Improving Database Learning with an Automatic Judge

Enrique Martin-Martin*, Manuel Montenegro†, Adrián Riesco‡, Rubén Rubio§
Dpto. Sistemas Informáticos y Computación, *Universidad Complutense de Madrid*, Madrid, Spain
Email: *emartinm@ucm.es, †montenegro@fdi.ucm.es, ‡ariesco@ucm.es, §rubenrub@ucm.es

***Abstract*—Databases are a key subject in several technical degrees. Because they have a strong practical nature, students require a large number of problems to master them. However, these problems are useful only if accurate and timely feedback is provided. In this paper, we present the learning improvements obtained by using LearnSQL, an automatic judge that has been designed to complement face-to-face lectures. We have measured the impact of this judge during the 2021/22 academic year and report promising results both in student engagement and final grades.**

***Index Terms*—Database Learning, Automatic Judge, SQL**

I. Introduction

Databases and data-processing systems are key topics in many technical degrees (see, e.g., the computer science curricula [1]). In fact, databases are ubiquitous and being able to manage them is a basic ability required by all companies. Database lectures include both theoretical aspects, in particular database design, and practical aspects, which include creating, searching, and indexing data. We will focus on these practical aspects, which score higher in Bloom's taxonomy [2]. In this sense, it is very important to provide the students with (i) as many programming exercises as possible and (ii) immediate feedback, so they are not blocked when an error occurs. However, although it is possible to generate a large amount of exercises, the high number of students in standard classrooms prevents instructors from giving this feedback timely. For this reason, it is important to provide automatic means to assess exercises, but it is also important that this feedback is as informative as possible, because the users are students that have not mastered the subject yet.

Automatic assessment [3] is a well-known field that tries to solve this problem by automatically generating a reply (the type of reply depends on the technique) to students' exercises. Among these techniques, we are interested in automatic judges [4], [5], which were initially developed for quickly evaluating programming exercises in competitions [6]. These judges, which only produced a plain correct/wrong answer, have been adapted to different environments (mainly to teaching of programming languages), easing the evaluation of a large number of students in a short time. In our case, we are not concerned with evaluation but with teaching, so we must provide a collection of exercises, including timely and concrete feedback, as large as possible. In this sense, we will take the main ideas of these systems and enrich them to focus on learning.

DOI reference number: 10.18293/SEKE2022-025

In this paper, we present the learning improvements obtained by using LearnSQL in our database course. LearnSQL is an automatic judge designed to complement face-to-face lectures with a wide set of SQL problems that students can try at any time. The judge supports not only problems about SQL queries but also procedural SQL problems where the student is expected to define procedures, functions, and triggers, so it covers a large part of the course syllabus. Moreover, LearnSQL provides detailed feedback pointing to the source of the error, so students can understand their mistakes and fix their code. When measuring the learning improvements, we are mainly interested in finding evidence that students that use the judge obtain better grades than those who do not. However, we are also interested in discovering the usage degree and student engagement with the judge.

The rest of the paper is organized as follows: Section II presents the context of the subject where LearnSQL is used. Section III introduces the judge, while Section IV evaluates the effects on the students' performance. Finally, Section V concludes and presents some topics of future work. The source code of LearnSQL is available at https://github.com/emartinm/lsql.

II. Context of the Course

We have used LearnSQL in the introductory course on databases which is part of the degree programs at the Faculty of Computer Science of the Complutense University of Madrid, Spain. These degree programs cover 4 years, being the databases course taught in the first semester of the 2nd year. Specifically, for this evaluation of the learning improvement, we used the judge in the academic year 2021–2022, i.e., from September 2021 to January 2022. The syllabus of the introductory course on databases covers the standard contents [7], [8]: relational model, entity-relationship model, SQL queries, procedural SQL (functions, procedures, and triggers), and transactions. The teaching in the introductory course on databases is face-to-face, organized in 30 lectures of 100 minutes each. Approximately 50% of the lectures are theoretical and the other 50% are practical sessions where the students solve exercises that require performing SQL queries and defining functions, procedures, or triggers in the database. In the year 2021–2022 there were 6 different groups in this course, with 40–80 students in each group.

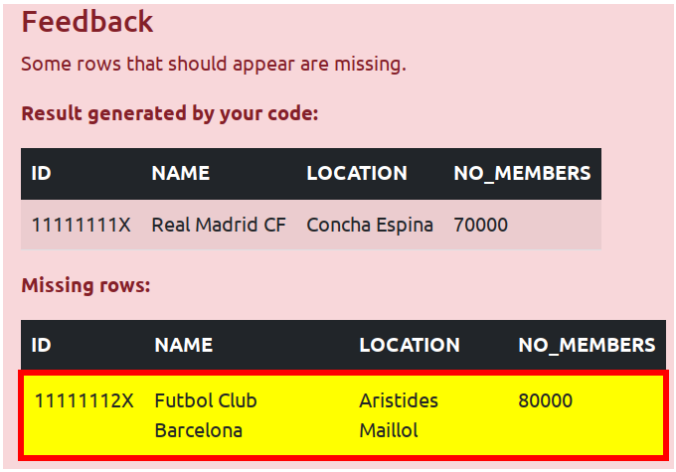

Fig. 1. LearnSQL feedback marking missing rows

For the evaluation of LearnSQL, we have used the judge in 3 groups of the course, taught by two different teachers, totaling 157 students. All the students in these selected groups have had access to the judge and all the exercises from the beginning of the course, concretely 166 problems about SQL queries and 23 problems about procedural SQL (functions, procedures, and triggers). We introduced the judge to the students in the first weeks of the course, and we encouraged them to use it to solve the exercises of the practical classes. However, the use of LearnSQL was completely voluntary for the students: the assignments in the practical classes were free practice exercises that were not assessed and did not have any impact on the final grade of the course. In other words, all students could freely use LearnSQL as a tool to practice SQL if they considered it was useful for them.

The 70% of the subject grade was obtained on the basis of a final written exam. This exam, with a total mark of 10 points, was composed of 3 parts. The first part covered the design of databases for a total of 3.5 points. The second part was the main component of the exam, with exercises about SQL queries and procedural SQL up to a total of 6 points. This is the part of the exam we will focus on when evaluating the impact of LearnSQL on students' learning (see Section IV). Finally, there was a final part about transactions with a value of 0.5 points.

III. LearnSQL

LearnSQL is an automatic judge for database problems. It is open-source software available at https://github.com/emartinm/lsql/ under the MIT license, so any instructor can deploy it in their server and adapt it to their needs. LearnSQL provides a simple web interface where students can browse the collections and problems, as well as submit their solutions by writing code in a text area. One of the key design features of LearnSQL is that it is an automatic judge *for learning*, so it provides detailed feedback when the student submits an incorrect solution. This detailed information helps students to understand their mistakes and fix their solutions, reinforcing positively their learning process. The received feedback ranges from syntax errors to differences in the schema of the solution (different number of columns, or wrong datatypes or names) or a detailed list of missing or incorrect rows, as shown in Figure 1.

LearnSQL supports several types of problems that are suitable for an introductory databases course:

- SQL queries: the student is asked to provide an SQL query that returns some expected results from the database.
- DML sentences: the student's code is expected to produce changes in the database by adding, removing, or updating rows.
- Function and procedure definitions: the judge asks for the definition of a function or procedure fulfilling some expected behavior when invoked.
- Triggers: the student has to write a trigger linked to a table manipulation, which must perform some modifications in the database.
- Discriminate SQL queries: the judge shows two SQL queries to the student that are very similar but not equivalent. The goal is that the student has to provide a set of rows in the involved tables such that the queries produce different results. This kind of problems is very interesting because it requires students to reach the highest levels of Bloom's taxonomy [2], as they must evaluate the queries and create a database state that could differentiate them.

To verify the correctness of a student's submission LearnSQL follows a simple approach: it executes the student's code in the DBMS and compares the obtained results to the expected ones generated by the official instructor's solution. This comparison can be done with more than one test case to increase the accuracy of the judgment. Currently, LearnSQL only supports Oracle 11g as DBMS for executing SQL code because is the system used in our introductory databases course. However, the SQL execution component follows a clear interface and could be easily replaced to connect to any other relational DB system like PostgreSQL or MySQL.

Apart from the judgment based on execution, LearnSQL also provides a richer feedback obtained by a semantic analysis of the student code. For that, it relies on the Datalog Educational System [9] (DES). DES is a deductive database system that supports many formats and query methods: Datalog, Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus, and SQL. When handling a query, DES performs a set of complex analyses on the code that can detect several semantic errors [10], e.g. unnecessary joins, inconsistent or tautological conditions, or unnecessary subqueries. This information is very relevant to students as it allows them to not only fix but also improve and simplify their SQL code, and gain more knowledge about their solution.

Finally, LearnSQL has been designed to be multilingual, so it could be used in any teaching environment. At the moment it only supports Spanish and English, but it could be easily adapted to any other language by providing a text file with the translation of all the strings used in the system. From the point

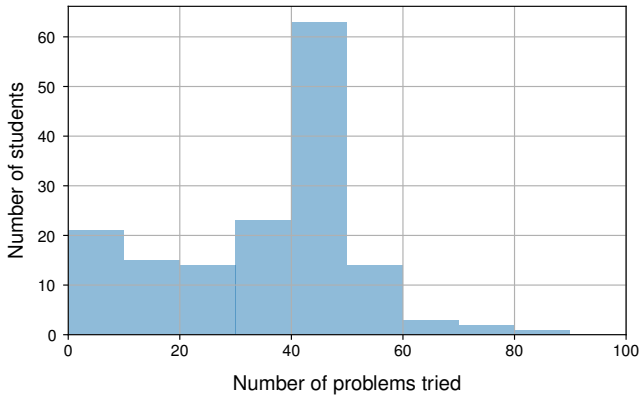

Fig. 2. Problems tried by student

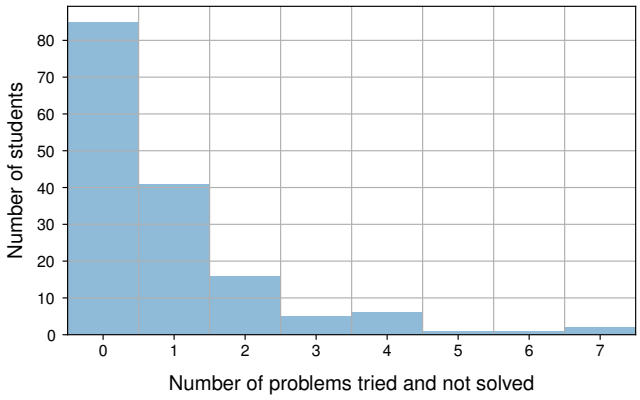

Fig. 3. Problems tried but not solved by student

of view of motivation and linked to gamification [11], LearnSQL supports an interesting feature: achievements. Teachers can define achievements that students obtain when solving a certain number of problems or collections, or a given number of problems of a concrete type. Apart from the satisfaction of obtaining recognition for their work, the achievements are shown as badges in the global ranking, fostering a healthy competition in the classroom.

IV. Evaluation

In this section, we first discuss how the judge was used according to its activity logs. Combining this information with the marks of the final assessed exam, we analyze whether there is quantitative evidence to support that the judge has had a positive effect on the students' performance and learning process.

Out of the 157 students registered in the course described in Section II, we have limited our analysis to the 130 who have attended the final exam, since their marks are useful for our analysis. However, among those who have not attended this exam, 15 have also practiced with LearnSQL, solving 25.8 problems in mean. The students have tried an unequal amount of exercises among the 189 available in the judge, as shown in Figure 2, and effectively solved the majority of those tried. The latter is illustrated in Figure 3, where it can be seen that just a few problems were left unsolved (7 at most, but most of the students left between 0 and 2), with an average of 8.18 attempts before giving up. The number of attempts for a student to solve a given problem is usually low ($\mu = 2.57$, $\sigma = 3.88$), the first try has been enough in the 60.54% of the cases, and the second attempt in a 15.64%. However, except for a student with a single submission, everyone has failed some attempts and obtained feedback from the judge.

We realize that most students have tried to solve around 40 exercises ($\mu = 38.78$, $\sigma = 16.28$), and only 8 of them have not used LearnSQL at all. Around 30 students have used the judge throughout the whole course duration, but the majority have concentrated their interaction during the first to second month of the lessons, two months before the exam. This time window coincides with the time when SQL queries and procedural SQL

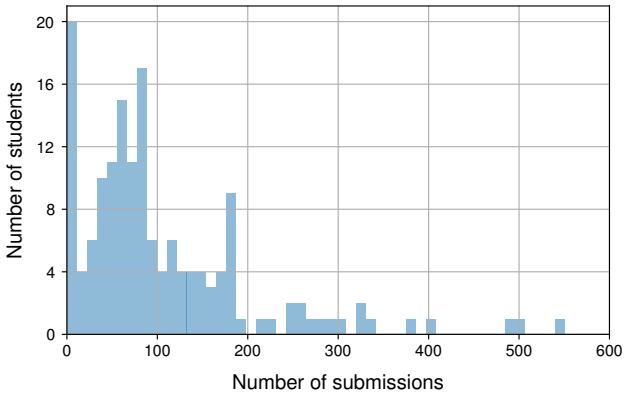

Fig. 4. Problems submitted by student

are taught in class. However, shortly before the final exam, the number of submissions has slightly increased. 85.91% of the submissions have been done outside class hours. It is also interesting to consider the information in Figure 4, which indicates that the judge has been used extensively. Although some students have not used the judge at all, those that have used it have submitted many times, and some of them sent more than 500 submissions. Combining this information with the one presented in Figure 3, we consider that students are engaged by the judge and do not stop until the problems have been solved.

In order to evaluate the effect on learning of LearnSQL, we compare the usage profile of the students with the marks they have obtained in the SQL-related exercises of the final exam. Figure 5 shows the distribution of those marks aggregated into five equally numerous subsets by increasing number of tried problems, where we can observe an increasing trend on the marks. The Spearman's and Pearson's correlation coefficients between these metrics are respectively 0.497 ($p = 1.8 \cdot 10^{-9}$) and 0.457 ($p = 4.6 \cdot 10^{-8}$), so there is statistically significant (linear) relationship between them. In summary, we consider that practicing with LearnSQL has a positive effect that becomes more noticeable as the amount of practice increases, but it is also observable with relatively little training.

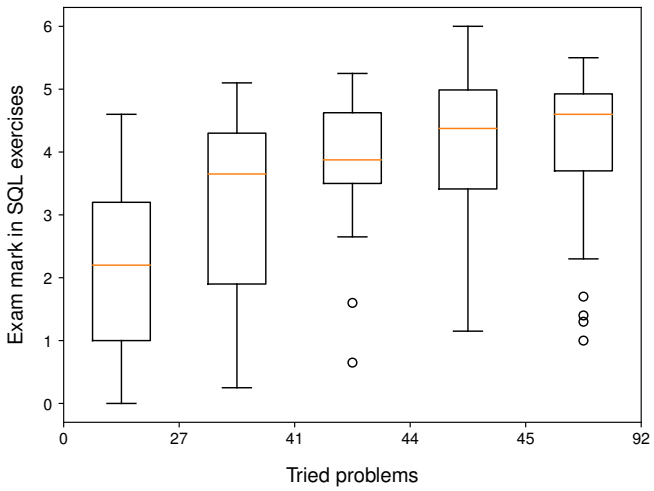

Fig. 5. Mean mark by tried problems

Regarding specific categories of exercises, the students who solved at least one problem on procedures in the judge have obtained a mean mark of 1.59 out of 2, while the mean mark of the rest is 0.99. Similarly, those who solved the five problems on triggers have obtained a mark of 0.95 out of 1 on average, compared to 0.45 for those who have not tried any problem of this kind.

We should acknowledge some threats to the validity of these conclusions. Since the usage of the judge was discretionary for the students, its hypothetical effect on the performance in the final exam could alternatively be explained by the general intuition that stronger and more enthusiastic students participate more in the activities of the course and also obtain better marks. However, the Spearman's correlation between the number of problems solved and the marks of exam exercises not related to SQL is much lower ($\rho = 0.25$, $p = 0.003$). Even if we assume that students have gained SQL skills by practicing with LearnSQL, we may wonder whether this same improvement could have been obtained with the classical non-assessed exercises, but more complex experiments would be required to analyze this.

V. Conclusion and Ongoing Work

In this paper, we have presented the learning improvements obtained by using LearnSQL in our database teaching. LearnSQL is a judge for automatically assessing the correctness of database exercises, including SQL queries, triggers, and procedures, among others. Unlike other automatic judges, LearnSQL has been designed to provide detailed explanations to students in order to help them detect their errors. LearnSQL has been used during the 2021/22 academic year and its usefulness has been measured. The results are promising, as there is statistical evidence that using the judge improves the final degree obtained by the students.

As future work, we plan to integrate the judge into Moodle, the learning platform used in our virtual campus, following an approach similar to the one in [12]. In this way, it would use LearnSQL to (possibly partially) validate assignments from the students and to integrate the marks obtained using our judge with the rest of marks, as well as presenting the students a single source for the whole subject.

In order to obtain further insight into the learning benefits of LearnSQL, we consider repeating the evaluation next year including other groups where classical non-assessed exercises are proposed to the students as control groups. Moreover, for confirming the conjectured positive effect of the informative feedback provided by the judge, we plan to offer to a separate student group a restricted version of LearnSQL where only a binary answer (correct/wrong) is obtained.

Finally, it might be interesting to add more problems and to test the tools in different degrees. In particular, we are interested in analyzing the impact of using this kind of judge with the students from the Mathematics degree.

Acknowledgements

This work has been supported by the teaching innovation projects of the Complutense University of Madrid INNOVA-Docencia 2020-21/18 and 2021-22/387.

References

[1] "Computing Curricula 2020," December 2020, https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf.

[2] B. S. Bloom, M. D. Engelhart, E. J. Furst, W. H. Hill, and D. R. Krathwohl, *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain*. David McKay Company, 1956.

[3] P. Ihantola, T. Ahoniemi, V. Karavirta, and O. Seppälä, "Review of recent systems for automatic assessment of programming assignments," in *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, ser. Koli Calling 2010. ACM, 2010, pp. 86–93.

[4] N. Gerritsen, T. Kinkhorst, and T. Werth, *DOMJudge 8.0 manual*, DOMJudge, 2022.

[5] L. Llana, E. Martin-Martin, C. Pareja-Flores, and J. Á. Velázquez-Iturbide, "FLOP: A user-friendly system for automated program assessment," *J. Univers. Comput. Sci.*, vol. 20, no. 9, pp. 1304–1326, 2014. [Online]. Available: https://doi.org/10.3217/jucs-020-09-1304

[6] M. A. Revilla, S. Manzoor, and R. Liu, "Competitive learning in informatics: The UVa online judge experience," *Olympiads in Informatics*, vol. 2, pp. 131–148, 2008.

[7] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems (7th Edition)*. Pearson Harlow, 2017.

[8] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database system concepts (7th edition)*. McGraw-Hill, 2019.

[9] F. Sáenz-Pérez, "DES: A deductive database system," *Electron. Notes Theor. Comput. Sci.*, vol. 271, pp. 63–78, March 2011. [Online]. Available: http://dx.doi.org/10.1016/j.entcs.2011.02.011

[10] F. Sáenz-Pérez, "Applying constraint logic programming to SQL semantic analysis," *Theory and Practice of Logic Programming*, vol. 19, no. 5-6, p. 808–825, 2019. [Online]. Available: http://dx.doi.org/10.1017/S1471068419000206

[11] J. Hamari, *Gamification*. John Wiley & Sons, Ltd, 2019, pp. 1–3. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781405165518.wbeos1321

[12] J. L. Brita-Paja, C. Gregorio, L. Llana, C. Pareja, and A. Riesco, "Introducing MOOC-like methodologies in a face-to-face undergraduate course: a detailed case study," *Interactive Learning Environments*, vol. 27, no. 1, pp. 15–32, 2019. [Online]. Available: https://doi.org/10.1080/10494820.2018.1451345