Kleerekoper, AJ and Schofield, A (2018)*SQL Tester: An online SQL assessment tool and its impact.* In: 23rd Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2018), 02 July 2018 - 04 July 2018, Larnaca, Cyprus.

Downloaded from: http://e-space.mmu.ac.uk/620119/

Publisher: ACM

DOI: https://doi.org/10.1145/3197091.3197124

Please cite the published version

https://e-space.mmu.ac.uk

SQL Tester: An Online SQL Assessment Tool and its Impact

Anthony Kleerekoper
SCMDT, Manchester Metropolitan University, UK
a.kleerekoper@mmu.ac.uk

Andrew Schofield
SCMDT, Manchester Metropolitan University, UK
a.schofield@mmu.ac.uk

ABSTRACT

Learning SQL can be surprisingly difficult, given the relative simplicity of its syntax. Automated tools for teaching and assessing SQL have existed for over two decades. Early tools were only designed for teaching and offered increased feedback and personalised learning, but not summative assessment. More recently, however, the trend has turned towards automated assessment, with learning as a side-effect. These tools offer more limited feedback and are not personalised.

In this paper, we present SQL Tester, an online assessment tool and an assessment of its impact. We show that students engaged with SQL Tester as a learning tool, taking an average of 10 practice tests each and spending over 4 hours actively engaged in those tests. A student survey also found that over 90% of students agreed that they wanted to keep trying practice tests until they got a "good" mark. Finally, we present some evidence that taking practice tests increased student achievement, with a strong correlation between the number of practice tests a student took and their score on the assessed test.

CCS CONCEPTS

• **Information systems** → **Structured Query Language**; • **Social and professional topics** → **Computer science education**; **Student assessment**;

KEYWORDS

Automated Assessment, Databases, SQL, Web-based Learning

ACM Reference Format:
Anthony Kleerekoper and Andrew Schofield. 2018. SQL Tester: An Online SQL Assessment Tool and its Impact. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

SQL is the standard querying language for relational databases and taught in all introductory database courses. In some ways, SQL is easier to learn than languages like Java or Python. It is syntactically smaller and more structured. Yet, students often find it more difficult to learn SQL [8, 12, 17].

A number of software tools have been produced over the last 25 years to help students learn SQL. The earliest tools were teaching focussed, providing students with increased feedback, hints and sometimes personalised learning [6, 8, 11, 17]. More recently produced tools have been focussed on assessment [10, 15, 16].

We have taught Database Systems to second-year students at Manchester Metropolitan University for a number of years. Initially, 10% of the unit assessment was based on a set of portfolio tasks that students completed and which were checked during lab sessions. The aim was to provide students with authentic tasks and assess their performance in person to provide timely formative and summative feedback. Lab sessions also included practice exercises which were not assessed.

However, our experience was that this system of assessment was not fit for purpose for a number of reasons. Firstly, it took a lot of tutor time to assess the work during the lab sessions. This meant that tutors were not available to students who were trying the practice lab exercises. Secondly, students were often ignoring the practice exercises and spending their time only on completing the portfolio tasks using a trial and error approach. The assessment format was not encouraging deep learning. Finally, there were issues with plagiarism and students working in groups on the assessment tasks and not completing them by themselves.

We therefore set out to overhaul our assessment methodology. We reasoned that replacing the portfolios with a one-time test would encourage deeper learning because students would have to make sure they had a good understanding of the topics to do well on the test. It would also solve the plagiarism problem and free up tutor time during the lab sessions.

Traditional options, such as a written test or online multiple choice quiz, were ruled out because they lacked authenticity [18]. Instead, we examined the literature on SQL assessment and adopted the approach of Prior and Lister with AsseSQL [15]. Their tool has been in use for over a decade and has received positive student feedback.

In this paper, we describe SQL Tester and assess its impact on student engagement and performance. Our description of the tool is in Section 3, where we also point out the main differences between SQL Tester and AsseSQL and explain our choices. The most significant differences are in the choice of question categories; the inclusion of the database schema and the randomisation of question order.

We also present an assessment of the impact of SQL Tester, with a focus on student engagement. We show that students took an average of 10 practice tests each, spending over 4 hours actively engaged in those tests (Section 4). We present the results of a student questionnaire which shows that students agreed that the tool motivated their learning and helped them revise (Section 5). Most interestingly, over 90% of the students agreed that they "wanted to keep trying" until they got the right answers and "good marks". We conclude our analysis of impact by presenting some evidence that the more practice tests a student took, the higher their mark on the final, assessed test (Section 6). This suggests that significant learning took place. Finally, we conclude with some suggestions for future work in Section 7.

2 RELATED WORK

Despite the relatively simplicity of SQL, students often struggle to write correct queries. Difficulties fall into two groups: syntactic and semantic. Kearns *et al.* suggest that students struggle to understand the basic concepts of relational databases [8]. This can result in errors such as mixing group functions with non-group attributes. Mitrovic adds that SQL presents an added burden of having to remember the database schemas leading to mistakes in naming tables or attributes [12]. These observations are supported by the quantitative analysis of Ahadi *et al.* [1].

Failing to understand the basic concepts can also lead to semantic errors such as missing join conditions, resulting in cross-joins rather than inner-joins [8]. Sadiq *et al.* point to the declarative nature of SQL which requires, in their words, thinking in "sets rather than steps" [17].

These difficulties have motivated the use of automated tutoring systems. The use of such systems for SQL arguably started 25 years ago with the work of Dietrich [6, 7]. She implemented a tool for students to practice a number of relational query languages, including SQL, and receive useful and immediate feedback.

Kearns *et al.* presented *esql*, a tool that displays the results of an SQL query and also pictures showing, step-by-step, how the query was evaluated [8].

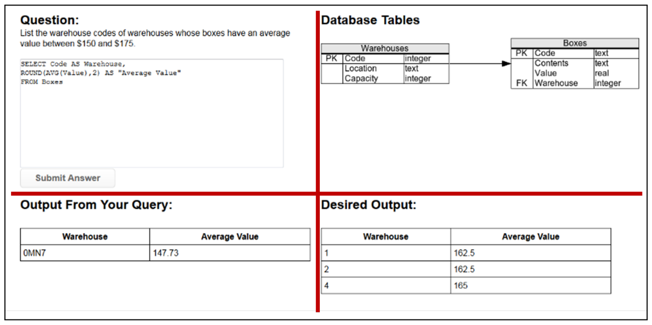
Mitrovic created the first Intelligent Tutoring System, called SQL-Tutor, which aimed to tailor its interactions to the individual student [11–13]. She argued that one of the main difficulties with learning SQL is that error messages are limited to generic syntax checking and do not consider the semantic meaning of queries. SQL-Tutor uses hundreds of constraints to offer semantic and syntactic error messages as well as hints to students, based on their submitted answers.

The learning experience of SQL-Tutor is individualised in two ways. Firstly, students can select the appropriate level of feedback. The lowest level is simply to mark the answer as correct or incorrect. At the highest level, a partial solution is provided. The second type of individualisation is that the system chooses which questions to pose based on the errors the student has made on previous questions.

SQL-Tutor was evaluated empirically, and students reported that it was easy to use and helpful [14]. It was also shown to improve learning, with students who used the system performing better than those who did not.

SQLator was created by Sadiq *et al.* [17]. Unlike SQL-Tutor, it does not tailor itself to different students, although questions can be grouped into difficulty categories. Nevertheless, it does allow students to practice SQL independently of a tutor and receive feedback (correct/incorrect). An empirical evaluation of the system showed that 62% of the enrolled students used the system for practice. 78.5% of the students used it to help them with their assessed work, which was to answer 10 questions (not necessarily through SQLator).

Prior and Lister argued that the existing systems supported teaching but not summative assessment and therefore lacked motivation for students [15]. They created AsseSQL, an online assessment tool. AsseSQL consists of a timed test in which students must answer 7 questions in 50 minutes. Students may make as many attempts as they wish in that time, and after every attempt they are shown the output of their query - either the rows returned from the RDBMS or the error message. The seven questions are drawn from seven categories (one question per category) and in each category there are a bank of questions so that neighbouring students are unlikely to receive the same set of questions.



List the warehouse codes of warehouses whose boxes have an average value between $150 and $175.

```
SELECT Code AS Warehouse,
ROUND(AVG(Value),2) AS "Average Value"
FROM Boxes
```

	Warehouses	
PK	Code	integer
	Location	text
	Capacity	integer

	Boxes	
PK	Code	text
	Contents	text
	Value	real
FK	Warehouse	integer

Warehouse	Average Value
0MN7	147.73

Warehouse	Average Value
1	162.5
2	162.5
4	165

Figure 1: The main window of SQL Tester showing the four main segments (clockwise from top-left): Question area, database schema, desired output and output from last attempt.

AsseSQL was empirically evaluated primarily through a student questionnaire. Students reported that they preferred the online test to written work, and found it more motivating. They also agreed that practising questions interactively helped them learn.

de Raadt *et al.* critique the marking methods of Sadiq *et al.* and AsseSQL [4, 5, 16]. They therefore augment the automated marking with peer review. This improves student learning but reduces the consistency of the marks. Kleiner *et al.* used a more complex, multi-step marking method in their tool, aSQLg, which also considers the cost and style of the student's answer [9].

Very recently, License presented testSQL which is very similar to AsseSQL [10].

3 SQL TESTER

SQL Tester is based around a 50-minute test, comprising ten questions. As with AsseSQL, students may address questions in any order and make as many attempts as they want within the time. The test time can be adjusted to longer than 50 minutes for students with special learning requirements.

The questions are drawn from the nine categories listed in Table 1, with two questions about Inner-Joins. Our choice of categories is influenced by AsseSQL which had seven categories. Since our test is worth 10% of the unit we opted to have ten questions and to include more questions of simpler types. We did not include correlated subqueries or self-joins because these are too complex (see [2]). For each test, there is a bank of questions for each category and questions are chosen at random from the bank. In most cases there are between 4 and 8 questions per category per test in the bank.

As well as randomly selecting questions, SQL Tester also randomises the order of the question categories for each test instance. We were concerned that if, for example, Question 1 always required a SELECT and WHERE, then some students would learn this pattern during the practice tests. Randomising the order of the questions makes this harder. We were also concerned that students might learn patterns in the question wording, therefore we tried to make all questions as similar as possible in style.

Fig. 1 shows the main test area of SQL Tester, which is divided into four segments. Clockwise from top-left, these are: the question and text box for writing an answer; the schema for the database being used for the test, the desired output and the output from the

Question Category	Success Rate
Simple SELECT	87.3%
WHERE	81.0%
ORDER BY	83.5%
Row Function	79.7%
Group Function	86.1%
Inner-Join	62.7%
GROUP BY	55.7%
GROUP BY with HAVING	48.1%
Simple Subquery	73.4%

Table 1: SQL Tester question categories and percentage succeeding on final assessed test in each category.

student's previous answer. Every answer a student submits is saved on the server so that if a student comes back to a question later, their last attempt is pre-loaded into the text box.

Unlike AsseSQL, SQL Tester displays the schema of the relevant database alongside the question and desired output. This is to address the concern raised by Mitrovic that students struggle to memorise the database schema and therefore make mistakes in spelling of table and column names [12]. This concern is supported by the findings of Ahadi *et al.* who found that 13% of all attempted answers in AsseSQL contained undefined columns [1].

We also used slightly simpler schemas than those reported by AsseSQL, in that ours consist of at most three tables, with no self-joins. When we display the schema, we show a simplified ERD that does not include multiplicities. We do this to reduce extraneous cognitive load, leaving students free to focus on the table and column names and on the connections between columns.

SQL Tester marks an answer as correct if it exactly matches the desired output. That is, the returned results must be the same, in the same order and in the same case. We insist on case-sensitivity because some of our questions involve using row functions to change case. We insist on the same ordering because some of our questions require a specific order. de Raadt *et al.* pointed out that checking the order could cause difficulties with the DISTINCT keyword which can partially order the results [4]. However, we argue that this is a positive outcome because students ought to be using DISTINCT where appropriate (rather than an unnecessary GROUP BY, for example).

SQL Tester requires a login to use and records every submitted answer, but only uses the latest answer for marking. All test instances are saved, and students can review their answers to completed tests, although they cannot make further attempts. Because all test answers are saved as they are submitted, if a student accidentally closes the browser window, they can return and continue the test.

At present there are four practice tests and one assessed test. We define a test as a schema with a bank of questions and a test instance as a schema with ten chosen questions. The practice tests and assessed test are identical in every respect except that each one has a different schema and therefore have different banks of questions. The questions in all tests are of approximately the same level of difficulty and written in the same style.

To prevent students taking the assessed test early, the assessed test is password protected. For security, SQL Tester uses standard techniques to prevent SQL injection attacks. It also conducts checks

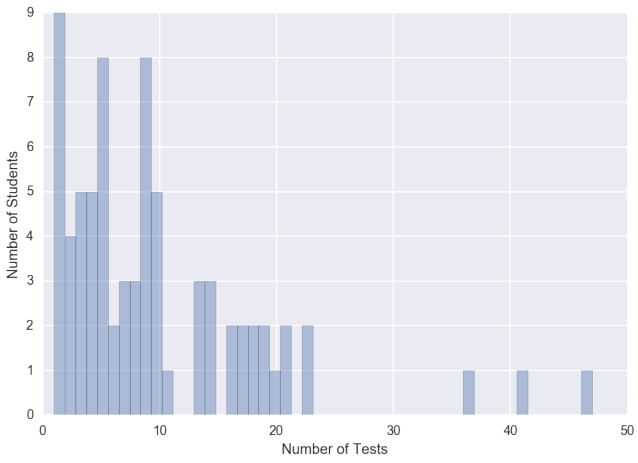


Figure 2: Distribution of the number of practice tests per student

that the only tables mentioned in the query are valid test database tables. This prevents students from attempting to modify the administration tables.

The source code for SQL Tester can be downloaded at www.github.com/kleerekoper/SQLTester.

4 ANALYSIS OF ENGAGEMENT

As mentioned, our motivation for creating SQL Tester was the desire to reduce plagiarism, free up time during lab sessions (both student and tutor time) and motivate deeper learning. We thought that students would engage with the provided practice lab exercises better in order to study for the final test. From our observations of the lab sessions, this was what happened. Students were not distracted by assessed work and therefore engaged with the exercises and completed them.

However, we also noticed that once they were introduced to SQL Tester (approximately three weeks before the assessed test), they started to spend considerable time using the system and taking practice tests.

In this section we present quantitative results showing high levels of student engagement. For the purposes of this analysis, we say that the time a student spent engaged with a test is the time between starting the test and their final submitted answer. Further, we only consider practice tests where the student engaged for at least two minutes. This excludes only 22 practice tests.

In total, 79 students were enrolled on the unit of whom 75 (95%) took at least one practice test. This exceeded the typical attendance at lab sessions which was at most 81% in the same period. It is also significantly more than the 78.5% of students report by Sadiq *et al.* who engaged with SQLator, a purely teaching tool, to help with their assignment [17].

Fig. 2 shows the distribution of the number of practice tests taken by the students. The average was 9.8 tests per student with a standard deviation of 8.9, though the distribution is skewed by a small number of students taking a very large number of tests (one student took 47 practice tests). The median was 8 tests per student.

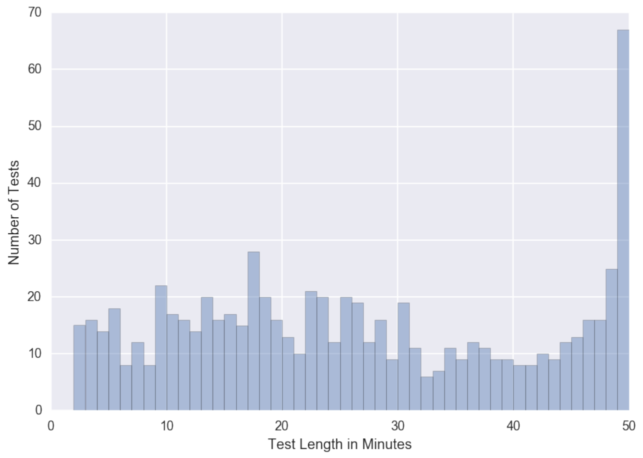


Figure 3: Distribution of the lengths of practice tests

Overall, there were 732 practice tests taken. Fig. 3 shows the distribution of the test lengths. On average, students spent 26 minutes engaged in each practice test, with a standard deviation of 14.8 minutes. The median was 24 minutes. On average, each student spent a total of 4 hours and 14 minutes on practice tests.

Further evidence of engagement is given by the number of attempted answers per test. On average, there were 31.9 answers submitted for each practice test, with a mode of 29 and a median of 28. There were six tests with more than 100 submitted answers, with the largest being 127.

These results show that students were doing far more than simply familiarising themselves with the test software. They were actively engaging with it, using it to test their understanding and practice their skills.

5 STUDENT QUESTIONNAIRE

As well as getting quantitative analysis of student engagement, we also asked students to answer an anonymous questionnaire. Of the 79 students, 34 responded (43%). The questionnaire consisted of eight statements with a five-point Likert scale.

Fig. 4 shows the statements and results. Statement 1 shows that students felt motivated to revise because of SQL Tester. In hindsight, this question could have been worded more in line with Prior and Lister's phrasing, asking whether students were motivated to "practice SQL", and perhaps include an explicit comparison to other types of assessment. A similar criticism applies to statements 4 and 8.

Over 90% agreed with statements 2 and 3, indicating that they wanted to spend longer working. 91% agreed that they wanted to keep taking tests until they could get a good mark, and 94% wanted to keep working on individual questions until they got the correct answer.

Interestingly, although more than 90% of students agreed with statements indicating a desire to spend longer learning SQL, a smaller proportion agreed with statements that asked this directly. 79% agreed that SQL Tester caused them to spend longer revising and 74% agreed that the Tester motivated them to revise. Perhaps this is the result of the wording of statements 1 and 4.

Statements 5 to 7 show that students found the different forms of feedback to be helpful in guiding them to the correct answer. Unfortunately, due to an error, the questionnaire did not include a similar statement about the usefulness of the database schema. It is probably not surprising that almost every student agreed that seeing the desired output helped guide them, given that matching the output was the criterion for an answer being correct.

Overall, the results from the questionnaire support the quantitative results in showing that SQL Tester motivated students to practice SQL and to spend longer than they would have done had the tool not been available for practising with.

6 STUDENT PERFORMANCE

The aim of this section is to provide some quantitative evidence that the students improved by using SQL Tester. However, we first remark that the average mark on the final, assessed test, was 7.5 with a standard deviation of 3.1, but this is somewhat skewed towards the lower end by some students who were unable to answer a single question correctly. The median score was 9. Table 1 shows the success rates for the different categories.

Interestingly, the average mark for the set of portfolio worksheet activities used in previous years, which SQL Tester has this year replaced, was 9.5 out of 10, with a standard deviation of 1.26. However, as discussed earlier, we believe the opportunity for students to have multiple attempts at their work based on tutor feedback and the tendency for students to collaborate on the worksheets makes these result not representative of the level of understanding of the students. In the authors' opinion, the students' understanding of SQL and relational database concepts and their ability to individually write correct SQL queries is generally superior having used the SQL Tester tool. At the end of this academic year, it will be interesting to compare this group of student's overall unit marks after having used SQL Tester with the marks from the last academic year, having completed the portfolio worksheets.

Fig. 5 shows the relationship between the number of practice tests each student took and their final mark on the assessed test. The results indicate that taking more tests led to better results. The Spearman Rank Correlation Coefficient is 0.52 ($p < 0.01$), indicating a strong correlation between the number of practice tests and the final score. Furthermore, taking the log of the number of tests (with 0 set to 0), Pearson's Correlation Coefficient is 0.6 ($p < 0.01$), indicating a strong logarithmic relationship between the number of practice tests and the final mark.

Further evidence that student performance increased as they took more tests is shown in Fig. 6. It shows the average mark for each test. "Test Number 1" represents each student's first practice test, number 2 their second and so on. The number of students taking each test is not constant as different students took a different number of practice tests. The graph is curtailed at 20 because after that there are only a few students shown.

The graph shows the trend that students did, on average, better after taking more tests. This shows that not only is there a correlation between the number of practice tests students took but that also students, on average, slowly improved.

These results are influenced by the increased likelihood that stronger students took more practice tests. This could mean that

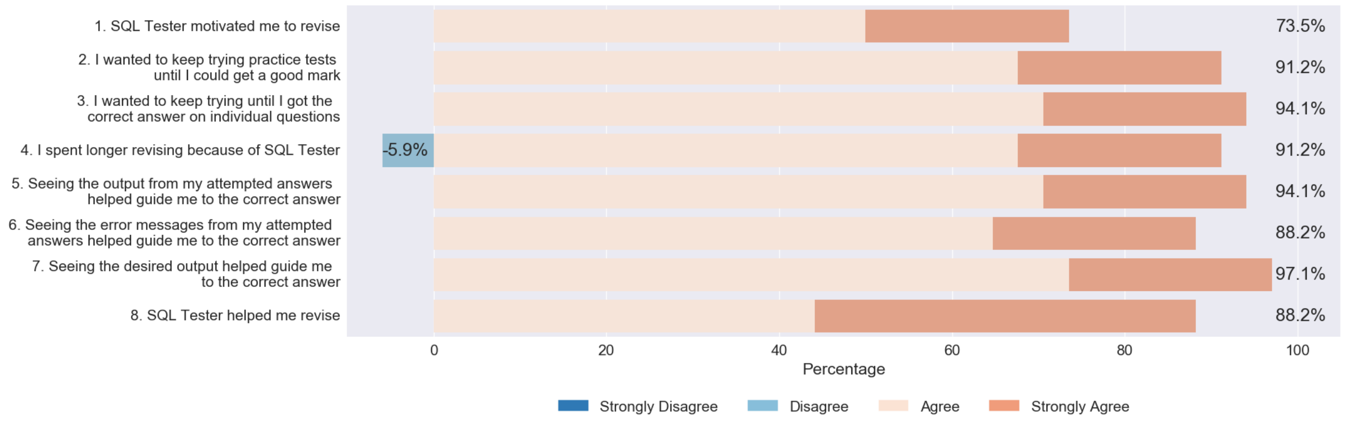


Figure 4: Plotting the number of practice tests against the student's final mark indicates that taking more practice tests increased performance.

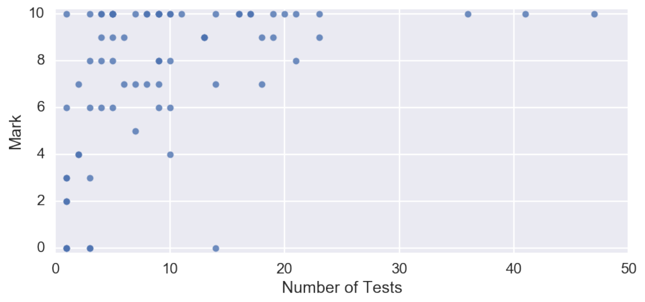


Figure 5: Plotting the number of practice tests against the student's final mark indicates that taking more practice tests increased performance.

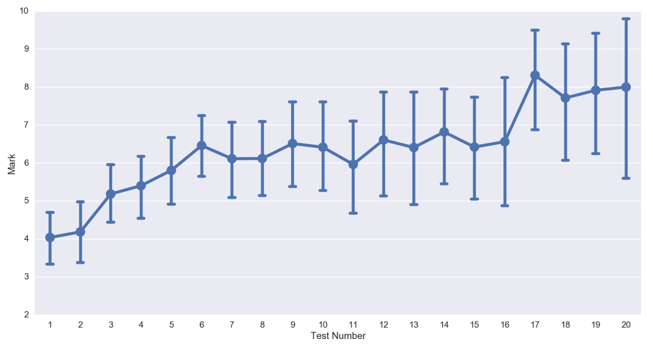


Figure 6: The average marks for student's practice tests show that the more tests they took the higher the average.

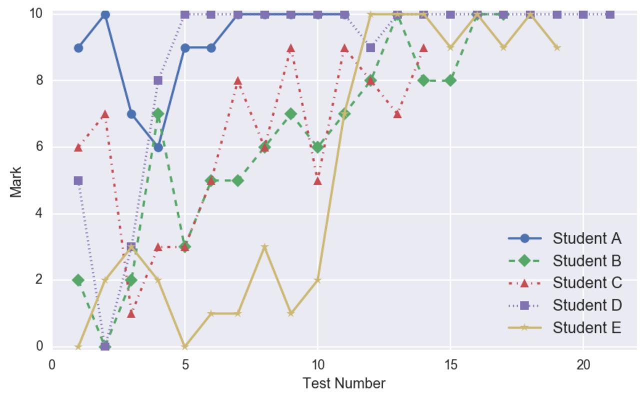


Figure 7: The marks for four the practice tests for four students and their final mark

the increased average is only the result of weaker students not taking more tests and therefore not bringing down the average mark. However, firstly, we note that a significant proportion of the students took a large number of practice tests. For example, 36 students took at least 10 practice tests, 19 took at least 15 and 10 took at least 20. We also note that in our questionnaire, two students reported that they did not revise at all because they are strong students. On the other hand, other students indicated that they only achieved good marks because of the practice tests.

To give an indication of how students used the system to improve, Fig. 7 shows the marks for five students on the practice tests they took followed by the final test. It shows that some students were high achievers to begin with but nevertheless continued to take practice tests to be sure (Student A). Others improved slowly over time (Students B and C), whilst still others showed very rapid improvement, either after only a small number of tests (Student D) or after a larger number of practices (Student E).

7 CONCLUSION AND FUTURE WORK

In this paper, we have described our online SQL assessment tool which we call SQL Tester. Our tool is very similar to AsseSQL with some differences relating to the number of questions, the question categories and the way the test is run.

We have evaluated student engagement with SQL Tester and presented results showing that students engaged strongly with it for learning. Specifically, we found that students spent over four hours taking practice tests, trying 10 tests each and submitting over 30 answers per test. We also provided results from a student questionnaire in which over 90% of the students agreed that they wanted to spend longer using the tool to get the correct answers and good marks. 75% agreed that the tool motivated them to revise and to spend longer revising. Finally, we presented some evidence that the tool improved student learning by showing that there is a correlation between the number of practice tests a student took and their final mark, with the average mark increasing with each extra test.

SQL Tester marks an answer as correct if it exactly matches the desired output. This heuristic has been criticised for allowing some form of cheating [4]. In AsseSQL, it was suggested that a second, hidden database could be used to check answers and avoid this problem. License recently suggested using a keyword check [10]. The problem with both of these options is that students cannot be made aware of why their answers are wrong with any level of detail. For example, they cannot be told which keyword is missing without giving away the answer. To say that a keyword is missing without specifying which one, also helps the student.

A better solution would be to design questions and table data that prevent any form of simple brute forcing of an answer. In this way, the only students who may be capable of "cheating", would probably be capable of getting the correct answer. How such questions could be designed is an interesting question.

SQL Tester was designed as an assessment tool and therefore no extra feedback was given, even during practice tests, other than the error messages from the RDBMS. Some students indicated that they would find hints useful. We are hesitant to include extra feedback because that reduces the authenticity of the assessment. In the real-world, one only gets the error messages from the RDBMS. On the other hand, it may improve learning and may help students understand error messages. This is an area that could be usefully examined further.

Along the same lines of retaining authenticity, we wonder whether or not students should be allowed to make full use of the Internet during the test. In the real-world, students would have access to online resources to help them solve a particular problem.

Boese argues that allowing students to use online resources results in them finding random bits of code and applying it to their problem, without proper understanding and context [3]. She suggests, therefore, that students should be allowed to use resources from the course during assessment but not online resources.

However, students will go on to use online resources during offline assignments and after graduation. We should, therefore, be training them how to use them correctly. In part, their ability to correctly use these resources will improve when their knowledge of the subject increases. In part, however, they must learn that blindly copying and pasting code will not lead to the correct answer. Giving them the opportunity to try it may help with that.

We therefore propose a future experiment in which a group of students take our assessed test twice - once with and once without access to online resources. We can then compare their results with and without access to these resources and evaluate whether their performance has increased or been hindered. Because of the question-bank model, there is less danger that their marks would increase naturally by having taken the test once. We could further reduce that prospect by having some take the test with resources first and some take it with resources second.

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 - SIGCSE '16.* 401–406. https://doi.org/10.1145/2839509.2844640

[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 - ITiCSE '15.* 201–206. https://doi.org/10.1145/2729094.2742620

[3] Elizabeth Boese. 2016. Just-In-Time Learning for the Just Google It Era. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16.* ACM Press, New York, New York, USA, 341–345. https://doi.org/10.1145/2839509.2844583

[4] Michael de Raadt, Stijn Dekeyser, and Tien Yu Lee. 2007. A system employing peer review and enhanced computer assisted assessment of querying skills. (2007), 163–178 pages. https://doi.org/10.1145/1315803.1315821

[5] Stijn Dekeyser, Michael de Raadt, and Tien Yu Lee. 2007. Computer assisted assessment of SQL query skills. In *Proceedings of the eighteenth conference on Australasian database-Volume 63.* Australian Computer Society, Inc., 53–62.

[6] Suzanne Wagner Dietrich. 1993. An Educational Tool for Formal Relational Database Query Languages. *Computer Science Education* 4, 2 (jan 1993), 157–184. https://doi.org/10.1080/0899340930040201

[7] Suzanne W. Dietrich, Eric Eckert, Kevin Piscator, Suzanne W. Dietrich, Eric Eckert, and Kevin Piscator. 1997. WinRDBI: A Windows-based Relational Database Educational Tool. In *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education - SIGCSE '97*, Vol. 29. ACM Press, New York, New York, USA, 126–130. https://doi.org/10.1145/268084.268131

[8] R. Kearns, S. Shead, and A. Fekete. 1997. A teaching system for SQL. In *Proceedings of the second Australasian conference on Computer science education - ACSE '97.* ACM Press, New York, New York, USA, 224–231. https://doi.org/10.1145/299359.299391

[9] Carsten Kleiner, Christopher Tebbe, and Felix Heine. 2013. Automated grading and tutoring of SQL statements to improve student learning. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research - Koli Calling '13.* 161–168. https://doi.org/10.1145/2526968.2526986

[10] Joshua License. 2017. testSQL: Learn SQL the Interactive Way. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE '17.* ACM Press, New York, New York, USA, 376–376. https://doi.org/10.1145/3059009.3072991

[11] Antonija Mitrovic. 1998. A knowledge-based teaching system for SQL. In *Proceedings of ED-MEDIA*, Vol. 98. 1027–1032.

[12] Antonija Mitrovic. 1998. Learning SQL with a computerized tutor. *ACM SIGCSE Bulletin* 30, 1 (1998), 307–311. https://doi.org/10.1145/274790.274318

[13] Antonija Mitrovic. 2003. An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education* 13, 2-4 (2003), 173–197.

[14] Antonija Mitrovic and Stellan Ohlsson. 1999. Evaluation of a constraint-based tutor for a database language. *International Journal of Artificial Intelligence in Education* 10 (1999), 238–256. http://ir.canterbury.ac.nz/handle/10092/327http://www.uic.edu/depts/psch/ohlson-1.html{%}5Cnhttp://ir.canterbury.ac.nz/handle/10092/327

[15] Julia Coleman Prior and Raymond Lister. 2004. The backwash effect on SQL skills grading. *ACM SIGCSE Bulletin* 36, 3 (2004), 32. https://doi.org/10.1145/1026487.1008008

[16] Michael De Raadt and Tien Yu Lee. 2006. Do students SQLify ? Improving Learning Outcomes with Peer Review and Enhanced Computer Assisted Assessment of Querying Skills. *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006* (2006), 101–108. https://eprints.usq.edu.au/1174/2/DeRaadt{_}Dekeyser{_}Lee{_}KOLI2006{_}PV.pdf

[17] S.W. Sadiq, Maria E. Orlowska, Wasim Sadiq, and Joe Lin. 2004. SQLator: An Online SQL Learning Workbench. *Proceedings of the Ninth Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (2004), 223–227. https://doi.org/10.1145/1007996.1008055

[18] G. Wiggins. 1990. The case for authentic assessment. - practical assessment, research & evaluation. *Practical Assessment, Research, & Evaluation* 2, 2 (1990), 1–3. https://doi.org/ED328611