TriQL Report

Abstract: In this report, we worked on conducting a survey to gather students' opinions about their experience using TriQL, a learning tool for SQL, MongoDB, and Neo4j query languages and structure. The effectiveness of this tool is evaluated based on queries in which the students select the attributes and conditions specified. The output from TriQL reports the queries in the three different syntaxes. The intent of this report is to develop a pilot study to see the effectiveness of the tool and determine how to proceed with an analytical scientific approach for evaluating TriQL as a learning module.

Introduction

As educational institutions teach database systems and different query languages there is not much research about learning secondary or tertiary database languages after the primary one. TriQL (i.e., Tribus linguist query, Latin for three query languages) [Alawini et al. 2022] is a tool for teaching query languages: SQL, MongoDB, and Neo4j based on a simple query generation user interface. The results displayed allow the students to see the differences among the three languages. The intent is for the student to focus on learning the three database query languages and the structures (schema) of these.

The popularity of database systems encourages students to be familiar with multiple languages. However, familiarity is least preferred because this system uses advanced database techniques, such as data integration and logical programming, to capture the core data operations common between the relation, graph, and document-oriented data models, including selection (filtering data), projection (redefining the output schema), grouping and aggregation [Alawini et al. 2022].

Related Work

The DBQA[1] system focused on evaluating subqueries and providing students with meaningful error messages but only focusing on a single database. The uniqueness of DBQA intends on advancing a student's understanding of more complex queries through subqueries and JOINs while providing understandable error messages. The user's-based interface provides all tables, primary keys, and foreign keys of the schema, to avoid the need of memorizing the schema, similar to how TriQL reports queries and shows the table(s) schema. A user study has not been completed at the date of this paper.

Testing of basic SQL query language knowledge is completed with the SQL Tester as tested by Kleerekoper, et al.[2] Students take practice tests and are allowed to revise their answers in a subsequent practice test. The output of SQL Tester illustrates four windows: 1) question, 2), database tables, 3) output from your query, and 4) desired output which allows the user to see results from the executed query. TriQL is similar in that it shows the query, the results from the query and the table schema. If a user repeats this process over time, in both cases students may

[1] Ryan Hardt, Esther Gutzmer. 2017. Database Query Analyzer (DBQA) - A Data-Oriented SQL Clause Visualization Tool. *SIGITE '17*, (October 4 - 7, 2017), 147 - 152.
[2] Anthony Kleerekoper, Andrew Schofield. SQL Tester: An Online SQL Assessment Tool and Its Impact. *ITiCSE ' 18*, (July 2 - 4, 2018), 87 - 92.

become familiar with question semantics while also increasing their knowledge of SQL syntax, whereas TriQL also exposes the user to MongoDB and Neo4J and over time would presumably increase the user's knowledge.

The problem-based learning approach along with the teacher's grading methodology influences how deep a student's knowledge is of SQL. Prior and Lister[3] examine how an online grading tool used in their university department reviews student work. This differs from TriQL because the approach as the article name mentions *backwash*, begins with the grading as the first dynamic assessed for learning and then designs lessons for learning SQL.

A virtual learning environment for SQL called LabDER is supported by Google Analytics and tracks students' responses worldwide to multiple choice or discursive questions about SQL or entity relationship diagrams (ERD). LabDER, a relational database virtual learning environment that has a set of functionalities for implementing a teaching plan and automatically evaluating students' responses, which can be multiple-choice or discursive or involve SQL or ERDs.[4] LabDER does not support MongoDB or Neo4J. However, it allows professors to build their instructional lessons into modules where students answer questions on the topics and receive feedback.

The student feedback received from answering SQL data definition (DDL) questions has improved grades. There are two modules, one for the student and another for the teacher to grade. The report measures performance based on minimum requirements for the accuracy of the schema. The benefit of this student learning tutorial as well as TriQL is providing access to the language for the student to receive feedback and/or provide the expected values. Yang measures the success of the tool based on improvement in student grades over time.[5]

Preliminary User Study Design

We worked on designing a preliminary user study to help gather TriQL users' feedback that will be used in improving the TriQL user experience. The study consists of 1) a set of queries that the user can run in TriQL. 2) a user study questionnaire to record the user experience and feedback that can help improve TriQL.

Methods

We started with designing the user study by breaking down our study to three basic steps:
1)Reading research articles about educational tools for teaching syntax and schema on databases. The articles we read focused on relational databases. All used SQL as the query

---

[3] Julia Coleman Prior, Raymond Lister. 2014. The Backwash Effect on SQL Skills Grading. *In ACM SIGCSE Bulletin*, (September 2004), 32 - 36.

[4] Adriano Lino, Alvaro Rocha, Luis Macedo, Amanda Sizo. 2019. LabDER - Relational Database Virtual Learning Environment. *In Creative Commons License Attribution 4.0*, (2019), 42 - 55

[5] J.C.Yang, et al. 2018. Semi-automated Assessment of SQL Schemas via Database Unit Testing. *In Proceedings of the 26th International Conference on Computers in Education. Philippines: Asia-Pacific Society for Computers in Education,* (2018).

language. The research studied how students interacted with multiple-choice questions and immediate feedback.

TriQL is a system for helping novices learn three major database systems, including relational (MySQL), graph (Neo4J), and document-oriented (MongoDB), and their query languages.

Figure 1: shows the main component of TriQL, Once a user submits a generic query*(represented as a JSON, JavaScript Object Notion, structure) via the Query Builder Interface (QBI).

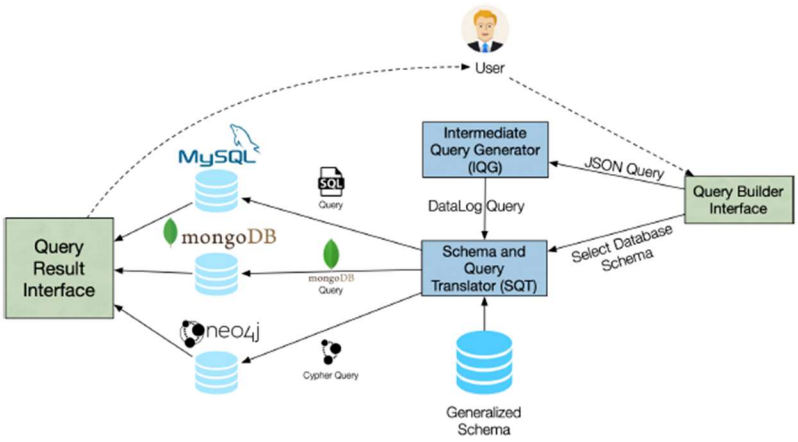


Figure 1: TriQL System Architecture: The Query Builder for building queries using a GUI; the Intermediate Query Generator converts user queries to DataLog; The Schema and Query Translator generates the schema of the three database and coverts the DataLog query into SQL, Cypher and MongoDB

2) Define our targeted student group (i.e., students with previous knowledge of database and one of the three query languages, students with no previous knowledge of database).

3) Designing the study protocol (i.e., in-person interviews, providing the targeted group with the TriQL link to learn about the tool, and then running the set of queries provided to them, followed by a brief survey).

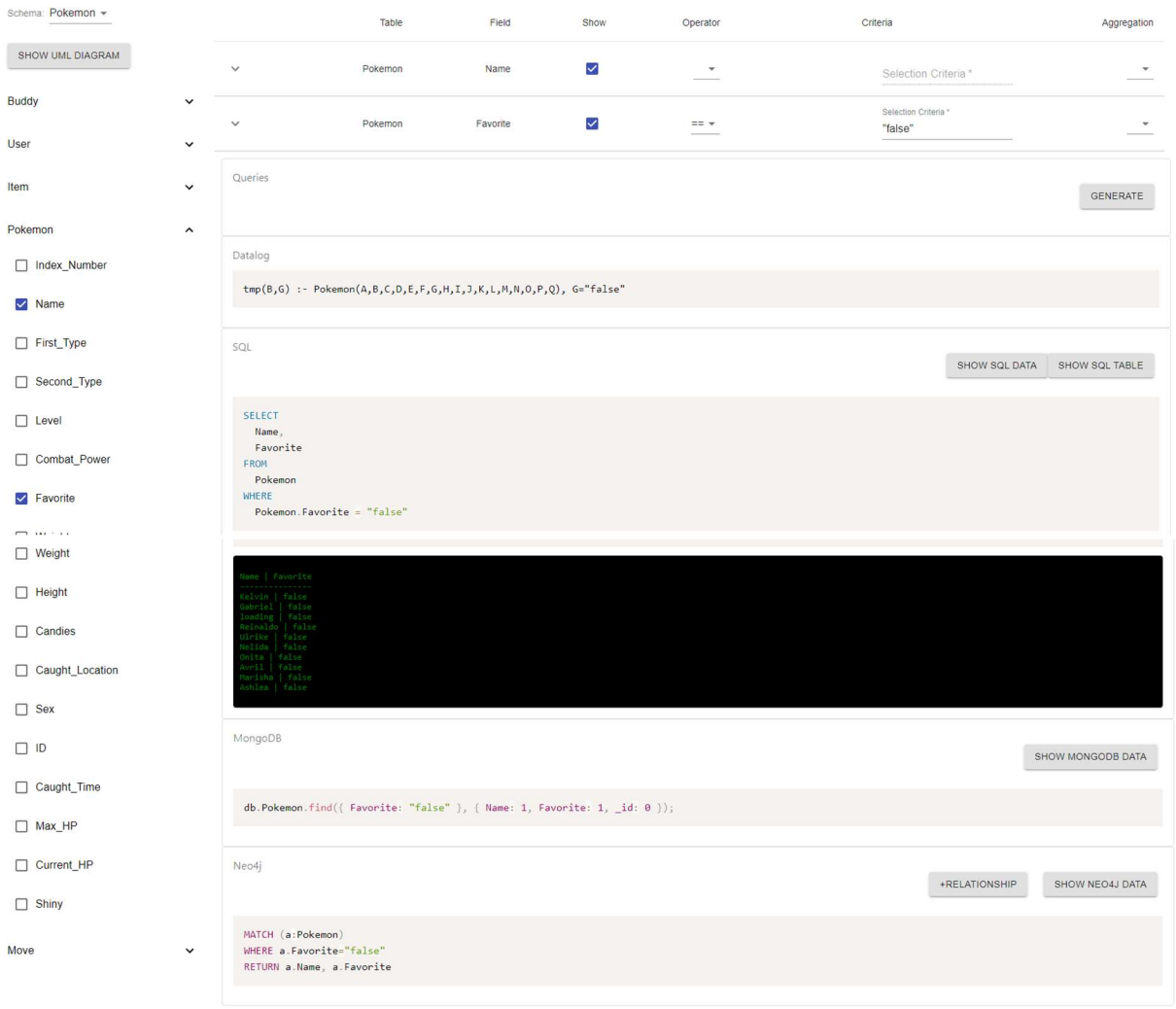
We explored the Pokemon database and designed queries to run our study. The Pokemon Database consists of 5 tables, 1) Buddy, 2) User, 3) Item, 4) Pokemon, and 5) Move. The Buddy table was disregarded because the primary key, ID, did not exist.

Based on the information above, we designed a set of queries for students to run when they work with TriQL which are listed below:

1. List the Pokeman Name(s) that are not Favorites.
2. Determine the average Level of all users and list the Names where their level is greater than 15.
3. List the User Name(s) and the Name of their Favorite Pokeman that plays above Level 28.
4. List the female non-shiny Pokemon.
5. Name all the Pokemon that the User Jeffery plays with and their Fast_TM_Power Moves.

6. Count all the Pokemon grouped by Favorite.
7. List the prescription Item_Name with a Charge_TM Move and their Shiny Pokeman status.

If we used the 1st query for example to display how the TriQL works the output will be as follows:



Our process while using the tool ran up against system issues in which we could only modify the user query a few times, this was permitted. To modify the query multiple times we had to refresh TriQL and return to the homepage and start the process again. Once we got familiar with TriQL, we were able to execute queries that worked and transcribed them into questions.

The survey questions were drafted in synchronization with the queries. While we played with TriQL we derived our questions from our experience and also whether TriQL would enhance the learning experience of database students. Another theme of our survey questions was questions related to TriQL's system design, such as user interface quality and ease of use. Since this is a pilot study, any lessons learned can be improved in the next iteration of the study.

We designed the survey to take about 10-15 minutes and wrote questions about TriQL's areas of improvement, learning usefulness perception, and system usability. Some questions require a yes or no answer and others require a short answer.

The questions listed below are used to create a google survey that can be reached here https://forms.gle/4eyTETqvVUu92qVG9 :

1. Do you feel TriQL helped you better understand the syntax differences among the three database languages (SQL, Neo4j, MongoDB)?
2. Did TriQL help you better understand the differences among the data models (i.e., relational, document, graph) of each database (i.e., MySQL, MongoDB, Neo4J)?
3. Did TriQL improve your ability in writing proper syntax for the three languages?
4. Do you think TriQL can help you reduce the time it takes to learn the syntax for any of the three databases?
5. Would you have preferred to use TriQL prior to or after learning each of the database languages?
6. Would you prefer to have TriQL at the beginning of the class or prior to the exam(s)?
7. Did TriQL help you prepare for the exam?
8. Was TriQL user-friendly or unnecessarily complicated?
9. Does TriQL increase your interest in learning the different database languages and structures?
10. Did TriQL improve your understanding of table joins?
11. What functionality would you like incorporated in TriQL that is not currently available?
12. What was your overall experience working with TriQL?
13. What, if any, were glitches you experienced while using TriQL?
14. Did you find errors in the syntax for any of the languages?
15. Can you easily distinguish the differences between the syntax?
16. Did you like the design of TriQL?
17. What are your recommendations for improving the front-end experience?
18. Could you have used TriQL if you did not have database knowledge?
19. Would you recommend TriQL to new students?
20. For how long have you used TriQL?
21. Would it be helpful to be able to export the query data (i.e., excel, notebook, etc.)

Conclusion and Future Work

TriQL identified positive learning outcomes for this research team. While we did not have prior experience or formal training on the three databases, we found that TriQL increased our knowledge on recognizing the differences among the syntax and different aspects of a database from the schema design, assigning and using the primary key in the JOIN function, defining the criteria of the query, and understanding the syntax of each query language.

A students interview and a qualitative analysis will be conducted in the future using the data will be collected from our survey to help with improving the TriQL and how it can be integrated into an introductory database curriculum as part of the database programming lab assignments.