Learning SQL in steps

Philip GARNER
School of Computing and Communications, Lancaster University
Lancaster, LA1 4WA, United Kingdom

and

John MARIANI
School of Computing and Communications, Lancaster University
Lancaster, LA1 4WA, United Kingdom

ABSTRACT

Learning SQL is a common problem for many Computer Science (CS) students, the steps involved are quite different to those mastered when learning procedural or object-oriented programming languages. The introduction of commercial products that include shortcuts into the learning environment can initially appear to benefit the student, however, transferring these skills to a textual environment can be difficult for many students. Computer Science students are required to build textual SQL queries because the demands of complex queries can quickly out grow the capabilities of graphical query builders available in many software packages. SQL in Steps (SiS) is a graphical user interface centred around the textual translation of a query; this combination of a GUI and a clear representation of its textual meaning has the potential to improve the way in which users gain an understanding of SQL. SiS allows for an incremental and evolutionary development of queries by enabling students to build queries step by step until their goal is reached. A planned evaluation of SiS hopes to quantify the extent to which the introduction of such a user interface into the learning environment can improve the students' understanding of the language.

Keywords: learning SQL, education, learning environments, graphical queries, relational databases

1 INTRODUCTION

Learning how to manipulate and retrieve information from a relational database is a core part of any CS course; however, it is well recognised that this is a skill that many students find difficult to master [24, 10, 2, 23, 22, 20, 16, 17, 11, 13]. There have been numerous attempts to simplify the process of learning SQL through the introduction of different software and educational techniques; however, these systems are not without fault.

In this paper we first highlight the problem of learning SQL before discussing some previous attempts to address this problem. In an attempt to avoid some of the flaws presented in the literature we present a web-based system called SQL in Steps (SiS) which is designed to allow students to independently build SQL queries and explore the functionality of the language. SiS utilises a graphical user interface which is closely coupled with the textual translation of queries to allow users to build queries in a graphical environment while gaining a concrete understanding of how to construct textual queries.

2 THE PROBLEM OF LEARNING SQL

Learning SQL is an important skill CS students are required to master, the ability to read, build and modify SQL is something which is required in many computing, and some non-computing, professions. Although learning SQL is a crucial skill it is well recognised that it is is a problematic area for many students.

There are many possible reasons that contribute towards the difficulties of learning SQL. The declarative nature of SQL can be a difficult concept for students to grasp, particularly if they are simultaneously learning procedural or object-oriented programming languages alongside SQL [12, 23].

Learning and visualising the database schema is also a problematic area for many newcomers to SQL [17]. To be able to effectively build a query over a relational database the user must first be able to visualise where the different elements of data required are and how they can be extracted. Join operations are common when querying relational databases, they are required to extract information from directly or indirectly connected tables. Visualising where these operations are necessary and how to execute them is also a common problem for students learning SQL [11, 17, 20].

SQL contains many powerful functions and misconceptions regarding any of these can lead to erroneous queries or those which may not return the desired results when executed on a different dataset [17].

3 RELATED WORK

The extraction of information from relational databases is a well recognised problem that has prompted a broad range of research over the past thirty years [24], two decades ago there began a series of attempts to improve the way students learn how to interact with databases (e.g. [21, 11]). The vast majority of the existing work can be divided into two categories: systems designed to analyse users queries and systems that use animation to illustrate the process of query execution.

The analysis of queries involves parsing the user's submitted query and comparing it against an often predefined desired solution. Once analysis is complete these systems can offer useful instructions or guidelines for the student to undertake or provide a mark for the query submitted.

The animation of queries involves taking a fully formed query and showing the various different steps taken to achieve the final results. This process can be used to highlight the purpose of the various different clauses within an SQL statement and allow the user to gain an understanding of each component of a query.

SQL allows users to build databases and manipulate their contents through the use of various different statements. Many of the educational tools reviewed focus only on the SELECT statement as it is the most commonly used and can become the most complex; there are also elements of the SELECT statement, such as the WHERE clause, that are used in other SQL statements such as UPDATE and DELETE [17, 22, 11, 4, 2, 20, 5]. By gaining an understanding of the SELECT statement the user is likely to find the transition to other statements easier.

3.1 Analysis of queries

There are numerous systems that analyse the input provided, the following is a brief overview of a selection of these existing systems.

SQLTutor [17] is an early example of a system designed to analyse queries submitted by students; the main focus of SQL-Tutor is to improve upon the basic error messages provided by the majority of database applications. Error messages provided by database applications are a well recognised problem for new users of SQL [17, 22, 20, 2], SQLTutor uses a series of manually predefined questions and answers to provide more meaningful and contextual error messages.

A more recent system which analyses users queries is LEARN-SQL [1], this system focusses on the fact that there are often multiple possible solutions to a query. Because of this LEARN-SQL operates by comparing the output of a users query against that of a model query rather than the contents of the query itself. An additional benefit of LEARN-SQL was an attempt to reduce the work load of teachers by introducing automatic marking.

The analysis of students queries is frequently coupled with a pool of predefined questions and answers [17, 22, 20], these are compiled by the teacher and every time a student uses the system they are allocated a question. Once a user attempts a question their answer is compared to the model answer and feedback or a score is provided to the user. If the input of questions is detailed enough and the system has a comprehensive knowledge of SQL then it can offer clear feedback as to where the user can improve their query. The downside of using pools of predefined queries is undoubtedly the time overhead of initially setting up the system as a teacher has to define and answer all the questions before the system can be used; it could also be argued that this approach limits the desire to explore the database and focusses on specific tasks.

The analysis of users queries for feedback is often continued into providing assessment support for teachers of the language. Assessment is used in varying degrees in SQLator [23], WinRDBI [6], AsseSQL [20], LEARN-SQL [1] and SQLify [5]. SQLator [23] and AsseSQL [20] use binary gradings while Russell et al. [22] offers a percentage, SQLify [5] continues this to offer peer reviewing between students.

3.2 Animation of queries

The animation of queries can be used to illustrate how the final result of a query is achieved rather than it simply being presented to the user without any intermediate steps. A typical set up for an animation based system would involve a textual input field where the query is entered; after execution the system will analyse the query and perform a series of animations illustrating how the final result set is found.

Work such as eSQL [11] and SAVI [2] adopt a system similar to a programming language debugger in that they contain step operations to advance to the next animation and continue operations to view the final result set. The user submits a query and, assuming the query contains no errors, the query is executed in stages allowing the user to visualise how the final result set is achieved. eSQL uses an algorithm to select a subset of the entire database which will accurately illustrate the execution of the query. The steps involved in the animation of a query are generally very similar and consist of animating each clause, these animations are often performed in an order different to that which they actually appear. The SELECT clause is the first component of a textual query but, for animation purposes, it is more logical to illustrate this step towards the end of the animation.

The problem of animated systems is that they require that a fully formed SQL query be submitted before animation can take place. These systems seem more appropriate for the illustration of examples rather then the process of learning SQL. Unless combined with meaningful error messages or examples it could be difficult for a novice user to *"get off the ground"* with many animated systems.

3.3 Commercial products

In an attempt to avoid using a textual interface with little support for teaching SQL many institutes revert to using database management applications such as Microsoft Access [15], HeidiSQL [9] or phpMyAdmin [19]. The main problem with using such systems stems from the aims of this software; they are designed for the management of information within relational databases, not to teach users how to construct textual queries. Such systems are *"not designed for educational purposes but for the professional management of databases"* [8] and as a result they often introduce problems when the users are required to transfer their skills to textual queries [14, 10]. In addition, it is possible for students to use the Query By Example builder available in many software packages to form queries and then use the SQL translation to submit the textual queries for grading without any understanding of their meaning [3].

Users can often fail to see the need for a textual system when, from their perspective, the graphical system performs all the necessary tasks with less opportunity for error. Although the initial perception of graphical systems may be that they offer the same capabilities of textual ones but with less chance of error but this is not the case, as students move to more advanced queries they are likely to build queries that are beyond the capabilities of graphical query builders.

4 SQL IN STEPS (SIS)

In an attempt to address the issues uncovered in the literature SQL in Steps (SiS) was developed. The main focus of SiS is to allow the student to build an SQL SELECT statement in small steps using a GUI while gaining an understanding of the textual query they are building in the background. The user interface is designed with the textual translation at the forefront. Every change made to the user interface prompts a change in the textual translation which, in turn, refreshes the results of the query in its current state.

Cembalo et al. observed that students often solve procedural problems by breaking them down into smaller steps and stated that this could not be done for SQL: *"this approach cannot be followed with SQL, because in a complex query there are no intermediate steps to solve separately, but instead temporary sets of*

data which result from the execution of the different operators of the same query"[2]. SiS introduces these intermediate steps and allows the student to easily visualise the temporary sets of data which they can build upon to refine their query.

Bringing the textual translation of a query to the forefront and avoiding an excessively abstract user interface we address the difficulty of transferring from a graphical to a textual environment.

Although SiS can loosely be categorised as an animated system it is different to others previously discussed in that it guides the user towards incrementally building a query before submitting it to view the results. This guidance through the process of building a query provides the user with the steps taken to achieve the final result set, this differs from other animated systems in that the steps are prompted entirely from the user and not through the use of predefined stages. The student can choose to build their SQL SELECT statement in any order they desire, for instance, they could build the WHERE clause (and view its effects) then add some ordering information before continuing to further refine the WHERE clause.

4.1 Textual translations and live results

The aim of teaching SQL is to provide students with an understanding of SQL such that they are capable of building textual queries independently, from the students perspective this can initially seem a daunting task. As previously discussed, attempts to simplify this process have involved introducing students to graphical systems before transferring to a textual user interface, however, this can make for a difficult transition.

SiS attempts to avoid this problem by allowing the user to build a query using a simple user interface while the textual translation remains clear and up to date. SiS' use of a textual translation in conjunction with live results enables the user to gain confidence in the different clauses of SQL, their syntax and their effect on the results. If a query were to contain an error or return no results this information is displayed to the user.

Through the development of AsseSQL Prior et al. identified the problem of students being unable to build a mental picture of the intermediate stages in building an SQL query: *"One of the difficulties for a student is conceptualising and visualising the result of an executed SQL statement"* [20]. This can lead students to make false assumptions about a query they have written only to build on those false assumptions until they execute their query and attempt to debug the problem. Enabling a user to clearly visualise the intermediate stages of a query is the basis upon which many animated systems are created but SiS continues this process further by illustrating every change made to the query rather than grouping them together. The live updating UI allows the user to immediately identify the effect of each change made and also increases the likelihood that they will identify the cause of any errors.

The user interface for SiS is intentionally not overly abstract and closely follows the structure of an SQL clause. This allows the user to easily map the functions of the user interface to the different components of an SQL statement. For instance, to select attributes for the SELECT clause involves selecting attributes using a check-box, selecting relations to be used in the query is done by clicking on a graphical representation of the database structure (Figure 1).

4.2 Database visualisation

When learning SQL students may be exposed to a wide variety of different database structures, they may be introduced to more complex schemas as their understanding increases. A common problem faced by many new users is the need to visualise the database schema before querying it; students should not be penalised for failing to remember the structure of a database but should be encouraged to learn how to best utilise the various different database schemas.

To familiarise the user with the structure of the database they are querying SiS presents them with a graphical representation of the database structure. This illustration can be zoomed in and out to discover details such as the attributes within a relation and the connections between relations. As previously discussed, this part of the user interface is not only used as a prompt when the user is interrogating the database but is also used as a means to build the FROM clause. The use of this visualisation, which is visible at all times, provides the user with the ability to quickly build queries and explore the database without studying lengthy descriptions of the database schema.

4.3 Automatic contextual help

Examples and demonstrations are an invaluable tool when learning any computer language [7], SQL is no exception. Descriptions of queries and examples of how their results can be achieved can provide an insight into new features of SQL that are difficult to understand otherwise. However, if a student is learning within one domain and their examples are given within another they can disorient and confuse the user. SiS includes the ability to provide contextual examples for all clauses within the database the user is using, SiS will analyse the structure and contents of the chosen database and automatically produce a series of simple queries that can provide the user with an insight into the correct usage of each clause.

4.4 Customisable

To ensure SiS remains relevant and doesn't overwhelm users large portions of it are customisable. Many of the features of SiS can be enabled/disabled within a configuration file; this allows an administrator to configure SiS such that it is only capable of simplistic queries using a limited set of SQL functionality if the more advanced features are not required.

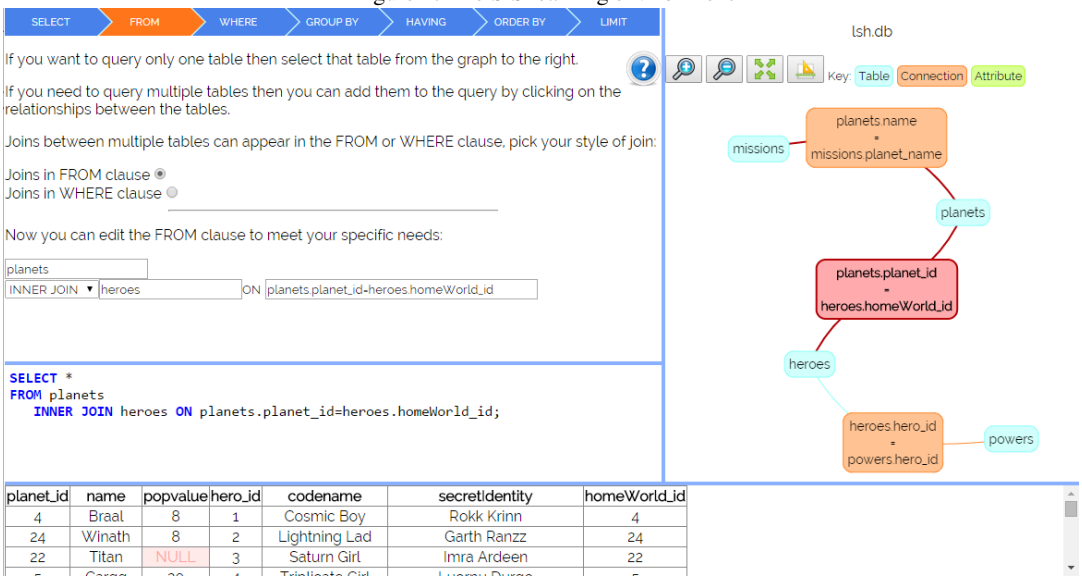
The implementation of SiS using SQLite enables a wide range of databases to be used by simply adding the database files to a directory which the system can access. This allows for the quick customisation of the system to meet the needs of various different users, as users become more confident with SQL more complex databases can be introduced, this allows teachers to easily expose students to a wide variety of different database structures while they learn SQL.

4.5 Web based

In recent years there has been a trend towards software that can be run in a web browser; ease of distribution and cross platform compatibility make it an appealing option for a wide array of applications.

Like SQLator [23], SAVI [2], AsseSQL [20] and SQLify [5] SiS has been developed from the ground up as a web based tool, built using HTML, SQLite, JavaScript and PHP. The use of web

Figure 1: The SiS learning environment



based software means that students do not have to install any specialist software and can access the tool from any computer connected to the internet.

4.6 Boolean visualisation

Building Boolean expressions is an important aspect of creating an SQL query, they are used in the WHERE and HAVING clauses of the SQL SELECT statement (along with the WHERE clause in other statements) and a good understanding of them is crucial in ensuring the desired results are returned. The understanding of Boolean expressions is something that many novice users struggle with [18], CS students must build a strong understanding of such expressions to become confident in many key skills including writing SQL statements.

To allow the user to visualise their Boolean expressions in a more graphical sense SiS introduces a graphical representation of Boolean expressions called the sandpit (Figure 2). As with all elements of SiS the idea of the sandpit is intentionally closely linked to its textual counterpart; each individual component of a Boolean expression is represented using a single widget on screen, the user can input their criteria on this widget. Individual widgets are combined together by drawing boxes around them, these boxes represent any Boolean expressions and their associated parenthesis, boxes can be in an AND or an OR state and can contain a mixture of both widgets and other boxes enabling them to build complex Boolean expressions in a graphical environment closely linked to its textual counterpart.

4.7 Limitations

Mitrovic et al. identified how many elements of the SELECT statement are also found in other statements [17]. Because of this (like [17, 22, 11, 4, 2, 20, 5]) SiS focusses entirely on the SELECT clause.

Due to limitations introduced by the graphical representation of sub-queries SiS currently has limited support for them, it allows for the inclusion of sub queries in the FROM clause alone.

5 TYPICAL USE CASE OF SIS

After selecting their chosen database the user is presented with a graphical representation of the database and the UI for building the SELECT clause. To explore the structure of the database the user can use the zoom buttons above the graphical representation of the database schema, this will offer more or less information as required. If the graph becomes too crowded for the screen space available they can maximise the graph and view it in full screen mode.

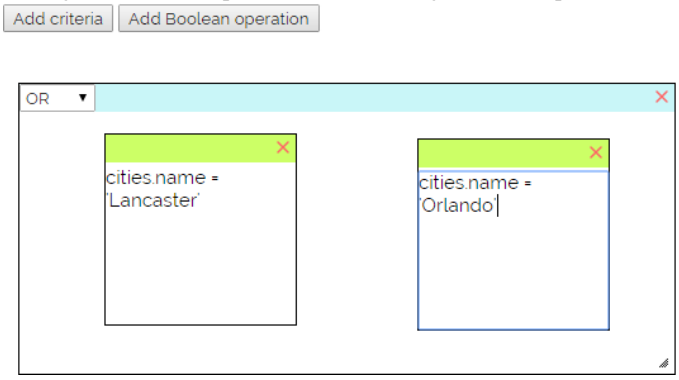
To build their first valid query the user must select a table (or series of connections for a query containing a join), as soon as this selection is made they can view the results of this simple query in the results panel below its textual translation. After selecting a table or series of tables the contents of these are immediately shown to the user allowing them to adjust their FROM clause to ensure they only use the information required for the query. Details of the SELECT clause can be refined by choosing various attributes from the tick boxes available or adding customised ones using the aggregate function builder. Selecting the FROM tab at the top of the page allows the user to customise the from clause, if the query only consists of one table this will offer little opportunity for customisation but if multiple relations are used the user can change the join type, again viewing the results as the changes are made.

To filter out some of the records shown the user can select the WHERE (or the HAVING tab for filtering by aggregate functions) tab and use the builder to construct a sandpit of widgets and boxes representing a Boolean expression, as with all aspects of SiS, and changes made in the sandpit prompt immediate changes to the textual translation and results.

The GROUP BY and ORDER BY tabs contain very similar user interfaces to that on the SELECT tab, a series of check boxes can be used to build a grouping or ordering clause. The LIMIT tab simply offers a simple input for two numbers, the number of desired results and any offset.

If the user is unsure of how any of the clauses should be constructed or what their purpose is they can click the question mark

Figure 2: The sandpit used for visualising Boolean expressions



icon found on every tab. This will open the contextual help which offers a described example of how to use the statement the user is struggling with.

6 FURTHER WORK

It is our intention to involve a selection of first year CS undergraduates with a trial involving the use of SQL in Steps. The user study will involve attempting to quantify the improvements gained by using SiS in conjunction with traditional teaching techniques over using traditional techniques alone. The undergraduate database course involves introducing the students to various theoretical and practical concepts, in the past the students have been taught using the textual interface of Microsoft Access and a text based interface to SQLite or MySQL. The structure of the study will involve a small assessment of the students ability at the start of the course (many will score very low as they may have little or no experience with relational databases), the sample will then be randomly divided into two groups, the first will continue without any use of SiS, the second will have access to SiS whenever they wish. The databases set up with SiS will be the same as those databases the students will be learning with. At the end of the course the students will again be tested, if a greater improvement is seen by those students using SiS then investigations will be made to deduce whether this is as a result of the involvement of SiS or a simple coincidence. As well as attempting to quantify the benefits introduced by SiS a focus group involving a selection of SiS users will take place after the study ends; the aims of this focus group will be to gather qualitative data regarding the positive and negative features of SiS as well as discussing future design iterations.

7 CONCLUSION

In this paper we have provided a brief overview of some existing technologies which have attempted to improve the way in which students learn SQL. This existing work can loosely be divided into two different categories, those systems that operate by analysing submitted queries for feedback and/or marking and those that use animation as a means to break down the process of building SQL statements into easy to process steps. After highlighting some strengths and weaknesses within the existing literature we introduce SQL in Steps (SiS), an online environment designed to improve the way in which users learn SQL by combining a graphical user interface centred around a textual translation with fast and frequent updates. Every change made to the user interface prompts a change in the textual translation of a query, this enables to user to clearly see how changes to the UI force changes in the textual query and, in turn, the results of the query. The frequency of updates allows the user to easily identify when errors have been introduced to the query allowing them to quickly identify and understand them. Transferring users from graphical user interfaces to textual ones is a well recognised problem, in an attempt to ease this the user interface in SiS is intentionally not hugely abstract, the close coupling between the UI and its textual equivalent is intended to make the transition from graphical system to textual as painless as possible.

The completion of a planned user study involving a series of first year undergraduates hopes to confirm the potential of introducing SiS into the learning process of SQL.

REFERENCES

[1] A. Abelló, M. E. Rodríguez, T. Urpí, X. Burgués, M. J. Casany, C. Martín, and C. Quer. **LEARN-SQL: Automatic Assessment of SQL Based on IMS QTI Specification**. In *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, pages 592–593. IEEE, 2008.

[2] M. Cembalo, A. De Santis, and U. Ferraro Petrillo. **SAVI: A new System for Advanced SQL Visualization**. In *Proceedings of the 2011 conference on Information technology education - SIGITE '11*, page 165, New York, New York, USA, Oct. 2011. ACM Press.

[3] J. Cigas and B. Kushan. **Experiences with online SQL environments**. *Journal of Computing Sciences in Colleges*, 25(5):251–257, May 2010.

[4] J. Danaparamita and W. Gatterbauer. **QueryViz: helping users understand SQL queries and their patterns**. In *EDBT/ICDT '11*, Uppsala, Sweden, 2011.

[5] M. de Raadt, S. Dekeyser, and T. Y. Lee. **A system employing peer review and enhanced computer assisted assessment of querying skills**. *Informatics in Education*, 6(1):163–178, Oct. 2007.

[6] S. W. Dietrich, E. Eckert, and K. Piscator. **WinRDBI: A Windows-based Relational Database Educational Tool**. *ACM SIGCSE Bulletin*, 29(1):126–130, Mar. 1997.

[7] T. V. Gog, L. Kester, and F. Paas. **Effects of worked examples, example-problem, and problem-example pairs on novices' learning**. *Contemporary Educational Psychology*, 36(3):212–218, 2011.

[8] A. Grillenberger and T. Brinda. **eledSQL**. In *Proceedings of the 7th Workshop in Primary and Secondary Computing Education on - WiPSCE '12*, page 101, New York, New York, USA, Nov. 2012. ACM Press.

[9] HeidiSQL. **HeidiSQL - MySQL and MSSQL made easy**. `http://www.heidisql.com/`. Accessed 28th May 2014.

[10] Karen Renaud and J. van Biljon. **Teaching SQL - Which Pedagogical Horse for This Course?** volume 3112 of *Lecture Notes in Computer Science*, pages 244–256. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[11] R. Kearns, S. Shead, and A. Fekete. **A teaching system for SQL**. In *Proceedings of the second Australasian conference on Computer science education - ACSE '97*, pages 224–231, New York, New York, USA, July 1996. ACM Press.

[12] V. Matos and R. Grasser. **Teaching Tip A Simpler (and Better) SQL Approach to Relational Division**. *Journal of Information Systems Education*, 13(2):85–88, 2002.

[13] V. Matos, R. Grasser, and P. Jalics. **The case of the missing tuple: teaching the SQL outer-join operator to undergraduate information systems students**. *Journal of Computing Sciences in Colleges*, 22(1):23–32, Oct. 2006.

[14] J. Mayes and C. Fowler. **Learning technology and usability: a framework for understanding courseware**. *Interacting with Computers*, 11(5):485–497, May 1999.

[15] Microsoft. **Microsoft Access - database software and applications**.

[16] A. Mitrovic. **A Knowledge-Based Teaching System for SQL**. In *ED-MEDIA 98*, pages 1027–1032, 1998.

[17] A. Mitrovic. **Learning SQL with a computerized tutor**. *ACM SIGCSE Bulletin*, 30(1):307–311, Mar. 1998.

[18] J. Nielsen. **Search and You May Find**. `http://www.nngroup.com/articles/search-and-you-may-find/`, 1997. Accessed: 21st March 2014.

[19] PhpMyAdmin. **phpMyAdmin**. `http://www.phpmyadmin.net/home\_page/index.php`. Accessed: 28th May 2014.

[20] J. C. Prior. **Online assessment of SQL query formulation skills**. pages 247–256, Jan. 2003.

[21] R. Rasala, V. K. Proulx, and H. J. Fell. **From animation to analysis in introductory computer science**. *ACM SIGCSE Bulletin*, 26(1):61–65, Mar. 1994.

[22] G. Russell and A. Cumming. **Improving the student learning experience for SQL using automatic marking.** In *Cognition and Exploratory Learning in Digital Age*, Jan. 2004.

[23] S. Sadiq, M. Orlowska, W. Sadiq, and J. Lin. **SQLator**. *ACM SIGCSE Bulletin*, 36(3):223, Sept. 2004.

[24] B. Shneiderman. **Improving the human factors aspect of database interactions**. *ACM Transactions on Database Systems*, 3(4):417–439, Dec. 1978.