LabDER - Relational Database Virtual Learning Environment

Adriano Lino[1[0000-0003-1403-2209]], Álvaro Rocha[1[0000-0002-0750-8187]], Luís Macedo[1[0000-0002-3144-0362]] and Amanda Sizo [1[0000-0002-9516-7991]]

[1] Department of Informatics Engineering
CISUC, Centre for Informatics and Systems
University of Coimbra
Coimbra, Portugal
{adlino, amrocha, macedo, sizo}@dei.uc.pt

Abstract. This paper describes a virtual learning environment for use in introductory database disciplines that aligns with the professor's teaching plan and aims to automatically evaluate students' responses to questions, which may be multiple-choice or discursive or may involve entity relationship diagrams (ERDs) or SQL. The main advantage of LabDER over previous automatic evaluation approaches for ERD and SQL is that it accepts multiple responses and encourages students to develop the best solution through semantic feedback based on compiler theories, software engineering metrics and supervised machine learning. This approach considers the distance of the student's response from the model response and provides semantic feedback via a blend of compiler and various other metrics, while predicting the student's grade using a machine learning algorithm. A case study was designed to confirm the approach and 15,158 students' responses were automatically evaluated. As a result, semantic feedback provided student self-learning through suggestions on the database concepts involved in each solution, which generated a considerable increase in student participation as well as an increase in their average grades. In future work, we will investigate how to include other database topics that can be automatically evaluated, such as query performance and relational algebra.

Keywords: Virtual Learning Environment, Automatic Evaluation, Entity Relationship Diagram, SQL, Compiler, Metrics, Supervised Machine Learning

1 Introduction

Entity relationship diagrams (ERDs) and Structured Query Language (SQL) represent a high proportion of the elective database components of baccalaureate courses and are included in the baccalaureate curriculum guidelines recommended by the Association for Computing Machinery (ACM) and the IEEE Computer Society [1]. These courses are supported by virtual learning environments (VLEs): systems that assist the teaching and learning process and provide resources for course assignments and access to material anywhere and at any time, using any device [16].

Microsoft [10] and Oracle [13] provide educational materials and tools on SQL and ERD topics, and some of these tools provide indicators for measuring the progress of user learning, often via multiple-choice activities. However, these tools have not been developed for the same purpose as VLEs and do not provide resources for the professor to implement the teaching plan and to manage and evaluate students and other academic requirements. Furthermore, they do not provide automatic evaluation of SQL and ERD.

Within the scope of this research, there are VLEs that enable the teaching of ERD and SQL with automatic evaluation. The literature review on this topic presents the following environments: SQL-Tutor [3], SQLator [14], MIST [4], OpenMark [15] and EER-Tutor [5]. However, these solutions for automatic evaluation of ERD and SQL are not complete, as they have evidence of limitations that can be overcome, i.e., an interface that limits the student answer at the same time as providing a model answer.

This article introduces 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. Unlike previous approaches, LabDER provides semantic feedback and a grade, based on compiler theory, software engineering metrics and supervised machine learning (ML). The automatic evaluation provides semantic feedback to the student on each submitted response, which in general is a measure of the distance between the student's response and the model response initially registered by the professor. This approach challenges and encourages students to refactor their solutions, to obtain, in addition to the answer that returns the correct result, semantic feedback with a grade that approximates that of the specialist. The main advantages of our approach are: 1) the student receives automatic feedback for each response submitted to the system and 2) the environment accepts equivalent computational solutions to the same problem and encourages the student to find more efficient solutions through semantic feedback.

This paper presents LabDER, describing its features, architecture and testing. The case study shows the system being used in a class providing more than 15,000 responses. Predictive models of SQL and ERD based on metrics and supervised ML have been described in previous studies [8],[9]. Based on this positioning, the previous perspectives will be analysed. Conclusions and references will close the work, as usual.

1.1 Literature Review

A literature review was conducted to assess the limitations of the computational approaches used to automatically evaluate ERDs in VLEs. Five electronic databases: ACM, IEEE, Springer, ScienceDirect and Web of Science, were used. The search terms used for eligible articles were "automatic evaluation", "automatic feedback", "automatic classification", "virtual learning", "diagram design", "entity relationship modelling", "ER modelling", "automated diagram evaluation" and their combinations.

The search criteria at the date of publication included all works published up to 7th March 2019, when the last search was made. Eligible documents were those in the English language where proposals were made for computational environments with automatic evaluation of ERDs. Textbooks, book chapters, theses, unpublished documents, non-English documents and exclusive works were excluded. Articles with no evidence

of system use were also excluded, as well as those that did not provide an analysis of results.

There are studies on automatic evaluation of ERDs without evidence of continuity, such as those on ERM-VLE [6] and DATsys [7], and some that are still in progress such as EER-Tutor [5] and OpenMark [15], originally known as KERMIT and ERD-Drawing. These studies provide references in the field of the achievement of the automation of ERD and are guides for the development of new technologies.

ERM-VLE is a text-based VLE for ER modelling, in which students design databases by browsing the virtual world. The virtual world consists of rooms, such as entity-creation rooms and relationship-creation rooms, and the students write text commands such as picking, dropping, naming, evaluating, creating and destroying, to manipulate objects. This tool proved to be very restrictive, and students reported discouragement at being forced to follow the path of the ideal solution.

The method used in DATsys [7] assesses the students' answer by running their solutions against a set of metric tools. When the marking subsystem runs the flow chart tool, the student's flow chart diagram is converted into BASIC code, which is then fed to the dynamic tool. Then, the dynamic tool runs the BASIC code and returns a grade and feedback to the students. However, this method does not address several equally valid model solutions with slightly different and mutually exclusive characteristics [7].

OpenMark [15] uses an approach for automatic evaluation of ERDs using NLP and heuristic techniques, in which a similarity measure is used to provide the marking. The results are close to those of the human evaluator, with 91% accuracy. A weak point is that the diagram used does not include all the ERD concepts, such as strong and weak entities, attributes and composition, and thus does not evaluate these concepts.

EER-Tutor [5] explores the effect of instructional feedback to simulate an evaluation of the EER and thus to provide feedback to detect and correct errors. However, for this purpose, the feedback in EER-Tutor is statically defined in the source code. Another disadvantage is the use of coloured visual highlights for various issues. This greatly simplifies the problem to be solved, and almost provides the system's expected response, highlighting the names of the tables, attributes and relationships.

In general, all these environments present some limitations that we can categorize as follows. **Restrictive responses** occur when students experience limitations in producing the diagrams because the environment accepts only a correct solution, and consequently students are directed to follow a standard response through the graphical interface. **Visual highlights** occur when systems visually highlight words in the description of an exam question, such as entities, attributes and relationships, then limit the student's reflection by providing the answers using the words expected by the system. **Ambiguity** occurs when students can guess the correct answer from among the alternatives offered and can get correct feedback for the wrong reason and **static feedback** occurs in an expert system that uses an approach limited by a set of pre-established rules, so that feedback is static for cases of error detection and correction. **Conceptual limitations** occur when the tools do not provide functionalities to evaluate database concepts, and consequently are limited to a restricted set of concepts. For example, they may evaluate only the entity name and relationship, and not cardinality, attributes or

domains. The literature review has shown few effective applications of automatic evaluation of ERD in the real world. Therefore, there are opportunities for improvement this field.

2 Method

LabDER[1] is the result of a doctoral research project. It aims to automatically evaluate students' responses, which may be multiple choice or discursive or may involve SQL or ERD. In addition, the system provides features similar to a VLE, enabling the professor to execute the teaching plan. The system is hosted on the cloud service of the PhD Programme in Information Science and Technology of the Faculty of Sciences and Technology at the University of Coimbra (FCTUC).

LabDER was developed following the standard software engineering process. Firstly, functional requirements, performance requirements and usability requirements were identified. The requirements list was built and validated with the support of key stakeholders for this project, i.e., the professor and the students. Performance testing was supported by OctoPerf[2] in all phases.

2.1 Features

The main functionalities of the system are automatic evaluations of SQL and ERD; however, LabDER has 89 functionalities, which together provide professors and students with an environment that helps in the database teaching and learning process.

The management features in the professor profile are create, read, update and delete data, and the following resource management modules are available (see **Fig. 1**):

Manage courses and classes: this module prepares LabDER for use by the professor and a class of students. The management features used to create a class are: professor, teaching institution, course, discipline and class.

Manage student: this module manages students' information. The features are: manage enrolment of students via CSV file or enable manual student enrolment through a standard form, manage student teams for performing activities and manage the student DB. Students have access to the system after agreeing to the legal terms of use and the privacy policy available on the LabDER website.

Manage evaluation and questions: this module manages the questions, answers and evaluations. There are four types of questions: discursive, multiple-choice, SQL and ERD crow's foot notation questions. Professors can share questions and answers with each other. The question group management feature is used to group the questions and gives the option to create activities with random questions. Normally, homework is created with the same questions for all students and should include all questions in the group of questions. Evaluations are usually random and should use a smaller number of group questions in order to be so. This avoids plagiarism.

---

1 LabDER is available at https://labder.dei.uc.pt/

2 Company for performance testing, see https://octoperf.com

Manage teaching material: the professor can edit the default digital book and add digital content, view reports to track student progress. Additionally, the professor can evaluate the student activities manually, replacing the automatic grade, or can give feedback in the format of virtual Post-it notes. In class, the professor can explain DB concepts using the SQL-Free and ERD-Free features. SQL-Free is an interface for connecting to a DB instance and running SQL code. ERD-Free is an interface for designing ERD crow's foot notation and saving it in a user DB or exporting it to SQL.

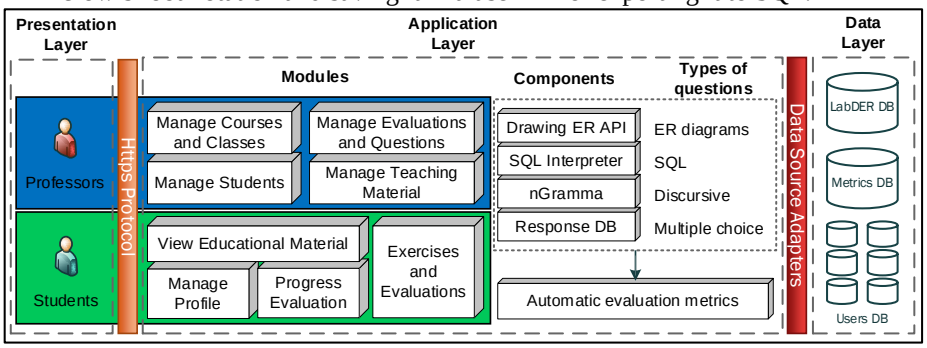

Fig. 1. General architecture of LabDER.

Furthermore, the student has the following functionalities.

View educational material: the student could access the interface for visualizing the digital book with six chapters, covering the topics of: introduction to DBMS, modelling, ERD, relational algebra, SQL and transactions and security. The student can export the materials in PDF format. Each student has a MySQL DB instance and can use SQL-Free to obtain access to the DBMS. The DBMS controls the security and the interface returns the SQL result. Students can use ERD-Free to design ERDs following the professor's instructions or can use it as needed. This interface has more than 59 features to improve the usability using shortcut keys and drag-and-drop.

Progress evaluation: the student can check their performance and the progress of each activity through reports.

Manage profile: the student can manage their profile by editing personal information and can reset the DB instance and monitor the access log using the tool.

Exercises and evaluations: this interface allow the student to visualize the activity and answer multiple-choice, discursive, SQL and ERD questions. All responses are evaluated automatically, and semantic feedback is provided instantly.

2.2 Architectural Overview

The environment uses the server-client architecture in three tiers (see **Fig. 1**). In the presentation layer, the professors and students access the application as clients via a browser. In the logic layer there are two application servers using load-balancing, via the Ubuntu and Apache HTTP service developed in PHP version 5.6.35. In the data layer, MySQL Server version 5.7.21 is installed on Ubuntu. The ERD design API[3] was

[3] API for ERD drawing, available at https://github.com/ondras/wwwsqldesigner

developed in JavaScript and is available on GitHub. In LabDER, this API is integrated for answering the ERD questions and designing freely using ERD-Free.

The professor uses the teaching plan to prepare the LabDER for the students through the modules, registering the course and class, questions and answers, summative, diagnostic and training activities and any other resources for the students to use.

Furthermore, the student visualizes the activities available through the exercise and evaluation module. The interface fits the question type, i.e., ERD, SQL, multiple-choice or discursive, and triggers the corresponding automatic evaluation component. The automatic evaluation component is triggered for each student's answer. In the case of SQL and ERD responses, the student's response is compared with the model response at three levels: compiler, metrics and the predictive model.

The compiler level uses a tokenizer, lexical analysis, syntactic and semantic analysis, a symbol table and error handler. This phase identifies each element of the student's response and analyses it, finally sending the feedback as a compiler. For example, the SQL semantic rule to detect the ORDER BY clause requirement is given by the model response. In the case of ERD, an example is the detection of the type of cardinality expected in the student's response, also given by the model response.

The metric-level evaluation computes measurements of the student's response and the model response. Most of the metrics used are DB metrics, such as the number of tables, columns or foreign keys, but we have also merged adaptations of software engineering metrics such as volume and McCabe [9]. At this level, one example of ERD feedback could be the lack of one strong entity and two columns of domain characters. In the case of SQL feedback, an example would be a solution having insufficient tables for the question.

In the prediction phase, a predictive function is triggered, using as input the metrics extracted from the student's response and the model response, and as output a grade that approaches the professor's evaluation. The predictive model was created in previous research [8], [9], using data collected from students and professors. It also describes DB metrics, with a demonstration of compiler phases and an error analysis.

2.3 Software Test

Appropriate software tests were defined for the goals to be achieved, using the environment. The first aim is to meet the functional requirements of the system and to find divergences between the execution environment and the system specification [11]. For this purpose, it must be ensured that all functionalities work as specified. A functionality test was made using a test-case template for 38 system functionality (see i.e. **Table 1**). The process followed a flow in cycles using five testers as recommended by Nielsen [12], and was completed when the feature obtained the status "passed".

The second aim was to ensure simultaneous access to LabDER in a classroom context, so that the system could be used by more than 500 students simultaneously in exams or class days. Thus, the performance testing aimed to keep users on the system, helping to find solutions to the problems of response time, load time, speed and low scalability. The system showed adequate performance, without bottlenecks or failures that could lead students and professors to leave the system.

Table 1. Functional test-case (TC) 65 - Answer ERD Questions.

Unique:	TC – 65
Test Case:	Answer ERD Questions
Location:	https://labder.dei.uc.pt/sql_avaliacao.php
Description:	Allows student to answer ERD questions in an assessment
Precondition:	The student is logged in and there is at least one assessment registered by the professor.
Procedure:	1: Select an assessment. 2: Select a question. 3: If the question is ERD, develop the ERD and click "Submit".
Expected Result:	The ERD is saved and semantic feedback is provided.
Result:	Passed

The performance test was undertaken using the OctoPerf software. The project was intended to support up to 1,000 concurrent users. This number was defined based on FCTUC's offer for the database discipline. There are about 450 students and 10 professors working together in the master's and baccalaureate courses in IT.

The main test scenario is the TC-Student ERD, which creates a complete simulation of the student answering an ERD question available in the assessment. **Table 2** shows the results of the test scenario in configurations with a 1vCPU and a 2vCPU. Load balancing (2vCPU) achieved response time of less than 4s using the 2vCPU for a thousand concurrent users and less than 1s for 1,000 concurrent users.

Table 2. Test case – student ERD performance comparison between 1vCPU and 2vCPU.

TC	Metric	Scenario – 1,000 VUs Configuration 1vCPU	Scenario – 1,000 VUs Configuration 2vCPU
Student ERD	Average Response Time	**28,175**	**1,822**
	Requests	**201,420**	**3,136,723**
	Latency	9,449	1,130
	Errors	942	50
	95% Percentile	**135,626**	**4,411**
	90% Percentile	129,932	3,992
	85% Percentile	52,770	3,142
	80% Percentile	32,729	2,844

The server received a total of 200,000 requests. After fixing the problems detected in the various stages of the performance test, we implemented load balancing to achieve the target response time. The best configuration received more than three million requests (see **Table 2**). In addition, in 95% of cases, with 1,000 concurrent users, the response time (previously 135s) had been reduced to 4s. The performance test was approved, since it achieved a 4s response time, less than the target response time for this test scenario, which was 10s. All test cases were approved, as shown in the technical report.

Table 3. The system usability scale questionnaire.

Please rate your level of agreement with each of the following statements:	**1. Strongly disagree**	**2**	**3**	**4**	**5. Strongly agree**
1. I think that I would like to use this system frequently.					
2. I found the system unnecessarily complex.					
3. I thought the system was easy to use.					
4. I think that I would need the support of a technical person to be able to use this system.					
5. I found the various functions in this system were well integrated.					
6. I thought there was too much inconsistency in this system.					
7. I would imagine that most people would learn to use this system very quickly.					
8. I found the system very cumbersome to use.					
9. I felt very confident using the system.					
10. I needed to learn a lot of things before I could get going with this system.					

The third objective was the usability test, to ensure that students and professors can perform their activities without experiencing difficulties in human-machine interaction.

The SUS questionnaire (see **Table 3**) was applied. This was originally written in English and later translated into Spanish and Portuguese.

LabDER was evaluated by 64 users, students and professors from Brazil, Portugal and Colombia, and scored an average of 72.97 on the SUS scale. Based on **Fig. 2**, it is observed that a system with a score of 70 points is acceptable and below 50 is unacceptable, according to the acceptability ranges scale. In adjective classification, a system evaluated with the adjectives "best imaginable", "excellent" or "good" is accepted. However, evaluation using the adjective "OK" may raise doubts about the acceptance and may lead to requirements for modifications to the system interfaces under analysis.

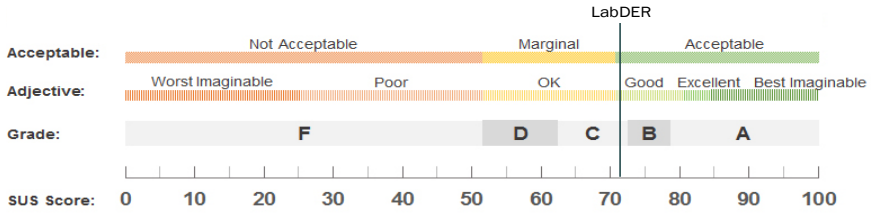

Fig. 2. LabDER usability ratings on the acceptability scale [2]. Scores below 50 are considered unacceptable.

Around 13 users (see **Fig. 3**) evaluated the system on the acceptability scale as "best imaginable" and 25 users agreed that the system was "excellent", while 21 users rated it as "good". Only one user rated the system as "worst imaginable". About 98% of the users classified the system as acceptable.

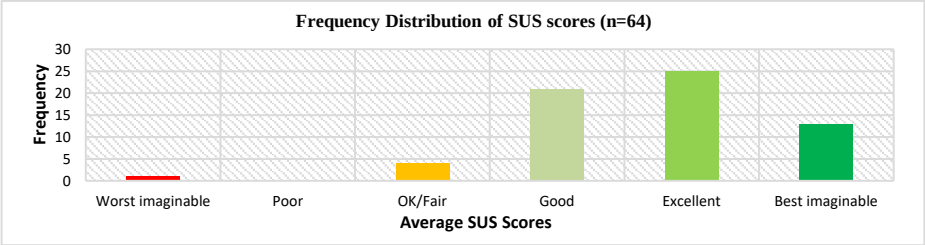

Fig. 3. Usability frequency distribution of LabDER for SUS usability adjective classification.

At the end of the software tests, the evidence was sufficient to show that LabDER is able to perform data collection accurately and can be used by students and professors.

3 Case Study

The case study was performed within the degree course in IT management at ISCAC – Coimbra Business School. The subjects consisted of 23 students. The course workload is estimated at 67.5 hours, for 15 weeks, with two face-to-face classes per week. The following section describes the preparation phase of the environment for using the system to collect the data and analyse the results.

3.1 Preparation

The preparation of the environment aims to release the system to be used by professors and students. The first preparation activity was to train the professor and prepare him to use the LabDER features, thus enabling him to execute his teaching plan.

The training had the professor's teaching plan as a requirement, as this helped to provide personalized training. The professor's teaching plan was converted into a set of executable activities in LabDER (see **Table 4**), and data were added to the environment, such as the course, discipline, questions, answers and activities. For reference, in this document SQL is labelled (S), ERD is labelled (E), discursive questions are labelled (D) and multiple-choice questions are labelled (M).

The students received the training on the first day. This was aimed at explaining the LabDER features and gaining initial access to the system. After completing the training, classroom support was provided on the first two days of class and on exam days.

Table 4. Course activity plan by period and content.

Act.	Quest.	Period dd/mm	LabDER Book Chapters	Content
A1	S 20 D 18 M 23	18/02 to 25/03	Module I	DB Concepts, DBMS, Relational Algebra, SQL overview.
			Module II	SQL Syntax Elements, DB Syntax Creation, Queries, Constraints, Sorting and Logical and Arithmetic Operators.
			Module III	DDL and DCL Statements, Create/Drop Objects, Insert/Update/Delete Statements, Constraint Syntax, Indexes, Views, Procedures, Triggers.
			Module IV	SQL GROUP BY Statement, Complex Queries with HAVING.
A2	S 17 D 06 M 28	27/03 to 15/04	Module V	Types of joins: cross, inner, outer, left, right. Subqueries: in a Column Expression, WHERE Clause as a Filter Criterion, FROM Clause as a Table, HAVING Clause as a Group Selector, Complex Queries Mixing All SQL Modules.
A3	E 25 M 35	17/04 to 10/06	Module VI	Design Concepts: The Relational Database Model, Entity Relationship (ER) Modelling, Advanced Data Modelling, Database Normalization.
E1	S 07 M 03	26/03	Modules I to, II, III, IV	Exam 1 – Evaluate All Introductory Database and SQL Concepts.
E2	S 07 M 03	16/04	Module V	Exam 2 - Evaluate the Advanced SQL Concepts.
E3	E 04 M 08	11/06	Module VI	Exam 3 - Evaluate All ERD Concepts.

With the help of the digital book and small real-world examples, students learned DBMS and SQL concepts for five weeks. The concepts taught are listed in **Table 4** (Modules I to IV). Meanwhile, they were given the first activity (A1) with 61 questions, (see **Table 4**). At the end of the period allowed for this activity, the students received an exam (E1) with 10 questions.

In the following three weeks, the concepts of joins and subqueries were taught to students, as described in detail in **Table 4** (Module V). Students received activity 2 (A2) for completion at the end of this three-week period. A2 mixed the concepts of joins and subqueries with aggregate functions and topics from previous modules and had 51 questions. At the end of A2, students received another exam (E2) with 10 questions.

In the next seven weeks, students were taught the database modelling concepts described in **Table 4** (Module VI). Students were given activity 3 (A3), with 60 questions, to be submitted at the end of this seven-week period. The students received the last exam (E3) after A3. This exam has 11 questions.

These six sets of data (A1, A2, A3, E1, E2 and E3) were collected (over a period of 15 weeks in total) and automatically assessed, as explained in the following subsections.

3.2 Results of Collection of Data

A total of 15,158 responses were collected. Of these, 2,539 were discursive, 2,435 were multiple-choice responses, 6,097 were SQL responses and 4,087 were ERD responses (see **Table 5**). The system log recorded a frequency of 1,915 logins and 26,918 iterations on the system. The iterations included digital book visualization, exercises, reports and any system window view.

Table 5. Number of responses submitted by question type, together with general system log.

Activity	Answers Submitted					User Log - Frequency			
(Total Questions)	Discursive	Multiple choice	SQL	ERD	Total	Login	Features Used	SQL-Free	ERD-Free
Activity 1 (61 q)	2,106	551	2,269	N/A	4,926	589	9,698	5,144	N/A
Activity 2 (51 q)	433	701	2,028	N/A	3,162	486	8,165	5,149	N/A
Activity 3 (60 q)	N/A	815	N/A	3,197	4,012	743	7,027	830	809
Exam 1 (10)	N/A	87	832	N/A	919	30	819	992	N/A
Exam 2 (10)	N/A	113	968	N/A	1,081	40	967	1,680	N/A
Exam 3 (11)	N/A	168	N/A	890	1,058	27	242	N/A	N/A
Total	**2,539**	**2,435**	**6,097**	**4,087**	**15,158**	**1,915**	**26,918**	**13,795**	**809**

In the training mode, more than 13,000 commands were submitted using SQL-Free, and around 800 diagrams using ERD-Free.

3.3 Results – ERD Evaluation

The results of the ERD evaluation are on a five-point scale, according to the percentage of hits (1 – Fail (0%-20%), 2 – Poor (21%-49%), 3 – Satisfactory (50%-69%), 4 – Good (70%-89%) and 5 – Excellent (90%-100%)).

Table 6 presents the frequency and percentages of submissions by DB concept, of the ERD design activities in A3 and E3. For example, activity 3 has 25 ERD questions (see **Table 4**), and the frequency of submissions for the relationship concept was 1,029, with 85% of these submissions being correct.

Table 6. Frequency and percentage of students successfully completing ERD.

	Database Concept																	
Activity	Table	Column	Column Numeric Domain	Column Character Domain	Column Date Domain	Primary Key	Unique Key	Relationship	Relationship Unary	Relationship Binary	Relationship Ternary	Relationship N-Ary	Relationship 1:N	Relationship M:N	Relationship Strong	Relationship Weak	PK Cardinality/ Participation	FK Cardinality/ Participation
A3	1,385	5,047	2,590	1,990	467	1,599	1	1,029	19	260	60	32	884	201	342	687	337	319
	90%	77%	82%	77%	96%	84%	100%	87%	98%	88%	99%	98%	85%	88%	88%	84%	76%	72%
E3	201	627	310	293	24	215	N/A	153	1	37	5	5	127	25	16	137	126	122
	78%	71%	73%	78%	92%	81%	N/A	86%	99%	87%	95%	99%	81%	82%	90%	84%	80%	77%

The professor interface (see **Fig. 4**) shows the student's answer to question 273, and the automatic feedback in red gives information about the lack of an M:N relationship

as well as the number of wrong cardinalities in each relationship. Additionally, the automatic feedback suggested grade 2, and the professor added manual feedback in yellow: “The relationship between benefit and plan is reversed. Fixing this will also solve the error of the missing M:N relation.”

The student interface is similar to that of the professor (see **Fig. 4**), but a response submit button is visible instead of the change grade button. Each student’s response is saved and evaluated automatically, and the results are presented in the semantic feedback panel and in the history panel. The student can view the professor’s evaluations through the history panel, submit a new response or add a note in Post-it format.

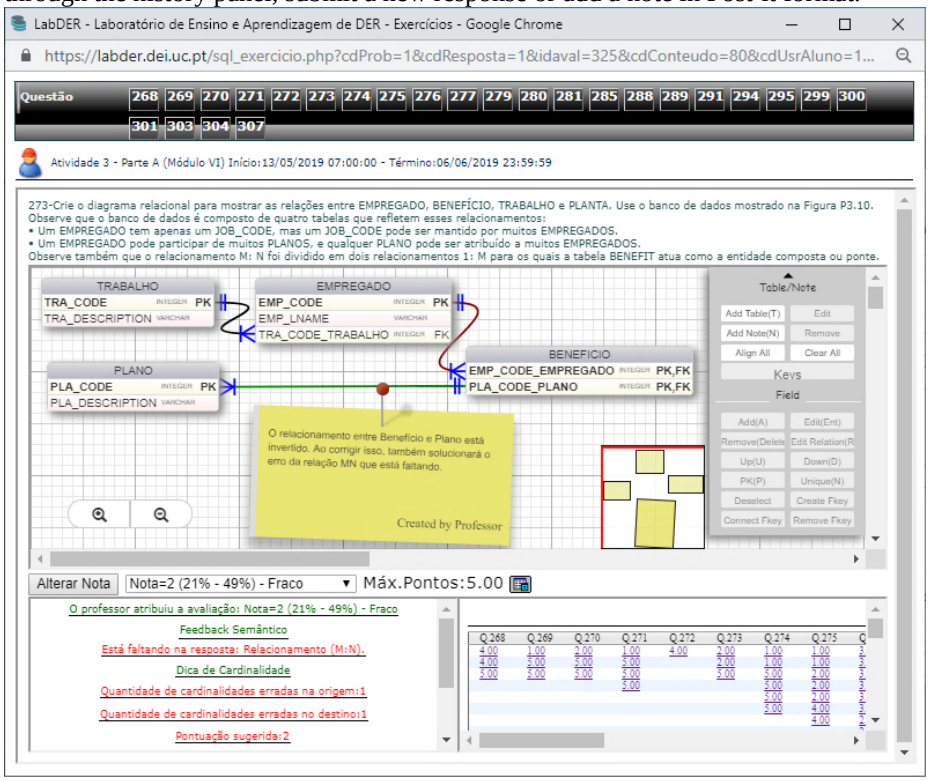

Fig. 4. Professor interface evaluating student’s ERD with manual feedback and grade.

The case study was carried out in Portugal, hence the language presented in the interface (see **Fig. 4**). However, the system was prepared for use in any language and has been used in Spanish, English and Portuguese.

3.4 Results – SQL Evaluation

The result to be analysed is the student’s answer to question 170. **Fig. 5** shows the professor’s interface when evaluating a student’s SQL command. In the feedback panel,

the result of the automatic evaluation can be seen, classifying the SQL as 92.43% correct and providing semantic feedback with SQL command improvement tips.

The student's solution to question 170, when compared to the system response, requires one more comparison expression to produce the same SQL result. Therefore, the automatic feedback hint notifies the student that the SQL has exceeded the number of comparisons operators, and indicates the exceeded operators in brackets; in this case the equals sign and the logical operator "and".

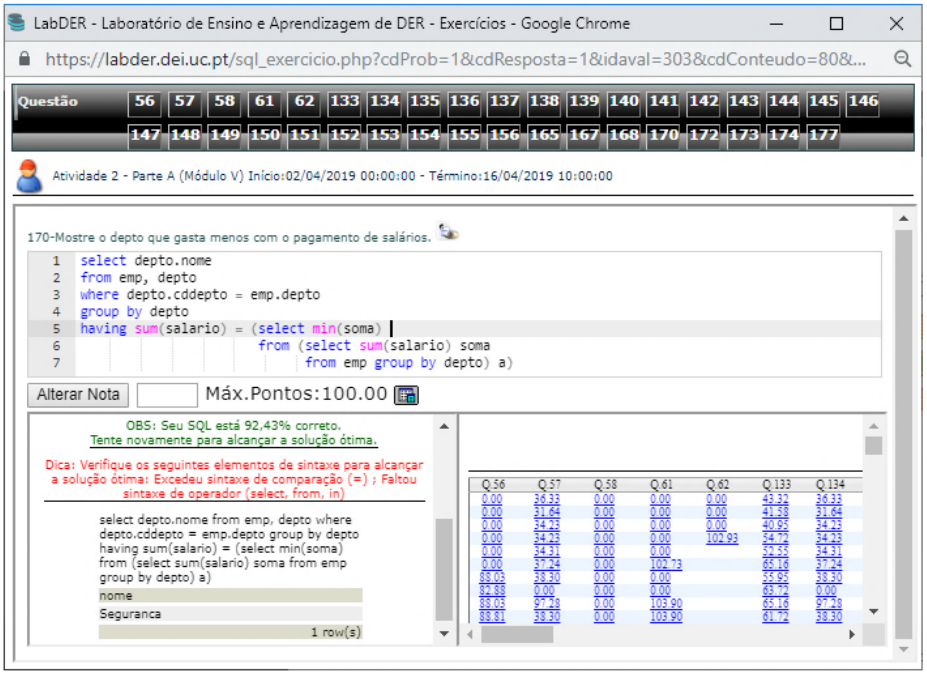

Fig. 5. Professor interface evaluating the SQL of the student's response.

The professor can change the automatic assessment, and the new grade is saved and is available in the student response history panel. This interface is also adapted for the discursive and multiple-choice questions. However, these are not presented via the automatic evaluation of SQL and ERD.

4 Discussion and Future Directions

The LabDER virtual relational database teaching environment was proposed to overcome the limitations of the ERD design found in previous work, such as restrictive responses, visual highlights, ambiguity, static feedback and conceptual limitations. LabDER overcomes the previous limitations using web technologies and can be aligned with the professor's teaching plan while providing an innovative approach to automatic assessment that accepts equivalent computational responses to an issue.

The approach improves on the state of the art of automatic evaluation of ERD and SQL because it accepts multiple responses and provides semantic feedback that encourages students to develop more efficient computational solutions. Semantic feedback mixes compilers, software engineering metrics and supervised machine learning to give grades and conceptual DB tips on each student's response. The integration of these techniques in the form of feedback demonstrates the system's accuracy to students and professors, since the notes and tips are consistent with the students' correct responses and their mistakes. Feedback proved to be useful to the students, because they used it to learn and test their skills in order to develop better solutions. For the professor, this approach allowed better orientation, allowing the students' most frequent conceptual doubts to be quickly identified. LabDER offers several ways to use this approach, in diagnostic, formative and summative evaluations, as well as offering a series of functionalities for mixing pedagogical strategies, and the overall result was very positive, with a correctness rate of above 95% for several DB concepts.

In addition to the case study, the project has been running in 13 other bachelor's and master's degree courses in computing in Brazil, Colombia, India and Portugal. Google Analytics has tracked access from 450 LabDER students in more than 16 cities in Portugal, 20 cities in Brazil, nine cities in Colombia and four cities in India, as well as tracking visitors from America, Europe and Asia. The software test showed that LabDER has the scalability and robustness to support this volume of access and provided evidence of self-learning and participation by users outside the classroom.

The LabDER approach is the opposite of traditional tools. This approach is used for decision-making: for example, the professor decides when to move on to the next learning topics. In previous approaches, the concern is to avoid user evasion and to ensure that the system continues to be used. This approach showed that students seek self-learning and try to solve all questions several times, until they can achieve the model response. In addition, the professor can make decisions, receive feedback from the students and decide which DB concepts should be taught again or whether the students can move faster.

In future work, we intend to investigate how to include new automatic evaluation modules, such as modules on relational algebra or database performance. Relational algebra can be converted into SQL, and then the framework can be used to develop the new module faster. Furthermore, the framework could be migrated to the teaching of UML class diagrams, because these are close to the ERD notation. Finally, we intend to include other ERD notations, as some professors want to teach modelling concepts using alternative notations.

Acknowledgement

We would like to thank OctoPerf for providing the software license and for support in performance testing. We also thank the IT staff of the University of Coimbra and ISCAC for their support in implementing the project and gratefully acknowledge the support of CNPq, the National Council for Scientific and Technological Development in Brazil.

References

1. ACM Computing Curricula Task Force ed: Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. ACM, Inc (2013). https://doi.org/10.1145/2534860.
2. Bangor, A. et al.: Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. J. Usability Stud. 4, 3, 114–123 (2009).
3. Chen, X. et al.: Does Adaptive Provision of Learning Activities Improve Learning in SQL-Tutor? Presented at the (2017). https://doi.org/10.1007/978-3-319-61425-0_44.
4. Dimitrieski, V. et al.: Concepts and evaluation of the extended entity-relationship approach to database design in a multi-paradigm information system modeling tool. Comput. Lang. Syst. Struct. 44, 299–318 (2015). https://doi.org/10.1016/j.cl.2015.08.011.
5. Elmadani, M. et al.: Investigating student interactions with tutorial dialogues in EER-Tutor. Res. Pract. Technol. Enhanc. Learn. 10, 1, 16 (2015). https://doi.org/10.1186/s41039-015-0013-1.
6. Hall, L. et al.: A virtual learning environment for entity relationship modelling. In: Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education - SIGCSE '98. pp. 345–349 ACM Press, New York, New York, USA (1998). https://doi.org/10.1145/273133.274327.
7. Higgins, C. et al.: Diagram-based CBA using DATsys and CourseMaster. In: International Conference on Computers in Education, 2002. Proceedings. pp. 167–172 IEEE Comput. Soc (2002). https://doi.org/10.1109/CIE.2002.1185893.
8. Lino, A. et al.: A Proposal for Automatic Evaluation by Symbolic Regression in Virtual Learning Environments. In: New Advances in Information Systems and Technologies. Advances in Intelligent Systems and Computing. pp. 855–865 Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-31232-3_81.
9. Lino, A. et al.: Application of clustering-based decision tree approach in SQL query error database. Futur. Gener. Comput. Syst. 93, (2019). https://doi.org/10.1016/j.future.2018.10.038.
10. Microsoft: Microsoft Online IT Training | Microsoft Learning, https://www.microsoft.com/en-us/learning/training.aspx.
11. Myers, G.J. et al.: The art of software testing. John Wiley & Sons (2012).
12. Nielsen, J., Landauer, T.K.: A mathematical model of the finding of usability problems. In: Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93. pp. 206–213 ACM Press, New York, USA (1993). https://doi.org/10.1145/169059.169166.
13. Oracle: IT Training: Certification for Oracle Technology, https://education.oracle.com/en/.
14. Sadiq, S. et al.: SQLator: An Online SQL Learning Workbench. In: ITiCSE '04 Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education. p. 5 ACM Press, New York, New York, USA (2004). https://doi.org/10.1145/1007996.1008055.
15. Thomas, P.: Online Automatic Marking of Diagrams. Syst. Pract. Action Res. 26, 4, 349–359 (2013). https://doi.org/10.1007/s11213-012-9273-5.
16. Vassilakaki, E.: New Trends in Higher Education: Can Information Professionals Rise to the Challenge? End Wisdom? 119–122 (2017). https://doi.org/10.1016/B978-0-08-100142-4.00012-9.