"There is no ambiguity on what to return": Investigating the Prevalence of SQL Misconceptions

Daphne Miedema
d.e.miedema@tue.nl
Eindhoven University of Technology
Eindhoven, The Netherlands

Michael Liut
michael.liut@utoronto.ca
University of Toronto Missisauga
Missisauga, Canada

George Fletcher
g.h.l.fletcher@tue.nl
Eindhoven University of Technology
Eindhoven, The Netherlands

Efthimia Aivaloglou
e.aivaloglou@tudelft.nl
Delft University of Technology
Delft, The Netherlands

ABSTRACT

In recent years, database education has been receiving more attention, with research in various directions such as the development of tools for education, the analysis of students' homework, and the exploration of misconceptions. Misconceptions are mistakes in student reasoning that lead to errors during problem-solving. Recent work has documented misconceptions and errors in SQL. In this study we test the prevalence of several of these misconceptions through a multiple-choice questionnaire, to see if they hold on a larger, more diverse, student population. We found that all misconceptions are held to some extent, with prevalence scores ranging from one to fifty-two percent of the student population. Additionally, we have uncovered previously unidentified areas of struggle, allowing us to identify new misconceptions.

CCS CONCEPTS

• **Information systems → Structured Query Language**; • **Social and professional topics → Computing education**.

KEYWORDS

Misconceptions, Data Systems Education, Databases, SQL

ACM Reference Format:
Daphne Miedema, Michael Liut, George Fletcher, and Efthimia Aivaloglou. 2023. "There is no ambiguity on what to return": Investigating the Prevalence of SQL Misconceptions. In *23rd Koli Calling International Conference on Computing Education Research (Koli Calling '23), November 13–18, 2023, Koli, Finland*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3631802.3631821

1 INTRODUCTION

Database Education as a research field has been on the rise in recent years [4]. Various areas of research have been focusing on the teaching and learning of the Structured Query Language (SQL); the errors students make have been identified and categorised [2, 38, 50], tools have been developed for supporting students in query writing [16, 28, 32, 33] and repair [26, 31], and visualisations have been proposed for assisting in query understanding [14, 23, 28]. Several existing works have concluded that SQL, simple as it may appear, is challenging for novices.

Recent research has identified misconceptions as underlying causes of student challenges and errors. This line of work was initiated by Taipalus [49], who also discussed the potential causes behind persistent SQL errors [50], and was continued by Miedema et al. [27], who collected and analyzed qualitative data on the thought process of 21 students while they were working on query formulation problems. In their work, they identified fourteen misconceptions in four categories: misconceptions based on previous course knowledge, generalization-based misconceptions, misconceptions based on language, and misconceptions due to an incomplete or incorrect mental model [27].

This paper builds on that work. It is inspired by the importance of research on understanding the problems novices face with SQL, and the design of interventions to support them. Currently, even though a set of misconceptions has been identified for SQL, it is still unclear how representative these misconceptions are for a wider population of students. This paper aims to investigate the prevalence of a subset of previously identified misconceptions among a diverse student population, as well as the reasoning behind erroneous student answers. Our goal is to answer the following research questions (RQs): **RQ1**: What is the prevalence of previously identified SQL misconceptions in a diverse student population? **RQ2**: What are students' reasonings when exhibiting misconceptions?

To answer our RQs, we created a set of multiple-choice questions (MCQs), similar to previous studies on misconceptions in other computer science domains [35, 48, 59]. Through our questionnaire, we tested 12 of the misconceptions identified by Miedema et al. [27]. We included several documented safety mechanisms to ensure that participants' responses are based on their knowledge and are not guesses or accidentally correct answers (e.g. certainty of response and an alternative question for each misconception). Question dependence was subsequently validated to ensure that each MCQ pair was capturing the same misconception. Moreover, we included an open-text field per question, for participants to elaborate on their answer selection, which provided us with additional qualitative data. The questionnaire was administered to

Koli Calling '23, November 13–18, 2023, Koli, Finland

ACM ISBN 979-8-4007-1653-9/23/11.
https://doi.org/10.1145/3631802.3631821

249 participants from six universities in four different countries on three different continents.

Our results show that all identified misconceptions were held to some extent, with prevalence scores ranging from one to 52% of our population. Moreover, we discover previously unidentified areas of struggle, leading to the identification of ten new misconceptions. We hope our findings will be helpful in improving SQL education.

2 RELATED WORK

2.1 SQL Education

Research on human aspects of Structured Query Language (SQL) started shortly after its inception in 1974. For example, in 1981, Phyllis Reisner examined the Human Factors that could play a role in understanding various query languages, including SQL [41]. In a later article, she examined the ease of use of SQL through performance in query formulation and query interpretation [42]. In that same article, she also provides some numbers on which SQL functions students seem to struggle with (such as GROUP BY and computed variables), and identifies some common syntax errors such as spelling mistakes, usage of synonyms and incorrect punctuation [42].

Much of the subsequent research on SQL in this area also focuses on errors, either on syntax [1], semantics [3, 7], or both [2, 38, 51]. A very extensive study was undertaken by Taipalus and Perälä [50], who analyzed more than 123,000 SQL queries to identify which error types were persistent. These are errors that were not corrected by the students. They found that logical errors (where the answer does not match the question that was asked) and complications were more persistent than syntax and semantic errors. Some of the most common persistent errors had to do with JOIN conditions, either being inconsistent, missing, or incorrect [50].

This error-focused research is useful for uncovering which concepts are bottlenecks for students' understanding. Complementary avenues for research remain open. Taipalus and Perälä indicate the need for further study of SQL with advanced SQL commands and realistic research settings [50]. We should also consider how we can increase students' understanding of SQL. In this paper, we elaborate more on cognitive understanding in subsection 2.2.

Finally, significant progress has been made in tool development for education. There exist various tools for supporting query formulation through different media [15, 16, 32]. Tools were also developed to give students feedback on their queries [11, 21, 33]. Other tools identify code smells to help students improve or repair their queries [26, 31], or use visual elements to remove some of the students' mental load during query formulation [9, 14, 23, 28].

2.2 SQL Misconceptions

Researchers have also started exploring the cognitive processes behind learning SQL. The main direction of this work has been (mis)conceptions, also sometimes called alternate conceptions. Misconceptions research is central to effective teaching, as we need to know the nature of the misunderstandings that result in difficulties or erroneous code being written in order to help our students repair these misunderstandings.

Misconceptions have been considered as "faulty extensions of productive prior knowledge" that should be identified and refined [20]. In programming, students' prior knowledge, especially from natural language and mathematics, has been found to be transferred and cause misconceptions [10, 40]. Linguistic transfer is especially relevant to our work on SQL errors, since programming keywords borrowed from the English language can have ambiguous or alternate meanings [6, 39]. One example is the word *and*, which is a conjunction in English but a Boolean operator in programming. Pea found that novice programmers believed that an *if* statement was continuously actively waiting for the Boolean condition to be true, as used in natural language [36]. Other works have attributed misconceptions to the misapplication of prior knowledge of mathematics, especially algebraic notation. For example, Bayman et al. found this knowledge to cause confusion on variable assignment statements [5] and Sorva found it to affect the understanding of execution sequences [46]. Prior exposure to different programming languages may also to cause misconceptions [43], for example, due to the differences in the notations and their use for the definitions of variables and arithmetic operators.

Research on programming misconceptions has focused on the understanding of concepts in imperative and object-oriented languages [25, 46] - an extensive list is presented by Sorva [47]. Apart from languages commonly taught in CS1, recent work on misconceptions has focused on visual programming languages used by younger learners [48], functional programming languages [13], as well as difficulties with the understanding of data structures [58].

Two main works on potential misconceptions in SQL exist. The first work was written by Taipalus [49], who discusses causes that could be behind the persistent errors explored they found before [50]. He maps the errors to four cognitive explanations introduced by Smelcer [45]. Although a good starting point, this work has the drawback that it is speculative: Taipalus used only plain text queries for analysis, without data exploring students' thought process [49].

More recent work on SQL misconceptions was undertaken by Miedema et al. [27]. They ran a think-aloud study in which they asked students to complete query formulation problems. This work identified four top-level categories: misconceptions based on previous course knowledge, generalisation-based misconceptions, language-based misconceptions, and misconceptions due to an incomplete or incorrect mental model.

2.3 Testing for Misconceptions

MCQs are a common way to test students' knowledge, both within a classroom environment and a research setting. There are many advantages to MCQs, such as ease of (volume) grading, heightened student confidence, being able to cover a wider range of topics than a traditional test, and (seeming) objectivity [22]. However, although students' performance on MCQs and constructed-response questions correlates, exclusively using MCQs to evaluate student performance in a classroom setting may not give the right perspective on a student's skills [22]. For example, students are not able to show their thought process, reason outside of the box, and cannot gain partial credit.

To reduce this effect, we can improve the design of MCQ tests while maintaining their advantages. Tamir introduced the concept of two-tier MCQ tests, which adds a justification question on top of each original question [52]. This justification can help to assess

learning and misconceptions, and students can use it to illustrate their thought processes. Alternatively, we could add a *certainty of response index* [17], which gives us information on whether our students are guessing the answers to the test, or actually (think they) know the answers. We can also combine these approaches into a three-tiered design as defined by [55]. Finally, to reduce the effect of guessing, we can ask students to select all answers that they think are correct. If we also have multiple correct answers, getting all correct answers right and not selecting any erroneous answers reflects mastery of a topic.

MCQ test designs have been used to test programming knowledge before. For example, Whalley et al. used one-tier MCQs to test programming comprehension [56]. Ma examined students' understanding of program execution and assignment statements by means of MCQs and students' notes [24]. Swidan et al. used MCQs to identify misconceptions, and included the *certainty of response index* [48].

Another application of MCQs in testing are Concept Inventories (CI): highly researched tests taken by students to investigate whether they understand the central concepts of an area of study. The first CI was the Force Concept Inventory [19], which centered on Newtonian Laws. More recently, CIs have been under development for various subfields of Computer Science. For basic programming knowledge, the FCS1 [53] and SCS1 [34] were developed, Herman et al. developed a CI for Digital Logic [18], Porter et al. created one for Basic Data Structures [37], and Wittie et al. worked on a CI for CS2 [57]. Research outcomes of applied MCQ studies can inform the development of future CIs.

3 METHOD

Our aim in this paper is to identify the prevalence of misconceptions on SQL in a widely varying student population. To this end, we created a multiple-choice questionnaire with questions that should evoke the misconceptions identified by Miedema et al. [27]. We used MCQs as they have been shown to work for identifying misconceptions for programming languages [24, 44, 48]. They also make it possible to run the questionnaire on a large scale. Additionally, the questions should exclusively evoke the misconceptions we are exploring. Open questions could make it difficult to quantify this.

Some of our questions had more than one correct answer. These are only counted as correct if the participant has exclusively selected correct answers. If not all correct answers are selected, we mark answers as incomplete.

3.1 Participants

The authors of this paper reached out to database course instructors within their network, to identify courses in which the questionnaire could be applied and helpful for the students. Instructors then shared the questionnaire with their students as study material for the exam. To add value for the participants, the questionnaire was set up as preparatory material. This meant that, upon completion of the questionnaire, the participants received an overview of their answers, the correctness per question, and some feedback on incorrect answers.

This study had 249 participants from six universities in four different countries on three different continents. We have gender data for 230 of these participants, where 195 participants identified as male, 45 identified as female, four participants preferred not to say, four identified as non-binary and one participant self-described their identity. The median age of our students was 21 ($\mu = 21.43$, $\sigma = 2.79$), with a minimum of 18 and a maximum of 45.

3.2 Materials

For the questionnaire we selected 12 different misconceptions from Miedema et al. [27]. We introduce four different question types: (1) Fill in the blank in the query, where options to fill in the blank were given. The participants could select more than one answer. (2) Select the correct query for the given natural language question. The participants could select more than one answer. (3) Given a natural language question and a SQL query, the participants were asked to evaluate whether this was the correct translation. (4) Given a natural language question and a SQL query, the participants were asked a meta-question about the query.

As we wanted to make sure we were measuring misconceptions, instead of guesses or accidental correct answers, we introduced four safety mechanisms.

(1) For each of the misconceptions tested, we generated two different questions that should evoke the misconception. These were typically of two different question types as mentioned above. The questions were not asked pair-wise but mixed up with questions on different misconceptions as long as these questions would not give hints on the correct answer to the following questions. To validate that the questions were capturing the same misconception, we calculated the dependence between the pairs with a χ^2 test.
(2) For each question, we asked how certain the participant was of their answer on a Likert scale from one through seven. This allowed us to disregard those answers that are indicating a misconception but were shown to be guesses. After the exclusion of guesses, we calculated the aforementioned dependence between each pair of misconceptions.
(3) We offered the participants a text field in which they could elaborate on their answers. Their answers to the question: "*Can you elaborate on why you think it is correct?*" were used to strengthen our interpretations for the misconception we were evaluating, as they regularly reflected faulty logic in the participants' thought processes.
(4) All questions were checked by four SQL educators who were not on the research team, to test for clarity of formulation, non-determinism of answers, plausible distractors, and possible coding errors.

Ten of the pairs of questions used in this questionnaire have been published previously as the Multiple-choice SQL Misconceptions Instrument (MSMI1) [30], with the exception of two pairs of questions that were found to be independent.

3.3 Pilot Study

To decide how many SQL misconceptions could reasonably be addressed during the study, and to make sure no problems were introduced upon digitizing the questionnaire, we ran a small pilot

Misconception name	certainty	χ^2	Question 1			Question 2			
			mis-conception	error/ incomplete	correct	m	e/i	c	%
BRACKETS	5.36	p=1.62e-13	21	0	228	32	0	217	11
NEQ	6.05	p=8.52e-70	17	112	120	13	102	134	6
IS	5.75	p=1.36e-99	71	3	175	70	2	177	28
SCOPING_WITH	4.60	p=1.62e-04	67	64	118	64	43	142	26
DISTINCT	4.82	p=8.83e-03	38	114	97	18	61	170	11
MISSING_ALIAS	5.46	p=4.31e-33	33	110	106	29	125	95	13
PK_DISTINCT	5.98	p=1.08e-06	84	NA	165	35	NA	214	24
UNDERSTANDING_PK	5.35	p=1.18e-06	10	17	222	15	122	112	5
EQ_PK	5.25	p=1.13e-03	38	29	182	76	18	155	23
ALIAS_SYNTAX	5.36	p=4.05e-11	1	56	192	3	123	123	1
SCOPING_SELFJOIN	5.16	p=0.105	24	34	191	46	68	135	14
MISSING_JOIN	4.37	p=0.232	144	NA	105	113	NA	136	52

Table 1: Counts per misconception question. pk stands for primary key, NA for not applicable. For the ten misconceptions in the top part of the table, the χ^2 score indicated that the questions were dependent. The bottom two are independent questions.

study. For this pilot, we used the materials as described above. However, instead of asynchronous administering of the questionnaire, we recruited a group of nine participants from the same target group as the main study and colocated them to simultaneously take the questionnaire.

The pilot started with the researcher explaining the purpose of the questionnaire: testing the participants' SQL knowledge. The participants were asked to take the test, focusing on both speed and accuracy. Then, if they found something they did not understand, or identified possible mistakes, they were asked to immediately raise their hand to share the problem with the experimenter. During the pilot study, we found that the questionnaire as written took our participants between 60 and 90 minutes and contained no major problems. We corrected minor errors, finalized the questionnaire and continued with recruitment and running the main study.

3.4 Data Analysis

For the quantitative data, we encoded each answer as either correct, erroneous, incomplete or containing a misconception. Various questions had the option to select more than one answer. For these questions, we coded the participant as 'having' the misconception if any of their selected answers contained the misconception. From this set we removed all participants that indicated they were unsure about their answer, counting only the ones that indicated a certainty of five or higher on a Likert-scale from one to seven.

We analyzed whether our pairs of questions could be considered dependent; whether a misconception in question one also meant a misconception in question two. We also checked if the misconceptions were dependent on one another; whether holding misconception one meant also holding misconception two. This was done through χ^2 tests.

The qualitative coding was done in pairs. The first author explored the free text answers to identify codes to start from. Then, the set of misconceptions was split in two, with each set independently being coded by two authors. Each author could create any new and extra codes that they felt were necessary. After coding, the pair of authors discussed any new codes they introduced and resolved any conflict between their coding.

4 RESULTS

In this section, we discuss the qualitative and quantitative results for the misconception categories in our questionnaire. Each category has its own section below. For an overview of our findings, you can find the misconception prevalences in Table 1, the relations between the misconceptions in Table 2, and the coding summary of qualitative answers in Table 3. Please note that all quotes that are printed in this paper are literal quotes and may contain spelling mistakes.

General results. Before we dive into each category, we first make some general observations.

The χ^2 tests of independence revealed that ten out of our twelve pairs of questions are dependent. The independent pairs are SCOPING_SELFJOIN and MISSING_JOIN. For the former, most students struggled much more with the second question than the first, so the questions may not have been of equal difficulty. For MISSING_JOIN, we later found out that there were some problems with the question (see subsection 4.12).

On top of the dependencies within pairs, we also calculated the dependencies between pairs to see if students holding one misconception are more likely to also hold another. For all scores, see Table 2. We found 19 pairs of misconceptions that seem to have a dependency, with eight of these being strong correlations.

Our codings for the qualitative answers are in Table 3. As can be seen in the table, of all elaborations on the misconceptions, only approximately half are informative, in that they mention something about the thought process of the student specific to the question. We analyze the question-specific answers per question in the sections below. The other half of the open answers can be divided in two. The majority of them are empty or filler answers ("nan", "nothing to say", "this is the correct answer"). Alternatively, they say something superficial about the thought process and decision making, such as it looking familiar, being taught that way, or that it was a guess. Overall, the number of open answers that indicate guessing are relatively low.

	BRACKETS	NEQ	IS	SCOPING_WITH	DISTINCT	MISSING_ALIAS	PK_DISTINCT	UNDERSTANDING_PK	EQ_PK	ALIAS_SYNTAX	SCOPING_SELFJOIN	MISSING_JOIN
BRACKETS	x											
NEQ		x										
IS	**		x									
SCOPING_WITH				x								
DISTINCT					x							
MISSING_ALIAS			*	*		x						
PK_DISTINCT	**			*			x					
UNDERSTANDING_PK	*		*				**	x				
EQ_PK									x			
ALIAS_SYNTAX	**	*								x		
SCOPING_SELFJOIN	**					**	*	*			x	
MISSING_JOIN			*				**	*	*		**	x

Table 2: χ^2 dependence scores between misconceptions. * for a p-value between 0.05 and 0.01, ** for a p-value <0.01.

Code	Frequency									
	BRACKETS	NEQ	IS	SCOPING_WITH	DISTINCT	MISSING_ALIAS	PK_DISTINCT	UNDERSTANDING_PK	EQ_PK	SCOPING_SELFJOIN
Empty or filler answer	16	10	43	11	21	18	44	18	31	23
In the book/slides/lecture notes/was taught this way	3		1				3			3
Looks familiar/right	2		7		7	1	1		2	
Guess/I don't remember/I don't know	10	2	12	13	9	9	11	4	16	14
This was not taught/I have not used this keyword before			3	3	1					3
This is the only answer that makes sense/exclusion strategy				6	1	1		2	10	9
All answers are correct				5		2				
Repeats question			1	7		1	1		2	4
Misinterpreted question							6			
Question-specific reason	5	16	74	54	17	29	53	1	53	14

Table 3: Associated explanations for answers containing misconceptions were assigned one of the codes in this table. In section 4 we describe all answers in the category *Question-specific reason*.

4.1 Using parentheses on the WHERE clause (BRACKETS)

Definition: Students believe it is syntactically required to use parentheses around the contents of the WHERE clause.

As identified in Table 3, 11% of our participants hold this misconception. They have a high certainty score for this question, with an average score of 5.36 out of 7.

Three participants explained that the WHERE clause requires brackets if it contains more than one condition: "there are multuple conditions" - participant 116

We also found a reasoning that identifies a *new misconception*. Participant 84 writes: "For WHERE (...) since we use SQL query output as a table we need to specify the SQL query boundaries so that we know where it starts and ends. COUNT() is a function that requires certain parameters - these parameters are passed inside the parenthesess." So, they believe that a WHERE clause requires brackets when we use it to output a table. In this perspective, brackets seem to almost be used as a packaging behaviour, marking the boundaries of what we do or do not use. In this sense, it may be related to the behaviour required for a subquery.

Finally, participant 17 did not care much about where they could (not) remove parentheses: "Fairly uncertain. I will be using parentheses anyway for clarity of my future code, so I did not care too much about where can I omit them." This is interesting, as not all

DBMSs allow for extra brackets, and as such, this behaviour may lead to syntax errors.

4.2 ≠ is valid syntax (NEQ)

Definition: Students believe that the ≠-sign is valid syntax to indicate inequality.

This misconception was relatively uncommon in this study, with only 6% of participants making this mistake. However, this still seems to be a high count for participants who are using their computers, who have access to their keyboard, to see that ≠ is not easy to type. Additionally, the participants' certainty in this question was the highest of all, with an average of 6.05 out of 7.

For the 30 misconception-related answers received, we had 16 participants who gave a question-specific answer. Of these, 15 participants mentioned that they felt all answer options signaled inequality. For example, participant 36 wrote: "<>, != and the other sign of the selected options mean the same (i.e. "different than")." As we can see, this participant did not stop to think that, although they knew the meaning of the symbols, they may not be valid syntax in SQL. Only one participant who selected the misconception answer did: "These all mean "unequal", I'm not sure if they can all be used in SQL though." - participant 15 Vice versa, participant 275 prioritized ≠ over <>, thinking that the former is valid syntax: "the not equal symbol I think it is correct, the <> I know that means that its not equal but I am not 100% sure that we can use it in SQL."

4.3 IS and == are valid syntax to indicate equality (IS)

Definition: Students believe that the ==-sign and IS-keyword are valid syntax to indicate equality.

Our prevalence scores show that the misconception answer was chosen 141 times over the two questions for 28% of the answers, with students being relatively certain about their answers (avg score 5.75).

Many participants chose is as a correct answer, as they felt it was familiar. For example, participant 194 wrote: "I think is can work like that, makes sense reading it aloud though, like 'city is london' makes sense in english so maybe that translates to how it functions in SQL" and participant 147 said that: "[...] is just made sense because if we can't use it under this context then when are we going to use it."

Additionally, many of our participants felt that is and = are equivalent (36) or that is is a valid operator (2). For example, participant 51 wrote: "The operator "=" and "is" is the exact same hence both will output the correct result."

This set of questions also covered the usage of == for equality, which is mentioned by Miedema et al. as a generalization from previous programming course knowledge [27]. Indeed, six participants referred explicitly to other programming languages, questioning whether this would generalize to SQL. For example, participant 263 wrote: "== is seen in many programming languages, but it might not be supported by SQL."

Interestingly, references to programming languages were also made for the is keyword: "I was never told about the possibility of using "is", but it sounds like a reasonable choice (also - it works like this in Python <3)" - participant 17

Furthermore, eleven participants wrote something about == being a valid operator. Finally, there were some participants who explained their answers with a statement suggesting that all three options (=, ==, is) indicate equivalence. Participant 33 wrote: "The second querie is wrong as it uses the different operator. All other queries are correct. The == is used in most programming languages as the conditional equality, but SQL also uses the = symbol, which at first I found strange. Using the keyword is also works."

The question for this misconception was: Which of the queries below answers the question: what are the IDs of customers who have ever purchased a product for the highest unit-price of all products?

The intended answer was:

```
WITH maxprice AS (
  SELECT MAX(unit-price) AS price
  FROM inventory)
SELECT t.cID
FROM transaction t, inventory i,
     transactionitem ti, maxprice
WHERE ti.pID = i.pID
AND t.tID = ti.tID
AND t.date = i.date
AND t.sID = i.sID
AND i.unit-price = maxprice.price
```

The misconception answer was:

```
WITH maxprice AS (
  SELECT MAX(unit-price) AS price
  FROM inventory)
SELECT t.cID
FROM transaction t, inventory i,
     transactionitem ti
WHERE ti.pID = i.pID
AND t.tID = ti.tID
AND t.date = i.date
AND t.sID = i.sID
AND i.unit-price = maxprice.price
```

Listing 1: Question two for SCOPING_WITH, with the intended answer and the misconception answer.

4.4 Not including the Common Table Expression (CTE) name in the FROM clause (SCOPING_WITH)

Definition: Students believe that the results from the CTE are accessible without including the table in the main query.

Our prevalence scores show that this misconception was the third-most common, with 26% of the answers containing the misconception. It has a certainty score on the low end, with many participants close to guessing (the average score was 4.60 out of 7). One of the two questions in this pair is included in Listing 1.

Out of the 131 misconception answers, we have 55 that present a question-specific reasoning. Approximately half of these are participants who were struggling with the question. Two participants explained that the question was too long for them (to read or understand), and 26 seemed to not have noticed the difference between the two versions of almost correct answers, one of which contained the misconception.

In the group of participants who did notice the difference (to (not) include the name of the CTE in the FROM clause), we identified three lines of reasoning:

(1) Adding the name of the CTE does not make sense/does not seem necessary. This perspective was reflected by the explanations of four participants. Participant 24 wrote: "maxprice were defined in the beginning, so mentioning it in where clause did not make sense"
(2) Adding the name of the CTE is redundant. Fourteen participants held this view. Participant 78 explained: "I believe you need to be comparing all ID values as the first two options do, but the second option includes maxprice in the FROM clause which is redundant"
(3) Adding the name of the CTE is not allowed. Eight participants held this view. In their very short explanation, participant 248 wrote that: "'FROM maxprice' is not allowed"

Finally, this question uncovered a *new misconception* held by two participants (attending different universities). They argued that the CTE was not a relation, and as such could not be called in the FROM clause. This raises the question of what they think the CTE is, if not a relation, and warrants further research and interviews.

The question for this misconception was: Suppose we want to find the names of all customers who have a shopping list for the first day in January 2022 that such a shopping list exists. Is this a correct query to answer this question? You can assume that dates are encoded as strings in dd-mm-yyyy.

```
SELECT cName
FROM customer c, shoppinglist s
WHERE c.cID = s.cID
AND date =
    (SELECT DISTINCT date
    FROM shoppinglist
    WHERE date LIKE "%-01-2022")
```

The answer options were:

(1) Yes, the query is correct (misconception)
(2) No, the query returns an incorrect answer (error)
(3) No, the query has a syntax error (correct)
(4) It is impossible to determine whether the query is correct (error)

Listing 2: Question one for DISTINCT.

4.5 DISTINCT will take the first item from a list (DISTINCT)

Our prevalence scores show that this misconception was on the rarer end of the spectrum, with 11% of the answers. It has one of the lowest certainty scores too, although the participants are still well above guessing (with an average score of 4.82 out of 7). One of the questions of this pair is available in Listing 2.

Miedema et al. hypothesized that this mistake originates from having only a few examples illustrating what DISTINCT does, leading to incorrect extrapolations of that knowledge [27]. This misconception was also described in a different form by Brass and Goldberg as *subquery term that might return more than one tuple* [7].

For this question we have found relatively few question-specific elaborations. Many participants indicated they guessed or found the answer by excluding other answers. We did find two who literally stated that DISTINCT takes the first item from the list, as indicated by Miedema et al. Participant 118 understands that DISTINCT filters, but also believes it takes the first item: "Since order by defaults to ascending order I believe any of these answers are correct. MIN will produce the min[,] distinct will produce the first distinct lowest sID[,] and s.sID will produce the first lowest."

Eight participants did not notice that the subquery returns multiple answers, or did not realize that = does not accept a list of inputs. E.g., participant 62 notices that the subquery uses LIKE and this will return a list of outputs, but does not connect this to its impact on the main query: "The like operator is going to return all results with that same type."

Taking this further, participant 36 showed us a *new misconception* about subqueries. They believe that using the structure of `attribute = (subquery)` will allow the main query to take the first item from the subquery's result table. They write: "This query will return the names of customers who have made a shopping list for first of Jan, because % matches the all dates of Jan 2022, but date = takes only the first "

Another *new misconception* came from participant 84, who believes that "% sign allows us to use current date." This is not so strange in and of itself, because many programming environments allow us to use placeholders to retrieve 'variables' from the real world. However, this does not explain what they think of the LIKE keyword next to it.

4.6 Missing alias in the SELECT clause (MISSING_ALIAS)

Definition: Students believe that it is not necessary to indicate which table they want to use an attribute from, and thus forego the required alias.

Our prevalence scores show that this misconception was not so common, with 13% of the answers, but, the certainty score indicates our participants were relatively sure of their answers (an average of 5.46 of 7). Question one for this misconception can be found in Listing 3.

Eighteen of the explanations indicate participants' beliefs that all of the answer variants point to the correct attribute (cID, s.cID, c.cID for Q1, sName, s.sName, p.sName for Q2). However, the one without an alias prefix will not work, given that the cID/sName will

be ambiguous. We also identified three *new misconceptions* through this question.

(1) According to some of our participants, the JOIN condition *will remove the ambiguity* that comes from not using the alias. For example, participant 196 writes that: "there is no ambiguity on what to return because we test c.cid = s.cid". What this means is not entirely clear. Perhaps, to these participants, the JOIN condition creates a literal connection between the two tables/attributes as it would for a variable or a key-value store, such that calling the one also calls the other. This misconception was held by six participants and indicated seven times in text.
(2) The second new misconception is that the JOIN condition of the query will create a unique new column. As such, no alias is required in the SELECT clause: "there is only one column called sName so the table doenst need to be specified" - participant 158. This misconception was held by three participants.
(3) Finally, one participant wrote that not using an alias will remove the duplicate entries within a column: "disticnt, so c.cID and s.cID would find them all, and cID would remove duplecites" - participant 248.

```
SELECT DISTINCT [ ]
FROM customer c, shoppinglist s
WHERE s.cID = c.cID
AND c.city = 'Berlin'
GROUP BY s.cID
HAVING COUNT(DISTINCT s.date) >= 5
```

The question for this misconception was: Find the IDs of customers from Berlin who have made at least 5 shopping lists for different dates. What is the correct expression to fill in the blank? (select one or more)
The answer options were:

(1) cID (misconception)
(2) s.cID (correct)
(3) c.cName (error)
(4) c.cID (correct)

Listing 3: Question one for MISSING_ALIAS.

4.7 Using DISTINCT on primary keys (PK_DISTINCT)

Definition: Students believe DISTINCT is required to retrieve unique elements, even if the attribute is a primary key.

As adding DISTINCT on a primary key does not lead to an incorrect query, but merely complicates it, in our question in this study we ask for the *shortest* correct query, making the answers containing DISTINCT explicitly incorrect. Our prevalence scores show that the misconception answer was chosen 119 times over the two questions for 24% of the answers, with students being very certain about their answers (with an average score of 5.98 out of 7).

Forty participants say that DISTINCT is required to remove duplicates. It seems that the main reason that students use DISTINCT is that this is a rule they have learnt: If the question mentions unique answers, then they apply DISTINCT. This is illustrated by the following concise explanation of participant 75: "unique = distinct". But even when students show to have a bit more insight into the workings of SQL, this template seems to hold up. As participant 236 writes: "needs second query in DBMS with bag semantics.", where the second query is the one including DISTINCT.

In our participants' elaborations, we also find a strong reflection of a lack of knowledge of primary keys. Seven participants wrote explanations that amount to the conclusion that IDs are not unique. For example, participant 11 wrote: "Query without select would just return all the cID's in the customer table which could be repetitions." And participant 275 wrote: "Distinct gets only once the store id and not the duplicates"

Related to these primary keys is the idea that SQL is not set-based (mentioned by three participants), and thus may have duplicates. As participant 292 wrote: "It depends on whether store is a set or a multi-set" A multi-set, or bag, is a variant of a set that allows for duplicates. However, from the database schema, the participant could have seen that store IDs are primary keys and thus do not contain duplicates.

Finally, there was one noteworthy explanation of why DISTINCT was chosen, that we think illustrates a *new misconception*. It hints at a relation between DISTINCT and the effects of a Cartesian product. The explanation, by participant 31, is as follows: "I think if you don't mention distinct it combines all sIDs with all store names." It seems that this participant confused DISTINCT with the JOIN conditions that should be included in the WHERE clause. This answer warrants further investigation into the understanding of DISTINCT.

The question for this misconception was: Which of the following queries is the shortest correct query to retrieve a list of names of the customers who have at least one transaction?
The intended answer was:

```
SELECT c.cName
FROM customer c, transaction t
WHERE c.cID = t.cID
```

The answer that most students chose:

```
SELECT c.cName
FROM customer c, transaction t
WHERE c.cID = t.cID
GROUP BY c.cName
HAVING COUNT(tID) >= 1
```

Listing 4: Question two for UNDERSTANDING_PK, with the intended answer and the erroneous answer chosen frequently.

4.8 Lack of knowledge of primary keys (UNDERSTANDING_PK)

Definition: Students do not understand the implication of the presence of a primary key.

Misunderstandings of primary keys were relatively rare for this question in our study. Only 5% of the answers could be considered

misconceptions. However, there were many errors in question 2, in which students suggested that using GROUP BY, HAVING and COUNT was better than using a plain query (see Listing 4). So, it seems that this query uncovered some other bias that we were not measuring.

For this question, only one student gave a question-specific reason for their answer, and this seems to be a *new misconception*. They confused the concept of the primary key with that of a JOIN condition. For question 1, participant 261 writes: "the names should be the same thus one of them should be part of the primary key."

4.9 (Not) equating primary keys (EQ_PK)

Definition: Students do not understand when they need to add join conditions and comparisons, and when they do not. For this category, we measured their tendency to compare a primary key and foreign key when it was not necessary.

Our prevalence scores show that the misconception answer was chosen 114 times over the two questions for 23% of the answers, with students being relatively certain about their answers (with an average score of 5.25 out of 7).

The most common reasoning for our participants including an exclusion is that *we need to make sure the items are not the same* (21 times). This template reasoning is mentioned by participant 279: "considering pID is the primary key, we definitely need to check it, but we should also check the pName, as 2 products with different id's can have the same name apparently." Indeed, in the question we are asking to return all different product names. As the participant mentions, indeed it could be the case that products with differing IDs have the same name. This is why they need to check for different names, but do not need to check for different IDs (as they are unique).

Related to this misconception is that *products with a different ID have a different name.* This was believed by at least ten participants and reflects a lack of understanding of primary keys. If one believes this, it is convenient to only check the IDs and ignore the names, as participant 272 expresses: "The second one is the correct option because if the products have different ID's they will definitely have a different name. This is because pID is the primary key of product."

Finally, there were many participants who thought that IDs were not unique. We saw this already in subsection 4.7, but it seems a deep-rooted issue. For the questions in this category, it meant that the participants believed that products with a different name may have the same ID (17 instances). Participant 268 writes as explanation: "we want to make sure the pName is not the same but also the pID is not the same otherwise pairs of the same pID are always returned."

4.10 Alias behind the attribute (ALIAS_SYNTAX)

Definition: Students believe a syntax of attribute.alias is correct.

The misconception answer for this question was given only four times and is thus relatively rare. This might, in part, be due to the nature of MCQs, where the alternative answers are shown which might trigger recognition.

Of the four misconception answers, one of these participants seems to have a previously undescribed, *new misconception*. Participant 280 writes: "All three options (a, b, and c) are correct because they all represent the column city, which is used to filter customers who live in Helsinki. The first two options (city and city.c) directly refer to the column city, while the third option (c.city) refers to the column city belonging to the table customer, referred to as c." It seems that this participant has different interpretations of the meaning of c, depending on the location within the query.

4.11 Only one copy of a table is required to compare within this table (SCOPING_SELFJOIN)

Definition: Students believe that comparing between rows can be done using only one copy of the table.

The prevalence table shows that this misconception occurred in 14% of our participants' answers, for 70 misconception answers. We identify three different question-specific reasonings. The first is that students think that any question that mentions counting elements requires the COUNT keyword. They have this template accessible and reach for it for every question, without checking whether the remainder of the query is correct. Four participants made arguments that *the COUNT keyword is required in order to count*; participant 85 wrote that: "only one with count in it and we need at least 2" and participant 196 writes: "we want at least 2 purchases, so we use count to determine the number of purchases."

Then there is another group of participants who also talk about counting, but are not explicit about needing the COUNT keyword. We coded these under the theme *COUNT can be applied in the WHERE clause*. Example quotes include "we can use count pid >2 and compare pid different to know there are at least 2 items in one transcation" by participant 101 and "count(pID) evaluates distinct type of products" by participant 236.

Finally, three participants suggest that *we can compare data within rows*. For example, participant 14 reasons about the answers: "Count means a customer has 2 houses in 2 different cities. Another one also checks for street. These are both wrong, I think we only have to fetch one table customers and in that table we can compare if c.city != c.city"

These participants believe that only one copy of the customer table is required to be able to find participants who live in different cities. The elaborations do not give us further insight into why they would believe this.

4.12 Missing JOIN conditions in the WHERE clause (MISSING_JOIN)

Definition: Students do not understand that JOIN conditions are required to retrieve the appropriate results, retrieving a Cartesian product instead.

This pair of questions aimed to identify students (not) using JOIN conditions. Unfortunately, despite our careful checking of the questions, it turned out there were ambiguities in question one. On top of that, students' explanations for this question mentioned that it was too long and too difficult to understand properly on a laptop screen. Although this was not reported by students in our pilot, it may have led to guessing and low certainty scores for this pair of questions. Although the prevalence of misconceptions on these questions seem high, we should disregard their results.

As to the question of how there was still a problem after all questions were checked by teachers and students during the pilot, this is potentially due to evaluating some questions without utilizing the schema. Question 1 for MISSING_JOIN had an redundant table in it, which was filtered out again by the remainder of the query. However, we assumed during construction that this table was not redundant, and thus required a JOIN condition. Some students correctly identified that this was not necessary, landing them in the misconception category.

5 DISCUSSION

In this section, we discuss our results and connect them back to our research questions.

First, we will compare our findings to those by Miedema et al. [27] that our questions were based on. We were able to find student answers reflecting the source misconceptions for nine out of twelve categories.

For BRACKETS, Miedema et al. found that students used brackets as a crutch [27, 29], and the misconception that students think the WHERE clause requires brackets. We find four participants in our population who believe this too. For NEQ, Miedema et al. [27] reason that this misconception is due to previous course knowledge, either from mathematics or studying Relational Algebra. Fifteen participants in our population mentioned that all answer options signaled inequality, and one felt that ≠ was more likely to be valid syntax than <>. For IS, Miedema et al. [27] state that the usage of is (NOT) is a language-based misconception, inferred as correct from the natural language use of this word. This is also reflected in the reasoning of 38 of our participants, especially of those who responded that is felt familiar. For SCOPING_WITH, Miedema et al. [27] make note that their students felt defining the CTE was sufficient to be able to use it. This is reflected by our participants arguing that adding the name of the CTE in the FROM clause could be seen as unnecessary (4), redundant (14), or even illegal (8). For MISSING_ALIAS, we find that 13% of our participants struggle with this concept, in our case in combination with a JOIN. These struggles with aliases have also been exposed by Miedema et al., who identified inconsistent use of aliases, complications using aliases, and scoping mistakes with regard to aliases [27]. Finally, on EQ_PK, we find that 21 participants say that "we need to make sure the items are not the same". In their results, Miedema et al. [27] also mention students applying a query template in unfitting cases.

There were also three categories that we were not able to confirm. These are ALIAS_SYNTAX, where we found other reasonings than Miedema et al. [27], MISSING_JOIN due to ambiguity in our questions, and UNDERSTANDING_PK because of low misconception prevalence.

A noteworthy observation about UNDERSTANDING_PK is that there were many submissions with errors. Especially for the second query, there were many participants who selected the most convoluted answer for a question asking for the shortest *correct* answer. As Listing 4 shows, the question involves a COUNT keyword. One possible explanation is that participants' template of counting was activated by the formulation of the question plus the presence of the answer, leading the participants to miss the correct answer. We confirmed this template-based thinking, first mentioned by Miedema et al. [27] in the questions EQ_PK as well as SCOPING_SELFJOIN, which also uses the COUNT keyword.

Finally, some of the errors that we identified require deeper insights which were not provided by our participants in their open-text responses. One important one is the relation between DISTINCT and the Cartesian product, which was deemed confusing by one participant. Another is to explore the characterization of CTEs, as some students believe that they are not relations. Finally, students are trying to find pairs of data using only one table, whereas they should use a self-join. We leave these directions for future research.

5.1 Implications for practice

This investigation provides quantitative evidence on the prevalence of SQL misconceptions and highlights the ones responsible for the biggest obstacles in learning SQL and their interactions. We believe that the implications of our results pertain to education practice, education research, and language development.

In education practice, the results highlight the concepts that require special attention in instruction. Educators might not be accurate in predicting the frequency of mistakes, as has been investigated in programming [8], but to improve upon SQL instruction, we first need to identify where possible problems in knowledge transfer occur [20]. The most common misconceptions that we identified were IS, PK_DISTINCT, and SCOPING_WITH, all three having to do with data relationships and table joins. For these concepts, one of the ways to support instruction and knowledge transfer could be the use of tools that visualize data relationships or intermediate query results such as [9, 14, 23, 28].

Research is required to address faulty knowledge refinement and reorganization for the misconceptions that this study identifies as most common. Techniques for preventing or mitigating misconceptions are under-explored for SQL. One promising direction could be SQL-specific notional machines - pedagogic devices that support explaining and understanding complex concepts through representations (such as visualizations or tool-supported representations of program executions) or analogies (such as concept metaphors, for example, a programming variable as a label or a box) [12]. Refutation texts should be researched for mitigating misconceptions; they should consist of either two or three elements: a misconception, an explanation of the correct concept, and (optionally) a cue, which helps the student understand that the misconception is incorrect [54].

5.2 Threats to validity

A threat to the external validity of our study concerns the students that answered the questionnaire, who might not be representative of all students of databases courses. This holds especially because misconception prevalence could be affected by the SQL instruction process, materials and style. To mitigate this, we recruited participants from different universities across the globe. More data could give further insights, which is why the MSMI1 is publicly available[1].

With respect to construct validity, respondents could possibly misinterpret or guess their answers to some questions. To mitigate

[1] https://doi.org/10.6084/m9.figshare.23097101

this, we included several safety measures, including the pilot test, and the certainty of response indicator, the results of which we took into account in the analysis. Unfortunately, our pilot did not catch the two pairs of questions that turn out to be independent, SCOPING_SELFJOIN and MISSING_JOIN. This is because the pilot was meant to capture errors in individual questions, and our sample size there was not large enough to calculate question dependence. As a result, we left these pairs out of our final published MSMI1 questionnaire.

6 CONCLUSION

In conclusion, in this paper we explored the prevalence of previously identified misconceptions among a diverse student population. We find that all previously identified misconceptions can lead to problems in a larger population to some extent, with misconception occurrences ranging from 1 to 52% of answers. The most commonly held misconception in our study population is that students think == and is are acceptable syntax for comparison between values IS. The least-identified misconception is ALIAS_SYNTAX with only 4 students selecting misconception answers, but which in turn had many incomplete answers. Through the open elaboration answers, we also identified ten new misconceptions:

(1) BRACKETS The WHERE clause requires brackets when it is part of a CTE.
(2) SCOPING_WITH CTE's are not relations and thus cannot be included in the FROM clause.
(3) DISTINCT The structure `attribute = (subquery)` allows the main query to take the first item from the subquery's result table.
(4) DISTINCT The %-sign in combination with a date field allows us to use the current date.
(5) MISSING_ALIAS The JOIN operation removes ambiguity such that aliases are not required.
(6) MISSING_ALIAS The JOIN operation creates a new column in the data.
(7) MISSING_ALIAS Not using an alias means the rows in the result table are distinct.
(8) UNDERSTANDING_PK Mixing up JOIN conditions and primary keys.
(9) PK_DISTINCT Not using DISTINCT leads to functionality equivalent to a Cartesian product.
(10) ALIAS_SYNTAX Placement of the alias (illegal placement is also considered) determines which column is accessed.

Our findings have implications for education practice, education research, and language development. From an educational standpoint, the results emphasize the concepts that require special attention in SQL instruction. Educators can benefit from a deeper understanding of the frequency and nature of misconceptions to improve knowledge transfer. Further research in SQL education is necessary to address the most common misconceptions identified in this study.

Moreover, this study contributes to the ongoing discussion on the SQL language itself. The findings provide quantitative evidence regarding the counter-intuitive aspects of SQL notation, syntax, and internal consistency. The prevalence of misconceptions related to the use of the IS keyword and equality signs highlights areas where SQL may benefit from further refinement and clarification.

Overall, this research advances our understanding of SQL misconceptions, informing educational practices, guiding future research endeavours, and contributing to the development of the SQL language. By addressing these misconceptions, educators and researchers can work towards improving SQL instruction and enhancing the mastery of this critical database query language.

Acknowledgments. The authors would like to thank all lecturers who helped us to gather this data by sharing the questionnaire with their students. Thanks to Alaaeddin Swidan for his help in the design of the study, and to Toni Taipalus, Andrew Petersen and Naaz Sibia for evaluating our questions.

REFERENCES

[1] Alireza Ahadi, Vahid Behbood, Arto Vihavainen, Julia Prior, and Raymond Lister. 2016. 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*. 401–406.
[2] Alireza Ahadi, Julia Prior, Vahid Behbood, and Raymond Lister. 2015. A Quantitative Study of the Relative Difficulty for Novices of Writing Seven Different Types of SQL Queries. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education*. 201–206.
[3] Alireza Ahadi, Julia Prior, Vahid Behbood, and Raymond Lister. 2016. Students semantic mistakes in writing seven different types of SQL queries. In *Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE*. 272–277. https://doi.org/10.1145/2899415.2899464
[4] Efthimia Aivaloglou, George Fletcher, Michael Liut, and Daphne Miedema. 2023. Report on the First International Workshop on Data Systems Education (DataEd '22). *SIGMOD Rec.* 51, 4 (jan 2023), 49–53. https://doi.org/10.1145/3582302.3582314
[5] Piraye Bayman and Richard E. Mayer. 1983. A Diagnosis of Beginning Programmers' Misconceptions of BASIC Programming Statements. *Commun. ACM* 26, 9 (Sept. 1983), 677–679. https://doi.org/10.1145/358172.358408
[6] Benedict Du Boulay. 1986. Some Difficulties of Learning to Program. *Journal of Educational Computing Research* 2, 1 (1986), 57–73. https://doi.org/10.2190/3LFX-9RRF-67T8-UVK9
[7] Stefan Brass and Christian Goldberg. 2006. Semantic errors in SQL queries: A quite complete list. *Journal of Systems and Software* 79, 5 (2006), 630–644.
[8] Neil C. C. Brown and Amjad Altadmri. 2017. Novice Java Programming Mistakes: Large-Scale Data vs. Educator Beliefs. *ACM Transactions on Computing Education* 17, 2, Article 7 (May 2017), 21 pages. https://doi.org/10.1145/2994154
[9] Maurizio Cembalo, Alfredo De Santis, and Ferraro Petrillo Umberto. 2011. SAVI: A new system for advanced SQL visualization. *SIGITE'11 - Proceedings of the 2011 ACM Special Interest Group for Information Technology Education Conference* (2011), 165–170. https://doi.org/10.1145/2047594.2047641
[10] Michael Clancy. 2004. *Computer Science Education Research.* Taylor & Francis Group, Chapter Misconceptions and attitudes that interfere with learning to program, 85–100.
[11] Michael De Raadt, Stijn Dekeyser, and Tien Yu Lee. 2006. Do students SQLify? Improving learning outcomes with peer review and enhanced computer assisted assessment of querying skills. *ACM International Conference Proceeding Series* 276 (2006), 101–108. https://doi.org/10.1145/1315803.1315821
[12] Sally Fincher, Johan Jeuring, Craig S. Miller, Peter Donaldson, Benedict du Boulay, Matthias Hauswirth, Arto Hellas, Felienne Hermans, Colleen Lewis, Andreas Mühling, Janice L. Pearce, and Andrew Petersen. 2020. Notional Machines in Computing Education: The Education of Attention. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education* (Trondheim, Norway) *(ITiCSE-WGR '20)*. Association for Computing Machinery, New York, NY, USA, 21–50. https://doi.org/10.1145/3437800.3439202
[13] Kathi Fisler. 2014. The Recurring Rainfall Problem. In *Proceedings of the Tenth Annual Conference on International Computing Education Research* (Glasgow, Scotland, United Kingdom) *(ICER '14)*. Association for Computing Machinery, New York, NY, USA, 35–42. https://doi.org/10.1145/2632320.2632346
[14] Kristin Annabel Torjussen Folland. 2016. *viSQLizer: An interactive visualizer for learning SQL.* Ph. D. Dissertation. Norwegian University of Science and Technology.
[15] Andreas Grillenberger and Torsten Brinda. 2012. eledSQL - A New Web-Based Learning Environment for Teaching Databases and SQL at Secondary School Level. In *Proceedings of the 7th Workshop in Primary and Secondary Computing Education*. 101–104. https://doi.org/10.1145/2481449.2481474
[16] Ryan Hardt and Esther Gutzmer. 2017. Database query analyzer (DBQA) - A data-oriented SQL clause visualization tool. *SIGITE 2017 - Proceedings of the*

18th Annual Conference on Information Technology Education (2017), 147–152. https://doi.org/10.1145/3125659.3125688

[17] Saleem Hasan, Diola Bagayoko, and Ella L. Kelley. 1999. Misconceptions and the certainty of response index (CRI). *Physics Education* 34, 5 (1999), 294–299. https://doi.org/10.1088/0031-9120/34/5/304

[18] Geoffrey L. Herman, Craig Zilles, and Michael C. Loui. 2014. A psychometric evaluation of the digital logic concept inventory. *Computer Science Education* 24, 4 (Oct 2014), 277–303. https://doi.org/10.1080/08993408.2014.970781

[19] David Hestenes, Malcolm Wells, and Gregg Swackhamer. 1992. Force concept inventory. *The physics teacher* 30, 3 (1992), 141–158.

[20] John P. Smith III, Andrea A. diSessa, and Jeremy Roschelle. 1994. Misconceptions Reconceived: A Constructivist Analysis of Knowledge in Transition. *Journal of the Learning Sciences* 3, 2 (1994), 115–163. https://doi.org/10.1207/s15327809jls0302_1

[21] Anthony Kleerekoper and Andrew Schofield. 2018. SQL tester: an online SQL assessment tool and its impact. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE 2018*. ACM Press, New York, New York, USA, 87–92. https://doi.org/10.1145/3197091.3197124

[22] William Kuechler and Mark Simkin. 2003. How Well Do Multiple Choice Tests Evaluate Student Understanding in Computer Programming Classes? *Journal of Information Systems Education* 14, 4 (2003), 389.

[23] Aristotelis Leventidis, Jiahui Zhang, Cody Dunne, Wolfgang Gatterbauer, H.V. Jagadish, and Mirek Riedewald. 2020. QueryVis: Logic-based diagrams help users understand complicated SQL queries faster. In *SIGMOD2020 (arXiv version)*.

[24] Linxiao Ma. 2007. *Investigating and Improving Novice Programmers' Mental Models of Programming Concepts*. Ph. D. Dissertation. Strathclyde. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=3c8efb0c95325ac2f6bb38bd3d56fdbe900e4892

[25] L. Ma, J. Ferguson, M. Roper, and M. Wood. 2011. Investigating and improving the models of programming concepts held by novice programmers. *Computer Science Education* 21, 1 (2011), 57–80. https://doi.org/10.1080/08993408.2011.554722

[26] Leonardo Mathon and Daphne Miedema. 2022. Increasing Awareness of SQL Anti-Patterns for Novices: A Study Design. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 2*. ACM, Lugano and Virtual Event Switzerland, 42–43. https://doi.org/10.1145/3501709.3544282

[27] Daphne Miedema, Efthimia Aivaloglou, and George Fletcher. 2021. Identifying SQL Misconceptions of Novices: Findings from a Think-Aloud Study. In *Proceedings of the 17th ACM Conference on International Computing Education Research*. 355–367.

[28] Daphne Miedema and George Fletcher. 2021. SQLVis: Visual Query Representations for Supporting SQL Learners. In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*.

[29] Daphne Miedema, George Fletcher, and Efthimia Aivaloglou. 2022. So many brackets! An analysis of how SQL learners (mis) manage complexity during query formulation. In *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*. 122–132.

[30] Daphne Miedema, Michael Liut, George Fletcher, and Efthimia Aivaloglou. 2023. MSMI1: Towards a Validated SQL Misconceptions Instrument. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 2*. https://doi.org/10.1145/3568812.3603471

[31] Csaba Nagy and Anthony Cleve. 2017. A Static Code Smell Detector for SQL Queries Embedded in Java Code. *Proceedings - 2017 IEEE 17th International Working Conference on Source Code Analysis and Manipulation, SCAM 2017* 2017-Octob (2017), 147–152. https://doi.org/10.1109/SCAM.2017.19

[32] George Obaido, Abejide Ade-Ibijola, and Hima Vadapalli. 2019. Generating SQL queries from visual specifications. *Communications in Computer and Information Science* 963 (2019), 315–330. https://doi.org/10.1007/978-3-030-05813-5_21

[33] Victor Obionwu, David Broneske, Anja Hawlitschek, Veit Köppen, and Gunter Saake. 2021. SQLValidator – An Online Student Playground to Learn SQL. *Datenbank-Spektrum* 21, 2 (2021), 73–81. https://doi.org/10.1007/s13222-021-00372-0

[34] Miranda C. Parker, Mark Guzdial, and Shelly Engleman. 2016. Replication, Validation, and Use of a Language Independent CS1 Knowledge Assessment. In *Proceedings of the 2016 ACM Conference on International Computing Education Research*. ACM, Melbourne VIC Australia, 93–101. https://doi.org/10.1145/2960310.2960316

[35] Wolfgang Paul and Jan Vahrenhold. 2013. Hunting high and low: instruments to detect misconceptions related to algorithms and data structures. In *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, Denver Colorado USA, 29–34. https://doi.org/10.1145/2445196.2445212

[36] Roy D. Pea. 1986. Language-Independent Conceptual "Bugs" in Novice Programming. *Journal of Educational Computing Research* 2, 1 (1986), 25–36. https://doi.org/10.2190/689T-1R2A-X4W4-29J2

[37] Leo Porter, Daniel Zingaro, Soohyun Nam Liao, Cynthia Taylor, Kevin C. Webb, Cynthia Lee, and Michael Clancy. 2019. BDSI: A Validated Concept Inventory for Basic Data Structures. In *Proceedings of the 2019 ACM Conference on International Computing Education Research*. ACM, Toronto ON Canada, 111–119. https://doi.org/10.1145/3291279.3339404

[38] Kai Presler-Marshall, Sarah Heckman, and Kathryn T Stolee. 2021. 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)*. 199–210. https://doi.org/10.1109/ICSE-SEET52601.2021.00030

[39] Ralph T. Putnam, D. Sleeman, Juliet A. Baxter, and Laiani K. Kuspa. 1986. A Summary of Misconceptions of High School Basic Programmers. *Journal of Educational Computing Research* 2, 4 (1986), 459–472. https://doi.org/10.2190/FGN9-DJ2F-86V8-3FAU

[40] Yizhou Qian and James Lehman. 2017. Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Trans. Comput. Educ.* 18, 1, Article 1 (Oct. 2017), 24 pages. https://doi.org/10.1145/3077618

[41] Phyllis Reisner. 1981. Human Factors Studies of Database Query Languages: A Survey and Assessment. *Comput. Surveys* 13, 1 (Jan 1981), 13–31. https://doi.org/10.1145/356835.356837

[42] Phyllis Reisner. 1986. Measurement of SQL Problems and Progress. Springer-Verlag Tokyo, Nemu-no-sato, Japan, 165–187.

[43] Z. Scherz, D. Goldberg, and Z. Fund. 1990. Cognitive Implications of Learning Prolog—Mistakes and Misconceptions. *Journal of Educational Computing Research* 6, 1 (1990), 89–110. https://doi.org/10.2190/UHFF-4LNQ-63VA-Q60C

[44] Simon and Susan Snowdon. 2011. Explaining program code: giving students the answer helps - but only just. In *Proceedings of the seventh international workshop on Computing education research*. ACM, Providence Rhode Island USA, 93–100. https://doi.org/10.1145/2016911.2016931

[45] John B. Smelcer. 1995. User errors in database query composition. *International Journal of Human-Computer Studies* 42, 4 (1995), 353–381.

[46] Juha Sorva. 2008. The Same but Different Students' Understandings of Primitive and Object Variables. In *Proceedings of the 8th International Conference on Computing Education Research* (Koli, Finland) *(Koli '08)*. Association for Computing Machinery, New York, NY, USA, 5–15. https://doi.org/10.1145/1595356.1595360

[47] Juha Sorva. 2012. *Visual Program Simulation in Introductory Programming Education*. Ph. D. Dissertation. Aalto University, Espoo, Finland.

[48] Alaaeddin Swidan, Felienne Hermans, and Marileen Smit. 2018. Programming Misconceptions for School Students. In *Proceedings of the 2018 ACM Conference on International Computing Education Research* (Espoo, Finland) *(ICER '18)*. Association for Computing Machinery, New York, NY, USA, 151–159. https://doi.org/10.1145/3230977.3230995

[49] Toni Taipalus. 2020. Explaining Causes behind SQL Query Formulation Errors. *Proceedings - Frontiers in Education Conference, FIE* 2020-Octob (2020). https://doi.org/10.1109/FIE44824.2020.9274114

[50] Toni Taipalus and Piia Perälä. 2019. What to expect and what to focus on in SQL query teaching. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (2019), 198–203. https://doi.org/10.1145/3287324.3287359

[51] Toni Taipalus, Mikko Siponen, and Tero Vartiainen. 2018. Errors and Complications in SQL Query Formulation. *ACM Transactions on Computing Education* 18, 3 (2018).

[52] Pinchas Tamir. 1989. Some issues related to the use of justifications to multiple-choice answers. *Journal of Biological Education* 23, 4 (1989), 285–292. https://doi.org/10.1080/00219266.1989.9655083

[53] Allison Elliott Tew and Mark Guzdial. 2010. Developing a validated assessment of fundamental CS1 concepts. In *Proceedings of the 41st ACM technical symposium on Computer science education*. ACM, Milwaukee Wisconsin USA, 97–101. https://doi.org/10.1145/1734263.1734297

[54] Christine D. Tippett. 2010. Refutation Text In Science Education: A Review Of Two Decades Of Research. *International Journal of Science and Mathematics Education* 8, 6 (2010), 951–970. https://doi.org/10.1007/s10763-010-9203-x

[55] Nurul Wahidah and Sigit Saptono. 2019. The Development of Three Tier Multiple Choice Test to Explore Junior High School Students ' Scientific Literacy Misconceptions. *Journal of Innovative Science Education* 8, 2 (2019), 190–198.

[56] Jacqueline L. Whalley, Raymond Lister, Errol Thompson, Tony Clear, Phil Robbins, P. K. Ajith Kumar, and Christine Prasad. 2006. An Australasian Study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies. In *Proceedings of the Eighth Australasian Computing Education Conference (ACE2006)*. 243–252. https://doi.org/10.1515/itit-2018-0010

[57] Lea Wittie, Anastasia Kurdia, Judy Peng, James Kelly, and Meriel Huggard. 2020. Developing a Concept Inventory for Computer Science 2: What should it focus on and what makes it challenging?. In *2020 IEEE Frontiers in Education Conference (FIE)*. Uppsala, Sweden. https://doi.org/10.1109/FIE44824.2020.9273941

[58] Daniel Zingaro, Cynthia Taylor, Leo Porter, Michael Clancy, Cynthia Lee, Soohyun Nam Liao, and Kevin C. Webb. 2018. Identifying Student Difficulties with Basic Data Structures. In *Proceedings of the 2018 ACM Conference on International Computing Education Research* (Espoo, Finland) *(ICER '18)*. Association for Computing Machinery, New York, NY, USA, 169–177. https://doi.org/10.1145/3230977.3231005

[59] Žana Žanko, Monika Mladenović, and Ivica Boljat. 2019. Misconceptions about variables at the K-12 level. *Education and Information Technologies* 24, 2 (2019), 1251–1268. https://doi.org/10.1007/s10639-018-9824-1