery confidence, and the participants are highly successful in fixing the errors, the error messages may be considered helpful. Second, if the error messages are not considered to increase error recovery confidence, and the participants are not likely to fix the errors, the error messages have neither negative nor positive effects on error fixing. For the sake of argument, if the error messages are considered to increase error recovery confidence, but the participants cannot fix the errors, the error messages possibly negatively affect error recovery, and provide a false sense of confidence. Finally, if the error messages are not perceived to increase confidence, but the participants are successful in fixing the errors, the error messages may be confusing, and possibly contradictory to the query writer's (in this case correct) understanding. In summary, a disparity between objective effectiveness and subjective confidence implies problems with the formulation

of the error messages, whereas uniformity implies the opposite. In this regard, the results show that none of the DBMSs studied generates error messages which hinder error recovery.

5.4. Limitations

There are four main limitations to this study. As such, these results should be interpreted in the microcosm they are measured. Although some DBMSs performed better than others by the four tested metrics, it is unknown how, e.g., lifting the discussed limitations would affect the results. First, we speculated in Section 5.1 that although perceived usefulness in error fixing was not reflected in the success rates, error fixing might affect the time needed to fix the error. However, time, as informative as it is, was not measured in this study. Second, and although we have argued for the importance of studying the performance of novices, the queries in our tests do not necessarily represent industry complexity queries. Furthermore, the queries each contained only one syntax error, which probably does not reflect real world problems in query formulation, but rather learning situations. Third, the query fixing setting only represented a scenario where the query writer (i.e., the participant) had to fix an erroneous query written by someone else (i.e., us). This does not reflect the typical query formulation process in which the query writer is responsible for formulating the query from start to finish. Fourth, in terms of error message effectiveness, we only measured binary success or failure. In fact, error fixing is a multi-step process, and error messages may guide the query writer towards a *more correct* query, before the error fixing process is completed. For example, after encountering an error message, a query writer might fix the expressions `name IN ('H%', 'K%')` to `name = 'H%' OR name = 'K%'`, which is more correct, yet still produces a syntax error. If this error fixing process was followed by the correct expressions `name LIKE 'H%' OR name LIKE 'K%'`, the error message would arguably have guided the query writer towards the correct solution, even though the solution was not reached in one fixing attempt. These four limitations should be taken into considerations in future studies, although accounting for points two and three potentially introduce severe threats to internal validity.

6. Conclusion

In this study, we set out to investigate four qualities of error messages of four popular relational database management systems. First, error message effectiveness in terms of query fixing success rates showed differences between MySQL, PostgreSQL, and SQL Server, in favor of the two latter, but not regarding Oracle Database. Second, perceived usefulness of error messages for finding the erroneous part of the query also showed differences between the DBMSs. Specifically, Oracle Database error messages were considered least useful in this regard. Third, PostgreSQL and SQL Server error messages were considered most helpful for fixing errors, and finally, we observed no statistically significant differences in error recovery confidence. Based on the results, we suggest DBMS vendors to consider recommendations regarding programming language compiler error messages, as some of the differences in the results between DBMS error messages may be explained by whether the error messages clearly points out the erroneous part of the query. For researchers, we propound the view that despite their age, SQL compilers are a fresh seedbed for error message studies, and a fertile ground to demonstrate how the results of programming language error message studies generalize to declarative languages.

CRediT authorship contribution statement

Toni Taipalus: Conceptualization, Methodology, Formal analysis, Investigation, Writing - original draft, Supervision, Project administration. **Hilkka Grahn:** Conceptualization, Formal analysis, Writing - review & editing. **Hadi Ghanbari:** Validation, Writing review & editing.

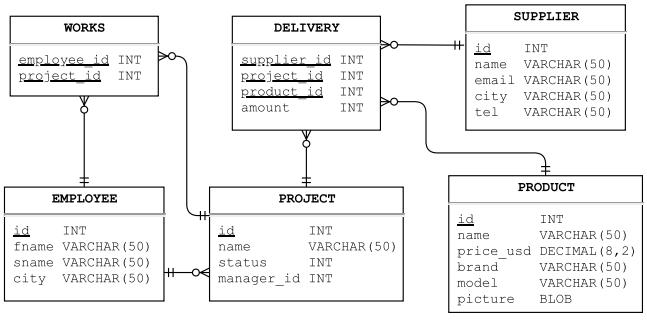
Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Data collection form

The data collection form consisted of four control questions followed by the form proper. The form consisted of six components — *(i)* database schema, *(ii)* data demand, *(iii)* erroneous SQL query, *(iv)* error message, *(v)* a question and a free form input field where the participant wrote the fixed version of the SQL query, and *(vi)* three questions answered with a Likert scale. Components *(i)*, *(v)*, and *(vi)* were repeated for each of the 16 tests, while components *(ii)* through *(iv)* were different for each test. The general structure of a form page is presented in Appendix A.1. The control questions and respective error messages are presented in Appendix A.2, and Appendix A.3 contains the questionnaire forms.

A.1. General structure



The database schema

Find the names of suppliers who have delivered at least one Apple product priced over 50 USD.

```
SELECT name
FROM   supplier
JOIN   delivery    ON (supplier.id = delivery.
    supplier_id)
JOIN   product     ON (delivery.product_id = product.
    id)
WHERE  product.price_usd > 50
AND    product.brand = 'Apple';
```

```
Error: ER_NON_UNIQ_ERROR: Column 'name' in field list
    is ambiguous
```

Please type the fixed SQL query here.

Please evaluate the error message (1 = strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = strongly agree).

	1 2 3 4 5
The error message was useful for finding the error	
The error message was useful for fixing the error	
The error message increased my confidence in error recovery	

A.2. Control questions

Before the tests, the participants were asked to fix four erroneous SQL queries. The database schema was the same as in the form. A made up (as opposed to a DBMS generated) error message was displayed.

Q1: Find the minimum, maximum and average prices of products which have been delivered by the supplier named 'DHL'.

```
SELECT MIN(price_usd), MAX(price_usd), AVE(
price_usd)
FROM product
WHERE id IN
  (SELECT product_id
   FROM delivery
   WHERE supplier_id IN
      (SELECT id
       FROM supplier
       WHERE name = 'DHL')
  );
```

```
Error: undefined function
```

Q2: Find the ids, names and prices of products which have been delivered at least 1000 pcs in total by a supplier from Kiev.

```
SELECT p.id, p.name, p.price_usd
FROM product p
WHERE 1000 <=
    (SELECT SUM(COUNT(d.amount))
     FROM delivery d
     WHERE p.id = d.product_id
     AND EXISTS
        (SELECT *
         FROM supplier s
         WHERE s.id = d.supplier_id
         AND s.city = 'Kiev')
    );
```

```
Error: aggregate functions cannot be nested
```

Q3: Find the names and status of projects which have received at least one Intel product.

```
SELECT j.id, j.name
FROM project j
JOIN delivery d ON (p.id = d.project_id)
JOIN product u ON (u.id = d.product_id)
WHERE u.brand = 'Intel';
```

```
Error: undefined correlation name
```

Q4: Find the names, brands, models and prices of products which have a price of over USD 500, and have been delivered to a project named 'Mastercraft'.

```
SELECT p.name, p.brand, p.model, p.price_usd > 500
FROM product p
JOIN delivery d ON (p.id = d.product_id)
JOIN project j ON (j.id = d.project_id)
WHERE j.name = 'Mastercraft';
```

```
Error: keyword not found where expected
```

A.3. Questionnaire forms

The questionnaire forms can be found as supplementary files. All the forms have a total of twenty pages. The first four pages contain the control questions which are the same in all the forms. The last sixteen pages contain the sixteen tests, and these pages are the same for each database management system, with the exception of the error messages.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jss.2021.111034.

References

Ahadi, A., Behbood, V., Vihavainen, A., Prior, J., Lister, R., 2016a. Students' syntactic mistakes in writing seven different types of SQL queries and its application to predicting students' success. In: Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE). ACM Press, New York, New York, USA, pp. 401–406. http://dx.doi.org/10.1145/2839509.2844640, URL: http://dl.acm.org/citation.cfm?doid=2839509.2844640.

Ahadi, A., Prior, J., Behbood, V., Lister, R., 2016b. 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 (ITiCSE). ACM Press, New York, New York, USA, pp. 272–277. http://dx.doi.org/10.1145/2899415.2899464.

Ahmed, U.Z., Sindhgatta, R., Srivastava, N., Karkare, A., 2019. Targeted example generation for compilation errors. In: 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, http://dx.doi.org/10.1109/ase.2019.00039.

Barik, T., Smith, J., Lubick, K., Holmes, E., Feng, J., Murphy-Hill, E., Parnin, C., 2017. Do developers read compiler error messages? In: 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE). IEEE, http://dx.doi.org/10.1109/icse.2017.59.

Becker, B.A., 2016. An effective approach to enhancing compiler error messages. In: Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16. ACM Press, http://dx.doi.org/10.1145/2839509.2844584.

Becker, B.A., Denny, P., Pettit, R., Bouchard, D., Bouvier, D.J., Harrington, B., Kamil, A., Karkare, A., McDonald, C., Osera, P.-M., Pearce, J.L., Prather, J., 2019. Compiler error messages considered unhelpful. In: Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education. ACM, http://dx.doi.org/10.1145/3344429.3372508.

Becker, B.A., Glanville, G., Iwashima, R., McDonnell, C., Goslin, K., Mooney, C., 2016. Effective compiler error message enhancement for novice programming students. Comput. Sci. Educ. 26 (2–3), 148–175. http://dx.doi.org/10.1080/08993408.2016.1225464.

Becker, B.A., Murray, C., Tao, T., Song, C., McCartney, R., Sanders, K., 2018. Fix the first, ignore the rest. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education. ACM, http://dx.doi.org/10.1145/3159450.3159453.

Blanca, M.J., Alarcón, R., Arnau, J., Bono, R., Bendayan, R., 2017. Non-normal data: Is ANOVA still a valid option? Psicothema 29 (4), 552—557. http://dx.doi.org/10.7334/psicothema2016.383.

Borthick, A.F., Bowen, P.L., Jones, D.R., Tse, M.H.K., 2001a. The effects of information request ambiguity and construct incongruence on query development. Decis. Support Syst. 32 (1), 3–25. http://dx.doi.org/10.1016/s0167-9236(01)00097-5.

Borthick, A.F., Bowen, P.L., Liew, S., Rohde, F.H., 2001b. The effects of normalization on end-user query errors: An experimental evaluation. Int. J. Acc. Inf. Syst. 2 (4), 195–221. http://dx.doi.org/10.1016/S1467-0895(01)00023-9.

Bowen, P.L., Rohde, F.H., Basford, J., 2004. Ex ante evaluations of alternate data structures for end user queries. J. Database Manage. 15 (4), 45–70. http://dx.doi.org/10.4018/jdm.2004100103.

Brass, S., Goldberg, C., 2006. Semantic errors in SQL queries: A quite complete list. J. Syst. Softw. 79 (5), 630–644. http://dx.doi.org/10.1016/j.jss.2005.06.028.

Brusilovsky, P., Sosnovsky, S., Lee, D.H., Yudelson, M., Zadorozhny, V., Zhou, X., 2008. An open integrated exploratorium for database courses. In: Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education. In: ITiCSE '08, ACM, New York, NY, USA, pp. 22–26. http://dx.doi.org/10.1145/1384271.1384280, URL: http://doi.acm.org/10.1145/1384271.1384280.

Brusilovsky, P., Sosnovsky, S., Yudelson, M.V., Lee, D.H., Zadorozhny, V., Zhou, X., 2010. Learning SQL programming with interactive tools: From integration to personalization. ACM Trans. Comput. Educ. 9 (4), 19:1–19:15. http://dx.doi.org/10.1145/1656255.1656257, URL: http://doi.acm.org/10.1145/1656255.1656257.

Carlebach, N., Yeung, N., 2020. Subjective confidence acts as an internal cost-benefit factor when choosing between tasks. J. Exp. Psychol.: Hum. Percept. Perform. 46 (7), 729–748. http://dx.doi.org/10.1037/xhp0000747.

Casterella, G.I., Vijayasarathy, L., 2019. Query structure and data model mapping errors in information retrieval tasks. J. Inf. Syst. Educ. 30 (3), 178–190, URL: http://jise.org/Volume30/n3/JISEv30n3p178.pdf.

Davis, F.D., 1985. A Technology Acceptance Model for Empirically Testing New End-User Information Systems: Theory and Results (PhD Dissertation). Massachusetts Institute of Technology.

Denny, P., Luxton-Reilly, A., Carpenter, D., 2014. Enhancing syntax error messages appears ineffectual. In: Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education. In: ITiCSE '14, Association for Computing Machinery, New York, NY, USA, pp. 273–278. http://dx.doi.org/10.1145/2591708.2591748.

Denny, P., Luxton-Reilly, A., Tempero, E., Hendrickx, J., 2011. Understanding the syntax barrier for novices. In: Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education - ITiCSE '11. ACM Press, http://dx.doi.org/10.1145/1999747.1999807.

Desender, K., Boldt, A., Yeung, N., 2018. Subjective confidence predicts information seeking in decision making. Psychol. Sci. 29 (5), 761–778. http://dx.doi.org/10.1177/0956797617744771.

Dietrich, B., Grust, T., 2015. A SQL debugger built from spare parts. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data - SIGMOD '15. ACM Press, http://dx.doi.org/10.1145/2723372.2735358.

Dong, T., Khandwala, K., 2019. The impact of "cosmetic" changes on the usability of error messages. In: Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems. ACM, http://dx.doi.org/10.1145/3290607.3312978.

Dunn, O.J., 1964. Multiple comparisons using rank sums. Technometrics 6 (3), 241–252. http://dx.doi.org/10.2307/1266041.

Espinosa, R., García-Saiz, D., Zorrilla, M., Zubcoff, J.J., Mazón, J.-N., 2015. Enabling non-expert users to apply data mining for bridging the big data divide. In: Ceravolo, P., Accorsi, R., Cudre-Mauroux, P. (Eds.), Data-Driven Process Discovery and Analysis. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 65–86.

Falessi, D., Juristo, N., Wohlin, C., Turhan, B., Münch, J., Jedlitschka, A., Oivo, M., 2017. Empirical software engineering experts on the use of students and professionals in experiments. Empir. Softw. Eng. 23 (1), 452–489. http://dx.doi.org/10.1007/s10664-017-9523-3.

Feldt, R., Zimmermann, T., Bergersen, G.R., Falessi, D., Jedlitschka, A., Juristo, N., Münch, J., Oivo, M., Runeson, P., Shepperd, M., Sjøberg, D.I.K., Turhan, B., 2018. Four commentaries on the use of students and professionals in empirical software engineering experiments. Empir. Softw. Eng. 23 (6), 3801–3820. http://dx.doi.org/10.1007/s10664-018-9655-0.

Fleming, S.M., Weil, R.S., Nagy, Z., Dolan, R.J., Rees, G., 2010. Relating introspective accuracy to individual differences in brain structure. Science 329 (5998), 1541–1543. http://dx.doi.org/10.1126/science.1191883.

Grust, T., Kliebhan, F., Rittinger, J., Schreiber, T., 2011. True language-level SQL debugging. In: Proceedings of the 14th International Conference on Extending Database Technology - EDBT/ICDT '11. ACM Press, http://dx.doi.org/10.1145/1951365.1951441.

Grust, T., Rittinger, J., 2013. Observing SQL queries in their natural habitat. ACM Trans. Database Syst. 38 (1), 1–33. http://dx.doi.org/10.1145/2445583.2445586.

Gutiérrez, F., Ochoa, X., Seipp, K., Broos, T., Verbert, K., 2019. Benefits and trade-offs of different model representations in decision support systems for non-expert users. In: Lamas, D., Loizides, F., Nacke, L., Petrie, H., Winckler, M., Zaphiris, P. (Eds.), Human-Computer Interaction – INTERACT 2019. Springer International Publishing, Cham, pp. 576–597.

Hannebauer, C., Hesenius, M., Gruhn, V., 2018. Does syntax highlighting help programming novices?. In: Proceedings of the 40th International Conference on Software Engineering. ACM, http://dx.doi.org/10.1145/3180155.3182554.

Hellerstein, J.M., Stonebraker, M., Hamilton, J., 2007. Architecture of a database system. Found. Trends Databases 1 (2), 141–259. http://dx.doi.org/10.1561/1900000002.

ISO/IEC, 2016a. ISO/IEC 9075-1:2016, "SQL - Part 1: Framework". ISO/IEC, URL: https://www.iso.org/standard/63555.html.

ISO/IEC, 2016b. ISO/IEC 9075-2:2016, "SQL - Part 2: Foundation". ISO/IEC, URL: https://www.iso.org/standard/63556.html.

Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society, 2013. Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. Technical Report, ACM, New York, NY, USA, http://dx.doi.org/10.1145/2534860, 999133.

Kölling, M., 1999. The Design of an Object-Oriented Environment and Language for Teaching (Phd Dissertation). Basser Department of Computer Science, University of Sydney.

Kontogiannis, T., 1999. User strategies in recovering from errors in man–machine systems. Saf. Sci. 32 (1), 49–68. http://dx.doi.org/10.1016/s0925-7535(99)00010-7.

Lachenbruch, P.A., Clements, P.J., 1991. ANOVA, Kruskal-Wallis, normal scores and unequal variance. Comm. Statist. Theory Methods 20 (1), 107–126. http://dx.doi.org/10.1080/03610929108830486.

Lawal, M.A., Sultan, A.B.M., Shakiru, A.O.S., 2016. Systematic literature review on SQL injection attack. Int. J. Soft Comput. 11 (1), 26–35.

Lee, M.J., Ko, A.J., 2011. Personifying programming tool feedback improves novice programmers' learning. In: Proceedings of the Seventh International Workshop on Computing Education Research. In: ICER '11, Association for Computing Machinery, New York, NY, USA, pp. 109–116. http://dx.doi.org/10.1145/2016911.2016934.

Lee, Y., Kozar, K.A., Larsen, K.R., 2003. The technology acceptance model: Past, present, and future. Commun. Assoc. Inf. Syst. 12, http://dx.doi.org/10.17705/1cais.01250.

Martino, B.D., Fleming, S.M., Garrett, N., Dolan, R.J., 2012. Confidence in value-based choice. Nature Neurosci. 16 (1), 105–110. http://dx.doi.org/10.1038/nn.3279.

McCall, D., Kolling, M., 2014. Meaningful categorisation of novice programmer errors. In: 2014 IEEE Frontiers in Education Conference (FIE) Proceedings. IEEE, http://dx.doi.org/10.1109/fie.2014.7044420.

Metcalfe, J., 2017. Learning from errors. Annu. Rev. Psychol. 68 (1), 465–489. http://dx.doi.org/10.1146/annurev-psych-010416-044022, PMID: 27648988.

Miao, Z., Chen, T., lexander Bendeck, Day, K., Roy, S., Yang, J., 2020. I-Rex: An interactive relational query explainer for SQL. Proc. VLDB Endow. 13 (12), 2997–3000. http://dx.doi.org/10.14778/3415478.3415528.

Miao, Z., Roy, S., Yang, J., 2019. Explaining wrong queries using small examples. In: Proceedings of the 2019 International Conference on Management of Data - SIGMOD '19. ACM Press, http://dx.doi.org/10.1145/3299869.3319866.

Migler, A., Dekhtyar, A., 2020. Mapping the SQL learning process in introductory database courses. In: Proceedings of the 51st ACM Technical Symposium on Computer Science Education. In: SIGCSE '20, Association for Computing Machinery, New York, NY, USA, pp. 619–625. http://dx.doi.org/10.1145/3328778.3366869.

Mills, R.J., Beaulieu, T.Y., Feldon, D.F., Olsen, D.H., 2020. Implications of prelecture material on cognitive load and instructional effectiveness in cross-disciplinary IS education: The nexus of Benford's law and SQL. Decis. Sci. J. Innov. Educ. 18 (2), 313–338. http://dx.doi.org/10.1111/dsji.12206, URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/dsji.12206. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/dsji.12206.

Pettit, R.S., Homer, J., Gee, R., 2017. Do enhanced compiler error messages help students? Results inconclusive. In: Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education. In: SIGCSE '17, Association for Computing Machinery, New York, NY, USA, pp. 465–470. http://dx.doi.org/10.1145/3017680.3017768.

Poulsen, S., Butler, L., Alawini, A., Herman, G.L., 2020. Insights from student solutions to SQL homework problems. In: Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education. ACM, http://dx.doi.org/10.1145/3341525.3387391.

Prior, J.C., 2003. Online assessment of SQL query formulation skills. In: Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20. In: ACE '03, Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 247–256, URL: http://dl.acm.org/citation.cfm?id=858403.858433.

Randolph, G.B., 2003. The forest and the trees: Using Oracle and SQL server together to teach ANSI-standard SQL. In: Proceedings of the 4th ACM Conference on Information Technology Curriculum (CITC). In: CITC4 '03, ACM, New York, NY, USA, pp. 234–236. http://dx.doi.org/10.1145/947121.947174, URL: http://doi.acm.org/10.1145/947121.947174.

Reisner, P., 1977. Use of psychological experimentation as an aid to development of a query language. IEEE Trans. Softw. Eng. SE-3 (3), 218–229. http://dx.doi.org/10.1109/tse.1977.231131.

Reisner, P., 1981. Human factors studies of database query languages: A survey and assessment. ACM Comput. Surv. 13 (1), 13–31. http://dx.doi.org/10.1145/356835.356837.
Reisner, P., Boyce, R.F., Chamberlin, D.D., 1975. Human factors evaluation of two data base query languages. In: Proceedings of the May 19-22, 1975, National Computer Conference and Exposition on - AFIPS '75. ACM Press, http://dx.doi.org/10.1145/1499949.1500036.
Sarkar, A., 2015. The impact of syntax colouring on program comprehension. In: Proceedings of the 26th Annual Conference of the Psychology of Programming Interest Group (PPIG 2015). URL: https://www.academia.edu/download/38888331/sarkar_2015_syntax_colouring.pdf.
van der Schaaf, T., 1995. Human recovery of errors in man-machine systems. IFAC Proc. Vol. 28 (15), 71–76. http://dx.doi.org/10.1016/s1474-6670(17)45211-6.
Schorsch, T., 1995. CAP: An automated self-assessment tool to check Pascal programs for syntax, logic and style errors. In: Proceedings of the Twenty-Sixth SIGCSE Technical Symposium on Computer Science Education - SIGCSE '95. ACM Press, http://dx.doi.org/10.1145/199688.199769.
Seifert, C., Hutchins, E., 1992. Error as opportunity: Learning in a cooperative task. Hum.-Comput. Interact. 7 (4), 409–435. http://dx.doi.org/10.1207/s15327051hci0704_3.
Shin, S.-S., 2020. Structured query language learning: Concept map-based instruction based on cognitive load theory. IEEE Access 8, 100095–100110. http://dx.doi.org/10.1109/ACCESS.2020.2997934.
Shneiderman, B., 1982. Designing computer system messages. Commun. ACM 25 (9), 610–611. http://dx.doi.org/10.1145/358628.358639.
Smelcer, J.B., 1995. User errors in database query composition. Int. J. Hum.-Comput. Stud. 42 (4), 353–381. http://dx.doi.org/10.1006/ijhc.1995.1017.
Taipalus, T., 2019. Teaching tip: A notation for planning SQL queries. J. Inf. Syst. Educ. 30 (3), 160–166, URL: http://jise.org/Volume30/n3/JISEv30n3p160.pdf.
Taipalus, T., 2020a. The effects of database complexity on SQL query formulation. J. Syst. Softw. 165, 110576. http://dx.doi.org/10.1016/j.jss.2020.110576.
Taipalus, T., 2020b. Explaining causes behind SQL query formulation errors. In: 2020 IEEE Frontiers in Education Conference (FIE). pp. 1–9. http://dx.doi.org/10.1109/FIE44824.2020.9274114.
Taipalus, T., Perälä, P., 2019. What to expect and what to focus on in SQL query teaching. In: Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE). In: SIGCSE '19, ACM, New York, NY, USA, pp. 198–203. http://dx.doi.org/10.1145/3287324.3287359.
Taipalus, T., Seppänen, V., 2020. SQL education: A systematic mapping study and future research agenda. ACM Trans. Comput. Educ. 20 (3), http://dx.doi.org/10.1145/3398377.
Taipalus, T., Siponen, M., Vartiainen, T., 2018. Errors and complications in SQL query formulation. ACM Trans. Comput. Educ. 18 (3), 15:1–15:29. http://dx.doi.org/10.1145/3231712, URL: http://doi.acm.org/10.1145/3231712.
The Joint Task Force on Computing Curricula, 2015. Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. Technical Report, ACM, New York, NY, USA, URL: https://dl.acm.org/citation.cfm?id=2965631.
Topi, H., Kaiser, K.M., Sipior, J.C., Valacich, J.S., Nunamaker, Jr., J.F., de Vreede, G.J., Wright, R., 2010. Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. Technical Report, ACM, New York, NY, USA, URL: https://dl.acm.org/citation.cfm?id=2593310.
Welty, C., Stemple, D.W., 1981. Human factors comparison of a procedural and a nonprocedural query language. ACM Trans. Database Syst. 6 (4), 626–649. http://dx.doi.org/10.1145/319628.319656, URL: http://doi.acm.org/10.1145/356835.356837.
Zapf, D., Reason, J.T., 1994. Introduction: Human errors and error handling. Appl. Psychol. 43 (4), 427–432. http://dx.doi.org/10.1111/j.1464-0597.1994.tb00838.x.
Zilligen, R., Hidayat, A., 2008. A misconception module to a database courseware. In: Proceedings of the 46th ACM Annual Southeast Regional Conference (ACM-SE). In: ACM-SE 46, ACM, New York, NY, USA, pp. 529–530. http://dx.doi.org/10.1145/1593105.1593250, URL: http://doi.acm.org/10.1145/1593105.1593250.

Toni Taipalus is a researcher and teacher at the Faculty of Information Technology, University of Jyväskylä, Finland. His research has been published in journals such as Journal of Systems and Software and ACM Transactions on Computing Education. His current research interests include query languages, database management systems, computing education, and data analytics.

Hilkka Grahn is a researcher and teacher at the Faculty of Information Technology, University of Jyväskylä, Finland. Her research has been published in journals such as International Journal of Human–Computer Studies and Accident Analysis & Prevention. Her current research interests include human–computer interaction, driver distraction, visual attention, and human factors.

Hadi Ghanbari is a postdoctoral researcher at the Department of Information and Service Management, Aalto University School of Business, Finland. Simultaneously, he is a visiting research fellow at FinEst Twins Smart City Center of Excellence, TalTech, Estonia. He mostly conducts empirical research on sociotechnical aspects of information systems development and digital innovation, with a focus on Technical Debt and information security.