The Backwash Effect on SQL Skills Grading

Julia Coleman Prior
University of Technology, Sydney, Australia
+61 2 9514 4480
julia@it.uts.edu.au

Raymond Lister
University of Technology, Sydney, Australia
+61 2 9514 1850
raymond@it.uts.edu.au

ABSTRACT

This paper examines the effect of grading approaches for SQL query formulation on students' learning strategies. The way that students are graded in a subject has a significant impact on their learning approach, and it is crucial that graded tasks are carefully designed and implemented to inculcate a deep learning experience. An online examination system is described and evaluated.

Categories and Subject Descriptors

H. INFORMATION SYSTEMS

H.2. DATABASE MANAGEMENT

H.2.3 Languages

Subject descriptor: Query languages

General Terms

Management, Measurement, Performance, Design, Reliability, Experimentation, Security, Human Factors.

Keywords

online examination, databases, SQL, SQL query formulation, learning approaches.

1. INTRODUCTION

Constructing database queries in Structured Query Language (SQL) is a pivotal skill required by many software developers. This paper reviews the effect that grading strategies have on the way that students learn and develop SQL query formulation skills, and how to design the grading in such a way that it encourages the students to engage deeply with the subject and to truly master these skills.

The next section looks at what we should aim to achieve when designing a database subject that includes SQL querying, and then the grading approaches most commonly used in universities are reviewed.

ITiCSE '04, June 28-30, 2004, Leeds, United Kingdom.

Section 3 reviews traditional, manual methods for testing students, and their limitations. Section 4 introduces an online system used for this grading in our department, and discusses how it works. Section 5 discusses the evaluation of this approach, and the conclusion summarises how the aims raised in Section 2 have been achieved. Further development and use of this online tool is also discussed in that section.

2. TEACHING, LEARNING AND GRADING SQL QUERY FORMULATION SKILLS

In our department's introductory database subject, one of the major learning outcomes is that a student is able to construct useful SQL queries.

Biggs' concept of alignment [1] suggests that to foster a deep learning approach by students, grading practices need to be integrated with teaching and learning activities and the learning outcomes. Biggs describes the impact that the grading method has on the student's learning approach as the 'backwash effect' [1]. The grading method should encourage students to take a deep learning approach, not enforce a surface one.

Together with Ramsden's suggestion [11] that the type of grading 'shapes the curriculum' and strongly influences the student's learning approach, it would seem that grading a student's SQL skills online, and in a manner similar to how they will use SQL as software professionals, would encourage them to adopt the same tactic in their learning approach.

Also, Toohey [13] states that giving students practical, professional tasks to perform for grading has 'clear relevance' to professional education. Ramsden [11] quotes Newble and Clarke who established the principle that problem-based learning simulates the type of problems met in professional life and is 'more likely to encourage students to adopt a deep learning approach'.

In the light of the above, we defined the following three-fold objectives for teaching SQL query formulation skills:

a) to grade students using an approach that accurately determines their individual SQL query formulation skills;

b) to grade students in a manner that closely replicates the way that they will use their SQL skills in real-world software development, as described; and

c) to encourage students to practice and develop their SQL skills online.

While designing the grading with Biggs' backwash effect in mind, we also considered Toohey's factors for selecting a grading method [13]:

i) validity of the grading, which is how accurately it reflects the learning objectives for the subject;

ii) reliability of the grading, where a highly reliable method is one where work submitted for grading on different occasions should return similar results;

iii) how well the grading leads to and enables real learning: as previously mentioned by other authors [1], [11] i.e. the way that work is graded has an enormous influence on the approach that a student takes to learning in a subject.

Usually only one SQL statement is necessary to get a useful result, and these statements are very short (relative to the complete computer program that would be necessary to fulfil a similar purpose using a conventional programming language). Thus, there is the perception that it must be relatively simple to learn to write SQL queries. In fact, it is a challenging skill and 'students have many difficulties learning it' [9]. Mapping from a problem statement describing what information is required from the database into an appropriate SQL statement is not easy, as when an SQL statement is executed the database software performs numerous operations that are imperceptible to the programmer. It is particularly difficult if one cannot see the result set that would be returned from the database when the query is executed.

Yet this is how students are often expected to construct SQL queries when they are being graded in this skill. The way that a professional software developer usually creates SQL is similar to Schön's 'talkback' [12], where a practitioner makes a design decision, tries it out and then modifies the solution according to the result of their interaction with the design situation. Professionals verify the results of this preliminary query once it has been executed online, and if it does not accurately return the required information, they refine the SQL query and re-execute it, repeating the verification and refinement steps until they are satisfied that the query is returning the desired results.

Ideally, then, in both learning and grading, it would be beneficial to students to be able to verify their solution for each question by executing the SQL statement and comparing their answer with the required results (data set). This immediate feedback on the validity of their solution would guide them to what amendments they need to make to their query design.

3. CONVENTIONAL SQL SKILLS GRADING

One grading approach is to give students a set of problems (descriptions of information that needs to be retrieved from the database) and to ask the students to construct SQL queries as the solutions. This may be in the form of a paper-based assignment to be submitted or as a supervised, written test. This is the grading route taken by numerous universities in their introductory database subjects [4], [7], [10] and [14].

The problem that we found in our department using this approach is that students were passing the subject, but they did not necessarily have the requisite SQL skills. As the assignment or test is submitted as a written document, it is not a motivation for students to practice their query construction skills online, and verifying them against a database, which is how they will use these skills professionally.

Constructing SQL queries is a practical skill, and cannot be gained without significant effort and repeated online practice. Most students do not put in this effort; after all, they are not graded in this way. As we have discussed, the type of grading that they will experience significantly influences students' learning strategies for a particular subject [1], [11]. Even students who have conscientiously practiced writing out queries will not develop their skills in a useful, long-term manner. One of the difficulties for a student is conceptualising and visualising the result of an executed SQL statement. Constructing queries online, executing them, visually verifying the result and, if necessary, modifying the query until it gives the correct result internalises the query formulation skill. It incorporates the idea of learning from one's mistakes. Immediate feedback is an important component in the learning loop; Mehta and Schelicht [8] describe this as one of the advantages of their computerised grading in large classes.

4. AN ONLINE ENVIRONMENT: AsseSQL

In our department we have introduced an online test to grade students' SQL skills in the introductory database subjects, using software that has been developed in-house specifically to address the issues raised in section 2. There appear to be several systems available that automate submission and testing of students' programs for grading, for example, BOSS [6] but the authors have not been able to find software for effectively grading SQL query formulation skills.

There are also numerous software packages designed specifically for teaching SQL query formulation skills, for example, WinRDBI [3] and SQL-Tutor [9], as well as several web sites (e.g. www.sqlator.com, www.sqlcourse.com) that enable students to practice formulating and executing queries and giving them immediate, individual feedback. These do not provide summative grading, however, and students need a convincing reason to motivate them to make use of such tools, or even database management system software, directly.

A description of the online test software, AsseSQL, follows. All the data about each test to be taken are stored in a database, for example, test date, duration, total number of marks, number of questions and type of SQL query to be tested in each question; in other words, the design of the test. Also stored in this database is a query pool – a selection of SQL problems and model answers (i.e. queries) that test different types of SQL statements. The structure of each test is such that although all the students in a class will do Test1, for instance, each student will be given their own unique version of Test1 when they actually take the test, as questions for each student are chosen at random from the pool.

Assume that we design Test1 so that there are 5 questions in total:

• question 1 is a SELECT on one table with one WHERE clause

• question 2 is a SELECT on one table with more than one WHERE clause, joined by logical operators

• question 3 is a SELECT on one table with a GROUP BY and a HAVING clause

• question 4 is a SELECT on two tables with a natural join

• question 5 is a SELECT with a sub-query containing a simple SELECT

In the query pool, there are a number of problems that could be used for question 1. When a particular student logs on to do the test, the program chooses one of these queries for this student's question 1, and similarly for each of the other questions in the test.

A second, 'scenario' database contains the tables against which both the model solutions (queries) and the students' attempts for each test question are executed. For example, there might be an Order Entry database containing Customer, Product and Order tables for Test1. The questions for a test would require queries to be constructed for querying data stored in this scenario database.

The students take the test in the faculty's computer laboratories under supervision. This is to ensure that it is the students themselves who take the test. The test software is web-based, residing on the faculty's intranet. Two levels of security need to be passed before a student can begin to take their test. The student must first login to the intranet and in order for them to actually start taking the test, a supervisor userid and password must also be entered. This userid and password are different for every test session, and are only valid for that test session. The student is thus only given these details in the laboratory, once every student is logged on and ready to begin the test, and no student may leave the test venue until the end of the test session. The test duration is fixed and is the same for every student, but each student's starting time is only recorded once they are through both authorisation stages, and their test will be available to them for the test duration (e.g. 60 minutes) from their individual starting time. When the student's time is up, their test is locked and the student is not able to submit any more answers.

Once the student's test is started, the first form presented to the student lists their particular set of questions for their test. The student may answer the questions in any order that they wish. Furthermore, students may attempt each question as many times as they wish, until it is marked correct, or their test time is up.

From this first form, the student clicks on the question that they wish to answer and are shown the answer form. This displays the question again, as well as the result set (of data) that should appear when a correct answer (query) is executed. Displaying the correct answer eliminates much of the potential for ambiguity in the question, and is particularly useful for those students for whom English is a second language. The students type their solution (i.e. an entire SELECT statement) into a textbox. They submit their answer and the SQL statement will be executed against the scenario database e.g. the Order Entry database.

If the submitted answer is syntactically incorrect, an error message is displayed. If the statement is executable, the data grid containing the result of the student's executed answer is displayed beneath the answer text box. If these results are not the same as the model solution's, a message stating this is shown and the student can compare their data result with the required one. The student can amend their SELECT statement and re-submit. Alternatively, they can elect to go back to the first form that lists all their test questions and choose to answer another question.

The program marks the student's answer by comparing the data set produced by the execution of the model answer to the data set that results from the execution of the student's answer. If the data sets are exactly the same, the student's answer is flagged as correct; otherwise it is flagged as an unsuccessful attempt.

If the student's answer is correct, they will be taken back to the first form again automatically. Any correctly answered questions will now have messages next to them stating this. Questions that have been attempted but are not yet correctly answered also have a relevant message next to them. The student can then click on the next question that they wish to answer.

The student may logout of the test at any time, but will be able to login again and attempt any incomplete or incorrect answers until their individual test time limit is up. In the same way, if their test window is closed accidentally, they will be able to login again and continue from where they left off, providing that their test time is not up.

The students are able to practice using the SQL test software. Ramsden [11] emphasises that a grading task should not be threatening and states that the lecturer should do everything possible to 'lessen the anxiety' raised by grading. A mock test is set up and the students are able to try this out as often as they wish, in a non-test atmosphere, for several weeks before the actual test near the end of the semester. The student may take the mock test as often as they wish. Thus, students who use the opportunity to practice with the online test software are quite comfortable with the approach at the actual test time, and are able to focus on constructing the queries to be graded, without having to be concerned about how the software works and how to interact with it. The mock test also gives the students further opportunity to practice their query formulation skills online. They are given a data model and description for the actual test's scenario database to study a week before the test date, so that they do not have to consider what the tables and relationships represent during the limited test time.

5. EVALUATION – STUDENTS

Housego and Freeman [5] point out that technology-supported teaching is effective only when based on teaching practices which motivate students to adopt a deep learning approach, not because information technology is used simply for its own sake. To verify that AsseSQL is effective, we have evaluated it using structured questionnaires, focus groups and an online discussion forum. Also, the manual remarking of a percentage of the submitted tests is undertaken each semester to verify that the marking was done fairly and as expected by the software.

A structured questionnaire is given to all students who take the online test. The students are asked to agree or disagree with each of the statements shown below in Table 1. In the most recent semester in which the test was run, 92% of the students who took the test completed the questionnaire. These results are indicated in Table 1, where the percentages of 'Agreed' responses for each statement are given.

Table 1. Percentage of 'Agreed' Responses to Statements in the Online Test Evaluation Questionnaire

	Statement	% Agreed
Q1	I was more motivated to practice SQL because of the online test than with a written assignment.	85
Q2	I was more motivated to practice SQL because of the online test than with a written test.	85
Q3	Practicing SQL queries interactively and online helped me to improve my SQL query skills.	92
Q4	I preferred taking the online SQL test to taking a written SQL test.	88
Q5	I preferred taking the online SQL test to submitting a written SQL Assignment.	84
Q6	I have an accurate idea of my ability to construct SQL queries after taking the online test.	78
Q7	The time given for the test was reasonable.	67
Q8	The marking was consistent and fair.	87

In the formal questionnaires, the focus group discussions and informal, open-ended feedback, a significant majority of students concurred that anticipating the online test influenced the way that they went about learning and developing SQL query skills. One student in a focus group commented that AsseSQL 'forced' him to develop SQL skills in a way that a written test would not necessarily do, partly because of the practice software but also because it was a more realistic approach and therefore more interesting. Other remarks included 'the online test pushed me to practice online as often as possible' and 'it [the online test] is really a good way to motivate students to learn SQL'. Clearly, this grading approach fulfils our third aim of encouraging students to practice and develop their SQL skills online, as well as Toohey's third factor referring to the grading's impact on real learning.

The evaluation process indicated that students consider the grading closely replicates the way that they will use their SQL skills in real-world software development, fulfilling our second aim. Some of the students were concurrently completing a semester of industrial practice with software development companies, which is a required part of their degree program. Significantly, this group of students were extremely positive about the use of AsseSQL for SQL skills grading, particularly in the context of this second aim.

In the focus group discussions, most students agreed that the first of Toohey's factors above - that the grading accurately reflects the learning outcomes - was fulfilled by the online test. Aligning the grading task with the learning outcomes for the SQL part of the subject was one of the major motivations for introducing the online test.

One of the advantages of using a computer to perform tasks is that it is consistent, and ideally suited to doing the same tasks over and over without the repetition adversely affecting its performance as it does with humans. When manually marking hundreds of students' answers it is extremely difficult, and in fact very unlikely, that an academic staff member will be able to mark students' answers completely consistently and fairly. With AsseSQL, answers producing the same results will always be marked reliably and accurately and it thus complies with the second of Toohey's factors, reliability. The students support this view (see Q8 in Table 1).

6. EVALUATION – TEACHERS

Among several advantages of using AsseSQL, perhaps the one most appreciated by teachers is the reduction in marking. With 1000 students enrolled in the last 3 semesters, the marking load would have been immense without AsseSQL. It is especially difficult and time-consuming to establish that a non-trivial SQL query is wrong without actually executing it.

Academic staff appreciate the re-use of some of the queries for different tests, thus saving test-setting time. After every test the testing space grows – more queries are added for each test, occasionally new databases are added as well, and this increases the 'randomness' of future customised tests.

AssesSQL automatically produces electronic records of every student's individual tests – all questions, all their answer attempts and their final marks, the latter saving the mundane task of marks entry.

Teachers can retrieve statistics on several aspects of the tests. For example, one statistic extracted is the number of attempts made by students for each question, an indication of which type of queries students struggle with. Ramsden [11] emphasises that one of the functions of grading should be that we use it as feedback to improve our teaching approaches.

Whilst limiting plagiarism is not the prime aim of the test, it is an added bonus. It would be extremely difficult for any student to share questions and/or answers with another student. Each student takes their own customised test, so it is very difficult for any student to help another one. The latter is also unlikely in a situation where each student will probably be more focussed on trying to get the answers correctly constructed online in a limited time.

Possibly the area that requires the most attention and care in AsseSQL is the setting of the test questions. It is crucial that the problem statements are precise and unambiguous, so that students are certain which information should be retrieved from the database. Although the displayed 'model answer' results help clarify this, it is nonetheless important to have several staff members work through and try to answer the questions without seeing the model answer to check for any imprecision or ambiguity. Whilst this may not reflect the real development world, where a client's ambiguous requirements can be clarified and verified, in a test situation students do not have the same opportunities to check that their interpretations are valid.

One concern that was raised by teaching staff was that students can design a contrived solution for a particular question that is only valid for specific results (i.e. those shown in the sample output). For example, where the student is required to code a reasonably complicated query that returns only a few rows and columns, one of which has the values '1, 5, 7, 9', the student could instead simply write:

```
SELECT <appropriate columns> FROM <table>
   WHERE <column> IN (1,5,7,9);
```

In the current version of AsseSQL, these attempts to effectively cheat are detected by a quick manual scan through an

automatically generated report. This scan is not terribly onerous (and considerably easier than conventional marking), as the report only shows student answers that differ significantly from the model answer queries. However, the next version of AsseSQL will make it very difficult to cheat in this manner: the model answer query and each student's answer will be run against a second scenario database, not visible to the student. This second database has the same model answer as the first database, but it has slightly different data. To be marked correct, a student answer will need to give the same answer as the model solution for that second database. The type of modifications to the data would be, for instance, different minimum and maximum values for groups of rows, and changes to the number of rows in a table.

The system uses binary marking – a student's answer is marked either 'correct' or 'incorrect', no partial marks are given. A concern has been expressed that a poor answer that is far from correct and an answer that only has a minor error will receive the same result (i.e. incorrect), and that this is unfair on the student with the 'better', almost correct answer. Firstly, as far as the computer is concerned, it does not accept partially correct instructions, a query (or any programming statement) is either right or wrong, and novice developers have to deal with that. Partial mark allocation tends to reward effort rather than correctness. Secondly, a student is given a reasonable indication of what is wrong with their query, and where the problem is in their query, by the system's error message given immediately after they have submitted their answer, and a student who has practised regularly and mastered the requisite skills will be able to identify and fix the problem timeously. As indicated by the student evaluation, most students are comfortable with the binary marking approach.

Finally, it has been suggested that encouraging students to take a 'hands-on' approach to developing their SQL query skills may have the undesirable side-effect of inculcating a 'trial-and-error' approach to query formulation without doing any prior design. We teach a defined, step-by-step process to designing any SQL query, reinforced in tutorials and laboratory classes. If a student has not mastered this type of design process by implementing it regularly in conjunction with online practice, they will not be able to formulate queries correctly, especially within a limited time as with the online test. This message is conveyed repeatedly to the students during the semester.

7. CONCLUSION

An online examination system for SQL queries was introduced and the results of its evaluation have been described. Our goal was to provide a system that aligned the grading strategy with the way students will use SQL after graduation, to encourage deep learning. The results of the students' evaluation indicate that our system achieves that goal. Future work will extend the tool for use in courses teaching material other than SQL queries.

8. REFERENCES

[1] Biggs, J. *Teaching for quality learning at University*. Buckingham, Open University Press,1999.

[2] Culwin, F. Web hosted assessment – possibilities and policy. In proceedings of *ITiCSE'98* Dublin, (Ireland, 1998), p55-58.

[3] Dietrich, S.W., Eckert, E. & Piscator, K. WinRDBI: a windows-based relational database educational tool. In proceedings of *SIGCSE'97*, (1997), p126-130.

[4] Grundy, F. Module description for COMP-207 : Databases, Department of Computer Science, Keele University, United Kingdom, 2001. Available online at: http://www.keele.ac.uk/depts/cs/modules/0102/level2/comp207.html [accessed 27/09/02].

[5] Housego, S. and Freeman, M. Case studies: integrating the use of web-based learning systems into student learning. *Australian Journal of Educational Technology,* 16(3), (2000), p258-282.

[6] Joy, M. and Luck, M. Effective electronic marking for on-line assessment. In *ITiCSE'98* (Dublin, 1998), p134-138.

[7] Maciaszek, L. Study guide for INFO603 Database Systems, School of Business and Informatics, Australian Catholic University, Sydney, Australia, 2001. Available online at: http://www.comp.mq.edu.au/~leszek/uni_courses/603_01StudyGuide.pdf [accessed 27/09/02].

[8] Mehta, S.I. and Schlecht, N.W. Computerized assessment technique for large classes. *Journal of Engineering Education*, (April 1998), p167-172.

[9] Mitrovic, A. Learning SQL with a computerized tutor. In proceedings of *SIGCSE'98,(*1998*),* p307-311.

[10] Paradis, F. and Barbour, R. Course outline for 0657.219B Database Practice and Experience. Department of Computer Science, University of Waikato, Hamilton, New Zealand, 2002. Available online at http://www.cs.waikato.ac.nz/Teaching/0657.219A/course_outline.html [accessed 27/09/02].

[11] Ramsden, P. *Learning to teach in higher education.* London, Routledge, 1992.

[12] Schön, D. A. *Educating the reflective practitioner: toward a new design for teaching and learning in the professions.* San Francisco, Jossey-Bass, 1987.

[13] Toohey, S. *Designing courses for Higher Education.* Buckingham, Open University Press, 1999.

[14] Webster, M. Course outline for ZXX-4506 Introduction to Databases with Oracle and SQL, Dept of Information Services, University of Bangor, Wales, 2002. Available online at: http://www.bangor.ac.uk/is/teaching/postgrad/postgrad.shtml [accessed 27/09/02].