POLITECNICO DI TORINO
Repository ISTITUZIONALE

Improving the effectiveness of SQL learning practice: a data-driven approach

Original
Improving the effectiveness of SQL learning practice: a data-driven approach / Cagliero, L., DE RUSSIS, L., Farinetti, L., Montanaro, T.. - STAMPA. - 1:(2018), pp. 980-989. (42nd IEEE Computer Society International Conference on Computers, Software & Applications (COMPSAC 2018), Symposium on Computer Education & Learning Technologies (CELT) Tokyo (Japan) July 23-27, 2018) [10.1109/COMPSAC.2018.00174].

Availability:
This version is available at: 11583/2709321 since: 2018-06-26T14:27:47Z

Publisher:
IEEE

Published
DOI:10.1109/COMPSAC.2018.00174

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

04 November 2026

Improving the effectiveness of SQL learning practice: a data-driven approach

Luca Cagliero, Luigi De Russis, Laura Farinetti
Politecnico di Torino
Torino, Italy
{luca.cagliero, luigi.derussis@laura.farinetti}@polito.it

Teodoro Montanaro
Istituto Superiore Mario Boella
Torino, Italy
montanaro@ismb.it

Abstract—Most engineering courses include fundamental practice activities to be performed by students in computer labs. During lab sessions, students work on solving exercises with the help of teaching assistants, who often have a hard time for guaranteeing a timely, optimized, and "democratic" support to everybody.

This paper presents a learning environment to improve the experience of the lab sessions participants, both the students and the teaching assistants. In particular, the environment was designed, implemented, and experimented in the context of a database course. The application designed to support the learning environment stores all the events occurring during a SQL practice lab, i.e., task progression, query submissions, error feedback, assistance requests and interventions, and it provides information useful both for use on-the-fly and for later analysis. Thanks to the analysis of these data, the application dynamically provides teaching assistants with a graphical interface highlighting where assistance is most needed, by considering different factors such as the progression rate, the percentage of correct solutions, and the difficulties in solving the current exercise. Furthermore, the stored data allow teachers later on to analyze and to interpret the behavior of the students during the lab, and to have insights on their main mistakes and misconceptions.

After describing the environment, the interfaces, and the approaches used to identify the students' teams that need timely assistance, the paper presents the results of different analyses performed using the collected data, to help the teacher better understand students' educational needs.

I. INTRODUCTION

Engineering curricula strongly rely on lab activities, where student can test, practice, and improve their learning achievements. Computer labs, specifically, require students to solve programming tasks working alone or in a team, with the support of teaching assistants in a controlled environment.

Large universities, with a high number of enrolled students and a low teacher per student rate, often have to face a big challenge to ensure everybody a valuable computer lab experience. In our university, B.S. computer science courses have an average number of 240 enrolled students, and each lab session has about 80 participants. Computer labs have 40-60 computers, and in general only 2 or 3 teaching assistants are available to help students during their tasks. In these conditions, students' experience is far from optimal, and this is reflected by the progressive decrease of lab participation during the course semester: the number of students that take part to the last computer lab session is generally less than half of the number of students that participated in the first one.

In such a context, students who have trouble in solving their tasks are often stuck waiting to get the teacher assistants' attention. On the other hand, teachers are often overwhelmed by many assistance requests and sometimes cannot guarantee adequate timely help to everybody. Besides, students' attitude can play an important role [1]: shy students tend not to ask for help even if they are in trouble, while others keep calling for assistance without even trying to solve problems autonomously.

The research presented in the paper aims at providing teaching assistants with an environment to support their task during computer science lab sessions. Specifically, we designed, implemented, and experimented a tool for SQL practices in lab that:

- anonymously records students' activities on all lab workstations;
- records students' requests for assistance, and teachers' assistance events;
- interprets activities and assistance requests in order to understand the performance level of the students, and specifically to extract the "difficulty level" they experience at any time;
- provides assistants with a dynamic comprehensive overview of the difficulty levels of every workstation with a simple "color status" interface, by suggesting where assistance is most needed;
- provides assistants with a detailed graphical view for each of the workstations, to highlight the reason why assistance is needed (e.g., many wrong tentative solutions, inactivity periods, …)

The idea behind this research is to provide teaching assistants with an optimized, prioritized, and "democratic" way for giving assistance to students, through an informed environment where they can easily spot who is more in trouble according to objective parameters (and not simply by raised hands). This research, furthermore, has a second objective: the recorded session data can be analyzed afterwards to extract useful information about students' behavior during SQL lab sessions, and about the most common mistakes and misunderstandings. Such an analysis provides an important feedback that teachers can exploit in future classroom lectures.

The remainder of the paper is organized as follows. Section II provides a discussion on the related works. Section III describes the designed environment, with a focus on the SQL lab experiment and on the teaching assistant interface, while

Section IV reports the analysis of the collected data, focusing both on students' experience and behavior during the lab, and on educational aspects related to the database course topics. Finally, Sections V draws the conclusions and highlights challenges and future perspectives of this work.

II. Related work

Supporting students in learning the SQL language is an established learning problem. For example, Mitrovic [2] proposed a tutoring system for guided SQL learning. The architecture of the proposed system is focused on a constraint-based model, which supports students in learning from errors by providing them with targeted hints. The SQLator Web-based interactive tool for learning SQL has been proposed in [3]. SQLator integrates an advanced function, based on a heuristic algorithm, to allow users to evaluate the correctness of their query formulation. In [4] the authors investigated the use of iconic metaphors in a higher level query language similar to SQL. The goal was to help users to learn and comprehend the relational data model.

More recently, the focus has moved to the proposal of new learning analytics environments able to collect significant user-generated data [5]. Learner-generated data can be used by professionals for the discovery of significant information. For example, the knowledge extracted from data acquired during SQL learning practices can help teachers and students to manage and monitor the teaching-learning process [6]. The learning environment proposed in this paper focuses on improving the interaction between students and teaching assistants during assisted SQL practices in lab. Unlike [5] and [6], the proposed environment relies on online monitoring and evaluation of the progress status of the SQL-based practice. It fosters effective student-teacher interactions based on the generation of specific alarms and on the graphical reporting of sequences of key activities performed by the students.

Parallel efforts have been devoted to quantitatively evaluating the difficulties in learning the SQL language [7] [8] and to categorizing the most common semantic mistakes in writing SQL queries [9] [10]. Unlike the above-mentioned work, the focus of this research is neither on categorizing common mistakes nor on evaluating the complexity of SQL queries, but to support both students and teachers during SQL practice in lab. To analyze students' interaction with the proposed learning environment, we categorized SQL query errors based on the categorization given in [9].

III. Educational context and experiment setting

The educational context for which the environment was designed, implemented, and experimented is the course on databases of the second year of the B.S. curriculum in Engineering and Management. This course has about 650 enrolled students, one of its main focuses is on the SQL language, and the lab activities involve SQL query exercises.

The environment was then designed to support students in practicing SQL queries with the Oracle DBMS [11]. It was tested in this specific course, but a database course is present in all the curricula in Engineering, and therefore is it highly reusable, with a potential number of 4,000 students every year.

During the 90-minutes weekly lab sessions, students are divided in 6 groups. The computer lab has 43 workstations with one or two working students (sometimes even three), and three teaching assistants provide students' support.

The lab session involved in this experiment was the second one, so that most students were already familiar with the type of practice to be completed. The task was to perform 13 SQL query exercises of medium complexity. Past years' experience showed that practically nobody was able to complete all the exercises, and most of the teams finish 6 or 7 of them. In these cases, students are strongly encouraged to finish the task later in autonomy.

A. Learning environment rationale and description

The developed learning environment, whose architecture will be detailed in subsection II.B, has been designed to support and record the students' activities while solving the 13 proposed SQL exercises. Students' interface (see Figure 1) proposes one exercise at a time, with the problem statement, the associated database schema and the table representing the correct results. The students enter their tentative query and the Oracle DBMS executes it, providing the feedback that is shown to the learners. Besides the Oracle message (useful for understanding query errors), when the query is syntactically correct the environment compares the expected result with the executed result.

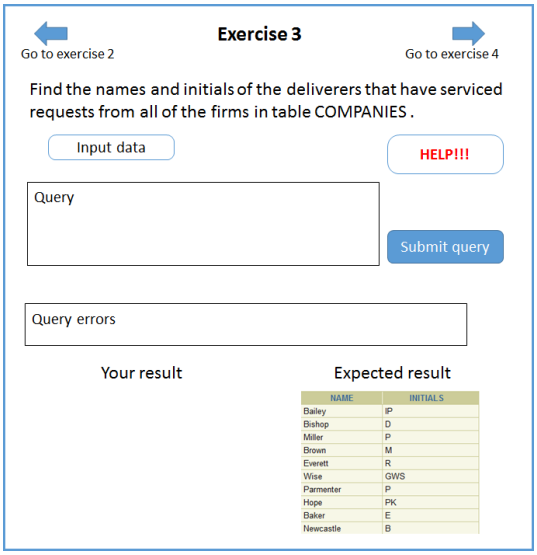


Fig. 1. Students' interface.

In addition, the interface allows teams to send a help request to the teaching assistants. The request will be displayed as an orange bell close to the requesting workstation icon (see subsection II.C) on the assistant interface.

The environment stores a number of event-related data in its database during each lab session, and precisely for every workstation:

- the text of every query submission, with the relative timestamp and the associated exercise identifier;
- the associated feedback message (i.e., correct query, Oracle message in case of syntactical error, application error in case of result error);
- the progressive number of the current exercise attempt (e.g., attempt 3 for exercise number 5);
- the requests for assistance, with the relative timestamp;

- the timestamps relative to the start and the end of every assistance intervention.

Thanks to these data, it is possible to extract useful information for interpreting the status of the students' teams, and specifically for understanding how confident and performant they are, and whether they need extra support.

Each workstation is dynamically assigned a color status (green, yellow, or red) that reflects the positive/negative performance according to a number of factors computed from the collected data:

1) The current exercise number, compared to the modal value of the current exercise numbers of all the workstations; this factor takes into account the relative progress of the students, and identifies the teams that are far slower than the average, i.e., that are potentially in trouble.
2) The percentage of correctly solved exercises: the students' interface, in fact, allows students to skip an exercise and to proceed to the next one; this factor takes into account the unsuccessful attempts and it is related to the perceived complexity of the exercises. A low percentage of correctly solved exercises definitely identifies teams that need extra support.
3) The time devoted to the current exercise; this factor is useful for timely identifying teams that have problems in solving a specific exercise.
4) The number of unsuccessful query submissions for the current exercise; like the previous one, this factor identifies problems related to a specific exercise, independently of the performance in the previous ones.

The color status depends on the values of these factors, combined in three components that are independently evaluated as "Good (G)", "Average (A)" and "Bad (B)" and then weighted to produce a single value mapped to one of the three colors. The three components are:

A. Current exercise status. It depends on factors 3) and 4) and specifically is:
   - G_A (Good) if ((t < 5 min) AND (NA < 4) OR (NA < 1)), i.e., the current exercise status is considered good if the team has been working on it for less than 5 minutes and the unsuccessful query submissions (NA=number of attempts) are less than 4, or if there are not query submissions yet (inactivity);
   - A_A (Average) if ((5 min <= t <= 8 min) AND (NA >= 1)), i.e., the current exercise status is considered average if the team has been working on it for more than 5 minutes but less than 8, and the unsuccessful query submissions are at least 1 (to ignore the case of inactivity);
   - B_A (Bad) if (t > 8 min) AND (NA >= 1), i.e. the current exercise status is considered average if the team has been working on it more than 8 minutes, and the unsuccessful query submissions are at least 1;

   Timing has been chosen by considering the average time required for solving the proposed exercises, thanks to the experience of previous years' lab sessions. The results discussed in Section IV will be used for an optimized timing choice in future experiments, according to actual students' average behavior for each exercise.
B. Past exercises status. It depends on factor 2) and specifically is:
   - G_B (Good) if (SE > 85%), where SE = percentage of correctly solved exercises, i.e., the past exercise status is considered good if the team has correctly solved more than 85% of the proposed exercises;
   - A_B (Average) if (70% <= SE <= 85%), i.e., the past exercise status is considered average if the team has correctly solved between 70% and 85% of the proposed exercises;
   - B_B (Bad) if (SE < 70%), i.e., the past exercise status is considered average if the team has correctly solved less than 70% of the proposed exercises.
C. Temporal progression status. It depends on factor 1) and specifically is:
   - G_C (Good) if (NCE > modal value +1), where NCE = number of current exercise, i.e., the temporal progression status is considered good if the team is progressing quicker than most of the other teams;
   - A_C (Average) if (NCE = modal value ± 1), i.e., the temporal progression status is considered average if the team is progressing like most of the other teams;
   - B_C (Bad) if (NCE < modal value - 1), i.e., the temporal progression status is considered bad if the team is progressing slower than most of the other teams.

"Good", "average" and "bad" statuses are assigned a coefficient of 0.8, 0.5 and 0.2, respectively. For the first two exercises only coefficient A (C_A) is considered, while for the other exercises C_A weights for 50%, coefficient B (C_B) for 20% and coefficient C (C_C) for 30%.

The final comprehensive coefficient C_T is then calculated as $C_T = 0.5 * C_A + 0.2 * C_B + 0.3 * C_C$, and the color status is Green if $C_T >= 0.55$, Yellow if $0.45 < C_T < 0.55$, and Red if $C_T <= 0.45$.

Values and thresholds have been chosen through simulation, and validated during the experiment.

B. The environment architecture

The developed application is structured in two parts, one running on each computer in the computer lab (*client*) and the other running on a dedicated central *server*.

The client is the only interface towards the students. It is composed of a Web application linked to a local database with the SQL exercises to be completed by the students and their progression (see Figure 1). The client contains and presents to the students the exercises to be performed during a session and allow them to execute the SQL queries as well to see the results (or the errors) of their operations. The client interacts with the central server for saving data (i.e., results, timestamps, errors, etc.) and for asking for the help of the teaching assistants. For the lab experiment described in the following subsection III.D, the client was developed as a PHP application

connected to the Oracle DBMS that was already running on each computer.

The central server, instead, provides a set of REST APIs to collect the data coming from each client, and a graphical interface for the teacher assistants. The server, therefore, can handle multiple requests and it is agnostic with respect to the specific exercise that are shown in the client interface. The user interface provided for the teaching assistants is illustrated in the following subsection III.C. In the lab experiment described in the paper, the server is implemented as a PHP application connected to a MySQL server. Finally, the data analysis reported and discussed in the following was extracted from the server.

C. *The teaching assistant interface*

The server provides a graphical interface for the teaching assistants, depicted in Figures 2 and 3. The goal of this interface is twofold: from the one hand, it shows how the laboratory is going, according to the metrics previously described (subsection III.A); on the other hand, it allows the teaching assistants to immediately catch any problems in the lab and act consequently. This second part is particularly important in a lab crowded by students, in which multiple requests may happen simultaneously and frequently. The user interface, developed as a Web application, is responsive so that the teaching assistants can check the progress of the lab from a computer, a tablet or a smartphone, without being stuck in a specific position in the lab.

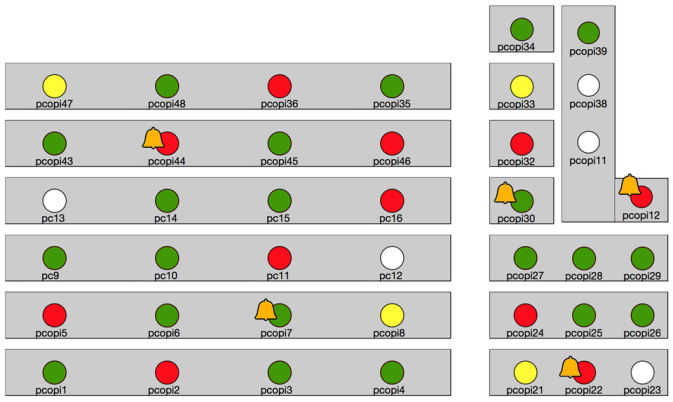


Fig. 2. Teaching assistant interface – map of the laboratory with the status of each workstation.

Figure 2 shows the main page of the teaching assistants' interface. It represents the map of the laboratory with the color status associated to each computer available in the lab. When a student asks for help through the client interface, an orange bell appears on the map, as for the *pcopi44* computer in figure. In this way, a teaching assistant, with a quick glance at the overall status of the lab, can decide to autonomously intervene in case of a "red" status, as she can also immediately see any request for assistance coming from the students. In the figure, the white color indicates a non-used workstation.

Figure 3, instead, depicts the detail page associated to each computer in the map, *pc10* in this case. The page shows four information:

1) Whether the students has requested any help, as indicated by the bell on the top of the page; if the bell is orange, then the student requested help for the current exercise.

2) The progression of the exercises. Completed and correctly executed exercises are in green, not correctly finished exercises are in red, while exercises still to do are indicated in white. In the case depicted in Figure 3, the team already completed 8 exercises, two of which are not correct. The progression bar also highlights the current modal value of the lab, shown by a black vertical line (located between exercise 5 and 6, in figure).

3) The number of times the student received any help (zero, in this case).

4) The possibility, for the teaching assistant, to intervene on the current exercise (the "Start" button on the bottom of the picture). To have precise data about assistance interventions, in fact, the teaching assistant could use this modality.

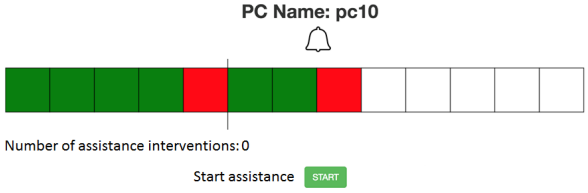


Fig. 3. Teaching assistant interface – detail of a single workstation.

D. *The lab experiment and the feedback*

The experiment, which took place in early November 2017, consisted of six 90-minutes lab sessions (in three different days of the same week) and involved five teaching assistants (three simultaneously in each session). Different students participated to each session, to accommodate in the 43-workstations lab all the interested students. The total number of students involved in the experiment was 370, because participation to labs is optional (even tough highly encouraged). The total number of participating team of students, i.e., the total number of used workstation, was 205. Table 1 resumes students' participation in each of the lab sessions, and the number of teams.

TABLE I. SUMMARY OF EXPERIMENT PARTICIPATION

Lab session	Students' teams	Students
1	35	50
2	43	95
3	25	39
4	37	79
5	26	35
6	39	72

The experiment was performed without any major issue and all the participants (students and teaching assistants) were able to use the tool and take advantage of the teaching environment.

After the experiment, we interviewed the teaching assistants, and the feedback was largely positive. After an initial adjustment to the new learning environment, they noticed an improvement of the service to the students, being less distracted by raised hands, and aware, thanks to the

interface, of the actual and objective difficulties of the working teams. They judged the new learning environment more dynamic and efficient, and also more "democratic": in fact, the interface updates the total number of requests from all workstations, allowing the assistants to give priority to students who have not been assisted yet.

They also reported that more than one third of the interventions were not initiated by a specific help request, but were consequence of the color status (red or yellow) of the workstations in the interface, thus providing a punctual and timely support to students that are potentially in trouble. This is an advantage of this environment with respect to traditional ones, were very little support is given to students that make no specific requests, mainly because it is difficult to understand the actual situation of a team by simply passing behind their back and looking at what they are doing.

The color status of the workstation helped the assistants also to prioritize help requests, processing first the ones that have an associated red color.

Besides, the teaching assistants reported a generally curious and positive attitude of the students towards the new learning environment, also confirmed by the sample interviews we did with the students at the end of each lab session.

On the other hand, the assistants complained about some delays in switching on and off the bell signal (help request), and they did not like to have to press the "start intervention" and "stop intervention" buttons on their interface every time they gave support (to record these events), especially because it was very easy to forget. These aspects will be considered in the next version of the learning environment.

Finally, they suggested that the application should display the list of workstations in order of assistance priority, considering not only the color status but also the timestamp of the help request (which presently is ignored). In case of many quasi-simultaneous help requests, in fact, they could not decide quickly were to go first.

About the students' feedback, from their point of view there was no major difference with respect to the traditional environment. The interface for query submission was slightly different than the Oracle's one and somebody complained about it because it was not familiar. The query text area was too small, but it will be enlarged in the next version of the interface. On the other hand, the interface included the expected table output to compare quickly with their solution output, and this feature was considered very helpful.

The approach for handling the "help requests" was highly appreciated, as soon as they realized that they did not need to spend much time attracting assistant's attention, but they could send a request and wait while keeping working on their tasks.

Both students and teaching assistants, finally, think that this environment can be effectively applied to other database courses, at least for all the labs involving SQL query exercises.

IV. ANALYSIS OF STUDENTS' BEHAVIOR

In this section, we analyze the data generated from the interaction between the teams of students and the learning environment. The aim of the analysis is to study the behavior of the students in solving exercises with different complexity. This information can be exploited for multiple purposes. On the one hand, teachers can understand the main students' difficulties and misconceptions. On the other hand, they can revise and improve the content of the practices based on the data collected during the past experiences. Specifically, the research questions we are addressing are:

1. *Are the duration and complexity of the practice appropriate for the current level of knowledge of the students?*
2. *Are the exercises proposed in the most suitable order?*
3. *In what situations are students most frequently asking for assistance?*
4. *What is the impact of assistance interventions on students' activities?*

To analyze the interaction between the students and the environment, we collected data about the time at which teams accessed each exercise (eventually multiple times), the submissions of solutions for each exercise, the correctness of the submitted solutions, and the most common mistakes. Hereafter, we will separately analyze data related to the following aspects:

a. The number of accesses to the interface for submitting solutions to the proposed exercises.
b. The number of attempts made by the teams of students to solve each exercise and the correctness of the submitted solutions.
c. The SQL queries submitted by the teams.
d. The time spent in solving each exercise.
e. The number of requests for assistance.
f. The temporal progression of the students' activities.

A. *Number of accesses to each exercise*

We analyzed the distribution of teams' accesses to the proposed exercises. In the guided practice, the exercises were proposed in order of increasing level of complexity. Although the students were recommended to solve the exercises in sequential order, the interface allowed them to move backward and forward, eventually reconsidering some of the previously accessed exercises.

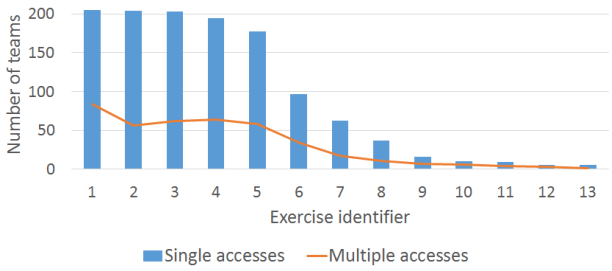


Fig. 4. Teams' accesses to SQL exercises.

The bars in Figure 4 show the number of teams who made a single access to each exercise. The overlapped line indicates, for each exercise, the number of students who performed multiple accesses.

The total number of teams interacting with the learning environment was 205. The majority of the students tried to solve the exercises from 1 to 5; approximately half of them solved exercises 6 and 7 as well, while only a few of them accessed exercises from 8 to 13. The number of students who

accessed the exercises more than once is maximal for exercises from 1 to 4, while it drops for exercises from 7 on because most students spent all of their time in solving the previous ones.

The results confirm that an average-level student would need more than the available 90 minutes to complete the practice. Most students solved all the simplest exercises plus a selection of two or three more complex ones (based on their preferences). Since some exercises were addressed only by few students, they were encouraged to solve the remaining exercises as home practice.

B. Number of query submissions

After accessing each exercise, teams submit their solution. The bars in Figure 5 show, for each exercise, the number of teams that submitted at least one solution (either correct or incorrect), while the overlapped line highlights the portion of teams who have submitted a correct solution through the environment. The success rate (i.e., the percentage of teams that submitted a correct solution) is above 70% for exercises 1, 3, and 6, while it is below 50% for all the other exercises. Notice that since the success rate is influenced by the number of attempts made, exercises with few attempts (e.g., exercises from 8 to 13) have relatively high rates even if their level of complexity is fairly high.

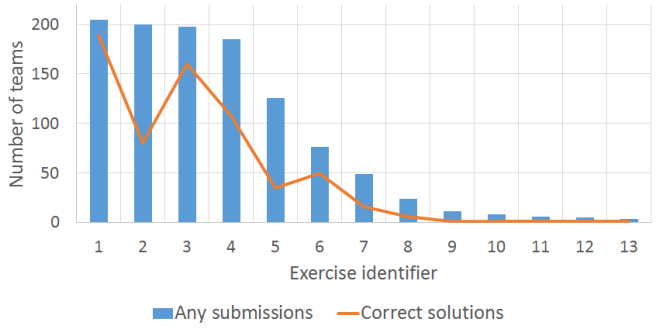


Fig. 5. Teams' submitted solutions (total vs. correct).

We analyzed also the statistics about the number of attempts separately for each team (see the chart in Figure 6). Specifically, for each exercise we first counted the total number of attempts, the total number of incorrect answers, as well as the fraction of incorrect answers associated with syntactical errors (Oracle messages). Then, we averaged all the counts over the students involved in the lab sessions.

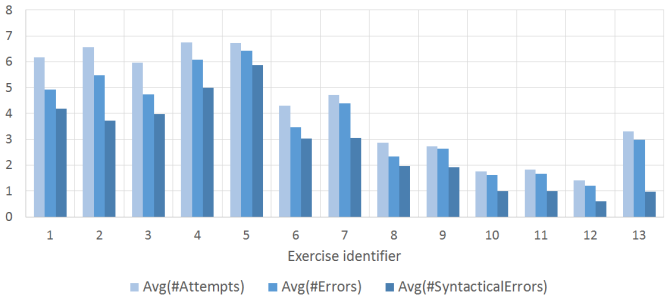


Fig. 6. Average number of attempts/errors per teams.

The results show that the involved students have made, on average, from 4 to 7 attempts per exercise for the first 7 exercises, and only 2 or 3 attempts for the subsequent exercises (mainly due to the lack of time). The average number of errors was approximately equal to the average number of attempts for exercises from 8 on, because most of the students did not succeed by the end of the practice. Conversely, the average number of syntactical errors is fairly high (between 4 and 6) in the first 5 exercises while it becomes less significant (between 1 and 3) from exercise 6 on. This result confirms that students got more experienced while solving exercises.

C. Students' SQL query analysis

We recorded all the queries (6,571 in total) submitted by the students during the lab sessions, for later analysis. The situation here is different from other studies such as [9] and [10], where the analysis regarded queries executed during exam sessions, and only final results (at the best of students' effort) are recorded. In our case every attempt is recorded, also the initial ones, when the student simply tries to understand how to proceed to the solution with a trial and error approach. This is demonstrated by a percentage of incorrect queries equal to 85% and a mean time between query submissions that frequently is less than one minute.

Even though the main contribution of the collected data is in the remainder of the analysis where the focus is on students' step-to-step progress toward the solution, the teacher can anyway extract meaningful knowledge about the major misconceptions and misunderstanding hidden in query errors.

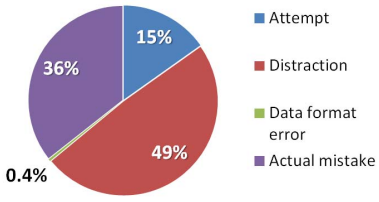


Fig. 7. Oracle reported error analysis – actual mistakes w.r.t. attempts and distraction errors.

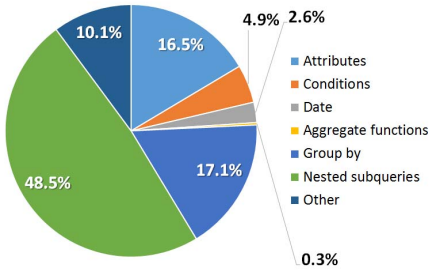


Fig. 8. Oracle reported error analysis – mistakes categories.

81% of the incorrect queries were associated to an Oracle compiler error, i.e., the syntax of the query was wrong. In the remaining 19%, the SQL syntax was correct but the result was not the expected one, in the sense that the output table was different. In the last category, most errors were related to problem statement misunderstanding and the consequent use of the wrong input tables, in the use of an incorrect aggregate

functions (e.g., SUM instead of COUNT), or in the use of wrong keywords in the predicates (e.g., AND between two conditions on the same attribute).

About the Oracle reported syntax errors, the analysis showed the categories in Figures 7 and 8.

Figure 7 shows that most of query errors (64%) are simply trial and error inputs: either they are an incomplete attempt or they are affected by severe lack of attention (e.g. misspelled table names, commas forgotten or inserted where not needed, wrong data formats, misspelled keywords, and so on).

The actual mistakes, analyzed in Figure 8, comprehend several categories. The most frequent errors are related to misconceptions about the structure of nested subqueries, followed by the *group by* construct and by problems with the attributes (such as domain inconsistencies or name ambiguity). Condition errors regard confusion between the WHERE and the HAVING statements. The "other" category includes miscellaneous errors such as string comparison errors or NULL value management.

The teacher used this analysis in the classroom right after the session lab, to discuss with the students the most frequent mistakes and to tailor the following exercise sessions to the actual needs of the students.

D. *Time spent in solving each exercise*

The time spent for solving each exercise is correlated with the number of attempts made to solve it. However, since the problem solving strategies adopted by the students can vary significantly based on their background, we measured and analyzed the amount of time dedicated to each exercise. Specifically, and separately for each exercise, we computed the dedicated time as the gap between the timestamp at which a team accessed the current exercise and the timestamp at which it accessed the next one. Notice that since exercises do not need to be solved sequentially, we considered as the next exercise the one with the closest access time (independently of the exercise number).

As shown in Figure 9, time spent for solving each exercise on average ranges between 10 and 20 minutes for exercises from 1 to 5, between 5 and 10 minutes for exercises from 6 to 8, while it is below 5 minutes for exercises from 9 to 13.

The results confirm that teams spent most of their time in solving the first 5 exercises, while addressing only a subset of the subsequent ones. Exercises from 3 to 5 took significantly more time than expected, since students performed several attempts before submitting a correct solution.

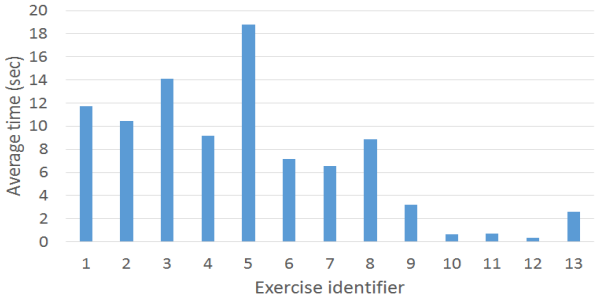


Fig. 9. Average time devoted to each exercise.

E. *Number of requests for assistance*

If teams encounter any problem in solving an exercise, they could ask for the intervention of a teaching assistant. Figure 10 plots, for each exercise, (a) on the left-hand side, the number of requests for assistance and (b) on the right-hand side, the percentage of teams that asked for it, with respect to the number of teams that actually worked on that exercise (refer to Figure 4 for the number of working teams).

The first 5 exercises were accessed by approximately the same number of teams. Therefore, a decreasing trend in the number of assistance requests for the first four exercises indicates an increasing confidence of the students on the SQL syntax, thus a learning progression. Exercise number 5 is an exception, and the large number of requests is related to its perceived higher complexity; this result is coherent with the highest number of query errors (see Figure 6) and the average time devoted to solve it (see Figure 9). Assistance requests for exercises from 6 on are very few, due to the fact that students had less time to work on them.

Figure 10 shows also that a large percentage of active teams asked for and received assistance. More than 45% of teams received assistance (one intervention or more) for the first exercise, even if the percentage decreases in the following ones, it remains quite high for all the exercises addressed by a significant number of teams. This result shows a "democratic" approach in students' assistance, which has been facilitated by the learning environment.

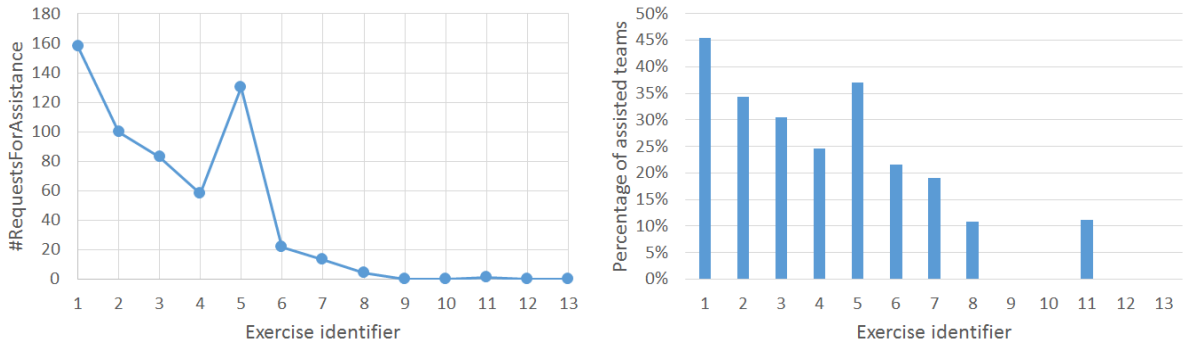


Fig. 10. (a) Number of assistance request and (b) percentage of teams that asked for (and received) assistance

F. Temporal progression of students' activities

We analyzed the temporal evolution of teams' interaction with the environment. Figures 10 to 12 plot the sequences of the key activities performed by three sample teams showing representative behaviors. Specifically, they indicate for each exercise the access time, the time when a correct or incorrect submissions was done, and the requests for assistance. To gain insight into the advanced functionalities of the environment, the color status associated with the team during all lab sessions is also shown in the horizontal bar at the bottom of the figures.

Team 1. The team whose activity is shown in Figure 10 represents an example of team able to work autonomously and with fairly high proficiency. The team correctly solved exercises from 1 to 4 by spending less than 5 minutes per exercise. It made one or two attempts per exercise not requesting any assistance. Then, it encountered problems while solving exercises from 5 to 8. In particular, exercise 5 appears to be critical for most of the students. The team spent on average 20 minutes each for solving exercises 5 to 8 and they made several attempts per exercise. At the end, they solved correctly exercises 6 and 7 without requesting any helps, while for solving exercise 8 they asked for assistance.

The analysis of the evolution of the color status complements the information provided by the graph. Specifically, it highlights an application-driven intervention made by the teaching assistants on exercise 5. Before that exercise, the color status was green. Due to the numerous incorrect attempts, the color status first changed from green to yellow and then from yellow to red. Consequently, the application raised an alarm thus suggesting an intervention by the teaching assistant. Therefore, even though the team did not explicitly make a request for assistance on exercise 5, the application has recognized and properly handled the critical situation.

Team 2. The second example (Figure 11) shows a team who demonstrated high proficiency (8 solved exercises vs. an average of 5), but that needed assistance in three cases. Specifically, after solving exercise 1 the team performed several incorrect attempts for exercise 2. Thus, it requested for assistance, which allowed them to overcome the issue. Similarly, they spent 15 minutes while trying to solve exercise 3. Due to the high number of incorrect attempts, a call for assistance is raised by the application. Finally, exercise 5 was solved thanks to the voluntary request for assistance.

Team 3. The last example, shown in Figure 12, refers to a team demonstrating fairly low proficiency, low autonomy, and a negative attitude in the interaction with the environment. The team performed accesses to only four exercises, it correctly solved only two of them and requested for assistance twice during the practice. Specifically, the first request for assistance was submitted while solving exercise 1. Contextually, the color status changed from green to red. After the assistance, they succeeded in solving the first two exercises, but then they requested again assistance for exercises 3 and 4.

By extending the analysis to all the teams, we are able to extract interesting information about both students' behavior and specific lab content characteristics.

a) *How frequent were the requests for assistance submitted? To which exercises are the requests most correlated?*
The position of the orange squares revealed the exercises for which students asked for assistance. The distance between these indicators reflects the attitude of the teams to work autonomously. The comparison between the results achieved for many students may highlight recurrent issues on specific exercises.

b) *Were the assistances helpful for solving the exercises?*
The closeness between the orange squares related to a request for assistance and any green circle (successful attempt) referring to the same exercise indicates to what extent the assistance intervention was helpful. The correlation between the orange squares and the green circles associated with the subsequent exercises could be deemed as interesting as well, because the knowledge provided by the assistance could help to solve the next exercises.

G. Discussion

The learner-generated data acquired and stored were used to analyze the interactions of the teams with the learning environment. The analyses allowed us to answer the research questions posed at the beginning of the section.

1. *Are the duration and complexity of the practice appropriate for the current level of knowledge of the students?*
The teacher can analyze the temporal sequence of teams' activities to estimate the duration and complexity of the SQL practice. Specifically, the positions of the requests for assistance and of the incorrect attempts allow teachers to easily identify the most critical exercises. The average percentage of solved exercises per team indicates the feasibility of the practice in terms of duration.
The SQL practice proposed to the students in this experiment was too long according to the scheduled time (13 exercises in 90 minutes), but the complexity of the proposed exercises was fair.

2. *Are the exercises proposed in the most suitable order?*
The temporal sequence of attempts indicates the preferred order according to teams' preferences.
The exercises of the SQL practice were sorted in order of increasing complexity. Based on the achieved results, the relative position of exercise 5 appeared to be inappropriate, as most teams spent a significantly higher amount of time compared to that devoted to subsequent exercises (e.g., exercises 6 and 7).

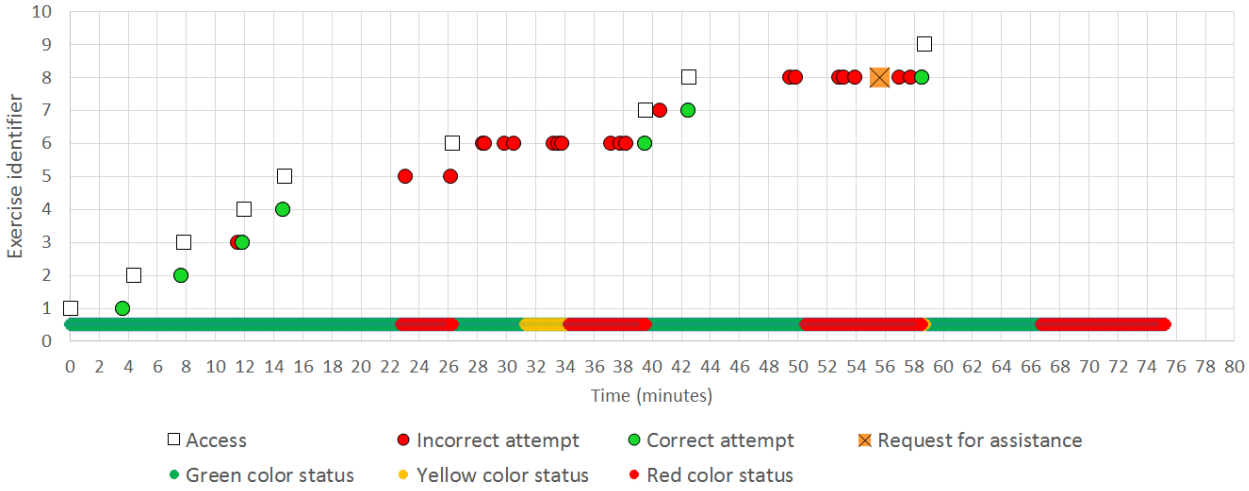


Fig. 11. Temporal sequence of the activities of the sample Team 1.

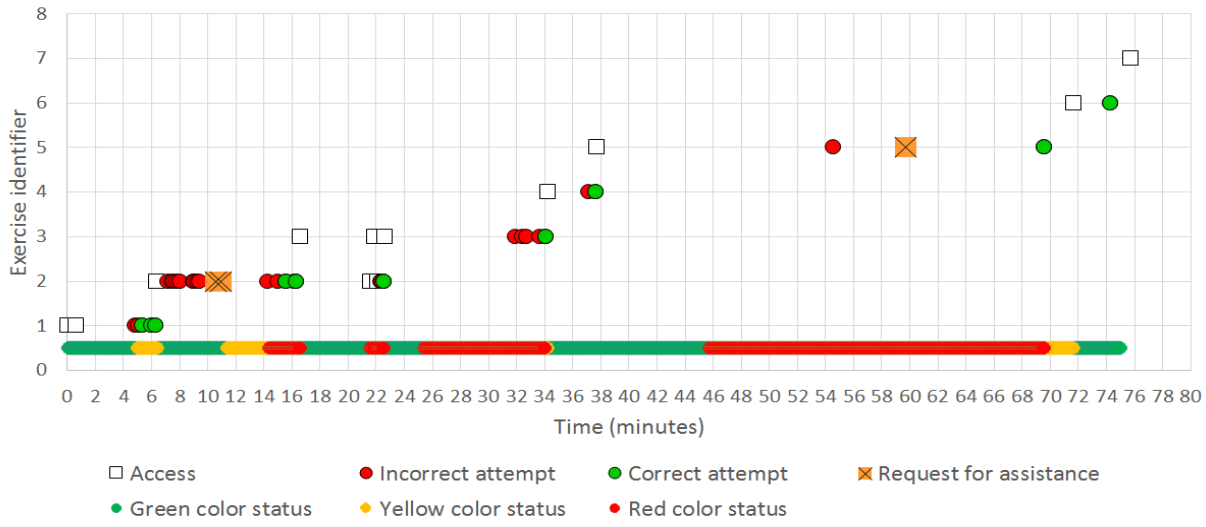


Fig. 12. Temporal sequence of the activities of the sample Team 2.

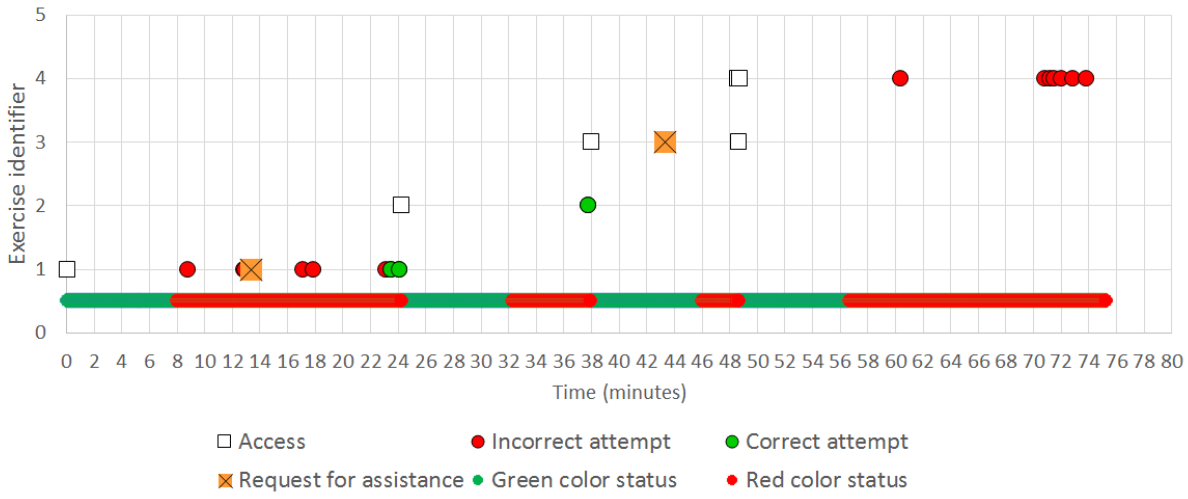


Fig. 13. Temporal sequence of the activities of the sample Team 3.

3. *In what situations are students most frequently asking for assistance?*

   The correlation between requests for assistance and exercises clearly comes out from the activity graphs (see, for example, Figures 10 to 12). Furthermore, the temporal distance between the consecutive requests indicates the attitude of the team to work autonomously. In our analysis, we identified three different team categories: (i) the teams working autonomously and with high profitability (i.e., many solved exercises and few requests for assistance), (ii) the teams needing frequent helps and working with fairly high profit (i.e., many solved exercises but many requests for assistance), and (iii) the teams with low profitability and autonomy (i.e., few solved exercises and many requests for assistance).

4. *What is the impact of assistances on students' activities?*

   The influence of the assistance on the teams' results can be analyzed by measuring the correlation between requests' and correct submission times. The assistance may help teams to solve not only the current exercise, but also the subsequent ones. In the examples in Figure 10 to 12, we identified several situations in which the benefits from assistance are not limited to the current exercise, but can be associated with the exercises addressed immediately after.

V. Conclusions and Future Work

In this paper, we presented a learning environment to improve the effectiveness of teaching assistance during computer science labs. Specifically, we designed, implemented, and tested a distributed, Web-based tool to support six guided practices of an undergraduate database course. Each practice consists of writing a set of queries in SQL language on the Oracle DBMS. The aim of this research activity was twofold:

1) The proposed environment provides teaching assistants with a clear picture of the ongoing status of each student team by storing and on-the-fly reporting the most significant indicators of teams' performance. Online monitoring teams' activities allows assistants to easily identify the teams who need for immediate help and to give the right priority to explicit teams' requests.

2) The environment collects learner-generated data related to the interaction between student teams and the environment. The offline analysis of these data allows teachers to identify the most common mistakes and misunderstanding of the students, possible flaws in the structure and organization of the lab practices, as well as the theoretical parts of the course that need for major revision.

Future work will entail the analysis of the learner-generated data acquired from the proposed environment by means of data mining techniques. Specifically, unsupervised techniques, such as time series analysis and clustering, can be integrated in the proposed environment to gain insights into the behavior of the students' teams during learning practices. For example, they can be exploited to identify groups of student teams with similar misconceptions. On the other hand, supervised techniques, such as classification and regression, can be exploited to on-the-fly predict for each team interesting learning indicators (e.g., the percentage of solved exercises, the level of comprehension of a given topic, the need for future assistance).

Acknowledgment

The authors would like to thank Luna Inguì for her contribution in implementing the application, and the teaching assistant, the tutors and all the students that participated in the lab experiment.

References

[1] H. Washizaki, Y. Sunaga, M. Shuto, K. Kakehi, Y. Fukazawa, S. Yamato, M. Okubo, and B. Tenbergen, "Combinations of Personal Characteristic Types and Learning Effectiveness of Teams", in *Proceedings of the 41st IEEE Computer Software and Applications Conference (COMPSAC)*, Turin, Italy, 4-8 July 2017. DOI: 10.1109/COMPSAC.2017.288.

[2] A. Mitrovic, "Learning SQL with a computerized tutor", in *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education (SIGCSE '98)*, Daniel Joyce and John Impagliazzo (Eds.). ACM, New York, NY, USA, 1998, pp. 307-311. DOI: http://dx.doi.org/10.1145/273133.274318.

[3] S. Sadiq, M. Orlowska, W. Sadiq, and J. Lin, . 2004. "SQLator: an online SQL learning workbench", in *Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education (ITiCSE '04)*. ACM, New York, NY, USA, 2004, pp. 223-227. DOI: http://dx.doi.org/10.1145/1007996.1008055.

[4] L. Aversano, G. Canfora, A. De Lucia and S. Stefanucci, "Understanding SQL through iconic interfaces," in *Proceedings of 26th Annual International Computer Software and Applications (COMPSAC)*, Oxford, UK, 26-29 August 2002, pp. 703-708. DOI: 10.1109/COMPSAC.2002.1045084.

[5] V.A. Romero-Zaldivar, A. Pardo, D. Burgos, C.D. Kloos, "Monitoring Student Progress Using Virtual Appliances: A Case Study", in Computers & Education, No. 58, pp. 1058-1067, 2012.

[6] A. S. Figueira, A. S. Lino, S. S. Paulo, C. A. M. Santos, T. S. A. Brasileiro and A. Del Pino Lino, "Educational data mining to track student's performance on teaching learning environment LabSQL", in *Proceedings of the 10th Iberian Conference on Information Systems and Technologies (CISTI)*, Aveiro, Portugal, 17-20 June 2015, pp. 1-6. DOI: 10.1109/CISTI.2015.7170395.

[7] A. Ahadi, J. Prior, V. Behbood, and R. Lister, "A Quantitative Study of the Relative Difficulty for Novices of Writing Seven Different Types of SQL Queries", in *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '15)*, Vilnius, Lithuania, 04-08 July 2015, pp. 201-206. DOI: http://dx.doi.org/10.1145/2729094.2742620.

[8] J. B. Smelcer, "User errors in database query composition", International Journal of Human-Computer Studies., Vol. 2, No. 4, April 1995, pp. 353-381. DOI: https://doi.org/10.1006/ijhc.1995.1017.

[9] A. Ahadi, J. Prior, V. Behbood, and R. Lister, "Students' Semantic Mistakes in Writing Seven Different Types of SQL Queries", in *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '16)*, Arequipa, Peru, 09-13 July 2016, pp. 272-277. DOI: https://doi.org/10.1145/2899415.2899464.

[10] S. Brass and C. Goldberg., "Semantic errors in SQL queries: A quite complete list", in Journal of System and Software, Vol. 79, No. 5, May 2006, pp. 630-644. DOI: http://dx.doi.org/10.1016/j.jss.2005.06.028.

[11] Oracle DBMS, https://www.oracle.com/database/index.html