ScrambleSQL: A Novel Drag-and-drop SQL Learning Tool

Chanapat Phewkum
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
62606063@kmitl.ac.th

Jittakan Kaewchaiya
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
58070012@kmitl.ac.th

Kazuki Kobayashi
Academic Assembly
Shinshu University
Nagano, Japan
kby@shinshu-u.ac.jp

Kanokwan Atchariyachanvanich
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
kanokwan@it.kmitl.ac.th

Abstract— **Structured Query Language (SQL) is a standard language for forming queries to access relational database systems, e.g. create table, as well as read, update, and delete data from the database. Thus, it is commonly taught in computer science classes. Technology growth helps developers to create many SQL tools which encourage students to learn effectively. Although many SQL learning tools have been developed, they are not suitable for some users who prefer to use a tablet for learning SQL, because the existing tools were not developed for a touchscreen device. Thus, we developed prototype ScrambleSQL, optimized for tablets. The purpose was to enhance learning, speed up SQL command writing and allow users to practice SQL commands anywhere and anytime, with the internet. We tested undergraduate students from an Information Technology program. They wrote SQL commands by ScrambleSQL and then completed a questionnaire. The participants were satisfied with its screen, learning and system capabilities. ScrambleSQL reduced typing errors and helped novice users to learn SQL commands with the provided keywords. In addition, the participants enjoyed learning with ScrambleSQL.**

Keywords—drag-and-drop interface, SQL learning, computer-aided instruction

I. Introduction

Data is usually stored in a database as a table and managed by SQL commands (SQL statements). SQL learning tools, which users use for writing an SQL command, have been developed significantly. They were developed for desktop computers, so a keyboard is needed to type the SQL command. However, nowadays, tablets are regularly involved in educational environments. They have advantages over desktop computers, e.g. portability, handheld and touch screen, whereas a desktop computer is heavy and needs a power supply. A tablet can enhance learning: learners can learn SQL commands anywhere and anytime they want with the internet which is easy to access. Typing on the tablet may lead to some typing errors for users who are not used to type on a touchscreen device, because the tablet screen is too large for typing, while user is holding it, and there are no physical buttons to indicate that a character was typed. Due to these problems, we developed an SQL learning tool that works in a drag-and-drop mode. Drag-and-drop can reduce the effect of typing errors [1] because users do not have to type commands. In addition, users can drag and drop any code blocks without using a mouse cursor but touch directly on the tablet.

ScrambleSQL is a new SQL learning tool which was designed by using the drag-and-drop concept. It was developed as an alternative tool to encourage users with a tablet to learn SQL commands. It was not aimed to take the place of an old typed interface SQL learning tool. Therefore, it may not suit users who prefer to type SQL commands by keyboard.

This paper presents ScrambleSQL, a new drag-and-drop interface SQL learning tool. The purpose was to enhance learning, speed up SQL command writing and allow users to practice SQL command anywhere and anytime, with the internet. We experimented with 30 participants who were students from Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang (KMITL) and finished a database system concepts class. They wrote SQL commands by using ScrambleSQL and then completed a questionnaire, which was adapted from the Questionnaire for User Interface Satisfaction (QUIS) [2].The results are shown in section V.

II. Related Work

A. Drag-and-drop concept

Drag-and-drop interfaces are used in many programing environments, because drag-and-drop interface can improve beginner performance significantly, on some programing activities, and leads to faster achievement of programing goals [3]. In addition, it can reduce typing errors and allow users to order the words easily and correctly [1].

In Heift's experiment [1], there were 27 participants, who were 19-26 year old students. They were divided into three equal groups. The first group took a multiple-choice test. The second group undertook the drag-and-drop exercise. The last group took a typed test. The exercise contained 20 scrambled sentences in German. The participants had to order the words correctly in each exercise. The drag-and-drop group performed significantly better than the multiple-choice group, and marginally better than the typed group.

Price and Barnes [3] used 31 participants. They took a programing exercise. They were divided into two groups. The first group consisted of 17 6th grade students. They used the

978-1-7281-2544-2/19/$31.00 ©2019 IEEE

drag-and-drop interface. The last group consisted of 14 7th grade students. They used the typed interface. Each group answered the same questions. The drag-and-drop interface improved beginner performance significantly on some programing activities and led to faster achievement of programing goals.

B. *Drag-and-drop learning tools*

In this study, we can divide the tools under programming environment into three groups:

The first group is represented by Scratch [4], a drag-and-drop programing tool, which emphasizes the programing algorithm. Users can program by dragging and dropping blocks. The blocks are joined vertically. After the user executes the command, each block is converted into a code statement. Users do not have to be concerned about code syntax but just put blocks together in the correct position. This method can eliminate syntax errors, because code statements are written by the tool. Therefore, Scratch is more suitable for users who want to learn an algorithm, than users who want to practice programing statements.

The second group consists of relational algebra tools, e.g. Bags, DB-Learn and DBsnap. Bags is a drag-and-drop relational algebra tool [5] like Scratch. Users can write relational algebra expressions by dragging and dropping blocks. The blocks are arranged vertically. After the user executes the command, all blocks are converted into a relational algebra expression. Then the query result is shown as a table. Similarly, DB-Learn [6] has more functions than Bags. It shows a relational algebra expression and query result as a table. It also allows users to store and retrieve projects in the cloud. DBsnap is another drag-and-drop relational algebra tool, which emphasizes the operator order [7]. Its blocks are designed as nodes, whereas Bags blocks and DB-Learn blocks are designed in a linear form. DBsnap blocks is easier to understand than the other two, because the node design shows a block structure as a tree. DBsnap shows the query result as the table and relational algebra expression immediately after the user moves the block. Relational algebra tools, with a drag-and-drop interface, help users to order the operators easily and quickly, because user does not have to waste the time to remove and type commands again.

The last group is represented by SQL Blockly [8], a drag-and-drop SQL tool like Scratch. Users can write an SQL commands by dragging and dropping blocks, which are put together vertically. Then, all blocks are converted into an SQL command. Users can also convert an SQL command into blocks. However, SQL Blockly does not show the query result. Therefore, SQL Blockly is suitable for users who want to write an SQL command but are not skilled in SQL syntax, because users do not have to write SQL commands directly and there are no syntax errors. Thus, user may not learn the SQL syntax effectively.

III. Requirements and General Strategy

A. *Requirements*

Nowadays, tablets are often involved in teaching. Most SQL learning tools use typed interfaces. We found that a typed interface SQL tool caused problems to some users, who used tablets to practice SQL commands. They had more typing errors than typing with a keyboard. Because there were no physical buttons on the tablet screen, users found it difficult to type on a large screen. Sometimes users did not notice that they had made typing errors, especially novice users. The error response from the SQL learning tool might not tell users that they had typing errors too. Some users solved the target problem by copying and pasting a word, e.g. table name and column name. Although this method reduced typing errors, it was inconvenient.

The requirements for ScrambleSQL were:

- The tool must show the query result after completing the correct SQL command.
- The tool must be accessible by any platforms via the internet.
- The tool must be convenient for users, who use a tablet to practice SQL commands.
- The tool must allow users to delete a code block or a whole blocks easily.

B. *Strategy*

We developed a web-based SQL learning tool with a drag-and-drop interface design. The tool was designed to order the blocks horizontally, because that looks similar to a typed interface SQL tool and similar to the usual representation of SQL commands, i.e. that found in texts. Thus, users, who got used to a typed interface SQL tool, could adapt to it easily. Conversely, users who had never used any SQL tools and used our tool as their first SQL tool, could adapt to a typed interface SQL tool easily too. As a web-based SQL learning tool, users can access it anywhere, anytime and on any platforms by any web browser. The drag-and-drop interface can help tablet users to practice SQL commands, because they can drag and drop blocks directly with their fingers. We used the Angular framework [9], because it is fast and efficient. Thus, the tool interacts directly with users.

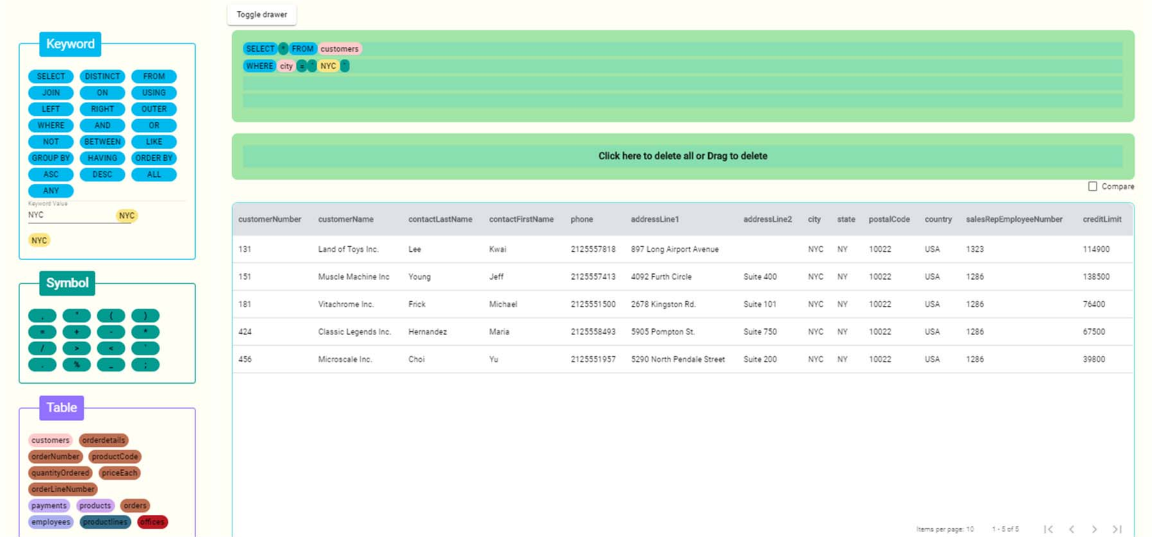
IV. ScrambleSQL Learning Tool

A. *Quick Guide*

The ScrambleSQL user interface was designed to be similar to and consistent with existing SQL learning tools, as shown in Fig. 1. It is divided into four panes:

Keyword pane - area at the left side, which contains many blocks, e.g. table name, column name, SQL keyword and SQL operator symbols. Users can create a SQL statement by dragging a block from the Keyword pane and dropping it on the Playground pane. The blocks are divided into three subareas.

- Keyword subarea contains many SQL keyword blocks. Users can create a custom value block by typing on the Keyword Value input. The custom value block is stored as a history after it has been dragged from the Keyword pane and dropped on the Playground pane. The history can store up to five custom value blocks.
- Symbol subarea contains many SQL operator symbol blocks.
- Table subarea contains many table name and column name blocks. The column name blocks are hidden first.



customerNumber	customerName	contactLastName	contactFirstName	phone	addressLine1	addressLine2	city	state	postalCode	country	salesRepEmployeeNumber	creditLimit
131	Land of Toys Inc.	Lee	Kwai	2125557818	897 Long Airport Avenue		NYC	NY	10022	USA	1323	114900
151	Muscle Machine Inc	Young	Jeff	2125557413	4092 Furth Circle	Suite 400	NYC	NY	10022	USA	1286	138500
181	Vitachrome Inc.	Frick	Michael	2125551500	2678 Kingston Rd.	Suite 101	NYC	NY	10022	USA	1286	76400
424	Classic Legends Inc.	Hernandez	Maria	2125558493	5905 Pompton St.	Suite 750	NYC	NY	10022	USA	1286	67500
456	Microscale Inc.	Choi	Yu	2125551957	5290 North Pendale Street	Suite 200	NYC	NY	10022	USA	1286	39800

Fig. 1. ScrambleSQL user interface.

If user presses a table name block, column name blocks which are related to the selected table name block are shown. They are hidden again when the user selects the table name block.

Playground pane - area at the top, which allows user to drop a block from the Keyword pane. Users can arrange blocks in this area freely. Whenever a block in this pane is changed, the SQL statement is executed. If the statement is correct, the query result is shown as the result table. Otherwise, a query error is shown at the top of this pane.

Drop pane - area between the Playground pane and the Result table pane, which allows the user to remove any blocks from the Playground pane. Users can drag a block from the Playground pane and drop it on this pane to remove that block. Users can also remove all blocks in the Playground pane by pressing on this pane.

Result table pane - area at the bottom, which shows result of the executed SQL statement. It is created by every block in the Playground pane, as a table.

B. *Architecture*

ScrambleSQL uses client-server architecture. The client side was developed with Angular [9], which has advantages of speed and performance. The server side was developed with Express [10], which allows ScrambleSQL's application page to be rendered on the server, using Server-Side Rendering (SSR). It was chosen over the Client-Side Rendering (CSR), for users with cheap or old tablets: generally, the server hardware will be more powerful than the tablet. We used MySQL as a database and used an Application Programming Interface (API) to query data from MySQL.

Each class of Angular (e.g. component and service) performs different tasks, which are defined by metadata. Component is a class, which binds a data to a template defined in the component metadata. Service is a class which manages (e.g. stores and retrieves) data between each class in the same module. Service was also used for sending and receiving data from an API. A flow diagram of the ScrambleSQL system is shown on Fig. 2. The individual components are:

- tool.component - main controller class of ScrambleSQL. It creates an application main page. There are many drag-and-drop containers (e.g. Keyword subarea, Symbol subarea, Table subarea, Playground pane and Drop pane) which were implemented using Angular Material's Component Dev Kit (CDK) [11]. It also contains result-table.component whose view is shown on the application main page.
- result-table.component creates a data table view. The data table was generated based on Angular Material's data table [11]. The query result is shown as a table by this class.
- keyword.service stores all keyword block objects and receives table names and column names which are available from the API.
- query.service sends the SQL statement which was created by the user to the API and receives the query data object from the API.
- Keyword - a model class. It defines the keyword block object properties.

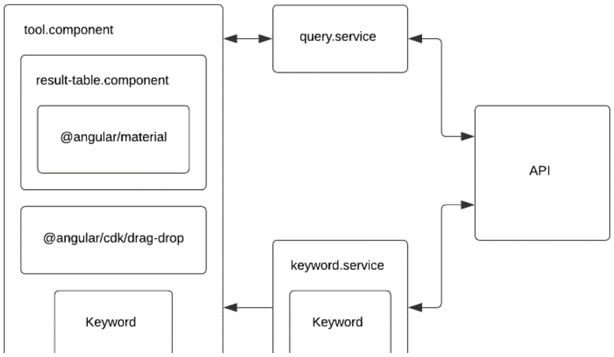


Fig. 2. ScrambleSQL data flow diagram.

C. *Workflow*

Fig. 3 shows the ScrambleSQL simple workflow. It can be explained as follows:

1. On the client, a user writes an SQL statement by dragging and dropping keyword blocks.
2. The values from the keyword blocks are concatenated to create an SQL statement string.
3. The SQL statement string is sent to the server.
4. On the server, the SQL statement string from the client is sent to query on the database.
5. The query result is sent back to the server.
6. A result object is created from the query result.
7. The result object is sent back to the client.
8. The result object is shown as a table.

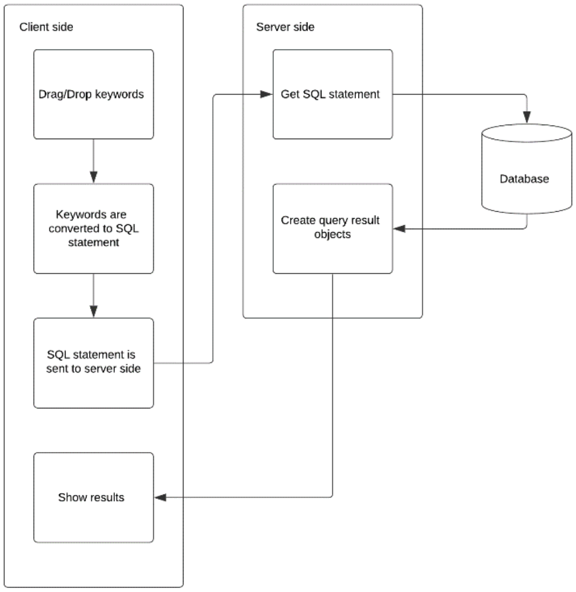

Client side
Server side
Drag/Drop keywords
Keywords are converted to SQL statement
SQL statement is sent to server side
Show results
Get SQL statement
Create query result objects
Database

Fig. 3. ScrambleSQL simple workflow.

V. Experiment

A. *Procedure*

We tested with 30 participants who were undergraduate students from Faculty of Information Technology, KMITL. They had finished a database system concepts class. We tested them individually. We took 10 minutes to guide the participants how to use the tool. Next, we gave the same basic SQL command exercise to all participants to solve with ScrambleSQL. After that, the participant answered a questionnaire about the tool.

B. *Result*

The questionnaire adapted from QUIS [2], used a 5-point Likert Scale, where 1 = "Strongly Disagree", 2 = "Disagree", 3 = "Neutral", 4 = "Agree", and 5 = "Strongly Agree". There were three dimensions, i.e. screen, learning and system capabilities. The question mean score was calculated from a normal mean method. The dimension mean score was calculated from sum of every question mean score and divided by the number of questions in the dimension. Questions in the tables have been translated to English from the original Thai for this paper.

The screen dimension was about the User Interface (UI). There were three questions, as shown in TABLE I. The participants were somewhat satisfied with the screen dimension because every question score was close to 4.0.

TABLE I. Screen Dimension Scores

Question	Mean scores	S.D.
Characters are seen clearly	3.96	0.96
Main components are highlighted properly	3.76	0.77
System layout is arranged properly	3.73	0.86

The learning dimension was about encouraging users. There were four questions, as shown in TABLE II. The participants are quite satisfied with "User learns to operate the system quickly", because the score was greater than 4.0. However, "User enjoys learning" score was less than 4.0, whereas its S.D. was 1.21 implying that participants enjoyment of the system varied from disagree to strongly agree.

TABLE II. Learning Dimension Scores

Question	Mean scores	S.D.
User learnt to operate the system quickly	4.26	0.63
Error messages were shown clearly	3.96	0.76
User remembered names and use of commands quickly	3.90	0.80
User enjoyed learning	3.90	1.21

The system capabilities dimension assessed performance of ScrambleSQL. There were three questions, as shown in TABLE III. The participants were quite satisfied with this dimension, except "Easy to use" score. This may be because participants, who could touch type quickly, did not prefer a drag-and-drop interface.

TABLE III. System Capabilities Dimension Scores

Question	Mean scores	S.D.
System is fast	4.16	0.69
System has good reliability	4.06	0.69
System is easy to use	3.56	1.07

The overall scores for ScrambleSQL were close to 4.0, as shown in TABLE IV, i.e. participants were satisfied with the screen, learning and system capabilities.

TABLE IV. OVERALL SCORES OF SCRAMBLESQL

Dimension	Mean scores	S.D.
Screen	3.82	0.12
Learning	4.00	0.17
System capabilities	3.93	0.32

VI. CONCLUSION

ScrambleSQL is a new drag-and-drop SQL learning tool. The prototype version was developed with Angular and tested with 30 participants, who were undergraduate students majoring in Information Technology and had finished a database concepts class. They were satisfied with screen, learning and system capabilities of ScrambleSQL. ScrambleSQL reduced typing errors and helped novice users to learn SQL commands with the provided keywords. In addition, the participants enjoyed learning with ScrambleSQL.

In the future, we plan to develop a query result comparison feature of SQL commands. It may help users to work out the difference and understand the SQL command clearly. Moreover, we will implement ScrambleSQL to use along with an e-leaning system for supporting instructors in a database concepts class.

REFERENCES

[1] T. Heift, "Drag or Type, But Don't Click: A Study on the Effectiveness of Different CALL Exercise Types," *Canadian Journal of Applied Linguistics,* vol. 6, no. 1, pp. 69-85, 2003.

[2] J. P. Chin, V. A. Diehl and K. L. Norman, "Development of an instrument measuring user satisfaction of the human-computer interface," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Washington, D.C., USA, 1988.

[3] T. W. Price and T. Barnes, "Comparing Textual and Block Interfaces in a Novice," in *Proceedings of the eleventh annual International Conference on International Computing Education Research*, Omaha, Nebraska, USA, 2015.

[4] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman and Y. Kafai, "Scratch: programming for all," *Communications of the ACM,* vol. 5, no. 11, pp. 60-67, 2009.

[5] J. Gorman, S. Gsell and C. Mayfield, "Learning relational algebra by snapping blocks," in *Proceedings of the 45th ACM technical symposium on Computer science education*, Atlanta, Georgia, USA, 2014.

[6] V. R, S. KP and P. Menon, "DB-Learn: Studying Relational Algebra Concepts by Snapping Blocks," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bangalore, India, 2018.

[7] Y. N. Silva and J. Chon, "DBSnap: Learning Database Queries by Snapping Blocks," in *46th SIGCSE Technical Symposium on Computer Science Education*, Kansas City, United States, 2015.

[8] M. Kolodziejczyk and H. Diel, "SQL Blockly," SPE Systemhaus GmbH, 18 April 2018. [Online]. Available: https://sql.spe-systemhaus.de. [Accessed 1 October 2018].

[9] Google, "Angular," Google, 20 September 2018. [Online]. Available: https://angular.io/. [Accessed 3 October 2018].

[10] Google, "Angular Material," Google, 21 September 2018. [Online]. Available: https://material.angular.io/. [Accessed October 4 2018].

[11] D. C. Wilson, "Express," StrongLoop, 13 March 2018. [Online]. Available: https://expressjs.com/. [Accessed 13 October 2018].