Courseware to Support the Teaching of SQL to High School Pupils

Ioan Covalcic

MSc in Information Technology (Business)

Supervisor: Monica Farrow

School of Mathematical and Computer Science

Heriot-Watt University

Aug 2010
ABSTRACT

This document outlines my MSc Dissertation, the purpose of which is the design of an eLearning courseware for the teaching of the basics of SQL to high school pupils. Given the gap in literature as regards design decisions for such a system, as well as no former eLearning software for the teaching of SQL in high schools to use as subject of investigation, this paper provides step-by-step explanations about how the system was developed and how it was validated with users. After outlining the way requirements were captured and describing them, as well as the derived use-cases, the paper motivates design decisions and presents the main functionality of the courseware by means of class, sequence and entity relationship diagrams. The implementation is described my means of screen shots and detailed explanations of functionality. At the end, and evaluation with high school pupils is used to reveal whether the project was successful and what further work is needed.
ACKNOWLEDGEMENTS

I would like to thank Ms. Monica Farrow for her ongoing help, support and advice during this dissertation.

I would also like to thank my family and my close friends who were always there for me and supported me in pursuing this Masters project. This dissertation is dedicated to them.
DECLARATION

I, Ioan Covalcic, confirm that this work submitted for assessment is my own and is expressed in my words. Any uses made within it of the authors of the works of any form e.g. ideas, equations, figures, text, labels, programs are properly acknowledged at any point of their use. A list of the references employed is included.
# TABLE OF CONTENTS

CHAPTER 1. Introduction 1
   1.1 Overview 1
   1.2 Rationale 1
   1.3 Goals and Objectives 1
   1.4 The Users of the Project 2
   1.5 Outline 2

CHAPTER 2. LITERATURE REVIEW 4
   2.1 eLearning 4
   2.2 eLearning for children 6
      2.2.1 Social differentiation 7
      2.2.2 The eLearning market for children 8
      2.2.3 Improvements 9
   2.3 Problems in learning SQL 10
   2.4 Usability in children’s applications 11
   2.5 Solutions for teaching SQL 12
   2.6 Professional, legal, and ethical issues 14

CHAPTER 3. REQUIREMENTS CAPTURE, SPECIFICATION AND ANALYSIS 16
   3.1 Requirements capture 16
   3.2 Requirements Specification 16
      3.2.1 Description of the Volere Template 16
      3.2.2 Project constraints (Volere [61]) 16
         3.2.2.1 Naming Conventions and Definitions (Volere [61]) 16
         3.2.2.2 Assumptions (Volere [61]) 17
         3.2.2.3 Functional Requirements (Volere [61]) 18
         3.2.2.4 Non-functional Requirements (Volere [61]) 22
         3.2.2.5 Open Issues (Volere [61]) 23
      3.2.2.6 Evaluation Requirements 24
   3.3 Analysis 24
      3.3.1 Use-cases 24
      3.3.2 Use-case Descriptions 25

CHAPTER 4. SYSTEM AND SOFTWARE DESIGN 35
   4.1 Choice of Technologies 35
   4.2 Choice of Tools and Hardware 36
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.4 Data Analysis</td>
<td>70</td>
</tr>
<tr>
<td>6.4.1 The method for data analysis and its implementation</td>
<td>70</td>
</tr>
<tr>
<td>6.4.2 The findings</td>
<td>71</td>
</tr>
<tr>
<td>6.4.2.1 Interface Presentation/ Layout</td>
<td>71</td>
</tr>
<tr>
<td>6.4.2.2 Transparency of the purpose and message of the courseware application</td>
<td>72</td>
</tr>
<tr>
<td>6.4.2.3 Navigation</td>
<td>72</td>
</tr>
<tr>
<td>6.4.2.4 Ease of Use</td>
<td>73</td>
</tr>
<tr>
<td>6.4.2.5 Clarity of Feedback</td>
<td>73</td>
</tr>
<tr>
<td>6.4.2.6 Help</td>
<td>74</td>
</tr>
<tr>
<td>6.4.2.7 Learning outcomes in the tutorial</td>
<td>74</td>
</tr>
<tr>
<td>6.4.2.8 Learning outcomes in training</td>
<td>75</td>
</tr>
<tr>
<td>6.4.2.9 Overall impressions</td>
<td>77</td>
</tr>
<tr>
<td>6.5 The Evaluation With Teachers</td>
<td>77</td>
</tr>
<tr>
<td>CHAPTER 7. SUMMARY</td>
<td>78</td>
</tr>
<tr>
<td>7.1 Outline</td>
<td>78</td>
</tr>
<tr>
<td>7.2 Achievements</td>
<td>79</td>
</tr>
<tr>
<td>7.3 Limitations</td>
<td>79</td>
</tr>
<tr>
<td>7.4 Future Work</td>
<td>80</td>
</tr>
<tr>
<td>BIBLIOGRAPHY / REFERENCES</td>
<td>81</td>
</tr>
<tr>
<td>APPENDIX A - Requirements Specification Using the Volere Template</td>
<td>86</td>
</tr>
<tr>
<td>APPENDIX B - Use-case Description</td>
<td>107</td>
</tr>
<tr>
<td>APPENDIX C – ER Diagram for Future Work</td>
<td>128</td>
</tr>
<tr>
<td>APPENDIX D - Table Creation Statements</td>
<td>129</td>
</tr>
<tr>
<td>APPENDIX E - The Object and Field Selection Sequence Diagrams</td>
<td>132</td>
</tr>
<tr>
<td>APPENDIX F- SQL Town Screen Shots</td>
<td>136</td>
</tr>
<tr>
<td>APPENDIX G- code from functions.as</td>
<td>139</td>
</tr>
<tr>
<td>APPENDIX H- The On-line Questionnaire</td>
<td>142</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Figure Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Use-cases (1)</td>
<td>26</td>
</tr>
<tr>
<td>2</td>
<td>Use-cases (2)</td>
<td>27</td>
</tr>
<tr>
<td>3</td>
<td>Use-cases (3)</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>ER diagram of the database</td>
<td>38</td>
</tr>
<tr>
<td>5</td>
<td>The design of the graphical interface prototype of the site</td>
<td>39</td>
</tr>
<tr>
<td>6</td>
<td>Sequence diagram 1: the user starts to use the learning area</td>
<td>41</td>
</tr>
<tr>
<td>7</td>
<td>Sequence diagram 2: The user does a tutorial</td>
<td>43</td>
</tr>
<tr>
<td>8</td>
<td>Sequence diagram 3: The user selects graphical objects from the game-like area</td>
<td>44</td>
</tr>
<tr>
<td>9</td>
<td>Sequence diagram 4: The user does training</td>
<td>46</td>
</tr>
<tr>
<td>10</td>
<td>Sequence diagram 5: the teacher configures a tutorial</td>
<td>47</td>
</tr>
<tr>
<td>11</td>
<td>Sequence diagram 6: The teacher configures a lesson from the tutorial</td>
<td>48</td>
</tr>
<tr>
<td>12</td>
<td>The class diagram</td>
<td>51</td>
</tr>
<tr>
<td>13</td>
<td>Sitemap for the learner</td>
<td>54</td>
</tr>
<tr>
<td>14</td>
<td>Sitemap for the teacher</td>
<td>54</td>
</tr>
<tr>
<td>15</td>
<td>SQL Town homepage</td>
<td>55</td>
</tr>
<tr>
<td>16</td>
<td>The confirmation of account deletion in ‘my account’ page</td>
<td>56</td>
</tr>
<tr>
<td>17</td>
<td>The initial state of the ‘Learn’ page</td>
<td>57</td>
</tr>
<tr>
<td>18</td>
<td>Confirmation on the deletion of a tutorial</td>
<td>59</td>
</tr>
<tr>
<td>19</td>
<td>Add a new lesson in the tutorial</td>
<td>60</td>
</tr>
<tr>
<td>20</td>
<td>The learning environment with all active elements displayed</td>
<td>61</td>
</tr>
<tr>
<td>21</td>
<td>The result of a correct query in the tutorial</td>
<td>64</td>
</tr>
<tr>
<td>E1</td>
<td>The sequence for the user selecting an object from a store</td>
<td>133</td>
</tr>
<tr>
<td>E2</td>
<td>The sequence for the user selecting a field from the store</td>
<td>134</td>
</tr>
<tr>
<td>E3</td>
<td>The sequence for the user moving on the map</td>
<td>135</td>
</tr>
<tr>
<td>F1</td>
<td>“Homepage” with its help page</td>
<td>136</td>
</tr>
<tr>
<td>F2</td>
<td>“Login” page with “The email is not valid” error</td>
<td>136</td>
</tr>
<tr>
<td>F3</td>
<td>‘Tutorials Configuration’ with the confirmation from when adding a new tutorial</td>
<td>136</td>
</tr>
<tr>
<td>F4</td>
<td>“Registration” page error</td>
<td>137</td>
</tr>
<tr>
<td>F5</td>
<td>A training situation example</td>
<td>137</td>
</tr>
<tr>
<td>F6</td>
<td>The message which appears while the user didn’t complete a task in the training area</td>
<td>138</td>
</tr>
<tr>
<td>F7</td>
<td>The message which appears when the user completed a task in the training area</td>
<td>138</td>
</tr>
<tr>
<td>F8</td>
<td>Error in the learning environment for a user query due its wrong format</td>
<td>138</td>
</tr>
<tr>
<td>H1</td>
<td>The on-line questionnaire’s information page</td>
<td>142</td>
</tr>
<tr>
<td>----</td>
<td>--------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>H2</td>
<td>The on-line questionnaire’s first page</td>
<td>142</td>
</tr>
<tr>
<td>H3</td>
<td>The online questionnaire’s second page</td>
<td>143</td>
</tr>
<tr>
<td>H4</td>
<td>The on-line questionnaire’s third page</td>
<td>144</td>
</tr>
<tr>
<td>H5</td>
<td>The on-line questionnaire’s fourth page</td>
<td>145</td>
</tr>
<tr>
<td>H6</td>
<td>The on-line questionnaire’s fifth page</td>
<td>146</td>
</tr>
<tr>
<td>H7</td>
<td>The on-line questionnaire’s sixth page</td>
<td>146</td>
</tr>
<tr>
<td>H8</td>
<td>The on-line questionnaire’s seventh page</td>
<td>146</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Table Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Comparison of the presented tools (a) in practice mode only (b) on two instances (proposal only) (c) for conjunctive queries. (d) planned for next version [41]</td>
<td>14</td>
</tr>
</tbody>
</table>
CHAPTER 1. INTRODUCTION

1.1 Overview

eLearning is a successful way for personal improvement which has grown in close relation with the evolution of computer usage in education [1]. Mainly because of its flexibility, more and more people from all over the world use eLearning to study a wide variety of domains. This is making eLearning one of the most attractive markets for software companies.

eLearning is also an important instrument of the “lifelong learning” concept, which encourages people to learn throughout their whole life, helping to create a knowledge-based society in which knowledge is a precondition for performance and competitiveness [2]. By individualizing the learned information and by not imposing any limitations regarding the time and location of learning, eLearning meets the requirements of a person’s education according to his current phase of life and previous knowledge.

Until soon eLearning was used especially by adults who want to increase their knowledge by acquiring higher qualifications in their free time while still being employed full-time [1]. With the increase of the number of computers and the Internet’s coverage, eLearning also gained a greater popularity within the younger generation. As revealed by researchers in [1], teenagers aged between 16 and 18 are very eager to use eLearning for their studies.

1.2 Rationale

Databases are everywhere! Every modern web site, every game, every complex application nowadays has a database working in its background! You can find a database behind any decent information system nowadays! That is what makes them so important and so useful to study, whatever a learner wants to do in the future.

We believe that SQL, the most used language for databases and one of the most used languages in IT overall, could be taught even from high school given the right tool for it. To this end, a courseware consisting of a fun environment and attractive graphics could make the basics of SQL easy to learn even by high school pupils. Moreover, by relieving the stress sometimes triggered by traditional teaching methods, it could encourage these learners to pursue the important subject of databases even more. This is the main problematic which motivated this project.

Even if the number of eLearning applications is continuously growing, the domain of databases lacks a proper tool dedicated for high school pupils. The current project will try to fill this gap.

1.3 Goals and Objectives

The main goal of the project is the design of an eLearning solution to be used with high school pupils for the teaching of the basics of SQL. Given the global use and importance of databases and our belief that a
subject whom is pleasurable to learn also indulges the learner to pursue it more, the courseware must stress on graphical attractiveness and ease of use. Designed as a fun and interactive game, the courseware must encourage the learning of basic SQL queries by means of interacting with graphical objects on the screen.

The courseware will be developed as a web site, thus it will be available to be used both during classes or laboratories in high schools or from anywhere else.

A default tutorial will take learners step-by-step through their first SQL lessons by means of indulging them to learn by interacting with the game area and observing the automatic generation of queries and then use what they learned to write their own queries. The training area will allow learners to practice their newly acquired skills by automatically generating tasks for them, while also providing learners with suggestions and SQL syntax information.

The project will allow for teachers to add and update their own tutorials with the lesson content of their choice and manipulate environment elements to support the lessons.

The project will benefit from the actual technological advantages of interactive multimedia. Moreover, animation and graphics will be used in order to convert the normal SQL language from a text-based form into a more graphical one filled with many examples in order to help in a better understanding of the topic, obtained in a much shorter period of time than required by any introductive course or tutorial in databases.

1.4 The Users of the Project

The users of the product would be both high school pupils and teachers. For pupils, the site will constitute a fun and relaxing way of studying the basics of SQL, hopefully helping them to easier understand its concepts. For teachers, the site will constitute a framework for building tutorials to be used in the first SQL classes with pupils. Older people such as university students or even adults who are beginners in SQL could be future users of the site if its initial outcomes within high schools are positive.

1.5 Outline

The following are the main chapters of this dissertation, reflecting the actual phases in the design of the courseware:

Chapter 2. Literature Review – presents the background on eLearning and its applications related to our subject.

Chapter 3. Requirements Capture, Specification and Analysis – provides an outline of the problems considered while capturing requirements and the main functional and non-functional requirements derived as part of a Volere ([61]) template; for the functional requirements, their analysis by means of use-cases is also provided and described by using the Alistair Cockburn template ([62]).
Chapter 4. System and Software Design – explains what technologies were used and the rationale for using each; it then describes the design decisions taken as departing from the requirements and use cases by means of sequence, class and entity relationship diagrams.

Chapter 5. Implementation - describes by means of screen shots the look and feel of the end system and how the functionality decided on by design decisions was implemented by using the decided-upon technology.

Chapter 6. Evaluation - discusses the choice of method for validating the courseware, how the participants to the evaluation were selected and contacted, how the evaluation method was implemented and its results.

Chapter 7. Summary – provides a discussion about the validity of the courseware to be employed as a tool for teaching the basics of SQL to high school pupils according to the findings from the evaluation; explains what limitations the chosen evaluation method had and how they should be further addressed; proposes future work.
CHAPTER 2. LITERATURE REVIEW

2.1 eLearning

E-Learning is a successful way for personal improvement which has grown in close relation with the evolution of computer usage in education [1]. Mainly because of its flexibility, more and more people from all over the world use eLearning to study a wide variety of domains. This is making eLearning one of the most attractive markets for software companies.

eLearning is also an important instrument of the “lifelong learning” concept, which encourages people to learn throughout their whole life, helping to create a knowledge-based society in which knowledge is a precondition for performance and competitiveness [2]. By individualizing the learned information and by not imposing any limitations regarding the time and location of learning, eLearning meets the requirements of a person’s education according to his current phase of life and previous knowledge.

eLearning can be defined as consisting of “communication and learning activities through computers and networks (or via electronic means)” [3] [4] [5] [6]. In addition to this general definition, more specific ones exist. eLearning is also defined as the “delivery of training and education via networked interactivity and a range of other knowledge collection and distribution technologies”, being also “knowledge delivered by online services as education and training” [7] [8]. Another definition states that eLearning is “self-learning thorough technology”, insisting on the fact that eLearning is also a collaborative type of learning [9].

An important part of eLearning is related to the way in which knowledge is stored and spread, duties realised by using the Internet and web technologies. Also, in addition to web technologies, eLearning needs multimedia based courseware to sustain knowledge [10] [11]. Very important for eLearning it is also the learning performance and efficiency of ICT [12] [13]. The learner will have the possibility to use two-way video, multi-way computer interaction, cable, Internet and many other features of ICT [14]. Good examples of the implication of ICT in the learning process are the learning from e-mail through subscription to online courses and online discussion on different domains and coaching by e-mail [15]. A final definition considers eLearning as a set of learning activities involving the use of computers, networks and multimedia technologies [16].

Because of the fast progress of Information and Communication Technology nowadays, eLearning is considered as the way forward in education. The main advantages of eLearning are related with the elimination of time and space constraints, deterministic for the traditional face-to-face education that affects the interaction between learners and instructor and also between learners and learners [17] [18] [19]. As a result, the performance of learners and also the one of the instructors is limited through the use of asynchronous and synchronous learning network models.

As we saw in most of the presented definitions, basic eLearning could be considered as consisting of a web-based system which allows the users or learners to have access to information or knowledge without
being restricted by time or geographic location. Reduced online learning time, reduced labour and resource advantages over traditional face-to-face education [20] do not explain the high number of failures of eLearning projects [21], which leads to the necessity that eLearning be studied at a deeper level by future eLearning solution developers.

From a different point of view, due to its importance for education in a modern society, eLearning has experienced a demand increase through businesses and universities, which can be summarised in a growth rate of 35.6% per year [22]. Even so, a high level of failure exists, which can be attributed to the fact that some users stop their interaction with the eLearning tool after an initial experience with it [21] [23]. Research in information system shows user satisfaction as being the key factor for the success in the implementation of an eLearning tool [24]. To understand user satisfaction with eLearning better and to avoid failure a set of factors are considered related with it, which can be grouped by using six factors: student, teacher, course, technology, system design, and environment [21] [24] [26] [28] [28] [30] [30] [31] [32]. Nevertheless, the proposed approaches cannot be viable for a business solution due to the big number of factors which are making the implementation very difficult and change almost impossible.

Nonetheless, eLearning advantages over face-to-face education are numerous [33]. These advantages are not totally obvious even if some of them, like time flexibility and lower cost, are well known by most of the people. Some of the most important advantages of eLearning as compared with face-to-face education are presented in the following list [56]:

- Less expensive – the cost of an untutored training program can be zero after the cost of the professional authoring software in which it was developed will pay off, or can be zero from the beginning by using free authoring software. For a tutored training program, in addition to the initial cost of the development there will be additional costs for the instructor supervising the class, but the total cost will still be lower than that for a traditional course, which requires in addition material resources and accepts a smaller number of participants.

- Not dependent on time and location – The learner can participate in training sessions from anywhere by using an Internet connection and in most of the cases at any time, excepting the cases when the course is time related with a specific event or depends on the availability of the instructors. This allows the learner to be employed and also to manage his time in order to make learning possible. This was not possible with traditional education.

- Self-paced – the training programs don’t have a fixed timeline for the content to be followed (except of the case in which a timeline is related with the purpose of the training), allowing the learner to manage how much time he spends on a part of the training in order to better understand and assimilate the presented information.

- Faster – the time required for an eLearning course to be completed is up to 50 percent shorter compared it with a traditional course. The difference appears in most of the cases because the course
is individualized and offers to learners the possibility to skip the parts they already know, allowing them to stress on the parts on which they need training.

- Increased retention – eLearning materials contain video, audio, quizzes and other strategies which improve the understanding of the message. Learners also have the possibility to replay certain parts of the training which were not clearly understood the first time.

- Easily managed large groups of students – eLearning courses are in most of the cases not limited to a specific number of participants, offering access to a training session to much more learners then it could be supported at once by traditional education.

- Consistent message – the problem regarding the different material on the same course on different sessions, which appears due to different instructors which teach the course at once, is eliminated. The message is consistent in the entire lifetime of the course and it can be modified just when it is required.

- Easily and quickly updatable – The course material can be updated and it can be made available on web-based eLearning applications by simply uploading the changes on the server. For applications using memory support the update is more difficult and time-consuming, but is still much cheaper than the change of course material in traditional education, where the reprinting of manuals and the retraining of instructors is required.

In addition to the presented advantages of eLearning, many others exists and well known disadvantages of eLearning like technophobia, boring text-based content, loneliness, can be avoided if the eLearning course is well designed [33].

### 2.2 eLearning for children

Informal learning, related to the development of competencies, is seen as the most important type of learning because it is the primary goal in the education of a child [2]. It is influenced both by the informal background in which a child grows, but also and very importantly by the actors involved in the child’s education, such as educational institutions. For this reason it is vital for a child that the decisive actors involved in his education be well trained and well networked from an educational point of view.

Depending on the context in which they are required, the characteristics of a child as a user of the computer or of the Internet may vary. This is one of the conditions which need to be considered when eLearning solutions dedicated to them are developed.

In order to have the possibility to use children as a target group to verify the educational influence of eLearning on and considering the previously presented condition for the development of an eLearning
solution, we need to respect child-related preferences [2]. For most of the children, the computer’s main purpose is to be used for playing games, and in this way to use it for informal learning.

A great number of studies and pedagogical approaches concerning child demands on what concerns eLearning were already undertaken and many solutions for applications implementation are already used [59]. Also, measures are undertaken to evaluate and guarantee the quality of eLearning products. Children and young people are already a very important target group if we consider the number of eLearning solutions created for them, and it is considered that they will become an even more important target group in the close future. As we can easily see, the market for eLearning solutions is already a very important one and it also presents signs of improvement for the future.

Due the market demand and future commercial potential of eLearning solutions for children, eLearning experienced an increase in research in the fields of multimedia education and child psychology in order to satisfy the needs of the target group [2]. The research is directed to discovering new possibilities for the improvement of eLearning, but also to discovering unhealthy influences and consequences of using eLearning instruments on children. An important part of the research is undertaken to increase the focus on pre-school children, a target group for which there are a much fewer number of solutions compared with the requirements of the market [58].

One of the most important areas, into which a great level of work is dedicated, is the evaluation of eLearning solutions for children and it is undertaken on different domains involved in eLearning, such as computer science, educational sciences and multimedia education [2]. The scope is not just to measure the success of eLearning software but also to verify how good the solution is in making the differentiation between age groups, using various didactic approaches, and creating the appropriate knowledge [2] [59].

2.2.1 Social differentiation

Nowadays, having access to a computer and Internet is not an issue for the majority of children and young people, even if it is at home or at school. Nonetheless, social inequalities can be noticed in the informal background of a child [57]. Children coming from a better environment from a social point of view have the possibility to use computers and the Internet at a great level to develop their multimedia skills, while children from underprivileged environments lack access to a computer at home and spend a great amount of time watching television. As a result of this, they have a harder time to complete the participation conditions for eLearning courses.

Children coming from an underprivileged environment have problems in the self-directed learning by using online resources [2]. This problem appears due to the lack of the required high-level of motivation, cognitive skills required for navigation and capacity to relate information coming from different sources. The problem is even worse for children who are slow learners, but less important for those who have high learning abilities. We must say that even children coming from a better environment who have access to a great number of resources can encounter problems without adult supervision.
2.2.2 The eLearning market for children

The eLearning market dedicated for children is dominated by transparency in what concerns the existing products and their providers [2] [58]. The market participants belong to different areas, ranging from being private persons to educational publishers. Even if most of the companies involved in the market develop free products dedicated to children, mainly meant to enhance the companies’ image or to attract further collaboration, several other providers choose to develop commercial solutions to gain profit on them. Institutions such as non-profit organisations or holding public contracts are also involved in the market.

The main types of eLearning solutions existing on the market for children are educational-entertainment services, straightforward learning applications and computer games [58]. The last type of eLearning implementation is very important if we think that most of the existent computer games have didactic features, helping children develop important skills such as reasoning, dexterity and strategic thinking.

The range of eLearning products dedicated to children comprising educational software, seminars, courses, Internet platforms, pages and forums as well as educational-entertainment products and games, is very diversified covering all of the children’s’ traditional requirements and much more [2]. This shows that children create a target group seriously considered from a commercial point of view. Furthermore, we can observe an ascending trend in the development of interactive interfaces for the eLearning product range ([2], [58]). This trend can be explained by the following [2]:

- applications are more complex, partially because of the increase in availability of high-speed Internet connections, which permits a better use of actual technology without affecting loading time.
- due their communicative parts, interactive interfaces are preferable than standard static interfaces.
- developers are also in favour of using interactive interfaces with highly graphic elements for attracting clients.

For a better distribution of eLearning solutions, providers try to increase the transparency of the market in order to highlight their products by using a greater range of distribution channels [2]. In addition to the traditional distribution channels like advertising on TV or radio are used the relative new channels like providers’ Internet pages and software databases.

Another way of distributing eLearning solutions, highly beneficial for the providers, is by relating their products to an institution, which will sponsor the solutions [58]. This was made possible by improving the quality of the eLearning solution, thus highlighting a clearer image of the product and making it attract sponsoring.
2.2.3 Improvements

In the present one philosophy that is used to define the modern types of education is the ‘lifelong learning’ philosophy. As the most important requirement for a knowledge-based society from an economic and social point of view, lifelong learning has become a philosophy adopted by many people to advance in new positions by developing new competencies [2]. This type of learning has its start from the school benches till the end of our life. In this context, eLearning becomes a very important addition, offering the possibility to fill the gaps left by traditional forms of education, due to the consideration of the individual needs of each learner when planning content. Furthermore, eLearning implementations are very individual learning tools.

Even if eLearning has already become an important part in the lifelong learning philosophy, on the whole it needs a lot of consolidation [2]. This situation appears because, in the case of many eLearning solutions, very important steps in their development were overlooked or badly completed due the hurry of completing the product as soon as possible and also to the lack of an initial deeper experience with the subject. The investigation of the learning and educational factors vital to develop an appropriate content is a step that is usually insufficiently considered in the development of an eLearning application or service.

To diminish the existing social inequalities, a better differentiation of the target groups of children and adolescents, by considering previous traditional learning conditions and using appropriate pedagogic approaches, is required [57]. The evaluation of educational software needs to consider in a much greater level the differentiation in target groups by age, how easy it is to learn the theory contexts, how good was the previous traditional didactic approach, and not just measure the success of the implementation.

Another important aspect that requires improvements is the insufficient access locations to computers and the Internet for children and the inappropriate use of computers, which are preconditions of eLearning [2]. Schools currently don’t benefit from enough resources to compensate the lack of access from children’s homes. In order to resolve these problems changes are required on different contexts [2]:

- at home, parents need to better supervise their children on how they spend their time using computers and try to increase children awareness regarding the information-oriented use of computers.

- schools need to give the possibility to children to have access to computers and Internet at an early age in order to start learning from them as soon as possible. While the courses are developed, it is necessary to foresee the social differences between children and provide methods to help the ones coming from an underprivileged environment to have the expected results.

- a better training of teaching staff is a necessity in order to prepare them for covering at an appropriate level didactic and technical aspect of using computers and the Internet in the classroom. Currently, the number of available training courses for staff members is very limited.
further research is needed to better understand how the teaching of computer and Internet courses should proceed according to the possibilities of each children age group.

A different direction for research is related with discovering the level in which eLearning solutions can be used by children coming from underprivileged environments and how good they are as compared with traditional education regarding informal learning [58].

2.3 Problems in learning SQL

We shall talk about the problems in learning SQL from a higher education point of view as we did not find any documented trial to teach SQL to high school pupils.

Structured Query Language (SQL) is the most important language existent on the relational database market and it is an essential topic covered in introductory courses regarding databases in higher education. The language raises a series of problems to students learning it even if its syntax is simple, structured and concise.

SQL consists of two distinct types of language: Data Manipulation Language (DML), used to manipulate data, and Data Definition Language (DDL), used to define the relational schema objects. Most of the introductory courses in higher education aim in developing in students a good knowledge of DML query expressions in SQL, a process often found as difficult. This is done by stressing on creating a logical connection between natural language questions and semantically correct SQL queries, as much as writing syntactically correct statements.

Many problems appear for students in the process of learning SQL. The most often encountered problems are listed below [34] [35] [36] [37] [38]:

- having a declarative nature, where you need to think in sets and not in steps, SQL is difficult to grasp when previous experience with other programming languages exists [34].

- students find it very difficult to remember the schema of a database, very often resulting in wrong solutions because of the use of wrong table or attribute names [35]. This problem forces the student to focus much more than necessary on low-level syntax and affects the more advanced feature learning.

- even if most of the introductive courses combine SQL basics with the relational data model, many students find it problematic to learn the logic behind a query and the basics of SQL, and even more the relational data model [34]. Some of the most difficult to learn seem to be the concepts of joins, grouping and aggregation.

- students perceive some query problems as much easier than they really are [38]. Due to the lack of experience in expressing questions in SQL, many students encounter the problem of discovering very
difficult to create queries from very clear and easy questions. Even if many queries can be written by using a relationally complete language, there are still difficulties in creating and understanding them.

To avoid or solve these problems, most of the courses provide a relational database management system in order for students to have an environment in which they can experiment and gain dexterity on creating queries without stressing on syntax [39]. This method is good in order to give fast feedback to the student’s work during training, but it is not useful for examinations, where semantic correctness is much more important than in training.

From the student’s point of view, it is necessary to have a simple work environment which accepts query expressions and returns feedback in order to learn SQL better [39]. From the instructor’s point of view, it is necessary to have a tool that can be used to support teaching and also be used in grading, diminishing the time required for marking the assignments. Both parts require a tool to support the practice of querying abilities in SQL in order to improve the result of SQL studying.

In the past few years, a set of tools has been developed which are used to sustain SQL learning and also return partial feedback. This is possible because relational query languages are not Turing complete and theoretically allow for immediate syntactic and semantic feedback. Several SQL tools will be presented in a future sub-chapter.

2.4 Usability in children’s applications

Even if often neglected by software designers, usability is one of the main points which need to be covered when developing a software system dedicated to eLearning. Usability was defined and related to overall system acceptability in 1993 by Jakob Nielsen [42]. Until then, the understanding of usability evolved from that of focus on efficiency and “user friendliness”, as defined in [43], to the definition published within the standard ISO 9241-11: “the effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments” [44].

For eLearning applications dedicated to children, the study of usability is even more important than in the cases of applications dedicated for adults. Children perceive things, in most of the cases, differently than an adult and have a much smaller level of patience, requiring better interfaces and control to keep them concentrated.

As it is the most widely used type of application by children, the websites were studied from their perspective from different points of view including usability [60]. It was discovered that teenagers and adults are performing differently on the web because of the teenagers’ insufficient reading and the difference in information research skills compared with an adult because of the fact they are less patient. Teenagers are also more intolerant with websites if they consider them as boring or difficult to understand. The importance of website graphics is much greater for them than for an adult. It was even demonstrated
that the loading speed of a website is even more important to them, teenagers preferring a website with low graphics but which load fast than one with heavy graphics but poor loading speeds. Other findings show that teenagers do not like to read too much in general and, especially, from the web and that they like small fonts even less than having to read. They do like interactive features though because of their interest to be seen as different from other people.

2.5 Solutions for teaching SQL

eLearning has become in the last few years one of the most popular types of Internet application, being sustained by a large number of specialists which support the transition from traditional education to this modern and active type of education [45] [46] [47] [48]. In addition, eLearning is one of the most active research areas related to the Internet, many researchers focusing on developing eLearning tools specialised on interactive and individualised learning [45],[46],[47]. In many domains, a number of powerful solutions already exist to aid teaching for students. Even if the number of good solutions is relatively low, the databases domain contains solutions like SQL-Tutor [36], [48], dedicated to higher education, with which many students already gain a better knowledge of SQL by solving generated personalized problems.

For high school education, not even one eLearning application for supporting SQL teaching and having as target group high school pupils exists. Under these circumstances, the research of SQL eLearning solutions will be made by referring to solutions from higher education.

Even if the number of solutions is continuously increasing, most of them cannot be used at a wider level than just in the academic environment [45]. The problem usually appears due the technical problems which emerge when trying to connect the application with existing Learning Management Systems. Rich interactive applications cannot be copied easily on the Internet, each one having individual requirements in order to be functional. The best known SQL courseware solution, SQL-Tutor [36], [48], needs a specialised server, on which the student needs to log in before starting his work, in order to be functional and to generate problems dynamically. After the work session, the system needs to save all the data about the student’s work in order to give details about the student progress to the teacher and to the student himself.

Studies undertaken on existing Learning Management Systems [49] revealed that these systems do not support interactive and personalized learning content, being capable to support just static content. The interactive content is available just by using independent systems using their own resources as SQL-Tutor [36], [48]. In addition with this problem, students may need to work with several independent solutions for the same course in order to manage to get the best features from all. The technical difficulties arisen by using several systems are in many situations much more important than any advantage for students and also for teachers.
In order to use several interactive systems in the same course, some problems need to be solved [45]. The first problem is related with the fact that usually each system has its own login function, which for an integrated solution needs to be unified. The second problem is concerned with the sharing of information on the student’s actions between the different systems. A third problem appears when the current level of the student is required and the statistics from all the systems need to be joined together.

One of the attempts in trying to solve the technical problems that appear when connecting interactive learning tools, was “integrated Exploratorium for database courses” [50]. The solution provides personalized access to several interactive learning activities, allowing the addition of other systems and the integration with the existent systems. Exploratorium currently contains annotated examples, self-assessed questions and an SQL laboratory.

Kennesaw database courseware is also a complete system dedicated to students for learning database basics by using interactive tools, animations, examples and sample tests available online [51].

Other projects exist which try to create architectures in which to allow the integration of several interactive eLearning systems. Some of them are MEDEA[54] and APeLS[55]. These projects tested different technologies in their attempt to find solutions, keeping an open communication with the eLearning community. Momentarily, the developers of both projects work together in their attempt to create a common framework which is hoped to also create a set of integration standards for interactive systems.

Even if the number of eLearning applications has an ascending trend and many interactive learning tools have been created for different computer domains, there are not too many solutions created for databases, especially for SQL [45]. The existing systems can be divided into two categories. The first category supports teaching of SQL basics by using interactive examples [51], [52], while the second is based on learning by practice, being based on problem solving and evaluation of the results [48], [53]. Both methods have proven themselves as effective for different situations. As presented in [48], students using SQL-Tutor have shown significant improvement on SQL problem solving.

Some of the most important of the tools that support SQL teaching are the following:

- eSQL (1997) was proposed in [34] as a helping tool to assist teaching of query processing concepts.

- SQL-Tutor (1998) was developed at the University of Canterbury, New Zealand [36]. It provides semantic feedback and is one of the most important tools created to aid the teaching of SQL, inspiring many of the following applications.

- SQLator (2004) was developed at the University of Queensland, Australia [16]. One of its most important features is the automatic grading of the queries submitted by the students.

- AsseSQL (2004) was developed at the University of Technology in Sidney, Australia [40] and is similar to SQLator.
- SQLify (2007) was developed at the University of Southern Queensland, Australia [41] and incorporates peer review and an improved system of automatic assessment in order to produce more comprehensive feedback for the students.

A cross-disciplinary review of the presented tools, created to reflect database theory and practice content as well as the pedagogical elements contained, is presented in Table 1 for the selected tools.

<table>
<thead>
<tr>
<th>Feature</th>
<th>eSQL</th>
<th>SQL-Tutor</th>
<th>SQLator</th>
<th>AsseSQL</th>
<th>SQLify</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modelling of student to individualize instructional sessions</td>
<td>X</td>
<td>√</td>
<td>x</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>Visualization of database schema</td>
<td>X</td>
<td>√</td>
<td>x</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>Visualization of query processing</td>
<td>√</td>
<td>X</td>
<td>x</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Feedback on query semantics</td>
<td>X</td>
<td>√</td>
<td>x</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Automatic assessment (using heuristics)</td>
<td>X</td>
<td>X</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Automatic assessment (using conjunctive queries’ query equivalence)</td>
<td>X</td>
<td>X</td>
<td>x</td>
<td>x</td>
<td>√</td>
</tr>
<tr>
<td>Use of peer review for assessment</td>
<td>X</td>
<td>X</td>
<td>x</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>Relational Algebra expressions support</td>
<td>X</td>
<td>X</td>
<td>x</td>
<td>X</td>
<td>√d</td>
</tr>
<tr>
<td>Special treatment of distinct and order by</td>
<td>X</td>
<td>X</td>
<td>x</td>
<td>X</td>
<td>√</td>
</tr>
<tr>
<td>SQL-injection attack countermeasures</td>
<td>X</td>
<td>X</td>
<td>x</td>
<td>X</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 1: Comparison of the presented tools (a) in practice mode only (b) on two instances (proposal only) (c) for conjunctive queries. (d) planned for next version [41]

2.6 Professional, legal, and ethical issues

The application developed within the current dissertation will be a relative small one, but even in this case it needs to be projected and implemented in a professional way. By doing this, the value of the final product will be greater and it will have a positive impact on the developer’s professional experience.

In order to create a professional application, during the project lifetime, a great importance will be allocated to the creation of diagrams and additional documentation to support a better design of the program. Also, the application will be tested during the whole implementation period and after that in order to create a product without functioning errors and appropriate for the requirements.
In addition to that an important part of the project will be to create a secure application which will store and control the user data without being possible to be alter by unauthorized persons.

The application will have as a guide of professional attitude the Code of Good Practice developed by the British Computer Society.

During the dissertation project time the Intellectual Property Law, the Data Protection Law, the Computer Misuse Law, and any other computer related regulation will be respected.

All the documents and computer application used during the dissertation development will be attributed to right owner in the reference area.

The application will have a private area for each user. The private area will be accessed by a username and a password allocated at the beginning of the activity on the project’s application.

All personal information of the test users will be kept until the final submission date of the dissertation project and deleted after that. Before then the test users will have the possibility to see their personal information and also to change their saved details.

The users will be informed from the beginning in how their personal information will be used during the application evaluation and the gathered data will be used according to the initial user approval. If the users’ data is required in a different part of the dissertation project the users will be informed about the changes and a new approval will be gain from them.
CHAPTER 3. REQUIREMENTS CAPTURE, SPECIFICATION AND ANALYSIS

In the previous chapters we have seen what the rationale and purposes of the project are, as well as an overview on the existing literature on eLearning and eLearning systems used in high schools. In this chapter we will describe the first phase of the work on developing the coursework for teaching basic SQL to high school pupils, that of capturing and analysing the user requirements. To this end, we will use the Volere template which is briefly described below. As Volere is normally used for commercial products, we have only considered the subchapters from the template which also apply to the case of the intended system. From reasons of space, we will only provide an outline of the main points in the Volere template ([61]) in this chapter and the whole requirements specification document in provided in Appendix A. We will also not present here the points which were already covered in Chapter 1 (point A from Volere) or which will be covered later in this paper.

3.1 Requirements capture

For capturing the teacher requirements, we discussed with our supervisor about the general issues related to the subject and her expectations from a courseware application teaching SQL. Next, we reviewed the literature on courseware applications on programming languages or even on SQL, but for university students, in order to discover their most important features and to decide what is important for our application. For capturing the learner requirements, we held informal discussions with fellow students and, for the primary target learners, with several high school pupils.

3.2 Requirements Specification

3.2.1 Description of the Volere Template

Volere is a professional commercial template used for the specification of requirements [61]. It identifies seventeen types of requirements categorized as product constraints and functional-non functional, allowing for software developers to maintain a checklist and logical structure of their requirements. The functional requirements from Volere are correlated with use-cases, thus allowing for the traceability of requirements. The template allows the developer to maintain a clear list of requirement rationales to which he can return at any point in the design, thus avoiding later misinterpretations, and also a clear list of assumptions and open issues which document the way a user requirement has initially been interpreted and the problems further requirements capture must address.

3.2.2 Project constraints (Volere [61])

3.2.2.1 Naming Conventions and Definitions (Volere [61])

User = anyone who is using the site; it might be a learner or teacher.
**Learner** = anyone who is using the site for learning purposes; it will initially be high school pupils, with the possible extension to university students or even adults; it does not matter what specialism the learners have as long as they use this site for learning SQL.

**Teacher** = any teacher of SQL (might be a high school or later on even a university teacher), from any specialism.

**Account** = the user’s personal interface to the system as a result of his use of it

**Registration** = the process of creating an account to one’s self as a user of the site

**Change account details** = the process of amending all the registration details, apart from his email address, which is used for user identification purposes.

**Lesson** = Short description of a topic as having been entered by the teacher and made available to the learner in the form of a short message within a tutorial.

**Tutorial** = learning guide made up by a series of lessons used by learners in their study of SQL.

**Training** = process of practicing SQL as a learner by solving tasks which were designed by the application.

### 3.2.2.2 Assumptions (Volere [61])

1. Although the site will be used by learners who are, as part of their school curricula, beginners to SQL, the site does not cater for the possibility of them having studied it otherwise than at school before. We therefore assumed that all the users have a beginner level.

2. We assumed that, for a beginner level, the learner would learn the basic notions of SQL: table, column, row, primary key.

3. Also concerning the beginner level, we assumed that a learner would be taught the basics of writing SELECT queries: using *, adding one or more fields, using WHERE, DISCTINCT and ORDER BY.

4. Also concerning the beginner level, we assumed that a learner would be taught only how to execute SELECT queries on single tables, thus we will not consider joins as functionality.

5. We assumed that the teacher would only use this site with beginner learners as defined by the assumptions above and that he would not therefore need to be able to extend the available types of supported queries. To cater for the possibility of later extension to the use with more advanced learners, we will propose additions as future work.

6. We assumed that the teacher might have several classes to which he wishes to use different tutorials.
7. We assumed that the teacher might want to check the way his tutorial gets displayed to learners.

8. We assumed that the teacher would like to correlate the graphical display in the site with SQL queries to better explain the queries that he is teaching.

9. We assumed that the teacher would prefer that the site builds the learner training tasks instead of having to write them down by him.

10. We assumed that a learner would only start training after he has read all the tutorials. The training will therefore allow him to practice all the knowledge acquired after having learned the tutorials and will not cater for learners who have not read the tutorials or try to get trained first thing when accessing the site.

11. We assumed that the learners would like to use the site at home as well, not only during classes.

12. We assumed that the site could be used by several learners at different browser accesses from the same computer while the computer is on as it could be used during separate classes by different groups within a laboratory.

13. We assumed that teachers’ accounts on the site would need to be verified by an administrator before the tutorial configuration facility be made accessible.

### 3.2.2.3 Functional Requirements (Volere [61])

The following is the shortened list of functional requirements. The extended list, contained in Appendix A, also contains the description and rationale of each requirement, its fit criterion (condition under which it is considered the requirement is respected), the use case to which the functional requirement applies to and the requirement’s dependencies with other requirements (whether the requirement is influenced or influences other requirements).

1. The site allows users to register.

2. In order to register, the site allows the user to enter his first name, last name and email address.

3. In order to register, the system allows customers to choose a user name and password.

4. In order to register, the system requires the user to enter the password twice.

5. The site allows users to log in.

6. In order to log in, the site allows users to enter their email address and password.

7. The system allows users to modify their registration details at any point while being logged in.
8. In order to modify registration details, the system allows users to modify to amend their first and last name.

9. In order to modify registration details, the system allows users to amend their first and last name.

10. In order to modify registration details, the system allows users to decide on a new username and password.

11. In order to change the password, the system requires users to enter it twice

12. The site allows the users to interact with the town.

13. While the user interacts with the town, the site allows him to move on the map.

14. While the user interacts with the town, the site allows him to select one store from the town.

15. When selecting a store, the site allows the user to observe what it sells.

16. While the user interacts with the town, the system allows him to deselect a store from the town.

17. While the user interacts with the town, the site allows him to select one object from within a store from the town.

18. When selecting an object from the store, the site allows the user to observe its details

19. While the user interacts with the town, the system allows him to deselect one object from a store from the town.

20. While the user interacts with the town, the site allows him to select several objects from within a store from the town.

21. While the user interacts with the town, the system allows him to deselect an object out of a list of several objects from a store from the town.

22. While the user interacts with the town, after having selected a store and object the site allows him to select one field of a store from the town.

23. While the user interacts with the town, after having selected a store and object the site allows him to deselect one field of a store from the town.

24. While the user interacts with the town, the site allows him to select several fields of a store from the town.

25. While the user interacts with the town, the site allows him to deselect several fields of a store from the town.
26. While the user interacts with the town, the site allows him to retrieve query information.

27. The site allows the user to write a query.

28. The site uses an SQL engine to execute a query.

29. The site allows the site to retrieve the results of a query.

30. The site allows the users to do a tutorial.

31. While the user does a tutorial, the site allows him to navigate through the list of lessons.

32. The site allows the users to get trained in SQL.

33. While the user does training, the site allows him to move forward in the list of tasks.

34. The site allows the users to explore the town without having to do a tutorial or training.

35. The site allows teachers to add a new tutorial.

36. As part of adding a new tutorial, the system allows the teacher to enter the tutorial’s name.

37. As part of adding a new tutorial, the system allows the teacher to decide whether the tutorial is to be active/inactive.

38. The site allows teachers to erase a tutorial.

39. The site allows teachers to edit a tutorial’s configuration.

40. As part of editing their tutorial’s configuration, the system allows the teacher to change the tutorial’s name.

41. As part of editing their tutorial’s configuration, the system allows the teacher to change the tutorial’s activity status.

42. The site allows teachers to edit a tutorial’s list of lessons.

43. As part of editing the tutorial’s list of lessons, the system allows teachers to add a new lesson.

44. As part of adding a new lesson, the system allows the user to enter the lesson’s text.

45. As part of adding a new lesson, the system allows the teacher to decide on the lesson’s position within the tutorial.

46. As part of adding a new lesson, the system allows the teacher to decide to clear all former selections from the town.
47. As part of adding a new lesson, the system allows the teacher to decide whether the lesson would block the learner’s advancement in the tutorial until a task is solved.

48. As part of adding a new lesson, the system allows the teacher to request that the tutorial advance automatically when a certain selection from the town area has been performed by the learner.

49. As part of adding a new lesson, the system allows the teacher to request that the tutorial advance automatically when a certain task (query) has been solved by the learner.

50. The SQL engine must check in the background if a learner executes the query the teacher would want him to execute in order for the tutorial to proceed to the next lesson.

51. As part of adding a new lesson, the system allows the teacher to request that certain SQL keywords be highlighted for the user during the display of that lesson.

52. As part of editing the tutorial’s list of lessons, the system allows teachers to edit a previously added lesson.

53. As part of editing the lesson, the system allows the user to amend the lesson’s text.

54. As part of editing the lesson, the system allows the teacher to change the lesson’s position within the tutorial.

55. As part of editing the lesson, the system allows the teacher to change his former option about clearing all former selections from the town.

56. As part of editing the lesson, the system allows the teacher to change his former option about whether the lesson would block the learner’s advancement in the tutorial until a task is solved.

57. As part of editing a lesson, the system allows the teacher to change his former option about whether to request that the tutorial advance automatically when a certain selection from the town area has been performed by the learner.

58. As part of editing the lesson, the system allows the teacher to change his former option on whether the tutorial advance automatically when a certain task (query) has been solved by the learner.

59. As part of editing the lesson, the system allows the teacher to change his former option about requesting that certain SQL keywords be highlighted for the user during the display of that lesson.

60. As part of editing the tutorial’s list of lessons, the system allows teachers to erase a previously added lesson.

61. As part of editing the tutorial’s list of lessons, the system allows teachers to move up or down within the tutorial a certain lesson.
3.2.2.4 Non-functional Requirements (Volere [61])

The following is the shortened list of non-functional requirements. The extended list, contained in Appendix A, contains the same contents as explained for functional requirements, only without the corresponding use-cases, as non-functional requirements are not system functions a user can have on the system represented in use-cases as for functional requirements.

1) Look and Feel Requirements (Volere [61])

62. The interface to the system is attractive to a teenage audience.

63. The interface to the system is friendly and colourful.

64. The interface to the system is interactive.

2) Usability and Humanity Requirements (Volere [61])

65. The learning part of the site is easy to use by teenagers.

66. The tutorial configuration part of the site is easy to use by adults.

67. The site should be usable by people with no former training on it.

68. The tutorial provides a clear navigation through the lessons within.

69. The default tutorial contains clear lessons about SQL basics.

70. The tutorial offers a means of highlighting the SQL keywords which are taught within the lesson the user is currently reading.

71. The tutorial offers a means of highlighting the SQL keywords which are required to be used by the user in order to solve a task.

72. The training tasks conform to what is taught in the default tutorial and what could maximally be covered by the teacher-configured tutorials.

73. The training tasks are clear and easy to understand by a beginner.

74. The training offers a means of highlighting the SQL keywords which are required to be used by the user in order to solve a task.

75. The system provides clear feedback on whether the user has solved a task from the tutorial or training.

76. The system provides clear error messages if a user has entered an incorrect query.
77. The system provides an easily accessible help facility.

78. The system provides a syntax remainder facility.

79. The system provides clear confirmation messages to the actions undertaken by the teacher.

80. The site provides clear navigation throughout its pages by using clear link labels and buttons.

81. The site provides a clear message about its purpose and the functions it supports.

82. The site provides clear error messages when a user is dealing with his account and entering incorrect or incomplete information.

3) Performance Requirements (Volere [61])

83. The system has an acceptable response time.

84. The system does not require many hardware resources.

85. The system supports at least 40 users using it at the same time.

4) Maintainability Requirements (Volere [61])

86. The system runs on any platform.

87. The system runs on any web browser.

5) Security Requirements (Volere [61])

88. The system protects user account information from unauthorized users.

89. The system allows only logged in the users to access the learning environment.

90. The system allows only logged in teachers to access the tutorial configuration area.

91. The system does not allow queries other than SELECT queries to be executed.

92. The system does not show the real names of the tables and their contents within the site.

93. The system does not allow the user to enter symbols which can interfere with the database in the URL.

3.2.2.5 Open Issues (Volere [61])

It remains an issue whether:
1. The teachers will want to build their own queries to be tested with the learner in training.

2. The teachers will want to use the tutorial only as an additional tool to the class or as a complete tool for learning SQL (which could be used to learn it even from home).

3. The teachers will need extra functions such as seeing reports on their pupils’ performance in the class or even give their students homework or tests.

4. A potential high school using this system would like to keep it only available within its intranet or be widely available to be used by the learners from home as well.

5. A potential high school using this system would not want for the system to offer access rights to its teachers automatically without requiring them to log in.

3.2.2.6 Evaluation Requirements

By the categorization of the requirements and the highlighting of the fit criterion for each (as can be seen from Appendix A), Volere ([61]) allows for designers to use the template as a checklist after the design respecting the requirements is implemented. To this end, for evaluating the accomplishment of functional requirements, we will apply Software Testing techniques: entering all the likely inputs and checking all the results with the expected outcomes. In what concerns the testing of non-functional requirements, look and feel requirements and usability and humanity requirements will be checked as part of the end evaluation with users, the results of which will be provided in Chapter 6. Performance requirements will be checked by testing the system on computers with different hardware capabilities. Maintainability requirements will be checked by trying the system on different platforms, different browsers and with a large number of users. Security requirements will be checked by trying to break the security of the system in the areas the fulfilment of requirements should provide this from happening.

3.3 Analysis

3.3.1 Use-cases

Use cases give context to functional requirements by showing in what way these requirements will be considered by the functionality of the system. The use-cases correspond to the functional requirements from the Volere template ([61]), the references to them from the functional requirements being highlighted in the specification from Appendix A. The use-cases were designed by using ArgoUML v0.28.1, a freeware tool.

All the use-cases could have been represented on the same figure, but this would have got much cluttered so we decided to break them into 3 figures: Fig. 1, Fig. 2 and Fig. 3. As already mentioned, the users of the system can be either learners or teachers. Moreover, the system also interfaces an SQL engine in order to require it to compute user or teacher queries (more details in the next subchapter). The actors from the use-
cases can therefore be a learner, a teacher or the SQL engine. The two kinds of actors corresponding to users have access to the same functionalities from the learning area, and the teachers also have access to the functionalities from the tutorial configuration area. This is represented in Fig. 1 in the form of a generalization relationship, where the “User” actor is the generalization of “Learner” and “Teacher” to represent the functionalities that the two actors have in common, while the extra functionalities attributed to the “Teacher” actor as use-cases to which he is the primary actor show the functionalities to which only the teacher has access. The system boundary, representing the functionalities which are computed within the actual system, is separated from the functionalities which take place as part of the SQL engine, thus not implemented by us.

3.3.2 Use-case Descriptions

The use case descriptions are based on Alistair Cockburn’s use case template, considered as good practice in Software Engineering [62]. The tables contain explanations about the following:

- The purpose of each use-case (“Goal in context”);
- The preconditions which normally involve “include” relationships to other use-cases;
- The success end condition: “the state of the world upon successful completion” [62]
- The failed end condition: “the state of the world if goal abandoned” [62]
- Primary actor: the actor who is doing the use case
- Secondary actors: other actors who are involved in the realization of the use case
- Trigger: the action which starts the use case
- Description: step-by-step outline of the normal actions undertaken to do the use case
- Variations: actions which can occur differently from the steps in the description
- Extensions: cases which extend the steps in the description

The most important use-cases, which also encompass most of the important functionality, have been underlined in red in the figures Fig. 1, Fig. 2 and Fig. 3. The following are the resulted descriptions for these use-cases. For the whole list of descriptions for all the use-cases see Appendix B.
Fig. 1 - Use-cases (1)
### Use Case 4: Interact with town

**Goal in Context:** This use case allows users to interact with the elements in the town.

**Preconditions:** Users have logged in.

**Success End Condition:** The user successfully manages to interact with the elements in the town.

**Failed End Condition:** -

**Primary Actor:** User

**Secondary Actors:**

<table>
<thead>
<tr>
<th>Trigger Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User moves on map</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>User chooses elements</td>
</tr>
</tbody>
</table>

**Variations or Extensions:**

### Use Case 6: Choose elements

**Goal in Context:** This use case allows users to generate an SQL query automatically by choosing the elements from the town they want to appear within.

**Preconditions:** Users have logged in.

**Success End Condition:** The user successfully generates the SQL query automatically by choosing elements from the town.

**Failed End Condition:** -

**Primary Actor:** User

**Secondary Actors:**

<table>
<thead>
<tr>
<th>Trigger Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User chooses elements</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>User gets query as automatically generated</td>
</tr>
</tbody>
</table>

**Variations or Extensions:**

<table>
<thead>
<tr>
<th>Extensions</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.1</td>
<td>User makes selections</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>User makes deselections</td>
</tr>
</tbody>
</table>

### Use Case 17: Compute query
<table>
<thead>
<tr>
<th>Goal in Context</th>
<th>This use case allows users to retrieve the results of an executed query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>Users are logged in and they have produced the text of the query (either by manually writing it or automatically by choosing elements)</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The user successfully retrieves the results of the query</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>User</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td>SQL engine</td>
</tr>
<tr>
<td>Trigger</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Variations or Extensions</td>
<td>Variation</td>
</tr>
<tr>
<td></td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>2.1</td>
</tr>
</tbody>
</table>

Use Case 18  
**Execute user query**

<table>
<thead>
<tr>
<th>Goal in Context</th>
<th>This use case allows the SQL engine to compute the user’s query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>The user is logged in and has produced the text of the query.</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The SQL engine successfully computes the user’s query</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>SQL engine</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Variations or</td>
<td></td>
</tr>
</tbody>
</table>
### Use Case 19: Give results to user query

**Goal in Context**
This use case allows the SQL engine to return the results of the user’s query to the system.

**Preconditions**
The SQL engine has computed the query.

**Success End Condition**
The SQL engine successfully returns the results of the query to the system.

**Failed End Condition**
- 

**Primary Actor**
SQL Engine

**Secondary Actors**

**Trigger**

**Description**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The SQL engine returns the results of the query to the system</td>
</tr>
<tr>
<td>2</td>
<td>The system hides the real table names for the user not to see</td>
</tr>
</tbody>
</table>

### Variations or Extensions

Use Case 22: Do tutorial

**Goal in Context**
This use case allows the user to do a tutorial for learning the basics of SQL.

**Preconditions**
The user has logged in.

**Success End Condition**
The user successfully uses the tutorial.

**Failed End Condition**
-

**Primary Actor**
User

**Secondary Actors**
SQL Engine

**Trigger**
The user chooses to do a tutorial

**Description**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user chooses to do a tutorial</td>
</tr>
<tr>
<td>2</td>
<td>The user interacts with the town</td>
</tr>
<tr>
<td>3</td>
<td>The user moves between lessons</td>
</tr>
<tr>
<td>4</td>
<td>The user computes queries</td>
</tr>
</tbody>
</table>

### Variations or Extensions
### Use Case 23: Move between lessons

**Goal in Context**
This use case allows the user to move through the lessons of the tutorial.

**Preconditions**
The user has logged in and is doing a tutorial.

**Success End Condition**
The user successfully moves through the tutorial’s list of lessons.

**Failed End Condition**
-

**Primary Actor**
User

**Secondary Actors**

**Trigger**

**Description**
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user moves through the lessons of the tutorial</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

**Extensions**
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user moves forwards freely</td>
</tr>
<tr>
<td>2</td>
<td>The user moves backwards freely</td>
</tr>
<tr>
<td>3</td>
<td>The user moves forwards automatically</td>
</tr>
</tbody>
</table>

### Use Case 26: Move forwards automatically

**Goal in Context**
This use case allows the user to move automatically to the next lesson in the tutorial when solving the task required by the teacher for the current lesson.

**Preconditions**
The user has logged in, is doing a tutorial.

**Success End Condition**
The user successfully moves forwards to the next lesson at the completion of the task required by the teacher for the current lesson.

**Failed End Condition**
-

**Primary Actor**
User

**Secondary Actors**
Teacher, SQL Engine

**Trigger**
The user computed a query.

**Description**
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The result of the user’s query as computed by the SQL engine is checked to have the same format like the query requested by the teacher.</td>
</tr>
<tr>
<td>2</td>
<td>The system moves the user to the next lesson automatically</td>
</tr>
</tbody>
</table>
Variations or Extensions

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>The result of the user’s query as computed by the SQL engine but the format of the query is not equal to the type of query required by the teacher and nothing happens</td>
</tr>
</tbody>
</table>

Use Case 27  
Do training

Goal in Context  
This use case allows the user to get trained for learning the basics of SQL.

Preconditions  
The user has logged in

Success End Condition  
The user successfully uses the training

Failed End Condition  
-

Primary Actor  
User, SQL Engine

Secondary Actors

Trigger  
The user chooses to do training

Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user chooses to do training</td>
</tr>
<tr>
<td>2</td>
<td>The user interacts with the town</td>
</tr>
<tr>
<td>3</td>
<td>The user computes queries</td>
</tr>
<tr>
<td>4</td>
<td>The system compares the result of the user query with the result of the task query</td>
</tr>
<tr>
<td>5</td>
<td>The system give user a feedback on the outcomes of his query</td>
</tr>
<tr>
<td>6</td>
<td>The user moves to next task</td>
</tr>
</tbody>
</table>

Varia lions or Extensions

Use Case 29  
Add tutorial

Goal in Context  
This use case allows teachers to add a tutorial for the teaching of the basics of SQL to their pupils

Preconditions  
The teacher is logged in

Success End Condition  
The teacher successfully adds a tutorial

Failed End Condition  
-

Primary Actor  
Teacher

Secondary Actors
<table>
<thead>
<tr>
<th>Trigger</th>
<th>The teacher chooses to add a new tutorial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td>Variations or Extensions</td>
<td>Variation</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Use Case 33  
Add lesson

<table>
<thead>
<tr>
<th>Goal in Context</th>
<th>This use case allows teachers to create a new lesson as part of a tutorial</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preconditions</td>
<td>The teacher is logged in and has previously added the tutorial he now wants to add a lesson to</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The teacher successfully adds a new lesson to the tutorial</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>Teacher</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>The teacher chooses to add a new lesson to the tutorial</td>
</tr>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
</tbody>
</table>

33
A lesson is triggered by the user’s action of executing a certain task.

7. The teacher decides whether the lesson will be accompanied by the blocking of the advancement to the next lesson until the user will have solved a certain task.

8. The system checks that the user has entered the lesson’s title and position.

9. The system confirms that the new lesson was successfully added to the tutorial.

### Variations or Extensions

<table>
<thead>
<tr>
<th>Variation</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td></td>
<td>If the system discovers the teacher has not entered the lesson’s title and/or position, it prompts the user about this and does not add the lesson.</td>
</tr>
</tbody>
</table>

### Use Case 35: Erase lesson

**Goal in Context**
This use case allows teachers to erase a previously added lesson from a tutorial.

**Preconditions**
The teacher is logged in, has previously added a tutorial and a lesson for it he now wants to erase.

**Success End Condition**
The teacher successfully erases the lesson.

**Failed End Condition**
-

**Primary Actor**
Teacher

**Secondary Actors**

**Trigger**
The teacher chooses to erase a previously added lesson.

**Description**
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The teacher chooses the lesson.</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses to erase the lesson.</td>
</tr>
<tr>
<td>3</td>
<td>The system confirms with the teacher the deletion.</td>
</tr>
<tr>
<td>4</td>
<td>The system confirms that the lesson was erased.</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

<table>
<thead>
<tr>
<th>Variation</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td></td>
<td>If the teacher refuses the lesson deletion, the lesson is not deleted.</td>
</tr>
</tbody>
</table>
CHAPTER 4. SYSTEM AND SOFTWARE DESIGN

In the last chapter we have seen which are the main functional and non-functional requirements for the site and showed how functional requirements would be supported by functionality and to what users by means of use-cases and their description. In this chapter we will describe how the functionality only conceptually sketched by use cases will be supported by design. This will first be described by briefly mentioning the technologies employed and for what functionality they will be used. We will then provide sequence, class and entity relationship diagrams to show the way the different parts of the system will work.

4.1 Choice of Technologies

For the project we will use different technologies to deal with the different parts of the courseware application:

- HTML for the presentation of the web site part of the application;
- JavaScript for the validation of forms used for registering, logging in users and allowing them to change their account details;
- PHP for the design of the dynamic content of the web site and for interfacing between the Flash part of the application and the MySQL database;
- We will use the Template Engine Smarty in order to split content and application logic from the presentation. All the pages will thus be divided into a php file and a tpl file. The php file will contain all the computation (forms, calls to external files and the database and other types of computation on the displayed information). The tpl file will deal with the presentation (fonts, colours, styles) and client-side methods and calls (JavaScript, Flash). This structuring will divide the server-side computation (dealt with by the php file, which will handle computation on the server and send selected results to be displayed to the tpl file) and the client-side computation (dealt with by the tpl file, which handles the display of the information in a html format on the client). Smarty thus allows for a separate and thus easier management and maintenance of computational code and the information presentation;
- Flash will be used for the design of the learning environment- the game-like interface. It will communicate with the MySQL database by using a set of specifically made PHP files which return the information in an accepted MySQL communication format;
- MySQL for the storage of the information regarding the general management of the web site (the pages, the user accounts) and the management of the learning environment (tutorials, lessons, tables used for the teaching of SQL).

We will use an Apache server which will host the web site, allow for it to be run on the localhost for aiding in the development and when the web site is finalized allow for it to run on-line.
4.2 Choice of Tools and Hardware

We will use Notepad++ v5.3.1 for creating the code for HTML, PHP and JavaScript. This tool is available under the terms of the GNU General Public License, it is compact and very easy to use, recognizing and highlighting PHP, HTML and JavaScript keywords so that the development is made easier.

We will use a trial version of Adobe Flash CS4 Professional Version 10.0 because it is part of an Adobe development suite often used for developing web sites: Adobe Web Premium CS4. The tool offers both graphical elements for the quick taking of design decisions and also the possibility of very easily adding action script code as attached to the graphical elements.

We will also use a trial version of Adobe PhotoShop CS4 Extended Version 11.0 for the editing of graphical data.

In what concerns the server, we will use a WampServer version 2.0, which is an all-in-one solution for hosting a web site. It supports by default, without the need for configuration or updates, PHP files and MySQL databases, while also offering useful tools such as phpMyAdmin which designers can use to graphically manipulate the database.

We will develop the web site on a Windows XP platform. The computer on which the development will be made will be the same with the hosting server and its hardware characteristics are the following: Intel Core 2 Duo E7200 processor with 2.53 GHz and 3GB of DDR2 modules RAM, videocard NVIDIA GeForce 9800 GT.

4.3 The Database

The ER diagrams for the site have been developed using Adobe Photoshop CS4 Extended Version 11.0 and Microsoft Office Word 2007.

Fig. 4 shows a description of the database for the courseware application as given by an ER (entity-relationship) diagram. The “Users”, “Administrators”, “Quest” and “Pages” tables are used by the general management part of the application, while the other tables are used for the learning process. The following is the description of each table:

- The “users” table – contains all the information about user accounts, differentiating between a learner and a teacher account by means of the “rights” field. It contains the users’ identification and login in details. The “last_entrance” field retains when the user has last accessed the site so that this could be used in the future if an administrator wishes to check when an account has last been accessed. The “completed_tasks” field retains how many tasks a user has accomplished from the training area of the web site;

- The “administrators” table – is very similar to the “Users” table. It does not allow people logging in as administrators to use the learning part of the web site. The administrator’s part of the web site will be
proposed as future work (we have not even considered requirements regarding it). Our vision of it is that it will allow special users (Administrators) to erase the database, to erase user accounts (for example for learners who have finished their course), to promote a logged in user to the status of teacher so that he can have access to the tutorial configuration area:

- The “pages” table - stores data related to each page of the web site and has the attributes id_page (the primary key), page_title (the title that appears on the blue line of the browser for each page), page (the name of the page in the form that it is given in the path), address (the path to the page from the server) and rights (which decide if the user is a learner or a teacher so that it differentiates accesses to pages);

- The “quest” table – stores the user responses to questions we will receive as part of a questionnaire in the evaluation phase of the project;

- The “main_tables” table – stores information about the tables from the database used as part of the learning part of the web site with the purpose of graphically representing the game-like area of the screen;

- The “main_benchstore”, “main_chairstore” and “main_chairsuperstore” tables – are all related to the learning environment of the website, representing the tables which the user uses to learn and practice SQL;

- The “tutorials” table – stores information on the details of the tutorials available for the learning environment, as decided upon by the teacher in his tutorial configuration area: name, status (shows if the tutorial is available to the users or not), last_change (allowing for the administrator to see when a last change to the tutorial area was made by a teacher- functionality which will be proposed as future work);

- The “lessons” table – stores information on the details of the lessons within the tutorial available for the learning environment, as decided upon by the teacher in his tutorial configuration area: text, action (description of the tasks the teacher requires the user to do to automatically advance to the next lesson), disableNextButton (the teacher’s option about whether the button used for advancing in the tutorial will be blocked until a task is solved by a user or not), clean_screen (the teacher’s option about whether to clean the query area by deselecting all the objects which the user has previously selected from the game-like area at the start of the current lesson), order_position (the teacher’s option about the position of the lesson within the tutorial), keywords_shown (the teacher’s option about SQL keywords to be highlighted while the lessons is displayed), last_change (allowing for the administrator to see when a last change to a lesson was made by a teacher- functionality which will be proposed as future work).

In addition to the presented tables, we have also designed the table structure for allowing users to create and manage their own tables (by using CREATE, INSERT and UPDATE operations). Moreover, this table structure will allow for teachers to add their own tasks for the user as part of the training and also be able to check what kind of tasks the user has solved. These functionalities will be proposed as future work and
further discussed in Chapter 7. The representation and outline of the ER diagram for this additional structure is provided in Appendix C.

The code used for generating the database and the tables within is provided in Appendix D.

Fig. 4- ER diagram of the database
4.4 User Interface Design

For the graphical development of the site, we have used Photoshop as it is very easy to create and edit graphics, and after that automatically generate the images at the appropriate sizes for the website by using it.

The interface was developed as to be easy to look at, intuitive and attractive, as the intended users are first of all high school children and we must respect the look and feel and usability requirements outlined for them.

In what concerns the chosen colour, departing from the usability guidelines for children’s’ applications from the literature and the captured requirements, we have chosen the chromatic options for our application based on a combination of yellow, blue and green, trying to keep it as easy to look at and not tiring as possible.

In what concerns the graphics used, we have taken the graphical objects for the game-like learning environment provided under the US Creative Commons Attributions 3.0 License from LostGarden [63]. Everything else (buttons, menus, pages, alerts, icons, etc.) have been designed by us. A picture of the prototype showing how the graphical interface to the site will look like is provided in Fig. 5.

![Fig. 5 - The design of the graphical interface prototype of the site](image-url)
To avoid confusion, we have decided to use a very simple navigation for the learning environment which would not go more than one page far from the homepage to get to the registration, login, change account page, tutorial, training area, information about the site (About us), terms and conditions or the help. In what concerns the tutorial configuration area of the site, we decided the teacher could get to it directly from any page on the site (we already noted that teachers also have access to the learning environment as any other user). Within the configuration area, the navigation will follow the basic way of solving the tasks of managing tutorials or lessons.

4.5 Code Design

4.5.1 Sequence diagrams

We will next provide explanations about the way the different parts of the site will work by means of sequence diagrams. These diagrams will cover the functionalities from the learning area (tutorial, training, general learning) and tutorial configuration areas, as we considered that the registering, logging in and modifying a user account actions are well known (they work as usually) and do not need further explanations. The tool used for the design of the sequence diagrams was Microsoft Office Visio 2007. Please note that the first entity in each sequence diagrams is always an actor, which should have been represented by a stick figure, but this is not possible in Microsoft Office Visio 2007.

4.5.1.1 A user starts using the learning environment

Fig. 6 shows the sequence for supporting the user action of starting to use the learning environment, which involves the loading of all the graphical elements on the screen for the user to manipulate.

When the user first accesses the site to start learning SQL, the game-like area on the screen (the background, all the external variables received at the start of the Flash containing the SQL learning area, all the internal variables) initialize. Flash then requires from the PHP Flash scripts to get information on the tables from SQL to which the user has access in his learning. In order to send information in small parts and thus not overload the server, all of this information is asynchronous requested from the database by first querying for the table ids and then retrieving the details attached to each table: the name of the object representing the table, its image and where to place it. Flash saves the information on the tables. The computation does not continue until all the information on the tables was loaded. Flash then requires from the PHP Flash scripts to get from SQL information on the entities contained in each table (the rows of the table) and their details (all the columns of the table), also in an asynchronous way. The computation does not continue until all the information on the entities within the tables was loaded. The list of all the available tutorials (those that have not been set as hidden by the teacher in his tutorial configuration area) is then loaded by retrieving them from SQL from the tutorial table in the same asynchronous manner via the PHP Flash scripts. The computation does not continue until all the information on the tutorials was loaded. Once all of this information is loaded, the graphics is activated for the user. The user can then start a tutorial, case in which the lessons from that tutorial and their details are retrieved from the SQL database
through the PHP Flash scripts in a similar way to what was described before. The user can then start the tutorial, which is described in the next subchapter. Once he finishes it, he can leave the learning environment.

Fig. 6 - Sequence diagram 1: the user starts to use the learning area
4.5.1.2 The user does a tutorial

Fig. 6 represents the sequence diagram describing how the fact that a user does a tutorial is supported by the system. First of all the fact that the user chooses to start a tutorial makes the system apply the tutorial settings. Then, how long there are lessons left, if the lesson was not blocked as set up by the teacher, the user can advance to the next lesson in the tutorial, case in which the system returns to the user the lesson details it had loaded upon the user entering the learning environment, as we have seen from Fig. 7. If the teacher has selected that the lesson is to clean the screen upon loading, all the previously selected objects from the screen are deselected. If the teacher has chosen that the lesson display will also trigger the highlighting of some SQL keywords, the system will make these keywords be highlighted. If the teacher has chosen that the “Next” button is to be blocked upon the display of the lesson until the user will have solved a certain task, the “Next” button’s status becomes “blocked”.

If the lesson has a task attached, the system retrieves the task required by the teacher and its type (it can be either a query task or the task to select something from the screen), then retrieves the format of the task required by the teacher, which can be either the click on a certain object within the screen or the format of an actual query. While the user has not completed the task, if the task is a query and he sends the query, the format of this query is compared against the format the teacher required for the query. If the formats are different, the user receives a feedback message about his query not being complete. Else, he receives a feedback message about the task being complete and, if the “Next” button’s status was blocked, it becomes normal, taking the user to the next lesson automatically. If the query was not graphical, the user will be taken automatically to the next lesson as soon as he clicks on the correct graphic element. Until he solves the task, the user can move around the map and select graphic elements.

If the lesson did not have a task attached, the user can freely move around the map and select graphical elements and will be allowed to click on the next button once he is done.

4.5.2 The user selects graphical objects from the game-like area

The sequence for supporting the action of making selections is presented in Fig. 8. When a user selects a store, if the wanted store was already open, the store is closed (the system sees the second click as a deselection) and the store’s status gets from “selected” to “normal”, thus allowing for another selection. In this case, the store details (objects within the store) are closed and the store does not graphically appear as selected any more (the highlighting is taken out). The query for the selected store is also cleared from the database.

If the store was not already open, if other stores were already selected, they are closed, the highlighting is taken out and the query is cleared for them so that only the current store is to be opened. The newly opened store then becomes the target position which is brought to the centre of the user’s view while also repainting the background and the stores back on the map. The newly selected store is opened, its status becomes “selected”, the store is highlighted and its details (the objects within) are showed to the user.
Fig. 7- Sequence diagram 2: The user does a tutorial
The user can then continue his selections by selecting objects or fields within the store. The sequence diagrams for these actions are represented in Appendix E.

Fig. 8 - Sequence diagram 3: The user selects graphical objects from the game-like area
4.5.3 The user does training

The functionality behind the user practicing his SQL skills in the training area is represented in the sequence diagram from Fig. 9. While the user is in training, the system automatically generates tasks for him. This is done by first choosing a random type of query from the existing ones (SELECT * FROM, SELECT field FROM, SELECT fields FROM, SELECT * FROM WHERE, SELECT field FROM WHERE, SELECT * FROM ORDER BY, etc.), then choosing random values for the task (for the conditions in the WHERE clause, the tables to do the query on, the fields, etc.), highlighting the keywords the task would require the user to use, showing the task’s text for the user. It also involves the computation of the correct results expected from the user’s query by using the PHP Flash scripts which query the SQL database for them. Please note that this computation first transforms the names of the tables used in the user’s query to the real names of the tables from the database and back into the names of the tables as they are visible to the user for security reasons, as already noted in the security requirements. After this initialization is computed, while the task is not completed and the user tries out queries, each of these queries is compared with the required query format and if it does not conform to it the user is sent an error message. If the user query conforms to the format, it is executed by the SQL engine via the PHP Flash scripts (involving, again, the transformation of the tables referred to by the user to the real table names in the database and back). If the SQL engine returned a problem with the query, the SQL engine’s error message is returned to the user. If the query does not have problems, the user query is compared with the task query’s result. If they are the same, the user is sent a confirmation message about the query being correct, otherwise he receives a message about the query not solving the required task.

4.5.4 The teacher configures a tutorial

The sequence diagram showing how the system supports the teacher in configuring a tutorial is given in Fig. 10. This sequence diagram shows all the options the teacher has for configuring a tutorial. If the user wants to add a tutorial, he is returned a form where he can add the tutorial’s text and status and upon submission of this form the system will add the tutorial to the database and send him a confirmation message about the tutorial being added. If the user wants to edit the tutorial’s configuration, he is returned by the system the details which he had already filled in. The user can then change the tutorial’s text and/or the tutorial’s status and, upon submitting the changes, the system saves the changes in the database and returns him a confirmation message. If the user wants to change the tutorial’s status directly from the list of tutorials, he can do this as well, then the system saves the change in the database and returns a confirmation message. If the user wants to remove a tutorial, the system requires confirmation and, if the user confirms, the system deletes the tutorial from the database and returns a confirmation that the tutorial was deleted. If the user cancels the confirmation, the tutorial will not be removed. If the user wants to edit the lessons the tutorial contains, this takes us to the next sequence diagram.
4.5.5 The teacher configures a lesson from within the tutorial

The sequence diagram showing how the system supports the teacher in configuring a lesson is given in Fig. 11. We presumed here that the teacher has already created a tutorial. He is now entering the lesson configuration page by having requested that the tutorial’s lessons be edited. He can add a new lesson, case in which he is given a form where he can fill in the lesson’s text, decide on a position for the lesson within the tutorial, optionally choose that the screen be cleaned of previous selections, optionally choose a task for the lesson (from within a list of query types and selection types), optionally choose that the
advancement to a new lesson be blocked for the user until he accomplishes the task, optionally choose that certain keywords be highlighted for the user together with the lesson’s display. Once the teacher submits the form, the lesson is added to the database and he receives a confirmation. If the teacher wants to edit the lesson, the lesson’s details are returned to him and he can then edit each of them. Once he submits the changes, they are also changed in the database and the teacher receives a confirmation message. The teacher can also change the position of the lesson within the list of lessons, for which he receives confirmation. He can also decide to remove the lesson, case in which he is asked for confirmation. If he confirms, the lesson is removed from the database and he receives a confirmation. In case he answers no, the lesson is not removed.

Fig. 10 - Sequence diagram 5: the teacher configures a tutorial
Fig. 11 - Sequence diagram 6: The teacher configures a lesson from the tutorial
4.6 Class Diagrams

The development environment provided by Flash offers different ways to design an application, each one having its advantages and disadvantages. You can design everything just graphically using the design interface with the timeline and scenes, you can use procedural functions and you can also use object-oriented programming. We only used object-oriented programming for the functionality to which it was a good solution, for the rest the other two ways of using Flash being more useful. These functionalities were the storage of information obtained from the database, which it was a good idea to place in classes and be able to manage together by using methods, and the creation of queries as a result of the user selecting objects from the learning environment. We only used public classes, methods and variables as the code was created by just one person and the code is compiled by Flash into a single compiled file by default., therefore it would not have been necessary to hide code from one file to another by using private or protected classes, methods or variables. The class diagram is provided in Fig. 12. The following is the description of the classes:

- **The “Game Manager” class** – manages all the other classes by containing arrays of objects from GameObject and SelectTable. It also contains two arrays of the Flash type Object which store information about the tutorials and the lessons for the currently selected tutorial. The class also has a variable of the Flash type Number named “activeTutorial” which shows which tutorial is currently active. The class contains methods for loading the information on tables, houses (stores), tutorials and lessons from the database, which synchronically call the specifically created PHP files to first get the ids of the elements, and then their details, as it was shown in the sequence diagram from Fig. 5. To control the loading of the elements from the database for tables, houses and lessons and make sure when they have been all loaded so that the Flash could continue to the next step in the computation (we have already noted for the sequence diagram in Fig. 5 that the loading of one type of graphical object does not start until the ones before it have all loaded in turn), we use the methods whose name starts with “getNoLoaded”, which return the number of elements already loaded in Flash. The loading methods know the total number of objects which need to be loaded by counting the ids for those objects retrieved from the database and they check this number with the one returned by the “getNoLoaded”-type methods to know when to proceed with the computation. The tutorial was not the case, as it cannot have so much information as to overload the server, therefore we load it without need of management. All the other methods have to do with the management of the object arrays: empty the array, find an object from the array, and add an object to the array. The exception is the “setCurrentLesson” method which receives as parameter the direction of reading the lessons in the tutorial and computes all the operations for the display of the next lesson: cleaning the screen if the teacher has required it, blocking the “Next” button if the teacher has required it or if the last lesson was reached, etc.

- **The “GameObject” class** – saves the information on the tables (stores) from the MainTables table we have seen in the ER diagram, containing variables representing the attributes from the table. It contains the “objStatus” variable showing whether the table (store) was selected or not and stores the fields of the actual tables (main_benchstore, main_chairstore, main_chairsuperstore) and the list of rows (objects) from
each of these tables into arrays. Most of its methods deal with the management of the rows/columns (objects/fields) in a table (store): they add new ones or remove them, find out their position in the array, find out the size of the arrays, check whether an object/field exists in the array. The class also contains a display method which paints graphical elements on the screen, “paintObjects”, and methods which change the object’s selection status to “normal” when it is not selected, and to “selected” when it is selected: the “changeStatus” method (for changing the status of the current store only) and “changeAllStatus” method (for changing the status of the current store and of all of its objects).

- **The “HouseObject” class** – contains variables saving the graphical details for dynamically representing the objects within each store on the screen: the name, the type, the URL from which the external image is loaded to represent the object, the position, the width and the height, the priority for displaying the object on a different layer on the screen. It also contains the “objStatus” variable which shows the status of an object (as presented before) and the “objDetails” array saving in name-value pairs the columns from the table to be displayed as details for the object. The class contains methods for setting the variables for graphical details and for the management of the “objDetails” array (“addObjectDetail”, “findObjectDetailPos”, “findObjectKey”, “removeObjectDetails”). In addition to these methods we have a method for changing the status of the current object and methods for the display and clearing of the graphical representation of the object details.

- **The “SelectTable” class** – is used for the creation of SELECT queries and contains the information which appears in the query regarding a table: the table name and the list of fields, the conditions on the fields of that table. It contains methods for the management of the fields and the conditions: adding fields/conditions, getting the number of fields/conditions, finding a specific field/condition, removing a certain field/condition.

- **The “SelectCondition” class** – stores a condition appearing in the WHERE clause of a query and all of its elements. It contains 5 variables: “fstElement”, which can be either a table’s field or a value and represents the first element of the condition, “fstTable”, representing the table whose field appears in the condition if the “fstElement” is a table’s field, similar for “sndElement” and “sndTable” and “relation” which is the logical relation to be used in the condition. Apart from the constructor, its methods check if a required condition is the current one in order to avoid duplicates (“findCondition”), return the work table for the current condition (“getWorkTable”- for example, if a condition refers to a field and a value the work table is the field’s table) and graphically display the condition (“toString”).

50
Fig. 12- The class diagram
CHAPTER 5- IMPLEMENTATION

In the last chapter we have decided on the technologies and tools used for the development and implementation and outlined the design of the database and object-oriented part of the project by means of entity-relationship diagrams and class diagrams. We have also shown how general functionality will be supported by the courseware by means of sequence diagrams. In this chapter we will describe how the implementation was carried out and what the results were. For the latter we will provide screen shots with clear explanations of how the code behind the images was implemented and how it works. We will also make an outline of the testing strategy used at the end of the implementation.

5.1 Implementation Strategy

For the implementation of the courseware application, we used iterative and incremental development. It was iterative in the since that we carried it out by developing the system in prototyping stages, each involving the following main phases of development: requirements, analysis, design, implementation, evaluation. At the same time, it was incremental as each new prototype involved the development of a new functionality. Each time new requirements were captured or refined by the help of our supervisor, our fellow colleagues and a few high school pupils. The analysis normally involved the extension or modification of the use-cases to cater for the new requirements. The new or modified use-cases were then considered within the design stage for the design of the functionality which would support their execution. The implementation added to the partly made prototype the new functionality and carried out new tests applying both to the added functionality as well as to how the whole system works as a result of its addition. The evaluation was only performed with our supervisor and with our fellow colleagues out of lack of time of also contacting the primary target user group- high school pupils, for this purpose. Evaluation with high school pupils was only undertaken during the last evaluation cycle.

5.2 The Final Application

Our website is called SQL Town. It runs on any browser and any operating system. It is available on-line at the following URL: http://94.7.212.252/SQLTown/

This sub-chapter provides only the most important screen shots of the application’s pages. Additional screenshots can be found in Appendix F.

5.2.1 The Website Part of the Application

5.2.1.1 Outline

The website was designed in PHP, combined with a MySQL database. As we also described in the previous chapter, we used the Template Engine Smarty to aid in the development and management of the content. In order to fully benefit from all the possibilities offered by Smarty, we have used the
“index.php” file for doing all the general computation from the website, while using the sets of .php and .tpl files for keeping individual information for the website’s pages.

The site was designed by using the XHTML standards and it is XHTML compliant.

The “index.php” file contains all the initialisations for the Smarty engine and loads all the necessary external files, for example the “settings” file, which contains the configurations for the website (the URL in which it is saved, the addresses for the image folders for increasing the application’s flexibility and the connection instructions for connecting to the database). “index.php” also checks the users’ access statuses in order to allow for the access to different pages according to the rights attached to each. The file also handles the logout functions if called with the parameter “logout=-1”. The most important function supported by “index.php” is though that of loading pages, which is accomplished by calling the file with the name of the page as the “page” parameter. When “page” is received as a parameter, information on the page (such as the page’s title, address and the required rights for access) is searched in the database. The .tpl and .php files corresponding to the page are then loaded. All the page access information is retrieved from the database, even for the first page displayed by default, as “index.php” displays the homepage when called without a parameter for the first time. Structuring the website in this way with “index.php” the only called file leads to a centralized approach, but also offers us more security as all the pages’ real names apart from “index.php” are hidden from the user’s view. In this way, the real files cannot be accessed.

In the “tpl” file for each requested page, other pages are called: “header.tpl” and “footer.tpl”. The logic behind this action is that of structuring the information into easily manageable chunks and of not repeating the information, the header and footer mainly remaining the same for all the pages.

The following is the code for the “index.php” file which is used for the loading of the pages:

```php
$sql = "SELECT address, page_title FROM pages WHERE page = ".$page.";
$res_page=mysql_query($sql);
$row_page=mysql_fetch_array($res_page);
$module=$row_page['address'];
$layout->template_dir = $module;
//send logged state into template for all pages
$layout->assign('logged',$_SESSION['logged']);
if($_SESSION['logged']==1) {
    $layout->assign("username", $_SESSION['username']);
    $layout->assign("user_id", $_SESSION['user_id']);
    $layout->assign("rights", $_SESSION['rights']);
}
$layout->assign('page'.$page);
$layout->assign('page_title',$row_page['page_title']);
include($module.$page.'.php');
$layout->display($page.'.tpl');
```

As the website currently supports two types of users, the learner and the teacher, we will present two site maps, one for each case. The site map for the learner type user is presented in Fig. 12 and that for the teacher type user is presented in Fig. 13.
5.2.1.2 The homepage

The homepage contains information which will appear in each of the site’s pages, such as the head with the SQL Town logo, the help, the main menu and the footer with the author’s name and the terms and condition link. The logo is a link which can always be clicked on to get back to the homepage. The help is a link to a help facility which contains adapted information for each page. The help is opened in a pop-up window. The main menu contains the “home” button which always allows one to get back to the homepage from anywhere within the site, the “About” button which leads to the “About us” page, the “Learn” button leading to the learning facility, the “Registration” button leading to the “Registration” page from within which a user can create an account and the “Login” button leading to the “Login” page for users who already have an account. After a user has logged in, “Registration” transforms to “My account” from which the user can modify his account details and “Login” turns into “Logout”. While a user is not logged in, the “Learn” button is blurred and the user cannot thus access the learning environment. The “Terms and Conditions” link leads to a new page opened in a separate tab informing the user on his rights within the website. If the user is logged in as a teacher, a special rounded button appears for him on the top of the screen, leading him to the tutorial configuration area of the site. This link causes the difference in the site map from the user’s and the teacher’s view of the site. The header and footer will be the same for the following pages we will describe, so we will not repeat this explanation.

The centre of the homepage contains a short description of the website’s purpose. Once a user is logged in, a shortcut appears to lead him to the learning environment. Fig. 14 represents the homepage.
5.2.1.3 The “About us” page

As information, in the centre of the page, the “About us” page introduces us and the motivations behind SQL Town.

5.2.1.4 The “Registration” page

The “Registration” page contains a form which the user is required to fill in order to create an account on the website. The form requires the user to enter his first and last names, email address, and choose a username and password, the latter needing to be filled in twice for security reasons. The page contains validations to ensure that the user has filled in all the fields, entered a correctly formatted email and entered the same password twice. The validation is both client-side validation written in JavaScript and prompting the user when something has not been filled in correctly and server-side validation written directly in PHP which returns a red message indicating what was wrong when reloading the page. The user must also tick that he has read the terms and conditions before being allowed by the validations to submit the form. When the user has filled in everything correctly and ticked the “Terms and conditions” checkbox, his details are saved in the database and a confirmation page appears. At the same time, the functionality for sending an automatic email to the user was implemented, but we would have needed an email server for making it work, which was impossible given the project’s timeframe. From the confirmation page, the user can click on a “Proceed” button so that he is logged in automatically (without needing to go to the “Login” page).
5.2.1.5 The “My Account” Page

This page is only accessible once a user has logged in and allows the user to change his account details. The page contains the same form as the “Registration” page, but this form is this time filled in with the user’s account details decided on when creating the account. The only detail which is not provided for the user is the password, which he needs to fill in again from reasons of security. The user can edit any of the details except for the email address, which is used so that he is identified within the site. The user does not need to tick that he has read the terms and conditions again when changing the account details. He is though given the extra functionality of deleting his account by ticking on a checkbox, case in which he is asked for confirmation and the account is only deleted if he has confirmed this action. The page contains client and server-side validation similar to the ones from the “Registration” page. After having submitted the edited form, the user is always sent a confirmation. Additionally, we have also designed the functionality for the user to receive an automated email with his new details, but, again, this functionality was not implemented. Fig.15 shows how the account deletion confirmation appears in my account page.

![Change details form](image)

Fig. 15- The confirmation of account deletion in ‘my account’ page

5.2.1.6 The “Login” Page

The “Login” page requires the user to provide his email address and password in order to access his account. It is the page the users will need to use upon new accesses to the site (as for the first login they could have simply clicked on the “Proceed” button from the registration confirmation). The form contains both client-side, JavaScript validations, and server-side, PHP validations providing error messages which are adapted to each error type.

5.2.1.7 The “Learn” Page

The “Learn” page loads the Flash object which is responsible for the learning activities. The Flash object is loaded by using a set of JavaScript functions which allow its adequate running on any browser and also
test for the existence of Flash support. If the browser does not have a Flash player, a download link is indicated instead of our object.

When the Flash object is called from the .tpl file to load the learning environment, it receives from PHP two attributes: the name of the current user and the number of tasks he has completed before the current work session. Also in the call to the Flash file, a parameter called “activeJoins” is put with a default value of “No”, indicating if joins are to be allowed or not in the learning environment (if selecting more than one store at a time is permitted). The “activeJoins” parameter was added with the prospect of adding joins as future work. With the functionality of being adaptable to different display sizes and resolutions as future work in mind, we designed a functionality which adapts the Flash object to our desired size. Fig. 16 shows the initial state of the learn page.

Fig. 17- The initial state of the ‘Learn’ page

The following is the code for the “game.tpl” page which is the page which is responsible for the loading of the graphics:

```html
{include file="../default/header.tpl"}
<div style="clear:both; width: 900px; height: 580px; padding-left: 7px; padding-right: 7px; background-image: url(graphics/SmallWorld/images/images2_17.gif); background-repeat: repeat-y;">
  <!--main-->
  <div style="float:center; width:900px; padding-top: 8px;">
    <div style="float:center;">
      {literal}
      <script language="JavaScript" type="text/javascript">
        var hasRightVersion = DetectFlashVer(requiredMajorVersion, requiredMinorVersion, requiredRevision);
        if(hasRightVersion) { // if we've detected an acceptable version
          // embed the flash movie
          AC_FL_RunContent(  
            'codebase', 'http://download.macromedia.com/pub/shockwave/cabs/swflash.cab#version=10,0,2,0',  
            'width', '${gameAreaWidth}'  
          );
      }
      {literal}
    </div>
  </div>
</div>
```

57
The "SQL Town Tutorials" page

This page can only be accessed by teachers by using the “Tutorial configuration area” link from the top of the screen. It contains information regarding the tutorials which are currently available in the website and the available operations for managing them. The page contains the list of current tutorials and their statuses, ordered in the list according to their date of creation. Each tutorial has an associated set of actions represented by icons: edit tutorial, remove tutorial and tutorial configuration, the latter referring to the list of lessons from within the tutorial. A separate button, “Add new tutorial”, opens up a form where a new tutorial can be added, the user deciding on his name and status: the tutorial can be “active” if it is to be visible by the learners or “hidden” if it is available to the teacher, but not available to the learners. The form has server-side validations ensuring that the name has been filled in. If the status is not decided on, it is automatically considered as hidden.

From within the list of tutorials, the teacher can choose to edit the configuration of a tutorial. In this case, the details of the tutorial will open up in the same form as for adding a new tutorial and the teacher will then be able to amend them and save the changes.

Also from within the list of tutorials, the teacher can choose to remove a tutorial, case in which he is required confirmation and the tutorial is only removed if he confirms this action. Together with the tutorial, all the lessons contained within are removed. Fig. 17 shows a required confirmation for deletion.
For each of the above actions, the teacher receives a confirmation message.

The teacher can decide to hide/activate a tutorial, case in which he notices its status changing from active to hidden and vice versa.

If the teacher clicks directly on the tutorial’s name or on the icon representing the option of editing the tutorial, he is taken to the “Tutorial configuration” page.

![Fig. 18-Confirmation on deletion of a tutorial](image)

5.2.1.9 The “Tutorial Configuration” Page

This page allows the teacher to manage the lessons from within the tutorial. It is here that he sees the whole list of lessons. He can add a new lesson by using the “Add new lesson” button, case in which a form is opened up in the page where the teacher can add the lesson’s text, its position within the tutorial, whether the lesson would clean the game area, block the “Next” button, require a task or highlight SQL keywords. It is mandatory for the teacher to add the lesson’s text and position, the form containing server-side validation and otherwise returning a clear error message. The lesson’s text is constrained to not more than 270 characters, so if the teacher adds more than that the whole text of the lesson will not appear in the list. When the lesson is added, the teacher receives a confirmation message.

From within the list of lessons, the teacher can choose a set of actions represented by icons: to edit the lesson, remove the lesson or move the lesson up or down in the list. If the teacher chooses to edit the lesson, he is taken to the same form as for the “Add new lesson” facility, but this time filled in with the details and options for the lesson as already set up. He can then change the lesson’s text, position and options as wanted, provided that he keeps the text and position fields filled in. If he does not, the form returns an error message from the server-side validation. Fig. 18 shows how you can add a new lesson in the tutorial.

If the teacher chooses to remove the lesson, a prompt comes up which asks him for confirmation and only if the teacher confirms the lesson is deleted.

In case of both of the previous operations the teacher receives a confirmation message.

If the teacher wants to move the lesson up or down in the list, he can do this by using the arrow icons provided as actions attached to each lesson. He will immediately see the position of the lesson changed.
The “Tutorial Configuration” page also contains two links which can take the teacher back to the list of tutorials: one is a clear “Back to the list of tutorials” link and the other is the displayed name of the tutorial.

Fig. 19 - Add a new lesson in a tutorial

5.2.2 The Flash Component of the application

Flash is also part of the website, but for a clearer explanation we used this division into two parts. For a better description of the Flash component, we will present it into two parts: design and code.

5.2.2.1 The Flash Component’s Design

1) Overview

The design is divided into two graphical parts:

A. The top part contains all the learning area’s graphical elements: an island, trees, buildings named stores, as well as buttons with different functionalities which we will describe below:

1) The island - is designed by means of a set of graphical circles with which one can interact in order to change the position on the island. When a certain area on the island is clicked on, that area gets to the centre of the Flash object by means of an animation;

2) The trees and other objects apart from stores - inactive graphical elements which are only used for the environment and are repainted once the position on the island is changed;

3) The clouds – graphical inactive animations which appear in the environment at a certain random time interval and are only used for the environment.

60
4) **The stores** – maybe the most important graphical elements from the application, helping us create the learning environment. Each store represents a table from the application’s database to which the user has access in order to learn SQL. The stores have their name written below and, once selected when clicked on, the focus changes to be placed on them and an area looking like a house interior appears. If the store is clicked on again, this action is considered as deselect and the interior image of the store is closed from the user’s view. Once a store is selected, a star with a number appears as attached to the store, showing the number of currently selected objects from within the store. In the new interior store area one can see the store’s name, the “X” button closing the interior view of the store, while not deselecting the store, two buttons indicating the store’s fields (corresponding to the columns from the database) and the store’s details (corresponding to the entries from the database). When clicking on the “Fields” button, a drawer opens up for the user containing the columns from the table representing the store, together with their types. Scroll bars are added to the drawer when the information within exceeds the drawer’s size limits. A field can be selected from the “Fields” drawer by clicking on it, which leads to its highlighting by means of a point sign in front of its name and emboldening of its row in the drawer, and deselected when clicking on it again, case in which the point sign and emboldening disappear. The central part of the interior area of the store contains the objects from the store, representing the rows from the table which the store represents. Fig. 19 shows all the active elements one gets when selecting a store.

![Fig. 19 - The learning environment with all active elements displayed](image)

5) **The store objects** – The objects are positioned within the interior area of the store according to the data obtained from the database. Their graphics, different from all the other graphical objects in the learning environment, is dynamically obtained from the exterior image folder, in this way new types of objects being easy to add without needing to change the code, but only by adding this information, containing the path to the image, in the database. When an object is selected, the previously inactive “Details” button opens up a drawer containing the entries from the table corresponding to the row represented by the selected object, as grouped according to the columns from which they belong. Scroll bars are added to the drawer when the information within exceeds the drawer’s size limits. At the same time with opening the drawer, the selection of an object makes the object be highlighted with yellow. Several objects can be
selected at the same time. For each newly selected object, the previous “Details” drawer closes and the one for the currently selected object opens. At the same time, all the selected objects become highlighted. If an object is clicked on twice, it is considered that it is deselected and its highlighting disappears.

B) **The bottom part** contains a small textarea for the submission of queries named SQL Translator. It is in this textarea where will appear the simulated SQL queries by means of graphical selections. On the right of the text area, the “Show” button can be used in order to execute the query and retrieve the results.

The selection/deselection of any active graphical object on the screen leads to the automatic generation of an SQL query in the SQL Translator textarea in the following ways:

- Selecting a store leads to the generation of an “SELECT * FROM store_name” query, where the store’s name is the one displayed below the store. If the store is deselected, this query gets erased.

- Selecting a field from the store limits the “*” sign from the SELECT query to only the selected field: “SELECT field_name FROM store_name”, where field_name is the name of the selected field. If several fields are selected, the query becomes: “SELECT field1, field2, ..., fieldn FROM store_name”, where field1, field2, ..., fieldn are all the selected fields. When a field is deselected, it disappears from this list of fields. If the only selected field is deselected, the field_name turns back to “*” in the query.

- Selecting an object from the store limits the SELECT query to only the results which have the ID corresponding to that object’s primary key: “SELECT * from store_name where ID=’object_id’”, where object_id is the object’s primary key. Here, we assumed that a field was not selected at the same time, but this combination is also possible in order to generate queries of the form: “SELECT field1 FROM store_name where ID=’object_id’” (or with a list of fields if several fields were selected). If several objects are selected, the query adds the different primary keys of the objects as conditions separated by “OR” (“...Where ID=’object_id1’ OR ID=’object_id2’ OR...OR ID=’object_idn’”).

- All these queries can be combined to generate the maximal query: “SELECT field1, field2, ... fieldn FROM store_name where ID=’object_id1’ OR ID=’object_id2’ OR...OR ID=’object_idn’”.

- If the “active_joins” variable sent from PHP to initialize the Flash is “Yes”, joins are also allowed by selecting more than one store at once. Otherwise, the clicking on a new store when another one is already selected involves the closing of the other store first.

If the user has not accessed the tutorial or the training, the fact that he has made a query involving a condition on the objects to be retrieved and without having selected any fields will only graphically open the store and show the objects within as selected. Other queries are executed in the normal way, which involves the opening up of an area containing the table with results. If the user is in the training or tutorial, any query will return results in the table-like representation. If the user enters an incorrect query (other
than SELECT), the Flash returns an “Unknown operation” error. If the user enters a SELECT query, but with an incorrect syntax, the Flash returns him a “Wrong operation format” error. If the user uses a SELECT query with a correct syntax, the Flash calls the PHP Flash scripts which change the name of the table to the one used in the database (out of security considerations) and communicate to the database the query. The query is then executed by the database and its results are returned to the PHP Flash scripts. The PHP Flash scripts check if the results are an error message or a table and in the first case simply return the error, in the second change the name of the table back to what the user uses and return the table to Flash. Flash displays the result as it came if it was an error within an error message (prompt), otherwise displays the results as a table in the result representation.

2) The Tutorial Area

Once the user is in the learning environment, he can see a list of tutorial names in the shape of buttons on the top left hand side of the learning environment. By clicking on any of these names, the tutorial having that name starts. If the tutorial was not previously loaded for a previous visualisation, it is loaded from the database and, while this happens, the user is warned about it by a message appearing in the top lesson area of the tutorial. After the tutorial loads, the first lesson appears in the lesson area. The user will read the lesson and, when he wants, move to the next lesson by means of a “Next” arrow. Once the second lesson has been reached, he will also be able to move back to the previous lesson by using the “Previous” arrow. The “Next” button is deactivated once the final lesson was reached. The lessons can trigger the blocking of the “Next” arrow until a selection action or query is accomplished by the user (in case in which the next lesson is automatically triggered) if this was decided on by the teacher. At the same time, the lessons can only require the action for automatically proceeding to the next lesson without blocking the “Next” button if this was decided by the teacher. The lessons can also make the main SQL keywords (SELECT, DISTINCT, FROM, WHERE, ORDER BY), placed in the tutorial and learning above the SQL Translator’s text area, become highlighted if the teacher has decided on this in order to show to the user what he is focusing on in the current lesson or what he required from him in order to solve a given task. On moving to a next lesson, the screen can be cleaned of previous selections for the user if the teacher has taken this decision.

We have proposed a default tutorial, SQL Basic, which we will now briefly describe. The tutorial was proposed as for users who are absolute beginners in SQL. It takes these users through the basic concepts related to databases (what a table, column, row is) by means of requiring him to make selections from the screen and observe the graphical elements which open up. It then explains how the basic SELECT queries can be written by, again, requiring the user to make selections from the screen and, this time, also observe the generated query from the text area. Given these generated queries as examples, supported by lessons on the syntax of each type of query and additional examples where it was considered as necessary, the user is then also required to write his own queries. We have not blocked the “Next” button for users in order to allow them to proceed in the tutorial even if they have not managed to do a query, thus avoiding possible irritation from the evaluators which we will use in the evaluation, as we will see, and encourage them to
look at all the parts of the website. At the same time, each time a user is taught a new type of query, the keywords related to the type of query it related to light up. They do the same in case the user is given a task. The tutorial was not designed as too long, taking at most one hour to solve according to our estimations, which we consider is appropriate as the tutorial is likely to be used during a class. The tutorial is proposed as an example for teachers, which they can change or remove if they want to. We will test the outcomes of this tutorial with users in the next chapter. Fig. 20 shows an example of query result which is also correct in order to continue the tasks.

Fig. 21- The result of a correct query in the tutorial

3) The Training Area

The training area is accessible by clicking on the “Start training” button from the top right corner of the screen when entering the learning environment (the “Learn” page). The training involves the automatic generation of random tasks. The Flash first chooses randomly between a list of query types, it then chooses values for the table, fields and conditions according to the respective query type and then creates the query, runs it by using the PHP flash scripts communicating to the database to find out the wanted result, creates the text of the task to be shown to the user and displays the highlighted keywords for the task. When a user is trying out queries, he is returned clear feedback about whether the task was accomplished or not, apart from the results pf the task being displayed (as a user might get results but his solution might not have been what was required). Underneath the task area on the left hand side there is a “Completed tasks” display which records and displays to the user how many tasks he has accomplished. The user only has the “Next” arrow available as we considered there would be no need for him to return to previous tasks. We have decided to leave the “Next” arrow for the next task as active so that the user could decide to change the task if he does not know how to solve it. The training generates an infinite number of tasks for the user.

5.2.2.2 The Flash Component’s Code

For making it easier to be developed and tested, the main code from the Flash file was developed within 7 frames. All the frames have a name so that they can be identified and called directly by using their names:
• The first frame is named “initialisation” and it loads the external files, initialises the global variables, and retrieves the variables sent from PHP: the name, the score and activeJoins, initialises the graphical elements for the environment and paints the background of the learning environment.

• The second frame is named “getTableData”, it has a single call for the “loadTables” method from the Manager class and it blocks the Flash movie from going to the next frame. The “loadTables” method will load the tables from the database via the specialised PHP Flash files. As soon as all the tables have been loaded from the database, the “loadTables” method will send an instruction for the Flash movie to move to the next frame.

• The third frame is named “getHouseObjects”, it has a single call for the “loadHouseObjects” method from the Manager class and it blocks the Flash movie from going to the next frame. The “loadHouseObjects” method works similar with the “loadTables” method previously described, but will load all the house objects (rows from the tables) for all the stores (tables ) together with all their information.

• The fourth frame is named “getLessons” and, in the same way, calls the “loadTutorials” method from the Manager class and loads all the active tutorial names in order for them to be displayed as tutorial titles on the buttons leading to the tutorials.

• The fifth frame is named “printActive”. Its main purpose is to graphically display all the tables and create an active graphical element for each existent table. When clicking on a store, this frame will make sure that the necessary changes are computed: the check of whether it has already been clicked on is made and, in that case, it is deselected, otherwise the target central position of the image is set up to be equal to the position of the selected object, while also closing other objects if they were previously selected. This frame also performs the operations for the queries submitted by the user, the frame checking if the query is a SELECT query and conforms with the correct syntax by means of the “computeManualQuery” method and, in case the query is correct, using the “populateSelectedElem” method to send the user query to the database and display the results. Another functionality of this frame is to display the interior area of the store when the central point of the image is equal to the position of the selected store.

• The sixth frame is named “decisionActiveInactive” and it is the main decision frame of the application. Its main purpose is that of checking whether the active graphics from the previous frame needs to be displayed (if the image’s central point is equal to the position of the selected store or to the area clicked on the map) or an inactive one presented in the next frame. The frame displays all the error messages received when a manual query is performed (no matter if they were generated by Flash because of an incorrect query or query syntax or returned from the database), as well as result area and the feedback for a task complete or incomplete from training and tutorial. The frame also verifies whether the user wants to advance in the tutorial and moves to the next/previous lesson if so.
• The seventh frame displays all the graphics as inactive except for the environment so that moving on
the map is possible (so that when a user clicks on the map he cannot select something while the map
moves). This frame also computes the advancement by one quarter of the distance between the target
and the current centre when it is called.

In addition to this code, there are also other independent code fragments directly attached to the graphical
elements from Flash in order to display the text within the result area, in order to compute the opening up
animations for showing selections, in order to give extra functionality to the buttons, etc. Apart from this
and the classes which were already presented within the class diagram in the previous chapter, there are
also two files containing functions:

• The first one is the “functions.as” file in which main types of static variables are defined and the
keywords, background, the star storing the number of selected objects from within the store, the name
of the store, etc. are displayed. It also contains all the methods for the generation of the tasks from
training.

• The second file is named “functionsManualQuery.as” and it contains all the functions for the
computation of a query. The code for the “functionsManualQuery.as” file is provided in Appendix G.

5.3 The Software Testing Strategy

The testing of the courseware application was performed as the final phase of the implementation in each
iteration to the development, when a new version of the prototype was under test. The testing strategy
followed the well established Software Testing practices, by carrying out system testing after each
development cycle, involving the use of the system in all the possible scenarios of use, giving all the
possible combinations of inputs, and comparing the correctness of the outputs against what was expected,
while also looking at intermediary messages by using the “trace” operation. We could have used some
tools for different situations, such as the ASUnit framework ([69]) in case the whole application was object
oriented or the TestComplete application ([70]) which offers the possibility to catch a default scenario and
after that run it time after time on your system to check that the system always behaves the same way.
These tools would have involved, though, that we learned and spent time on a deeper testing, which was
not possible due to the short time frame given for the Masters project.
CHAPTER 6. EVALUATION

In the previous chapter we have seen how the developed courseware application, SQL Town, was implemented, its look and feel and how it was tested. As no testing is enough until the piece of software is tried out with end users, the next and final step is the evaluation of the software with users to decide whether it is a good solution for them. To this end, we will describe and motivate the option for a certain type of evaluation and method for carrying it out. We will then show how the method was implemented, how the participants were selected and contacted and how the data was recorded. The option for the method to analyse the data will then be motivated and its implementation will be described. The results will be shown and conclusions will be derived about the outcomes of the SQL Town on-line courseware.

6.1 The type of evaluation

According to the time in the development of the software when evaluation is undertaken, there can be the following types of evaluation:

1) Formative evaluation- is carried out while the piece of software is still under its development with a small number of users in order to “test run” different parts which could have escaped the developer’s tests [64]. Its purpose is mainly that of strengthening improvement of the piece of software and making sure it respects its goals.

2) Summative evaluation- is carried out at the end of the development of the system in order to test with end users the system’s efficacy (whether it does what it was designed to do) [64]. It is used to assess the overall impact of the piece of software for its intended user population.

For evaluating the SQL Town coursework application, we will be using formative evaluation as we are interested in both assessing if the software reaches its goals as defined in Chapter 1 and also if and where it can be improved, which will be proposed as future work. We consider that this evaluation, even if carried out on a fully functional application, is not the final one. In order to develop a totally usable and useful application for the users, which to respect their exact needs and be easy to employ in high schools, it would have been necessary to involve the users’ right from the initial design (which is called user-centred design in usability engineering). Nevertheless, the short time frame proposed for this project made this impossible. We therefore consider that the courseware application should still be refined by involving users. Only before officially launching the application it should also be evaluated in a summative manner. Within the current evaluation, we will be focusing on its ease of use, learning outcomes, level of attractiveness for the primary intended user population: high school pupils.

6.2 The evaluation method and its implementation

The on-line questionnaire was the best option for the evaluation method for our case. First of all, this is because, as we were evaluating the site with users who are not necessarily 18 years old yet, meeting them
in person for the evaluation would have involved the need for legal approval in the UK, which takes long
to obtain and was not an option in the short time frame given for a Masters project. Secondly, as we did not
know any high school pupil from the UK to ask him to help us, we needed to ask pupils for help from our
home country, therefore an Internet-based questionnaire was the best option.

We opted for a semi-structured questionnaire, thus containing both open-ended and closed questions. We
did not use a structured questionnaire as we believe this would have restrained the users to some fixed
responses which would not have necessarily fit with their opinions and also not allowed them to have their
say about the problems they encountered. We did not use an unstructured questionnaire either as this
would have obliged the users, who are teenagers and thus likely to be impatient, to state ideas and
comments everywhere, thus encouraging the less patient ones to write too short answers or even give up to
the questionnaire.

6.3 Preparing and conducting the evaluation

6.3.1 The design of the questionnaires

The questionnaire was designed by us as a website by using PHP and MySQL. It was hosted on an Apache
server named WampServer. The server was kept on 24/7 so that users could fill in the questionnaires when
they had the time to. The evaluation was carried out within a period of a week. The screen shots of all the
pages of the questionnaire are provided in Appendix H.

The closed questions focused on gathering data about the user, which ensured that he is part of the target
user group for the system, and on finding out exact responses regarding the users’ opinions about the
different parts of the site, which allowed for the easier interpretation of such data. Open-ended questions
normally required the users to provide suggestions of improvement for the different areas of the site or to
give details about the parts of the site which they had problems using or disliked. All the questions which
covered areas of interest were designed as compulsory, prompting the users if not filled in, while those
which only required additional comments or suggestions were optional. The questions first of all required
the user to state his age, grade and whether he had previously had experience with SQL so that we could
make sure he was part of the intended user group. We then focused around the interface and its layout and
colouring to find out how attractive the users considered the interface of the courseware application. The
next set of questions focused on the clarity of navigation around the site and using its functionalities
and on whether the feedback the system and the help provided to the user made this clearer. We then asked
questions regarding the learning outcomes of the tutorial and training areas and the ease with which the
users used them given the support offered within each page. The last set of questions asked the users to
conclude on whether they liked the site and whether they found it useful.

As the intended user population as participants for the evaluation, as we will see below, were high school
pupils, we decided that the questionnaire should not take more than 15 minutes to fill in. Therefore, the
number of questions was not too large and we filled in the questionnaire ourselves and added a thinking time to estimate the time it would take to fill in by users.

6.3.2 The protocol

Although we did not meet face-to-face with the participants and only used an on-line questionnaire, thus legal aspects not being involved, we still considered that it would be more ethical if we had a participant’s approval to our evaluation before he starts filling in the questionnaire. This was accomplished by designing an information sheet for the participants whom we added as the first page of the on-line questionnaire. It contained a background about us, the purpose of the evaluation, its contents and duration. It also reassured the participant that his personal details will be kept as anonymous and are not even required as part of the questionnaire and informed him that we would be happy to show him the results of the evaluation at the end of the Masters project. All this information was, of course, provided in an informal manner for high school pupils to find easy to read. The user was then asked to proceed to the next page in the questionnaire only if he agreed with participating to our evaluation. This information sheet can be seen as the first page of the questionnaire screen shots provided in Appendix H.

6.3.3 The user selection and contacting process

As the primary users of the site would be high school pupils before the extension with older learners would even be considered, we decided that it is this user class that we must focus on. We therefore only selected users who were in between 15 and 19 years of age. To avoid the bias of having the courseware application be easily understood because the users are familiar with programming languages and computer systems as would be the case for pupils from a high school which is specialised in Informatics, we decided we would only be considering users from high schools with other profiles, which would be more or less accustomed with using computers, but would not in any case be familiar with programming on them. In this way, by catering for users from high schools other than those specialised in Informatics we are also ensuring that the Informatics high school pupils would not have problems in using our courseware application. In what concerns their computer literacy, we did not impose any norm in this respect in order to cater for the normal distribution of computer skills in a high school of another profile than Informatics. A consideration of trends of answers according to the classification of the users by gender in male and female would have potentially be relevant, but we did not consider it as we believe that we would have too few participants to be able to derive an accurate conclusion in this respect. In what concerns the familiarity with SQL, we decided that we would only evaluate the site with users who have never before studied it, as it is this class of users we mainly intended the courseware for.

The first user we contacted was a friend’s brother, who is studying in a high school specialized in Biology. We then asked him to ask all his friends, from any other high school than an Informatics-specialised one, to evaluate our system. The result was that we managed to gather 15 responses to our questionnaire, which we consider is a good result. From the users who replied, we got an almost equal number of males and females: 7 females and 8 males, which is a good result as we can thus derive general conclusions without
the possible bias of being, influenced more by the replies of users of a certain gender. Most of the users were from the 11th grade (in our country, the high school starts in the 9th and finished in the 12th grade) - 7 of the users, 3 were from the 10th grade and 5 were from the 12th grade. This is not necessarily relevant as to consider that they would have more experience with computers because, as we already mentioned, we had users from high schools other than Informatics-specialised ones. None of the participants had previously studied SQL; therefore they were all absolute beginners.

6.3.4 How the data was recorded

The replies the user gave to the questionnaire were recorded by our database into columns for each question. To proceed to the data analysis phase, we made a list of all the questions for each user which we stored in a Microsoft Office Word document and we added the answers from the table columns to each question.

6.4 Data Analysis

6.4.1 The method for data analysis and its implementation

According to the format of the data that it involves, data analysis can be quantitative (involving working with numbers) and qualitative (involving working with words, documents, images, narratives, text or other non-numerical types of data) [65].

For our evaluation the data analysis will be qualitative as we were interested in finding out critiques, opinions and suggestions from the users, therefore we will be analysing words.

From the methods used in qualitative evaluation to analyse data the most widely used is content analysis [66]. It uses codes to reflect the ideas existent in the text at the end of the data collection process. Content analysis offers the option of coding the data in two different ways: of coding down (where, starting from the theory, the themes are decided before the analysis phase starts) or by coding up (where the themes emerge from the data) [67]. The evaluator goes through the whole set of data by assigning codes to the different parts of the user answers. The selected codes can be words or numbers.

We used content analysis together with a combination of coding down and coding up methods. As we have used a semi-structured questionnaire, the user answers will fit into some clear themes which represent the areas on which our questions focused. We wanted to find out answers from these areas as we were interested in assessing ease of use, learning outcomes and interface attractiveness, all of which were derived as important as a result of our previous studies on usability from our undergraduate and Masters Studies, as well as the literature read for the Masters project on usability for courseware applications. The first themes were therefore derived from theory on usability areas to focus on for courseware applications. We also used a coding up method by deriving new themes from the data as we noticed they would not fit in the existing ones.
The themes correspond to the subchapters used in the next section of the paper. The findings are revealed by means of counting the common opinions users had and supporting them with citations from user replies to the open-ended questions. Although in general the ideas which were supported by more than half of the participants are considered as a finding, in the case of suggestions individual ones might also be considered if we believe they could potentially be a good idea for future work (which would, of course, involve further evaluation to test if other users also think it is a good idea).

6.4.2 The findings

6.4.2.1 Interface Presentation/ Layout

Regarding the **colours** used throughout the website, all the participants liked them. All the users also liked the **graphical objects** (stores/objects within) used within the game-like interface. We received though a number of suggestions from 4 of the users:

„*A bigger number of objects are better looking!!!*” (Male, 11th grade, aged 18)

„*Animate 2D painted objects would make it come alive and make it more attractive.*” (Male, 12th grade, aged 19)

„*Maybe look a bit more real*” (Female, 10th grade, aged 16)

„*Smotherer edges.*” (Male, 10th grade, aged 16)

As can be seen from these suggestions, no two users had the same suggestion; therefore we cannot derive any conclusion on what would be seen as a good solution. Nevertheless, the first suggestion was also considered by ourselves but, as already pointed out, more objects would have made the loading of the Flash environment harder and we decided on a smaller number as we cannot be aware what hardware and software platforms and Internet connection speeds our prospective users have. Although only suggested by one person, we believe that the second suggestion is very interesting and we will propose it as future work.

In what concerns the environment in which the town is placed, most of the participants (13/15) liked it. The other two had some interesting comments to make:

„*Too simple*” (Female, 10th grade, aged 16)

„*I consider the environment used too simple and 3D-objects free.*” (Male, 12th grade, aged 18)

When asked whether they would have liked some other kind of environment, these two users suggested having more objects and a town-like feel to the game-like environment:

„*Roads, people around, to seem more real*” (Female, 10th grade, aged 16)
“A more graphical environment with many other 3D objects around than the stores, that is why it is called SQL Town.” (Male, 12th grade, aged 18)

Other participants also had suggestions to make:

“...like more an environment closer to reality. Something like grass and trees on the ground. Just for the more entertainment.” (Female, 12th grade, aged 19)

“...like clouds with changing colours once clicked.” (Male, 10th grade, aged 16)

“...To see some trees” (Male, 11th grade, aged 18).

Although these suggestions were only made by 1/3 of the users and are personal opinions about how the game-like environment should look like, we could though conclude that a more complex and more entertaining graphics would add a plus to our learning environment. **This will be proposed as future work.** Nevertheless, we must still consider the technical problems this could entail as the graphics would load harder and users with slower Internet connections and older computers might have to wait longer to see the results of their actions.

In what concerns the display of the fields within the “Fields” drawer, all the users except one liked it. The comment made by the single user not liking it will be provided in the “General learning outcomes” theme, as the user considered the effect of the display on learning outcomes. The results were nevertheless positive.

6.4.2.2 Transparency of the purpose and message of the courseware application

When asked whether they understood the **purposes of the site**, all the users except one considered it clear enough. This one user had not read the homepage or “About us” page. Other 2 users who have not still considered the message clear. All the users who have read the **homepage or “About us” page** considered it clear enough. We therefore consider that the site makes its purpose clear easily enough, without necessarily needing for the users to read the homepage or “About us” page, but if they do these pages make it even clearer.

6.4.2.3 Navigation

11 of the participants had some initial problems with getting to the learning environment. We will provide 3 pf the citations:

“I did not know how to start to use the site after logging in and I had to figure out I had to click on ‘Game.’” (Male, 11th grade, aged 18).

“Once in ‘Game’ I did not initially know where the tutorial is” (Female, 12th grade, aged 19)
“I had to click around the site to understand how to get to the tutorial. I did not know SQL basics is the tutorial” (Female, 10th grade, aged 16)

We have considered all of these problems and, as an improvement resulted from the evaluation phase, changes the “Game” button to “Learn”, added a clearly named link from the homepage to the learning area and added arrows to show that “SQL Basics” is the tutorial where users should start their learning.

6.4.2.4 Ease of Use

We asked users how easy it was for them to create an account as this is the function they needed to use in order to be able to try out the learning environment. All the responses were positive, which can be explained by the fact that the form is very similar to any other registration form most of the users are familiar with from creating email or social website accounts. We also asked users how easy it was to understand how to use the SQL Translator facility, as this function is directly related to interacting with the learning environment. There were no problems in this respect either which is probably because the text area is a clear affordance (visual indication of the action which can be undertaken on it) by being a text area.

6.4.2.5 Clarity of Feedback

We asked users about how clear it was for them to perceive the results they received from the system when trying to solve queries. These results could have been either errors coming from the system if the query did not use a correct syntax, errors coming from the SQL engine if the query was incorrect (using not existent names for tables, fields, etc.) or actual results in the form of tables. 3 of the users mentioned having initial problems with now understanding error messages in case they were entering incorrect formats for queries (please note that the initial error message for incorrectly formatted queries was: ”Wrong operation format”):

“I did not understand at first how I should solve the error of wrong operation format” (female, 10th grade, aged 16)

“I could not solve a query because it always gave back ‘Wrong operation format’. It took me a while to figure out I could look for the format in ‘Syntax’” (Male, 12th grade, aged 18)

“Wrong operation format is not a clear error message” (Female, 12th grade, aged 19)

As only these errors were considered as hard to understand, we changed the error message as a result to the evaluation so that the word “format” is replaced with “syntax” and a clear indication of where to look for the correct syntax is provided: “Wrong operation syntax. Look for the correct syntax by using the ‘Syntax’ button on the left of SQL Translator”. Although not detected as a problem, we made the same change for the error given if the operation is totally incorrect.
None of the users had problems with knowing if they had accomplished a task in training.

6.4.2.6 Help

Many of participants to the questionnaire (10/15) stated having used the help for trying to find out how to do a query. The help did not provide the answer for them, as we have configured the help only for explaining how the site can be used. To solve this problem, we proposed as a result to the evaluation the addition of a clear and coloured (red) message on top of any help page about the purposes of the help: “The help only gives you guidance for using the site. If you have problems in writing your queries, please use the ‘Syntax’ button on the left of the SQL Translator”. As future work, we will propose adding an extra button in the main menu for documentation on SQL.

6.4.2.7 Learning outcomes in the tutorial

In what concerns the level of difficulty of the tutorial’s lessons, more than half of the participants considered that the tutorial is “so and so”. 6 of the rest of the participants considered it easy and one participant considered it very easy. The results therefore show that none of the participants perceived the tutorial as being hard. The large percentage who considered it moderate demonstrate that the tutorial is challenging, but also doable, which is a good result as learners need to be challenged in order to remember what they learned.

3 of the participants provided suggestions for making the lessons easier to learn:

„A richer example of the syntax. Something like self editing examples.” (Female, 12th grade, aged 19)

„Always try expanding your ideas to make the small descriptions more clear. Do this by adding more details and examples.” (Male, 12th grade, aged 18)

„There would be nothing more to do than making an extra PDF available for download where things could be larger explained and exemplified so that people that are having difficulties in understanding general SQL (like me), could make a better image.” (Male, 10th grade, aged 16)

While the third suggestion was already considered by us by proposing as future work the addition of a special documentation button in the main menu, we consider that the first suggestion (having self editing, that is the site supporting of automatic generation of the results if the user cannot) would reduce the quality of the tutorial as it would not be challenging at all any more. The second suggestion was considered for the default tutorial by adding an extra example for the SELECT... FROM... ORDER BY query as a result to the evaluation, but the default query is only an example in itself, therefore it is the teacher’s decision on the amount of details and examples he will provide in his own tutorials.

In what concerns the difficulty of the tasks given as part of the tutorial, 2/3 of the participants (10/15) considered they were easy after having read the examples in the tutorial. 4 of the other participants considered them so and so and one of them as very easy. This distribution demonstrates that, although the
default tutorial may be challenging in itself to the users, they do manage to solve the tasks easily enough after having understood it. This will, of course, vary after the change of the tutorial and addition of new ones by the teacher.

In what concerns the **length of the tutorial**, more than half of the users (8/15) considered it is just right. Other 6 considered it short, while one user considered it long. This shows that this opinion depends on personal preference but that, nevertheless, most of the participants have seen the tutorial as either just right in size or short, thus demonstrating that it is usually doable by a novice to SQL.

In what concerns the **overall usefulness of the tutorial**, 13 of the participants considered it as either useful (5) or very useful (8), while only 2 of them considered it as “so and so”. This demonstrates that the great majority of the users generally found the outcomes of the tutorial as useful. The results might be a little biased to having more “useful” replies than “very useful” ones as the question used a Likert scale to which some people never give the best reply as it is not in their nature to ever give the greatest grade to anything.

In what concerns the **suggestions for improving the tutorial overall**, there were 4:

„*The tutorial can be improved by showing the advantages of using this kind of language in simple cases*” (male, 12th grade, aged 19)

„*Show approximate time until completion, approach more difficult commands*” (Female, 10th grade, aged 16)

„*It was excellent for such a small tutorial to give such good results. Yet, i believe that some extra documentation could be designed in order to achieve better results. For me this tutorial raised my interest for SQL. I am planning more for the future.*” (Male, 10th grade, aged 16)

„*Browsing on many tutorial websites, i found something interesting during my search on a C++ tutorial website. It was called the show me link. Like in your tutorial of Basic SQL, the C++ one included exercises. But it also had something more. For example, i was given to write in the C++ translator the basic #include <iostream.h> library. But, lets propose I didn’t get how to do that from the earlier pages so, all that i had to do was to press the show me, and the syntax appeared. This could be also useful in your tutorial.*” (male, 10th grade, aged 18)

The last suggestion refers to the self editing problematic mentioned before by the same user and already discussed above. The first suggestion is a good idea to be considered, but it will of course depend on the teacher writing or editing the actual tutorial. The second suggestion will be proposed as future work as it would add useful information. The third suggestion was already proposed when suggesting the documentation button for future work.

### 6.4.2.8 Learning outcomes in training

In what concerns the **understanding of the tasks provided in the training area**, none of the users had problems in this respect.
In what concerns the level of difficulty of the tasks after having done default the tutorial, most of the users (10/15) considered them easy, which is a good result as we did not want to challenge the user even more as part of the training, but to make it easy for him to practice his skills. 4 other users considered that they were so and so and one user that they were easy, therefore nobody found the tasks hard to solve after having done the tutorial.

All of the users considered the highlighting of the SQL keywords as useful for them when solving the tasks, but 4 of them had even further suggestions to make about making it easier to solve the tasks:

„Yes, at the beginning of the training course you could introduce the highlighted bar and its utility.” (Male, 12th grade, aged 18)

„The highlighting of keywords could be better if it could light in different colors. Something like blue when not used, green when used properly and red when something is wrong.” (Male, 10th grade, aged 16)

„Progressive highlighting as you pass by.” (Female, 12th grade, aged 19)

„I didnt realise what were that highlighted keywords trying to tell me. Only after completing the first set of exercises i finally realised what were the highlighted keywords useful for.” (Male, 12th grade, aged 18)

The suggestions show that, first of all, for some (2 users) the highlighting’s purpose was not immediately clear. Although this problem was only mentioned by two users, we believe that making the highlighting words glow upon first use in the tutorial (if the teacher chooses to have words highlighted) and training would make this highlighting stand up more and be clearer for everybody and it is this functionality that we will propose as future work. Although having the highlighting presented within the training as one user suggested would have been a good idea, it would have needed to be introduced in the tutorial as well as the highlighting is first gives there if the teacher chooses to, but this latter case depends on the teacher, therefore we did not consider it. In what concerns the progressive highlighting of words or the use of different colours to progressively show that the user is writing the query correctly/incorrectly, we believe that the use of colours is a better option as it would also highlight the keywords initially to indicate what keywords the user must use in his query. We will therefore propose this progressive colouring of the highlighting keywords as well as future work.

In what concerns the clicking on stores for training to get the results instead of writing the queries, most of the users (12/15) stated they have. This can be seen as a problem as the purpose of the training was to indulge them to start writing their own queries. To avoid this clicking for users who have already solved a certain number of tasks (let’s say 5 tasks), we will propose as future work that the stores become inaccessible once this number has been reached.

In what concerns the overall usefulness of the training part of the site, the great majority of users (13/15) considered it either useful (9) or as very useful (4), while only 2 considering it as “so and so”. Again, users could have replied to “useful” simply because they are not used to giving too high grades to
Likert scales, or they might have simply seen the training as less useful than the tutorial. Nevertheless, the overall results are positive.

In what concerns the suggestions for improving the training part of the site, there were 4:

„Extra exercises where you are not allowed to help yourself by clicking on objects.” (Female, 12th grade, aged 19)

„I consider the Training part could be better if a difficulty level was added. For example, level 1 could include very basic instructions to use. As you pass, you could reach the final difficulty level. This way you are sure that you got everything right!” (Male, 10th grade, aged 16)

„The highlighting keywords bar is indeed very useful, but it could turn itself off, after finishing a set of exercises (in my opinion).” (Male, 12th grade, aged 18)

„The training tasks should be step by step, from something very easy to a very hard situation” (Male, 11th grade, aged 18)

All these suggestions therefore had a common idea behind: that of having levels of difficulty in the training area. Although these suggestions have been made only by 4 users and more evaluation would be needed to decide if the site, being for absolute novices in SQL, it is a good idea to have levels of difficulty, we will propose having them and further evaluating with users if they are useful as future work. We would see this by having queries which are harder and harder according to the total number of queries solved by a user. This would also be accompanied with stores which become inaccessible after a certain number of tasks are accomplished, as already mentioned. The suggestion made by the third user is also very interesting, so we will also propose it for future work.

6.4.2.9 Overall impressions

In what concerns the level of satisfaction with using the site, all of the users mentioned liking using the site (8) or even liking it very much (7). Moreover, in what concerns its level of usefulness, the results were also very positive; 7 users found it useful and 8 found it very useful. We can therefore conclude that the overall impressions about our site were positive.

6.5 The Evaluation with Teachers

After having evaluated the courseware application with high school pupils as it was outlined in the previous subchapter, we have also evaluated it with our supervisor within an informal discussion to find out if the application suits her needs. After having tried out the application and all of its features, our supervisor being happy of the result. The outcome of this evaluation was therefore positive as well.
CHAPTER 7. SUMMARY

7.1 Outline

In this project we have described the process involved in the development of a courseware application for the teaching of the basics of SQL to high school pupils. We have first outlined the motivation for this project: the fact that SQL, the first language one learns when studying databases, is not taught within high schools and our belief that it could be taught given an attractive and easy to use application which would make pupils have fun and learn at the same time. We then decided on the purpose and objectives of the project: that of developing such an application which would be fun and attractive website, usable by both high school pupils and teachers and supporting a tutorial, training environment and a facility where the teacher could add and edit his own tutorials for his pupils. We decided that the learners would initially be high school pupils, with the extension of the courseware to be used by older students and even adults if the results with high school pupils are positive. We then started developing the application by first gathering requirements from users belonging to each class of potential users. The requirements obtained were specified, motivated and, for the functional ones, related to the corresponding use-cases, by the use of a professional commercial template, Volere [61]. The use-cases for requirements analysis were provided and the most important of them explained by means of use-case descriptions given as following Alistair Cockburn’s template [62]. Departing from the use-cases, which defined the main functionalities the site should support for each class of users, we then explained what technologies and tools were used in the design and demonstrated how the functionalities are designed by means of sequence diagrams. ER diagrams were also proposed in order to explain the design of the database storing information used for all of the user functionalities. Class diagrams were also provided to explain the main structure of the classes in the Flash component of the application, dealing with the storage of the information from the database in a structured manner. Once the design phase was complete, we explained that in fact there was not one single design phase, and not even one single requirements capture and analysis phase, but that we used iterative and incremental prototyping, passing through these phases, as well as through implementation and evaluation repeatedly by developing parts of the functionalities representing certain use-cases and then going back to the users with the prototype for a new evaluation, after which the cycle was repeated, adding or amending functionality in every such cycle. We described by means of detailed explanations and screen shots the implementation of the website part of the application and that of the Flash component of the application. We gave an outline of the testing method used, which conforms to general Software Testing practice of performing system testing by means of inputs for which we check the outputs. After we considered that the evaluation was finalised, we went to the primary users of the application- high school pupils with which we did not have the opportunity to evaluate the application before because of lack of time, to evaluate the software. We explained that we used formative evaluation as we considered that more refinements and evaluations with high school pupils would be necessary in order to launch the courseware application. We used an on-line semi-structured questionnaire allowing us to avoid the problematic of legal issues from the UK and the fact that we did not know any high school pupil from here. We received a good number of replies to the questionnaire (15 users) which we decided to analyse at the end of the data
gathering process, by means of a content analysis. We generated the themes as departing from theory and asked questions, as well as directly from the data. The results showed that overall the users were happy with the application and that they found it useful for their learning. Found problems and individual suggestions were catered both as a result to the evaluation in the available time, as well as proposed as future work.

7.2 Achievements

The following were the original contributions we brought to this project:

- A good outline of recent literature on eLearning, eLearning for children, general problems in learning SQL, usability in children’s applications and eLearning solutions for high schools;

- The proposal of an eLearning application for high school pupils for learning SQL, the first of this kind as far as we are aware; the application received good reviews from its users;

- The proposal of an attractive and fun environment which relieves the stress associated to learning, makes learning more pleasurable and indulges the user to pursue the subject; this environment was indeed seen as pleasurable by the users with which it was evaluated.

- A solution for representing queries by means of manipulating of graphical elements from the screen;

- A solution for making on-line tutorials interactive and clear by means of step-by-step lessons accompanied by the support for a game-like interaction combined with teacher-defined tasks;

- The SQL Basics default tutorial which was considered as challenging by users but useful for understanding how to write queries in SQL;

- A training environment which allows for users to practice their SQL skills by automatically generating infinite random tasks which are totally assessed by the computer;

- A solution for allowing the teacher to personalise his own tutorials and decide on the options which would make them as interactive as he wishes;

- An outline of user comments and suggestions which would make the above solutions even better;

7.3 Limitations

Although the results to our projects were mostly positive, there are some limitations to our solution:
The fact that the solution was designed by capturing requirements from a small number of users out of lack of time. We cannot be sure that the solution captures the requirements high school pupils and teachers would have as a group.

The fact that the solution was only evaluated with the primary end users at the end of the development, also out of lack of time. Although the results were mostly positive, there was a great risk that this evaluation could have proven the solution wrong and the redesign would have involved much more work;

The fact that we only evaluated the solution with 15 users; although this is a good number as Nielsen showed that most of the usability problems are discovered with the first 5 users [68], the courseware’s implementation in high schools would require much larger scale evaluations;

The fact that the for the selected users differences in the level of computer literacy and gender, as well as previous knowledge, were not considered, as this would have given us too few results to be able to derive viable conclusions. Our results could therefore be biased by having more users of a certain profile.

The fact that we did not formally evaluate the courseware application with a reasonable number of teachers. Our solution was not properly tried out with teachers, thus we cannot be absolutely sure about its outcomes in this respect.

Due to all of these limitations, we believe that the courseware application should continue to be refined and evaluated with end users: both high school pupils and teachers. Moreover, the users’ profile should be closely considered and any kinds of differences taken into account.

7.4 Future Work

We propose as future work for this project:

- Adding full support for joins
- Adding support for other types of operations: CREATE, UPDATE, INSERT INTO, DROP, DELETE FROM.
- Adding the possibility for students to solve complex tasks as a team work.
- Allowing students to compete against each other by proposing queries to one another.
- Adding more advanced graphics for the learning environment.
- Adding an extra button in the main menu for documentation on SQL.
- Adding a timeline to show how many lessons a tutorial still has for the user
- The accentuation of keywords highlighting for the first tutorial/training where it is used
- The use of progressive keyword highlighting combined with colouring to show the user as he writes if his query is correct or not
- Consideration of levels for users so that the tasks in training are harder and harder, selections are blocked and highlighting is not offered any more after a certain number of solved tasks
BIBLIOGRAPHY/REFERENCES


[69] Asunit. *ASUnit, the only open-source Unit Test Framework that supports Flash players 6, 7, 8, 9 and 10* [Internet]. [cited 5 Aug 2010], Available from: http://www.asunit.org/#learn-more

APPENDIX A - Requirements Specification Using the Volere Template

A. PROJECT DRIVERS

1. The Purpose of the Project

The overall goal of this project is the design of a friendly and interactive eLearning site to be used for the teaching of basic SQL. The site would be initially used in high schools for teaching pupils who are beginners in SQL. If the initial outcomes of this project are positive, it will also be proposed as a solution for teaching basic SQL to university students or even adults. For learners, the project will constitute a more pleasant and less stressful way of learning by combining theoretical ideas with a game-like layout which would be fun to use. For teachers, the purpose is for the web site to support the addition of tutorials so that teachers could decide on their own way of using the site with their pupils during the first SQL classes.

2. Client, Customer and Stakeholders

As the project is built as part of a Masters dissertation for Heriot-Watt University, it does not have a customer funding it, but it will be submitted for the university’s own use.

3. Users of the Product

As already mentioned within the project’s purpose, the users of the product would be both high school pupils and teachers. For pupils, the site will constitute a fun and relaxing way of studying the basics of SQL, hopefully helping them to easier understand its concepts. For teachers, the site will constitute a framework for building tutorials to be used in the first SQL classes with pupils. As also mentioned within the project’s purpose, older people such as university students or even adults who are beginners in SQL could be future users of the site if its initial outcomes within high schools are positive.

B. Project constraints

1. Naming Conventions and Definitions

User = anyone who is using the site; it might be a learner or teacher.

Learner = anyone who is using the site for learning purposes; it will initially be high school pupils, with the possible extension to university students or even adults; it does not matter what specialism the learners have as long as they use this site for learning SQL.

Teacher = any teacher of SQL (might be a high school or later on even a university teacher), from any specialism.

Account = the user’s personal interface to the system as a result of his use of it

Registration = the process of creating an account to one’s self as a user of the site

Change account details = the process of amending all the registration details, apart from his email address, which is used for user identification purposes.

Lesson = Short description of a topic as having been entered by the teacher and made available to the learner in the form of a short message within a tutorial.

Tutorial = learning guide made up by a series of lessons used by learners in their study of SQL.

Training = process of practicing SQL as a learner by solving tasks which were designed by the application.

2. Assumptions

1. Although the site will be used by learners who are, as part of their school curricula, beginners to SQL, the site does not cater for the possibility of them having studied it otherwise than at school before. We therefore assumed that all the users have a beginner level.

2. We assumed that, for a beginner level, the learner would learn the basic notions of SQL: table, column, row, primary key.

3. Also concerning the beginner level, we assumed that a learner would be taught the basics of writing SELECT queries: using *, adding one or more fields, using WHERE, DISCTINCT and ORDER BY.
4. Also concerning the beginner level, we assumed that a learner would be taught only how to execute SELECT queries on single tables, thus we will not consider joins as functionality.

5. We assumed that the teacher would only use this site with beginner learners as defined by the assumptions above and that he would not therefore need to be able to extend the available types of supported queries. To cater for the possibility of later extension to the use with more advanced learners, we will propose additions as future work.

6. We assumed that the teacher might have several classes to which he wishes to use different tutorials.

7. We assumed that the teacher might want to check the way his tutorial gets displayed to learners.

8. We assumed that the teacher would like to correlate the graphical display in the site with SQL queries to better explain the queries that he is teaching.

9. We assumed that the teacher would prefer that the site builds the learner training tasks instead of having to write them down by himself.

10. We assumed that a learner would only start training after he has read all the tutorials. The training will therefore allow him to practice all the knowledge acquired after having learned the tutorials and will not cater for learners who have not read the tutorials or try to get trained first thing when accessing the site.

11. We assumed that the learners would like to use the site at home as well, not only during classes.

12. We assumed that the site could be used by several learners at different browser accesses from the same computer while the computer is on as it could be used during separate classes by different groups within a laboratory.

13. We assumed that teachers’ accounts on the site would need to be verified by an administrator before the tutorial configuration facility be made accessible.

C. Functional Requirements

Requirement number: 1

Requirement description: The site allows users to register.

Rationale: To only offer access to the site to users who are authorized to. To differentiate between the access rights of each kind of user. To identify these user within the site

Requirement type: Functional.

Fit criterion: A user will be able to register.

Use cases: “Register” use case.

Dependencies: 2-61, 65-76, 78-79, 82-92

Requirement number: 2

Requirement description: In order to register, the site allows the user to enter his first name, last name and email address.

Rationale: The first and last names will help identify the user within the site and write status messages personalised for him. The email address will allow the user to access his account when using the site again.

Requirement type: Functional

Fit criterion: A user is able to enter his first name, last name and email address.

Use cases: “Register” use case

Dependencies: 1, 3-61, 65-76, 78-79, 82-92

Requirement number: 3

Requirement description: In order to register, the system allows customers to choose a user name and password.
Rationale: The chosen username will help identify the user within the site and write status messages personalised for him. The password will allow users to access their account when using the application again.

**Requirement type:** Functional  
**Fit criterion:** A customer is able to enter his username and password.  
**Use cases:** “Register” use case  
**Dependencies:** 1-2, 4-61, 65-76, 78-79, 82-92

---

**Requirement number:** 4  
**Requirement description:** In order to register, the system requires the user to enter the password twice.  
**Rationale:** To make sure the user has not misspelled it and will therefore remember it in the correct way.  
**Requirement type:** Functional  
**Fit criterion:** A customer is able to enter his password the second time.  
**Use cases:** “Register” use case  
**Dependencies:** 1-3, 5-61, 65-76, 78-79, 82-92

---

**Requirement number:** 5  
**Requirement description:** The site allows users to log in.  
**Rationale:** To allow them to access their account and thus be identified by the system.  
**Requirement type:** Functional  
**Fit criterion:** A user is able to log in  
**Use cases:** “Login” use case  
**Dependencies:** 1-4, 6-61, 65-76, 78-79, 82-92

---

**Requirement number:** 6  
**Requirement description:** In order to log in, the site allows users to enter their email address and password.  
**Rationale:** To allow them to access their account and thus be identified by the system.  
**Requirement type:** Functional  
**Fit criterion:** A user is able to enter his email address and password.  
**Use cases:** “Login” use case  
**Dependencies:** 1-5, 7-61, 65-76, 78-79, 82-92

---

**Requirement number:** 7  
**Requirement description:** The system allows users to modify their registration details at any point while being logged in.  
**Rationale:** To allow users to modify misspelled information about their name and choose a different username and password.  
**Requirement type:** Functional  
**Fit criterion:** A user is able to modify his registration details.  
**Use cases:** “Modify registration details” use case  
**Dependencies:** 1-6, 8-61, 65-76, 78-79, 82-92

---

**Requirement number:** 8
**Requirement description:** In order to modify registration details, the system allows users to modify to amend their first and last name.

**Rationale:** To allow users to modify misspelled information about their

**Requirement type:** Functional

**Fit criterion:** A user is able to amend his first and last name.

**Use cases:** “Modify registration details” use case

**Dependencies:** 1-7, 9-61, 65-76, 78-79, 82-92

---

**Requirement number:** 9

**Requirement description:** In order to modify registration details, the system allows users to amend their first and last name.

**Rationale:** To allow users to modify misspelled information about their

**Requirement type:** Functional

**Fit criterion:** A user is able to amend his first and last name.

**Use cases:** “Modify registration details” use case

**Dependencies:** 1-8, 10-61, 65-76, 78-79, 82-92

---

**Requirement number:** 10

**Requirement description:** In order to modify registration details, the system allows users to decide on a new username and password.

**Rationale:** To allow users to be identified with a different name within the site and be given status messages with this changed name. To allow users to enter a different password when logging in.

**Requirement type:** Functional

**Fit criterion:** A user is able to enter a new username and password.

**Use cases:** “Modify registration details” use case

**Dependencies:** 1-9, 11-61, 65-76, 78-79, 82-92

---

**Requirement number:** 11

**Requirement description:** In order to change the password, the system requires users to enter it twice

**Rationale:** To make sure the user has not misspelled it and will therefore remember it in the correct way.

**Requirement type:** Functional

**Fit criterion:** A user is able to enter the password the second time.

**Use cases:** “Modify registration details” use case

**Dependencies:** 1-10, 12-61, 65-76, 78-79, 82-92

---

**Requirement number:** 12

**Requirement description:** The site allows the users to interact with the town.

**Rationale:** To help the user deduce queries by using the graphical representation of the town.

**Requirement type:** Functional

**Fit criterion:** A user is able to interact with the town.

**Use cases:** “Interact with the town” use case

**Dependencies:** 1-11, 13-26, 28-32, 34, 48, 57, 64-65, 67-68, 83-89
Requirement number: 13

**Requirement description:** While the user interacts with the town, the site allows him to move on the map.

**Rationale:** To allow the user to get to the area on the map he wants.

**Requirement type:** Functional

**Fit criterion:** A user is able to move on the map.

**Use cases:** “Move on map” use case

**Dependencies:** 1-12, 30-32, 34, 64-65, 67, 83-89

---

Requirement number: 14

**Requirement description:** While the user interacts with the town, the site allows him to select one store from the town.

**Rationale:** To help the user simulate a SELECT * FROM store query graphically.

**Requirement type:** Functional

**Fit criterion:** A user is able to select a store from the town.

**Use cases:** “Select store” use case

**Dependencies:** 1-12, 15-26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

---

Requirement number: 15

**Requirement description:** When selecting a store, the site allows the user to observe what it sells.

**Rationale:** To allow the user to have an easy view of the objects within (representing the rows in the store’s table).

**Requirement type:** Functional

**Fit criterion:** A user is able to observe what a store sells.

**Use cases:** “Select objects” use case

**Dependencies:** 1-12, 14, 16-26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

---

Requirement number: 16

**Requirement description:** While the user interacts with the town, the system allows him to deselect a store from the town.

**Rationale:** To help the user erase a SELECT * FROM store query in a graphical way.

**Requirement type:** Functional

**Fit criterion:** A user is able to deselect a store from the town.

**Use cases:** “Deselect store” use case

**Dependencies:** 1-12, 14-15, 17-21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

---

Requirement number: 17

**Requirement description:** While the user interacts with the town, the site allows him to select one object from within a store from the town.

**Rationale:** To help the user simulate a SELECT object FROM store query graphically.

**Requirement type:** Functional

**Fit criterion:** A user is able to select one object from a store from the town.

**Use cases:** “Select objects” use case

**Dependencies:** 1-12, 14-16, 18-21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89
Requirement number: 18

**Requirement description:** When selecting an object from the store, the site allows the user to observe its details

**Rationale:** To allow the user to have an easy view of the details of the object (representing the columns and its contents of the row representing the object)

**Requirement type:** Functional

**Fit criterion:** A user is able to observe what the details of the object selected from within the store are.

**Use cases:** “Select objects” use case

**Dependencies:** 1-12, 14-17, 19-21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 19

**Requirement description:** While the user interacts with the town, the system allows him to deselect one object from a store from the town.

**Rationale:** To help the user erase a SELECT object FROM store query in a graphical way.

**Requirement type:** Functional

**Fit criterion:** A user is able to deselect one object from a store from the town.

**Use cases:** “Deselect objects” use case

**Dependencies:** 1-12, 14-17, 20-21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 20

**Requirement description:** While the user interacts with the town, the site allows him to select several objects from within a store from the town.

**Rationale:** To help the user simulate a SELECT object 1, object 2, ..., object n FROM store query graphically.

**Requirement type:** Functional

**Fit criterion:** A user is able to select several objects from a store from the town.

**Use cases:** “Select objects” use case

**Dependencies:** 1-12, 14-18, 21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 21

**Requirement description:** While the user interacts with the town, the system allows him to deselect an object out of a list of several objects from a store from the town.

**Rationale:** To help the user erase object n from a SELECT object 1, object 2, ..., object n FROM store query in a graphical way.

**Requirement type:** Functional

**Fit criterion:** A user is able to deselect several objects from a store from the town.

**Use cases:** “Deselect objects” use case

**Dependencies:** 1-12, 14-19, 21, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 22

**Requirement description:** While the user interacts with the town, after having selected a store and object the site allows him to select one field of a store from the town.

**Rationale:** To help the user simulate a SELECT field FROM object graphically

**Requirement type:** Functional

**Fit criterion:** A user is able to select one fields of a store from the town.

91
Use cases: “Select fields” use case

Dependencies: 1-12, 14-16, 23-26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 23

Requirement description: While the user interacts with the town, after having selected a store and object the site allows him to deselect one field of a store from the town.

Rationale: To help the user erase a SELECT fields FROM object query graphically

Requirement type: Functional

Fit criterion: A user is able to deselect the field of an object from the town.

Use cases: “Deselect fields” use case

Dependencies: 1-12, 14-16, 22, 24-26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 24

Requirement description: While the user interacts with the town, the site allows him to select several fields of a store from the town.

Rationale: To help the user simulate a SELECT field 2, field 2, ..., field n FROM object query graphically

Requirement type: Functional

Fit criterion: A user is able to select several fields of a store from the town.

Use cases: “Select fields” use case


Requirement number: 25

Requirement description: While the user interacts with the town, the site allows him to deselect several fields of a store from the town.

Rationale: To help the user erase field n from a SELECT field 1, field 2, ..., field n FROM object query graphically

Requirement type: Functional

Fit criterion: A user is able to deselect several field of a store from the town.

Use cases: “Deselect fields” use case

Dependencies: 1-12, 14-16, 22-24, 26, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 26

Requirement description: While the user interacts with the town, the site allows him to retrieve query information

Rationale: To help the user understand a query by observing it as a result of being simulated graphically.

Requirement type: Functional

Fit criterion: A user is able to retrieve query information.

Use cases: “Get query” use case

Dependencies: 1-12, 14-25, 28-32, 34, 46, 48-50, 52, 55, 57-58, 64-65, 67-68, 83-89

Requirement number: 27

Requirement description: The site allows the user to write a query

Rationale: To allow him to practice what he learns
Requirement type: Functional
Fit criterion: A user is able to enter a query.
Use cases: “Write query” use case

Requirement number: 28
Requirement description: The site uses an SQL engine to execute a query
Rationale: To allow the user to retrieve the results of his query
Requirement type: Functional
Fit criterion: The query gets executed
Use cases: “Get query results” use case

Requirement number: 29
Requirement description: The site allows the user to retrieve the results of a query
Rationale: To allow him to see the results of the query either so that he checks how they look like or to check if he has solved a task correctly.
Requirement type: Functional
Fit criterion: A user is able to retrieve the results to a query
Use cases: “Get query results” use case

Requirement number: 30
Requirement description: The site allows the users to do a tutorial.
Rationale: To teach the user his first lessons in SQL.
Requirement type: Functional
Fit criterion: A user is able to do a tutorial.
Use cases: “Do tutorial” use case

Requirement number: 31
Requirement description: While the user does a tutorial, the site allows him to navigate through the list of lessons.
Rationale: To help the user pass to a new lesson or go back to a previous one (in case he has not fully understood the last lesson).
Requirement type: Functional
Fit criterion: A user is able to move backwards and forwards in the lessons in the tutorial
Use cases: “Move between lessons” use case

Requirement number: 32
Requirement description: The site allows the users to get trained in SQL.
Rationale: To allow the user to practice his skills
Requirement type: Functional
Fit criterion: A user is able to get trained on the site
Use cases: “Do training” use case


Requirement number: 33

Requirement description: While the user does training, the site allows him to move forward in the list of tasks.

Rationale: To help the user proceed to a new task when he either solved a previous one or chooses to skip the previous one

Requirement type: Functional

Fit criterion: A user is able to move forwards in the list of tasks in the tutorial

Use cases: “Move to next task” use case

Dependencies: 1-11, 32, 65, 67, 83-89

Requirement number: 34

Requirement description: The site allows the users to explore the town without having to do a tutorial or training.

Rationale: To allow the user to experiment with the town freely.

Requirement type: Functional

Fit criterion: A user is able to explore the town

Use cases: “Explore town” use case

Dependencies: 1-29, 65, 67, 83-89

Requirement number: 35

Requirement description: The site allows teachers to add a new tutorial.

Rationale: To allow the teacher to add his personalised tutorial for the first SQL classes.

Requirement type: Functional

Fit criterion: A teacher is able to add a new tutorial

Use cases: “Add tutorial” use case


Requirement number: 36

Requirement description: As part of adding a new tutorial, the system allows the teacher to enter the tutorial’s name.

Rationale: To allow the teacher to show a certain tutorial name to the learners.

Requirement type: Functional

Fit criterion: A teacher is able to enter the tutorial’s name

Use cases: “Add tutorial” use case


Requirement number: 37

Requirement description: As part of adding a new tutorial, the system allows the teacher to decide whether the tutorial is to be active/inactive.

Rationale: To allow the teacher to decide whether to show a certain tutorial to the learners.
Requirement type: Functional
Fit criterion: A teacher is able to choose whether the tutorial is active/inactive
Use cases: “Add tutorial” use case

Requirement number: 38
Requirement description: The site allows teachers to erase a tutorial
Rationale: To allow the teacher to take out of the list of tutorials the ones which he does not need any more.
Requirement type: Functional
Fit criterion: A teacher is able to erase a tutorial
Use cases: „Erase tutorial” use case

Requirement number: 39
Requirement description: The site allows teachers to edit a tutorial’s configuration
Rationale: To allow the teacher to change the tutorial’s name/activity status.
Requirement type: Functional
Fit criterion: A teacher is able to edit a tutorial’s configuration
Use cases: „Edit tutorial configuration” use case

Requirement number: 40
Requirement description: As part of editing their tutorial’s configuration, the system allows the teacher to change the tutorial’s name.
Rationale: To allow the teacher to correct a misspelled tutorial name/ change his mind about the name.
Requirement type: Functional
Fit criterion: A teacher is able to change a tutorial’s name.
Use cases: „Edit tutorial configuration” use case

Requirement number: 41
Requirement description: As part of editing their tutorial’s configuration, the system allows the teacher to change the tutorial’s activity status
Rationale: To allow the teacher to change his mind about the tutorial’s activity status
Requirement type: Functional
Fit criterion: A teacher is able to change a tutorial’s activity status
Use cases: „Edit tutorial configuration” use case

Requirement number: 42
Requirement description: The site allows teachers to edit a tutorial’s list of lessons
Rationale: To allow the teacher to write lessons, configure them to optionally use certain keywords or require user task, decide the order in which they appear

Requirement type: Functional

Fit criterion: A teacher is able to edit a tutorial’s list of lessons

Use cases: „Edit tutorial” use case


Requirement number: 43

Requirement description: As part of editing the tutorial’s list of lessons, the system allows teachers to add a new lesson

Rationale: To allow the teacher to write his own lessons.

Requirement type: Functional

Fit criterion: A teacher is able to add a new lesson

Use cases: „Add new lesson” use case


Requirement number: 44

Requirement description: As part of adding a new lesson, the system allows the user to enter the lesson’s text.

Rationale: To allow the teacher to decide what contents the lesson should have.

Requirement type: Functional

Fit criterion: A teacher is able to enter the lesson’s text

Use cases: „Add new lesson” use case


Requirement number: 45

Requirement description: As part of adding a new lesson, the system allows the teacher to decide on the lesson’s position within the tutorial

Rationale: To allow the teacher to decide the order for his lessons

Requirement type: Functional

Fit criterion: A teacher is able to decide for the order of the lesson within the tutorial

Use cases: „Add new lesson” use case


Requirement number: 46

Requirement description: As part of adding a new lesson, the system allows the teacher to decide to clear all former selections from the town.

Rationale: To allow the teacher to clear the town area so that giving the student a task he needs to solve by looking at the area does not interfere with the student’s previous selections from this area.

Requirement type: Functional

Fit criterion: A teacher is able to decide whether the town area is to be cleared of all former selections as part of the new lesson.

Use cases: „Add new lesson” use case

96
Requirement number: 47

Requirement description: As part of adding a new lesson, the system allows the teacher to decide whether the lesson would block the learner’s advancement in the tutorial until a task is solved.

Rationale: To allow the teacher stop the learner from continuing in the tutorial until he has solved a task and thus learned something.

Requirement type: Functional

Fit criterion: A teacher is able to decide whether the advancement in the tutorial is to be blocked by the new lesson.

Use cases: „Add new lesson” use case


Requirement number: 48

Requirement description: As part of adding a new lesson, the system allows the teacher to request that the tutorial advance automatically when a certain selection from the town area has been performed by the learner.

Rationale: To allow the teacher to make sure the user has made a certain selection from the town area so that the tutorial can continue to the next lesson

Requirement type: Functional

Fit criterion: A teacher is able to request that the tutorial advance automatically when a certain selection from the town area has been performed by the learner.

Use cases: „Add new lesson” use case

Dependencies: 1-12, 14-27, 30-31, 35-43, 52, 57, 60, 66-67, 69, 72, 75, 79, 83-88, 90

Requirement number: 49

Requirement description: As part of adding a new lesson, the system allows the teacher to request that the tutorial advance automatically when a certain task (query) has been solved by the learner.

Rationale: To allow the teacher to make sure the user has solved a certain task so that the tutorial can continue to the next lesson

Requirement type: Functional

Fit criterion: A teacher is able to request that the tutorial advance automatically when a certain task has been performed by the learner.

Use cases: „Add new lesson” use case


Requirement number: 50

Requirement description: The SQL engine must check in the background if a learner executes the query the teacher would want him to execute in order for the tutorial to proceed to the next lesson.

Rationale: To make sure the learner has learned how to do a certain query

Requirement type: Functional

Fit criterion: The SQL query returns to the system whether the query yielded similar results to allow for the proceeding to the next lesson.

Use cases: „Execute teacher query” use case

Requirement number: 51

Requirement description: As part of adding a new lesson, the system allows the teacher to request that certain SQL keywords be highlighted for the user during the display of that lesson.

Rationale: To allow the teacher to give to the user a clear highlighting of the SQL keywords he is teaching in the lesson or give clues as to what SQL keywords to be used to solve query given within the lesson.

Requirement type: Functional
Fit criterion: A teacher is able to request that certain SQL keywords be highlighted for the user during the display of that lesson.

Use cases: „Add new lesson” use case


Requirement number: 52

Requirement description: As part of editing the tutorial’s list of lessons, the system allows teachers to edit a previously added lesson

Rationale: To allow the teacher to change the text and options related to the lessons.

Requirement type: Functional
Fit criterion: A teacher is able to edit the previously added lesson

Use cases: „Edit lesson” use case


Requirement number: 53

Requirement description: As part of editing the lesson, the system allows the user to amend the lesson’s text.

Rationale: To allow the teacher to make corrections to the text.

Requirement type: Functional
Fit criterion: A teacher is able to amend the lesson’s text

Use cases: „Edit lesson” use case


Requirement number: 54

Requirement description: As part of editing the lesson, the system allows the teacher to change the lesson’s position within the tutorial

Rationale: To allow the teacher to change the order for his lessons

Requirement type: Functional
Fit criterion: A teacher is able to change the order of the lesson within the tutorial

Use cases: „Edit lesson” use case


Requirement number: 55

Requirement description: As part of editing the lesson, the system allows the teacher to change his former option about clearing all former selections from the town.

Rationale: To allow the teacher to change his decision about clearing the town area so that giving the student a task he needs to solve by looking at the area does not interfere with the student’s previous selections from this area.

Requirement type: Functional
**Fit criterion:** A teacher is able to change his former option about whether the town area is to be cleared of all former selections as part of the new lesson.

**Use cases:** „Edit lesson“ use case

**Dependencies:** 1-11, 14-27, 30-31, 35-43, 46, 60, 66-67, 69, 72, 79, 83-88, 90

**Requirement number:** 56

**Requirement description:** As part of editing the lesson, the system allows the teacher to change his former option on whether the lesson would block the learner’s advancement in the tutorial until a task is solved

**Rationale:** To allow the teacher change his opinion about stopping the learner from continuing in the tutorial until he has solved a task and thus learned something.

**Requirement type:** Functional

**Fit criterion:** A teacher is able to change his former option on whether the advancement in the tutorial is to be blocked by the new lesson.

**Use cases:** „Edit lesson“ use case

**Dependencies:** 1-11, 30-31, 35-43, 47, 60, 66-67, 69, 72, 79, 83-88, 90

**Requirement number:** 57

**Requirement description:** As part of editing a lesson, the system allows the teacher to change his former option about whether to request that the tutorial advance automatically when a certain selection from the town area has been performed by the learner.

**Rationale:** To allow the teacher to change his decision about making sure the user has made a certain selection from the town area so that the tutorial can continue to the next lesson.

**Requirement type:** Functional

**Fit criterion:** A teacher is able to change his former option on whether the tutorial to advance automatically when a certain selection from the town area has been performed by the learner.

**Use cases:** „Edit lesson“ use case

**Dependencies:** 1-12, 14-27, 30-31, 35-43, 48, 60, 66-67, 69, 72, 79, 83-88, 90

**Requirement number:** 58

**Requirement description:** As part of editing the lesson, the system allows the teacher to change his former option on whether the tutorial advance automatically when a certain task (query) has been solved by the learner.

**Rationale:** To allow the teacher to change his opinion about making sure the user has solved a certain task so that the tutorial can continue to the next lesson.

**Requirement type:** Functional

**Fit criterion:** A teacher is able to change his former option about whether the tutorial advances automatically when a certain task has been performed by the learner.

**Use cases:** „Edit lesson“ use case

**Dependencies:** 1-11, 14-31, 35-43, 49, 60, 66-67, 69, 72, 75, 79, 83-88, 90

**Requirement number:** 59

**Requirement description:** As part of editing the lesson, the system allows the teacher to change his former option about requesting that certain SQL keywords be highlighted for the user during the display of that lesson.

**Rationale:** To allow the teacher to change his opinion about giving to the user a clear highlighting of the SQL keywords he is teaching in the lesson or give clues as to what SQL keywords to be used to solve query given within the lesson.
**Requirement type:** Functional

**Fit criterion:** A teacher is able to change his former option about whether certain SQL keywords be highlighted for the user during the display of that lesson.

**Use cases:** „Edit lesson“ use case

**Dependencies:** 1-11, 30-31, 35-43, 50, 60, 66-67, 69-72, 79, 83-88, 90

---

### Requirement number: 60

**Requirement description:** As part of editing the tutorial’s list of lessons, the system allows teachers to erase a previously added lesson

**Rationale:** To allow the teacher to change his opinion on whether the lesson should be displayed to the learner as part of the tutorial.

**Requirement type:** Functional

**Fit criterion:** A teacher is able to erase a previously added lesson

**Use cases:** „Erase lesson“ use case

**Dependencies:** 1-11, 30-31, 35-59, 61, 66-67, 69, 72, 79, 83-88, 90

---

### Requirement number: 61

**Requirement description:** As part of editing the tutorial’s list of lessons, the system allows teachers to move up or down within the tutorial a certain lesson.

**Rationale:** To allow the teacher to change the ordering of the lessons within the tutorial.

**Requirement type:** Functional

**Fit criterion:** A teacher is able to move up or down a lesson within the tutorial.

**Use cases:** „Change lesson order“ use case

**Dependencies:** 1-11, 30-31, 35, 38-39, 42-43, 60, 66-67, 69, 72, 79, 83-88, 90

---

**D. Non-functional Requirements**

1) **Look and Feel Requirements**

### Requirement number: 62

**Requirement description:** The interface to the system is attractive to a teenage audience.

**Rationale:** Because it is such an audience we mainly want it to be used with (high school pupils)

**Requirement type:** Look and feel requirement

**Fit criterion:** Teenagers have a good opinion about the interface.

**Dependencies:** 63, 66-67

### Requirement number: 63

**Requirement description:** The interface to the system is friendly and colourful.

**Rationale:** Because such an interface would attract learners and make learning be more fun

**Requirement type:** Look and feel requirement

**Fit criterion:** The interface uses a large range of colours which go together.

**Dependencies:** 62

### Requirement number: 64

**Requirement description:** The interface to the system is interactive

**Rationale:** To allow users to learn by clicking on different objects in the screen
Requirement type: Look and feel requirement

Fit criterion: The interface allows the clicking on objects in the interface and gives feedback on it.

Dependencies: 12-27

2) Usability and Humanity Requirements

Requirement number: 65

Requirement description: The learning part of the site is easy to use by teenagers

Rationale: To allow this target group to easily use it.

Requirement type: Usability and humanity requirement.

Fit criterion: It is easy for teenage users to use it.

Dependencies: 1-34, 69-76, 78

Requirement number: 66

Requirement description: The tutorial configuration part of the site is easy to use by adults

Rationale: To allow teachers to use it.

Requirement type: Usability and humanity requirement.

Fit criterion: It is easy for adult users to use it.


Requirement number: 67

Requirement description: The site should be usable by people with no former training on it.

Rationale: To allow users to understand how it should be used at first glance.

Requirement type: Usability and humanity requirement.

Fit criterion: It is easy for users to use the site without needing training.

Dependencies: 1-62, 66, 68-76, 78-79

Requirement number: 68

Requirement description: The tutorial provides a clear navigation through the lessons within.

Rationale: To allow the learner to learn each notion within in turn and thus assimilate the information easier.

Requirement type: Usability and humanity requirement.

Fit criterion: The user can easily understand how to navigate through the lessons within the tutorial.

Dependencies: 1-12, 14-31, 66-67, 69-72

Requirement number: 69

Requirement description: The default tutorial contains clear lessons about SQL basics.

Rationale: To help the user understand his first lessons.

Requirement type: Usability and humanity requirement.

Fit criterion: The user can easily understand the lessons in the default tutorial.

Dependencies: 1-11, 30, 35-61, 65-68, 70-71

Requirement number: 70

Requirement description: The tutorial offers a means of highlighting the SQL keywords which are taught within the lesson the user is currently reading.
**Rationale:** To highlight what is required for the user to remember/use in his task.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user can easily perceive the highlighted SQL keywords and these keywords help him understand what is taught in the lesson he is currently reading.

**Dependencies:** 1-11, 30, 35, 38-39, 42-43, 50, 59, 65-69, 71

---

**Requirement number:** 71

**Requirement description:** The tutorial offers a means of highlighting the SQL keywords which are required to be used by the user in order to solve a task.

**Rationale:** To give the user hints about how he can solve the task

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user can easily perceive the highlighted SQL keywords in the tutorial and these keywords help him understand what is required from him for solving the task.

**Dependencies:** 1-11, 30, 35, 38-39, 42-43, 50, 59, 65-70

---

**Requirement number:** 72

**Requirement description:** The training tasks conform to what is taught in the default tutorial and what could maximally be covered by the teacher-configured tutorials

**Rationale:** To allow the user to practice exactly what he has learned as part of the tutorial.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user finds the same problematic as taught in the tutorial in the training

**Dependencies:** 1-11, 32, 35-62, 65-68

---

**Requirement number:** 73

**Requirement description:** The training tasks are clear and easy to understand by a beginner.

**Rationale:** To allow the user to solve them.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user clearly understands the tasks in the training

**Dependencies:** 1-11, 32, 65-67, 74-76, 78

---

**Requirement number:** 74

**Requirement description:** The training offers a means of highlighting the SQL keywords which are required to be used by the user in order to solve a task.

**Rationale:** To give the user hints about how he can solve the task from the training.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user can easily perceive the highlighted SQL keywords in the training and these keywords help him understand what is required from him for solving the task.

**Dependencies:** 1-11, 32, 65-67, 73

---

**Requirement number:** 75

**Requirement description:** The system provides clear feedback on whether the user has solved a task from the tutorial or training.

**Rationale:** To help the user understand if he has done well and whether in the tutorial he is allowed to advance (if the blocking option had been set by the teacher)

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** A user can easily perceive feedback from his solution to solving a task.
Requirement number: 76

Requirement description: The system provides clear error messages if a user has entered an incorrect query.

Rationale: To show the user that he has made a mistake and give an indication about the kind of mistake he has done.

Requirement type: Usability and humanity requirement.

Fit criterion: A user always understands the error messages he receives when writing incorrect queries and knows what part of the query he has written incorrectly.


Requirement number: 77

Requirement description: The system provides an easily accessible help facility.

Rationale: If a user becomes confused about any part of the system, there should be a help facility to guide him through the process.

Requirement type: Usability and humanity requirement.

Fit criterion: A user of the system always finds assistance about any part of the system easily.

Dependencies: -

Requirement number: 78

Requirement description: The system provides a syntax remainder facility.

Rationale: If a user forgets the SQL syntax at any point, there should be a facility to remind it to him.

Requirement type: Usability and humanity requirement.

Fit criterion: A user of the system will have access at any moment to the syntax required to solve a task.

Dependencies: 1-11, 65-67, 73

Requirement number: 79

Requirement description: The system provides clear confirmation messages to the actions undertaken by the teacher.

Rationale: To confirm to the teacher that his requested actions have been performed.

Requirement type: Usability and humanity requirement.

Fit criterion: The teacher can easily perceive the confirmation messages and be sure his actions have been successfully confirmed.

Dependencies: 1-11, 35-61, 66-67

Requirement number: 80

Requirement description: The site provides clear navigation throughout its pages by using clear link labels and buttons.

Rationale: To allow the user to seamlessly navigate without wondering how to get somewhere.

Requirement type: Usability and humanity requirement.

Fit criterion: The user can easily navigate throughout the site.

Dependencies: -

Requirement number: 81
**Requirement description:** The site provides a clear message about its purpose and the functions it supports.

**Rationale:** To allow users accessing it for the first time to easily understand what they can do on it.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user can easily understand what he can do on the site.

**Dependencies:** -

Requirement number: 82

**Requirement description:** The site provides clear error messages when a user is dealing with his account and entering incorrect or incomplete information.

**Rationale:** To allow for the user to understand where he has made a mistake in dealing with his account.

**Requirement type:** Usability and humanity requirement.

**Fit criterion:** The user can easily perceive and understand the error messages given when he is dealing with his account.

**Dependencies:** 1-11

3) Performance Requirements

Requirement number: 83

**Requirement description:** The system has an acceptable response time.

**Rationale:** Fast response times increase productivity of all users on the system.

**Requirement type:** Performance requirement.

**Fit criterion:** The system responds to any request within 10 seconds.

**Dependencies:** 1-61

Requirement number: 84

**Requirement description:** The system does not require many hardware resources.

**Rationale:** To allow for even users with older (up to 6-year old) computers to use it.

**Requirement type:** Performance requirement.

**Fit criterion:** The system runs on a Pentium 4 processor.

**Dependencies:** 1-61

Requirement number: 85

**Requirement description:** The system supports at least 40 users using it at the same time.

**Rationale:** To allow for a class of learners and a few teachers to use it simultaneously

**Requirement type:** Performance requirement.

**Fit criterion:** The system runs without crashing with at least 40 users.

**Dependencies:** 1-61

4) Maintainability Requirements

Requirement number: 86

**Requirement description:** The system runs on any platform

**Rationale:** Customers using the system may be using a variety of operating systems, and the system must accommodate these.
Requirement type: Maintainability and support requirement.
Fit criterion: The system runs on Windows, SOLARIS MAC and Linux platforms.
Dependencies: 1-61

Requirement number: 87
Requirement description: The system runs on any web browser
Rationale: Learners and teachers may be using a variety of web browsers, and the system must accommodate these.
Requirement type: Maintainability and support requirement.
Fit criterion: The system runs on Microsoft Internet Explorer, Mozilla Firefox, Opera, Apple Safari, Google Chrome.
Dependencies: 1-61

5) Security Requirements

Requirement number: 88
Requirement description: The system protects user account information from unauthorized users.
Rationale: Users should have their account as personal.
Requirement type: Security requirement.
Fit criterion: The System complies with the UK Data Protection Act 1998.
Dependencies: 1-61, 89-90

Requirement number: 89
Requirement description: The system allows only logged in the users to access the learning environment
Rationale: Only learners and teachers who use the site for the study of SQL should use it.
Requirement type: Security requirement.
Fit criterion: Users can only access the learning environment after they have logged in
Dependencies: 1-34, 88

Requirement number: 90
Requirement description: The system allows only logged in teachers to access the tutorial configuration area.
Rationale: Only teachers who are logged in as SQL teachers should be allowed to create tutorials for their learners.
Requirement type: Security requirement.
Fit criterion: Teachers can only access the tutorial configuration area once they have logged in.
Dependencies: 1-11, 35-61, 88

Requirement number: 91
Requirement description: The system does not allow queries other than SELECT queries to be executed.
Rationale: We assumed that only SELECT queries will be taught as the basics of SQL.
Requirement type: Security requirement.

Fit criterion: Other queries are rejected with error messages and do not reach the SQL engine.

Dependencies: 1-11, 28-29

Requirement number: 92

Requirement description: The system does not show the real names of the tables and their contents within the site.

Rationale: To cater for the possibility of users trying to crack the database.

Requirement type: Security requirement.

Fit criterion: The site only shows tables and the objects within graphically.

Dependencies: 1-11, 28-29

Requirement number: 93

Requirement description: The system does not allow the user to enter symbols which can interfere with the database in the URL

Rationale: To cater for the possibility of users trying to crack the database.

Requirement type: Security requirement.

Fit criterion: The system replaces symbols from the URL with something which does not harm.

Dependencies: -

E. Project Issues

1. Open Issues

It remains an issue whether:

1. The teachers will want to build their own queries to be tested with the learner in training.

2. The teachers will want to use the tutorial only as an additional tool to the class or as a complete tool for learning SQL (which could be used to learn it even from home).

3. The teachers will need extra functions such as seeing reports on their pupils’ performance in the class or even give their students homework or tests.

4. A potential high school using this system would like to keep it only available within its intranet or be widely available to be used by the learners from home as well.

5. A potential high school using this system would not want for the system to offer access rights to its teachers automatically without requiring them to log in.

2. Waiting Room

As extensions possibilities for this system:

1. Adding support for joins.

2. Adding support for other types of queries: CREATE, UPDATE, INSERT INTO, DROP, DELETE FROM.

3. Adding the possibility for students to solve complex tasks as a team work.

4. Making queries in training be differentiated according to the learner’s level any maybe as a response to the level acquired after each tutorial is done by him.

5. Making queries in training advance the learner to a different level and become harder according to this level.

6. Allowing students to compete against each other by proposing queries to one another.
APPENDIX B - Use-case Description

<table>
<thead>
<tr>
<th>Use Case 1</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows users to register on the SQL Town site so that they are allowed to log in and learn SQL by using it</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>User has access to the site and chooses to register</td>
</tr>
<tr>
<td><strong>Success Condition</strong></td>
<td>User is successfully registered and is able to log in and start using the learning facility.</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>As user did not provide the system with the necessary details or has provided them in the wrong format, he is not registered and he is given error messages.</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>User</td>
</tr>
<tr>
<td><strong>Secondary Actors</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>User chooses to register</td>
</tr>
</tbody>
</table>

### Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User chooses to register</td>
</tr>
<tr>
<td>2</td>
<td>User enters his personal details: first name, last name and email address</td>
</tr>
<tr>
<td>3</td>
<td>User enters username and twice the password</td>
</tr>
<tr>
<td>4</td>
<td>User confirms having read the terms and conditions</td>
</tr>
<tr>
<td>5</td>
<td>The system checks if all the required information has been filled in, if the email address has the right format and if the two entered passwords are identical</td>
</tr>
<tr>
<td>6</td>
<td>The system thanks the user for joining SQL Town</td>
</tr>
<tr>
<td>7</td>
<td>The system lets the user to log in</td>
</tr>
</tbody>
</table>

### Variations or Extensions

#### Variations:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>If any of the required information is missing the system tells the user to fill it in or it does not continue</td>
</tr>
<tr>
<td>5.2</td>
<td>If any of the required email address was provided in a wrong format, the system prompts the user to correct it or it does not continue.</td>
</tr>
<tr>
<td>5.3</td>
<td>If user enters two passwords that do not match, system prompts him to enter them again</td>
</tr>
</tbody>
</table>

Use Case 2

<table>
<thead>
<tr>
<th>Use Case 2</th>
<th>Log in</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows the user to log in the system, so that he is allowed to use it for learning SQL</td>
</tr>
</tbody>
</table>
### Use Case 2

**Modify account details**

**Goal in Context**
This use case allows users to modify their account details in case there was a name misspelling or if they want to change their username and password.

**Preconditions**
Users are logged in

**Success End Condition**
The user’s registration details are updated.

**Failed End Condition**
User details are not modified because the user has not provided all the required information or has provided it in the wrong format.

**Primary Actor**
User

**Secondary Actors**

**Trigger**
User presses the button requesting for his registration details to be modified.

**Description**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User requests to modify his account details.</td>
</tr>
<tr>
<td>2</td>
<td>User makes necessary changes.</td>
</tr>
<tr>
<td>3</td>
<td>User chooses to save the changes.</td>
</tr>
<tr>
<td>4</td>
<td>System checks if the user has filled in the compulsory information and if he has used the appropriate format.</td>
</tr>
<tr>
<td>5</td>
<td>System confirms to user the change of his account details</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

Variations:

- 4.1 If the changes do not respect the format requested email address field, the system...
prompts the user to correct them and does not allow the user to save them (keeps the old ones and gives error message).

| 4.2 | If the user emptied compulsory fields, the system prompts him to fill them in and does not allow the user to save them (keeps the old ones and gives error message). |
| 4.3 | If the user decided to modify his password and provides two different passwords, the system prompts him that the passwords are different and does not allow the user to save them (keeps the old ones and gives error message). |

### Use Case 4 Interact with town

**Goal in Context**
This use case allows users to interact with the elements in the town

**Preconditions**
Users have logged in

**Success End Condition**
The user successfully manages to interact with the elements in the town

**Failed End Condition**
-

**Primary Actor**
User

**Secondary Actors**

**Trigger**

**Description**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User moves on map</td>
</tr>
<tr>
<td>2</td>
<td>User chooses elements</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

### Use Case 5 Move on map

**Goal in Context**
This use case allows users to move the map so that it focuses on the area with the objects they need

**Preconditions**
Users have logged in

**Success End Condition**
The use gets the area with the objects he needs as central on the screen

**Failed End Condition**
-

**Primary Actor**
User

**Secondary Actors**
## Use Case 6
### Choose elements

#### Goal in Context
This use case allows users to generate an SQL query automatically by choosing the elements from the town they want to appear within.

#### Preconditions
Users have logged in

#### Success End Condition
The user successfully generates the SQL query automatically by choosing elements from the town.

#### Failed End Condition
- 

#### Primary Actor
User

#### Secondary Actors

### Trigger

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>User chooses elements</td>
<td>1</td>
<td>User chooses elements</td>
</tr>
<tr>
<td>User gets query as automatically generated</td>
<td>2</td>
<td>User gets query as automatically generated</td>
</tr>
</tbody>
</table>

### Variations or Extensions

#### Extensions

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>User makes selections</td>
</tr>
<tr>
<td>1.2</td>
<td>User makes deselections</td>
</tr>
</tbody>
</table>

## Use Case 7
### Make selections

#### Goal in Context
This use case allows users to select the store/objects/fields, he needs to appear in the query

#### Preconditions
Users have logged in

#### Success End Condition
The user successfully selects all the objects and, optionally, fields he wants to use for the query

#### Failed End Condition
- 

#### Primary Actor
User

#### Secondary Actors
Use Case 8  
Select store

Goal in Context  
This use case allows users to select the store (only one allowed) they want to appear in the query

Preconditions  
Users have logged in

Success End Condition  
The user successfully selects the store he wants to use for the query

Failed End Condition  
-

Primary Actor  
User

Secondary Actors

Trigger

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User selects store</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>System adds store to query</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>User selects object</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>System adds object to query</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>User selects field</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>System adds field to query</td>
</tr>
</tbody>
</table>

Variations or Extensions

Use Case 9  
Select objects

Goal in Context  
This use case allows users to select the objects that they want to appear in the query

Preconditions  
Users are logged in and have selected a store

Success End Condition  
The user successfully selects the objects to appear in the query

Failed End Condition  
-

Primary Actor  
User

Secondary Actors
Use Case 10  
**Select fields**

**Goal in Context**
This use case allows users to select the fields of the store that they want to appear in the query.

**Preconditions**
Users are logged in and have selected a store.

**Success End Condition**
The user successfully selects the fields of the store to appear in the query.

**Failed End Condition**
- 

**Primary Actor**
User

**Secondary Actors**

**Trigger**

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User clicks on object</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>System opens object</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>System adds the object to the query after WHERE (if more than one object selected, uses &quot;AND&quot;)</td>
<td></td>
</tr>
</tbody>
</table>

**Variations or Extensions**

Use Case 11  
**Make deselections**

**Goal in Context**
This use case allows users to deselect the store/ objects/fields, they do not need to appear in the query any more.

**Preconditions**
Users are logged in and have selected elements.

**Success End Condition**
The user successfully deselects the store/ objects/ fields he does not want to use for the query any more.

**Failed End Condition**
- 

**Primary Actor**
User

**Secondary Actors**

**Trigger**

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User opens store fields</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>User clicks on field</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>System adds the field to the query after SELECT (if more than one field selected, uses &quot;*,&quot;)</td>
<td></td>
</tr>
</tbody>
</table>

**Variations or Extensions**
<table>
<thead>
<tr>
<th>Secondary Actors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger</td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Step</strong></td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

<table>
<thead>
<tr>
<th>Use Case 12</th>
<th>Deselect store</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows users to deselect the store if they want to erase the query</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>Users are logged in and have previously selected a store</td>
</tr>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The user successfully deselects the store and the query is erased.</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>User</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Secondary Actors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trigger</td>
</tr>
<tr>
<td><strong>Description</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**Variations or Extensions**

<table>
<thead>
<tr>
<th>Use Case 13</th>
<th>Deselect objects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows users to deselect the objects that they do not want to appear in the query any more.</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>Users are logged in, have selected a store and have selected an object from it.</td>
</tr>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The user successfully deselects the objects, which are taken out of the query.</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>User</td>
</tr>
</tbody>
</table>
### Secondary Actors

<table>
<thead>
<tr>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Step</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

### Variations or Extensions

<table>
<thead>
<tr>
<th>Use Case 14</th>
<th>Deselect fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal in Context</td>
<td>This use case allows users to deselect the fields of the store that they do not want to appear in the query any more.</td>
</tr>
<tr>
<td>Preconditions</td>
<td>Users are logged in, have selected a store and have selected a field.</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The user successfully deselects the fields of the store which he wants to be removed from the query</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>User</td>
</tr>
</tbody>
</table>

### Use Case 15 | Write query |
<p>|Goal in Context| This use case allows users to enter the query manually. |
|Preconditions| Users are logged in. |
|Success End Condition| The user successfully writes the query |</p>
<table>
<thead>
<tr>
<th>Failed End Condition</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Actor</td>
<td>User</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td>-</td>
</tr>
<tr>
<td>Trigger</td>
<td>-</td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Variations or Extensions</td>
<td>-</td>
</tr>
</tbody>
</table>

Use Case 16  Get query

Goal in Context  This use case allows users to produce the text of the query

Preconditions  Users are logged in.

Success End Condition  The user successfully produces the text of the query

Failed End Condition  -

Primary Actor  User

Secondary Actors  -

Trigger  -

Description  | Step | Action |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User produces the text of the query</td>
</tr>
</tbody>
</table>

Variations or Extensions  Extensions

Step  | Action
|------|--------|
| 1.1  | User writes query (manually)
| 1.2  | User chooses elements (generating the automatically production of the text of the query)

Use Case 17  Compute query

Goal in Context  This use case allows users to retrieve the results of an executed query

Preconditions  Users are logged in and they have produced the text of the query (either by manually writing it or automatically by choosing elements)

Success End Condition  The user successfully retrieves the results of the query

Failed End Condition  -

Primary Actor  User
<table>
<thead>
<tr>
<th>Secondary Actors</th>
<th>SQL engine</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Trigger</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User requires to see the results of the query</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>System retrieves query and checks it for correct syntax</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>System obtains query result from SQL engine</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>System returns query results to user</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variations or Extensions</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.1</td>
<td>If syntax is incorrect (is other than SELECT) system returns error message to user</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Case 18</th>
<th>Execute user query</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Goal in Context</th>
<th>This use case allows the SQL engine to compute the user’s query</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Preconditions</th>
<th>The user is logged in and has produced the text of the query.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Success End Condition</th>
<th>The SQL engine successfully computes the user’s query</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Failed End Condition</th>
<th>-</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Primary Actor</th>
<th>SQL engine</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Secondary Actors</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Trigger</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>The system gives correct names to the tables in the user query</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>The system gives the user’s query to the SQL engine</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>The SQL-engine computes the query</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variations or Extensions</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Use Case 19</th>
<th>Give results to user query</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Goal in Context</th>
<th>This use case allows the SQL engine to return the results of the user’s query to the system</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Preconditions</th>
<th>The SQL engine has computed the query.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Success End Condition</th>
<th>The SQL engine successfully returns the results of the query to the system</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Failed End Condition</th>
<th>-</th>
</tr>
</thead>
</table>
### Use Case 20: Execute task query

**Goal in Context**
This use case allows the SQL engine to compute the generated task for training query.

**Preconditions**
The user is doing a task.

**Success End Condition**
The SQL engine successfully computes the training’s task query.

**Failed End Condition**
-.

**Primary Actor**
SQL engine

**Secondary Actors**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The system gives the training’s task query to the SQL engine</td>
</tr>
<tr>
<td>2</td>
<td>The SQL engine computes the query</td>
</tr>
</tbody>
</table>

### Use Case 21: Give results to task query

**Goal in Context**
This use case allows the SQL engine to return the results of the training’s task query to the system.

**Preconditions**
The SQL engine has computed the query.

**Success End Condition**
The SQL engine successfully returns the results of the query to the system.

**Failed End Condition**
-

**Primary Actor**
SQL Engine

**Secondary Actors**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The system returns the results of the query to the system</td>
</tr>
<tr>
<td>2</td>
<td>The system hides the real table names for the user not to see</td>
</tr>
<tr>
<td>Trigger</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>

### Variations or Extensions

<table>
<thead>
<tr>
<th>Use Case 22</th>
<th>Do tutorial</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows the user to do a tutorial for learning the basics of SQL.</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>The user has logged in</td>
</tr>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The user successfully uses the tutorial</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>User</td>
</tr>
<tr>
<td><strong>Secondary Actors</strong></td>
<td>SQL Engine</td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>The user chooses to do a tutorial</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

### Variations or Extensions

<table>
<thead>
<tr>
<th>Use Case 23</th>
<th>Move between lessons</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows the user to move through the lessons of the tutorial</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>The user has logged in and is doing a tutorial</td>
</tr>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The user successfully moves through the tutorial's list of lessons</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>User</td>
</tr>
<tr>
<td><strong>Secondary Actors</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Step</td>
</tr>
</tbody>
</table>
Use Case 24  
Move forwards freely

Goal in Context
This use case allows the user to move to the next lesson in the tutorial manually

Preconditions
The user has logged in and is doing a tutorial

Success End Condition
The user successfully move forwards through the tutorial’s list of lessons

Failed End Condition
-

Primary Actor
User

Secondary Actors
Teacher, SQL Engine

Trigger
The user chooses the option to move forwards

Description
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user chooses the option to move forwards</td>
</tr>
<tr>
<td>2</td>
<td>The system moves him to the next lesson</td>
</tr>
</tbody>
</table>

Variations or Extensions
Variations

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>If the teacher had set that the lesson is to be blocked until the user has solved a certain task the user cannot move forwards</td>
</tr>
<tr>
<td>1.2</td>
<td>If the user is at his last lesson in the tutorial he is not allowed to move forwards any more</td>
</tr>
</tbody>
</table>

Use Case 25  
Move backwards freely

Goal in Context
This use case allows the user to move to the previous lesson in the tutorial manually

Preconditions
The user has logged in and is doing a tutorial

Success End Condition
The user successfully moves backwards through the tutorial’s list of lessons

Failed End Condition
-

Primary Actor
User

Secondary Actors
<table>
<thead>
<tr>
<th>Trigger</th>
<th>The user chooses the option to move backwards</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Variations or Extensions</td>
<td>Variations</td>
</tr>
<tr>
<td></td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
</tr>
</tbody>
</table>

**Use Case 26**  
**Move forwards automatically**

**Goal in Context**  
This use case allows the user to move automatically to the next lesson in the tutorial when solving the task required by the teacher for the current lesson

**Preconditions**  
The user has logged in, is doing a tutorial

**Success End Condition**  
The user successfully moves forwards to the next lesson at the completion of the task required by the teacher for the current lesson.

**Failed End Condition**  
-

**Primary Actor**  
User

**Secondary Actors**  
Teacher, SQL Engine

**Trigger**  
The user computed a query

<table>
<thead>
<tr>
<th>Description</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>The result of the user’s query as computed by the SQL engine is checked to have the same format like the query requested by the teacher.</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>The system moves the user to the next lesson automatically</td>
</tr>
</tbody>
</table>

**Variations or Extensions**  
Variations

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>The result of the user’s query as computed by the SQL engine but the format of the query is not equal to the type of query required by the teacher and nothing happens</td>
</tr>
</tbody>
</table>

**Use Case 27**  
**Do training**

**Goal in Context**  
This use case allows the user to get trained for learning the basics of SQL

**Preconditions**  
The user has logged in
Success End Condition

| The user successfully uses the training |

Failed End Condition

- 

Primary Actor

User, SQL Engine

Secondary Actors

Trigger

The user chooses to do training

Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user chooses to do training</td>
</tr>
<tr>
<td>2</td>
<td>The user interacts with the town</td>
</tr>
<tr>
<td>3</td>
<td>The user computes queries</td>
</tr>
<tr>
<td>4</td>
<td>The system compares the result of the user query with the result of the task query</td>
</tr>
<tr>
<td>5</td>
<td>The system gives the user feedback on the outcomes of his query</td>
</tr>
<tr>
<td>6</td>
<td>The user moves to next task</td>
</tr>
</tbody>
</table>

Variations or Extensions

Use Case 28

Move to next task

Goal in Context

This use case allows the user to move to the next task in the training

Preconditions

The user has logged in and is doing a training

Success End Condition

The user successfully moves forwards to the next task

Failed End Condition

- 

Primary Actor

User

Secondary Actors

Trigger

The user chooses the option to move forwards to the next task

Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The user chooses the option to move forwards to the next task</td>
</tr>
<tr>
<td>2</td>
<td>The system moves him to the next task</td>
</tr>
</tbody>
</table>

Variations or Extensions

Use Case 29

Add tutorial

Goal in Context

This use case allows teachers to add a tutorial for the teaching of the basics of SQL to their pupils
### Preconditions

The teacher is logged in

### Success End Condition

The teacher successfully adds a tutorial

### Failed End Condition

-

### Primary Actor

Teacher

### Secondary Actors

#### Trigger

The teacher chooses to add a new tutorial

#### Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The teacher chooses to add a new tutorial</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses the title of the tutorial</td>
</tr>
<tr>
<td>3</td>
<td>The teacher chooses whether tutorial is active/inactive</td>
</tr>
<tr>
<td>4</td>
<td>The teacher submits his options</td>
</tr>
<tr>
<td>5</td>
<td>The system checks if the name has been added</td>
</tr>
<tr>
<td>6</td>
<td>The system confirms that the tutorial was successfully added</td>
</tr>
</tbody>
</table>

#### Variations or Extensions

<table>
<thead>
<tr>
<th>Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
</tr>
<tr>
<td>Move</td>
</tr>
</tbody>
</table>

If the system finds that the title of the tutorial was not added, it prompts the teacher about this and does not allow him to add the tutorial until the title is given.

### Use Case 30

#### Erase tutorial

#### Goal in Context

This use case allows teachers to erase a previously added tutorial (please note that I considered the case where the default tutorial, SQL Basics, is not erased)

#### Preconditions

The teacher is logged in and has previously added the tutorial he now wants to erase

#### Success End Condition

The teacher successfully erases the tutorial

#### Failed End Condition

-

#### Primary Actor

Teacher

#### Secondary Actors

#### Trigger

The teacher chooses to erase a previously added tutorial

#### Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The teacher chooses the tutorial</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses to erase the tutorial</td>
</tr>
<tr>
<td>3</td>
<td>The system confirms with the teacher the</td>
</tr>
</tbody>
</table>
The system confirms that the tutorial was erased

Variations or Extensions

<table>
<thead>
<tr>
<th>Variation</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>If the teacher refuses the tutorial deletion, the tutorial is not deleted.</td>
<td></td>
</tr>
</tbody>
</table>

Use Case 31  
Edit tutorial configuration

Goal in Context
This use case allows teachers to edit the configuration of a previously added tutorial (please note that I considered the case where the default tutorial, SQL Basics, does not have its configuration edited)

Preconditions
The teacher is logged in and has previously added the tutorial he now wants to edit its configuration

Success End Condition
The teacher successfully edits the tutorial’s configuration

Failed End Condition
-

Primary Actor
Teacher

Secondary Actors

Trigger
The teacher chooses to edit the tutorial configuration of a previously added tutorial.

Description

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The teacher chooses the tutorial</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses to edit the tutorial’s configuration</td>
</tr>
<tr>
<td>3</td>
<td>The teacher makes the necessary changes</td>
</tr>
<tr>
<td>4</td>
<td>The system checks that the title field has not been left blank</td>
</tr>
<tr>
<td>5</td>
<td>The system confirms that the tutorial’s configuration was successfully edited</td>
</tr>
</tbody>
</table>

Variations or Extensions

<table>
<thead>
<tr>
<th>Variation</th>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>If the system discovers that the tutorial field was left blank, it prompts the teacher about this and does not save his new tutorial configuration</td>
<td></td>
</tr>
</tbody>
</table>

Use Case 32  
Edit tutorial

Goal in Context
This use case allows teachers to make modifications to the list of lessons and lessons within an already added tutorial (please note that I considered the case where the default
### Tutorial, SQL Basics, does not have its list of lessons and lessons within edited

<table>
<thead>
<tr>
<th><strong>Preconditions</strong></th>
<th>The teacher is logged in and has previously added the tutorial he now wants to edit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The teacher successfully edits the tutorial</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>Teacher</td>
</tr>
<tr>
<td><strong>Secondary Actors</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>The teacher chooses to edit a previously added tutorial.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Step</strong></td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>

### Variations or Extensions

<table>
<thead>
<tr>
<th><strong>Use Case 33</strong></th>
<th><strong>Add lesson</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Goal in Context</strong></td>
<td>This use case allows teachers to create a new lesson as part of a tutorial</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>The teacher is logged in and has previously added the tutorial he now wants to add a lesson to</td>
</tr>
<tr>
<td><strong>Success End Condition</strong></td>
<td>The teacher successfully adds a new lesson to the tutorial</td>
</tr>
<tr>
<td><strong>Failed End Condition</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>Primary Actor</strong></td>
<td>Teacher</td>
</tr>
<tr>
<td><strong>Secondary Actors</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>The teacher chooses to add a new lesson to the tutorial</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Step</strong></td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
</tbody>
</table>
| | 5 | The teacher decides whether the lesson will be accompanied by the highlighting of one or
The teacher decides whether the lesson will be accompanied by the requirement that the next lesson is triggered by the user’s action of executing a certain task.

The teacher decides whether the lesson will be accompanied by the blocking of the advancement to the next lesson until the user will have solved a certain task.

The system checks that the user has entered the lesson’s title and position.

The system confirms that the new lesson was successfully added to the tutorial.

Variations or Extensions

<table>
<thead>
<tr>
<th>Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step</strong></td>
</tr>
<tr>
<td>8.1</td>
</tr>
</tbody>
</table>

**Use Case 34**

**Edit lesson**

**Goal in Context**

This use case allows teachers to edit an already added lesson.

**Preconditions**

The teacher is logged in, he has previously added a tutorial and a lesson he now wants to edit the lesson.

**Success End Condition**

The teacher successfully edits the lesson.

**Failed End Condition**

-.

**Primary Actor**

Teacher

**Secondary Actors**

**Trigger**

The teacher chooses to edit a previously added lesson.

**Description**

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The teacher chooses the lesson</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses the edit option</td>
</tr>
<tr>
<td>3</td>
<td>The teacher makes the necessary amendments</td>
</tr>
<tr>
<td>4</td>
<td>The system checks that the lesson’s title and position were not left blank</td>
</tr>
<tr>
<td>5</td>
<td>The system confirms that the lesson has been successfully edited</td>
</tr>
</tbody>
</table>

**Variations or Extensions**

<table>
<thead>
<tr>
<th>Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step</strong></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
If the system discovers the teacher has left the lesson’s title and/or position blanc, it prompts him about this and does not save the changes.

<table>
<thead>
<tr>
<th>Use Case 35</th>
<th>Erase lesson</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal in Context</td>
<td>This use case allows teachers to erase a previously added lesson from a tutorial</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The teacher is logged in, has previously added a tutorial and a lesson for it he now wants to erase</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The teacher successfully erases the lesson</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>Teacher</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>The teacher chooses to erase a previously added lesson</td>
</tr>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
</tr>
<tr>
<td>Variations or Extensions</td>
<td>Variation</td>
</tr>
<tr>
<td></td>
<td>3.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Use Case 36</th>
<th>Change lesson order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal in Context</td>
<td>This use case allows teachers to change the position of a lesson through the tutorial</td>
</tr>
<tr>
<td>Preconditions</td>
<td>The teacher is logged in, has previously added a tutorial and a lesson for it he now wants to move</td>
</tr>
<tr>
<td>Success End Condition</td>
<td>The teacher successfully changed the lesson position</td>
</tr>
<tr>
<td>Failed End Condition</td>
<td>-</td>
</tr>
<tr>
<td>Primary Actor</td>
<td>Teacher</td>
</tr>
<tr>
<td>Secondary Actors</td>
<td></td>
</tr>
<tr>
<td>Trigger</td>
<td>The teacher chooses to change the lesson order.</td>
</tr>
<tr>
<td>Description</td>
<td>Step</td>
</tr>
<tr>
<td></td>
<td>The teacher chooses the lesson</td>
</tr>
<tr>
<td>---</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>2</td>
<td>The teacher chooses to move lesson down or up through the tutorial</td>
</tr>
<tr>
<td>3</td>
<td>The system confirms that the lesson position has change</td>
</tr>
</tbody>
</table>

### Variations or Extensions

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The move is not possible.</td>
</tr>
</tbody>
</table>
APPENDIX C – ER Diagram for Future Work

This ER diagram C1 shows additional tables already created as part of the project in order to sustain future work. The description of these tables is the following:

- **The “user_tables” table** – it is similar with the “main_tables’ table represented in Fig. 4, managing the list of tables and their graphical representation details for the tables created by learners;
- **The “users” table** – already represented in Fig. 4, repeated here to show the point;
- **The “tasks” table** – stores details on a task the teacher can give to users as part of the training area: name, description (the actual task given to the user), solution (the query the user must solve to solve the task) and last_change (allowing for the administrator to see when a last task was added by the by a teacher);
- **The “user_works” table** – stores details about the teacher tasks accessed by the user, which will allow teachers to check what tasks were solved or not by his learners at a later date: solution (the query entered by the users to solve the task), correct (whether the results to this solution conform to the results of the task required by the teacher), date (the date and time the user has accessed the task- no matter if he has solved it or not).
APPENDIX D - Table Creation Statements

CREATE DATABASE `sql_town` DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
USE `sql_town`;

CREATE TABLE IF NOT EXISTS `administrators` (
    `user_id` int(5) NOT NULL AUTO_INCREMENT,
    `first_name` varchar(30) NOT NULL,
    `last_name` varchar(30) NOT NULL,
    `username` varchar(30) NOT NULL,
    `password` varchar(32) NOT NULL,
    `rights` int(1) NOT NULL,
    `email` varchar(40) NOT NULL,
    `last_entrance` datetime NOT NULL,
    PRIMARY KEY (`user_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;

CREATE TABLE IF NOT EXISTS `lessons` (
    `lesson_id` int(5) NOT NULL AUTO_INCREMENT,
    `text` varchar(270) NOT NULL,
    `action` varchar(50) NOT NULL,
    `disableNextButton` varchar(3) NOT NULL DEFAULT 'no',
    `clean_screen` varchar(3) NOT NULL DEFAULT 'no',
    `order_position` int(3) NOT NULL,
    `keywords_shown` varchar(40) NOT NULL,
    `tutorial_id` int(3) NOT NULL,
    `last_change` datetime NOT NULL,
    PRIMARY KEY (`lesson_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;

CREATE TABLE IF NOT EXISTS `main_benchstore` (
    `ID` int(3) NOT NULL AUTO_INCREMENT,
    `Image` varchar(100) NOT NULL,
    `Type` varchar(30) NOT NULL,
    `Name` varchar(50) NOT NULL,
    `xPosition` int(4) NOT NULL,
    `yPosition` int(4) NOT NULL,
    `Color` varchar(40) NOT NULL,
    `Weight` float NOT NULL,
    `Material` varchar(20) NOT NULL,
    `Places` varchar(20) NOT NULL,
    `Price` int(4) NOT NULL DEFAULT '0',
    `shadowWidth` int(4) NOT NULL DEFAULT '53',
    `shadowHeight` int(4) NOT NULL DEFAULT '28',
    `date` datetime NOT NULL,
    PRIMARY KEY (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;

CREATE TABLE IF NOT EXISTS `main_chairstore` (
    `ID` int(3) NOT NULL AUTO_INCREMENT,
    `Image` varchar(100) NOT NULL,
    `Type` varchar(30) NOT NULL,
    `Name` varchar(50) NOT NULL,
    `xPosition` int(4) NOT NULL,
    `yPosition` int(4) NOT NULL,
    `Color` varchar(40) NOT NULL,
    `Weight` float NOT NULL,
    `Material` varchar(20) NOT NULL,
    `Price` int(4) NOT NULL DEFAULT '0',
    `shadowWidth` int(4) NOT NULL DEFAULT '22',
    `shadowHeight` int(4) NOT NULL DEFAULT '35',
    `date` datetime NOT NULL,
    PRIMARY KEY (`ID`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;

CREATE TABLE IF NOT EXISTS `main_chairsuperstore` (
    `ID` int(3) NOT NULL AUTO_INCREMENT,
    `Type` varchar(50) NOT NULL,
    `Image` varchar(100) NOT NULL,
    `Name` varchar(50) NOT NULL,
    `xPosition` int(4) NOT NULL,
    `yPosition` int(4) NOT NULL,
    `Color` varchar(40) NOT NULL,
    `Weight` float NOT NULL,
    `Material` varchar(20) NOT NULL,
CREATE TABLE `main_tables` (
    `id` int(10) NOT NULL AUTO_INCREMENT,
    `name` varchar(50) NOT NULL,
    `symbol` varchar(50) NOT NULL,
    `xPosition` varchar(6) NOT NULL DEFAULT '0',
    `yPosition` varchar(6) NOT NULL DEFAULT '0',
    `date` datetime NOT NULL,
    PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `pages` (
    `page_id` int(2) NOT NULL AUTO_INCREMENT,
    `page_title` varchar(100) NOT NULL DEFAULT '',
    `page` varchar(50) NOT NULL DEFAULT '',
    `address` varchar(100) NOT NULL DEFAULT '',
    `rights` int(1) NOT NULL DEFAULT '0',
    PRIMARY KEY (`page_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
CREATE TABLE `date` datetime NOT NULL, PRIMARY KEY (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `tasks` (
  `task_id` int(5) NOT NULL AUTO_INCREMENT,  
  `task_name` varchar(100) NOT NULL,  
  `description` text NOT NULL,  
  `solution` blob NOT NULL,  
  `supervisor` int(5) NOT NULL,  
  `last_change` datetime NOT NULL,  
  PRIMARY KEY (`task_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `tutorials` (
  `tutorial_id` int(3) NOT NULL AUTO_INCREMENT,  
  `name` varchar(24) NOT NULL,  
  `status` varchar(6) NOT NULL DEFAULT 'hidden',  
  `last_change` datetime NOT NULL,  
  PRIMARY KEY (`tutorial_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `user_tables` (
  `id` int(10) NOT NULL AUTO_INCREMENT,  
  `name` varchar(50) NOT NULL,  
  `symbol` varchar(50) NOT NULL,  
  `owner` int(5) NOT NULL,  
  `xPosition` varchar(6) NOT NULL DEFAULT '0',  
  `yPosition` varchar(6) NOT NULL DEFAULT '0',  
  `date` datetime NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `user_works` (
  `work_id` int(11) NOT NULL AUTO_INCREMENT,  
  `user_id` int(5) NOT NULL,  
  `task_id` int(5) NOT NULL,  
  `solution` blob NOT NULL,  
  `correct` char(1) NOT NULL DEFAULT '0',  
  `date` datetime NOT NULL,  
  PRIMARY KEY (`work_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `users` (
  `user_id` int(5) NOT NULL AUTO_INCREMENT,  
  `first_name` varchar(30) NOT NULL,  
  `last_name` varchar(30) NOT NULL,  
  `username` varchar(30) NOT NULL,  
  `password` varchar(32) NOT NULL,  
  `rights` int(1) NOT NULL,  
  `email` varchar(40) NOT NULL,  
  `completed_tasks` int(4) NOT NULL DEFAULT '0',  
  `last_entrance` datetime NOT NULL,  
  PRIMARY KEY (`user_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;
APPENDIX E - The Object and Field Selection Sequence Diagrams

E1. The user selects an object from the store

The functionality behind the action of the user selecting an object from a store is represented in Fig. E1. This sequence is referred by the sequence regarding the selection of graphical objects represented in Fig. 7, so it departs from the presumption that a store has already been selected (an object cannot be selected from the store without having the store selected first). While the user wants to select objects, if he clicks on an object, if the object was already selected, it is deselected, its status turns from “selected” to “normal” to allow for new selections, its details are closed, its graphical highlighting is removed and the object is removed from the query (from the FROM statement referring to the primary key identifying it from the table corresponding to the selected store). If the object was not already selected, if other objects were selected their object details are closed from the user’s view. The status of the newly clicked on object becomes “selected”, the object is added to the query in the FROM statement, it is graphically highlighted in the user’s view and its details are open for the user to see.

E2. The user selects a field from the store

The functionality behind the action of the user selecting an object from a store is represented in Fig. E2. Again, this sequence diagram was referred from the one in Fig. 7 and departs from the presumption that a store had already been selected. While the user wants to select fields, if he wants to open a field’s area and clicks to open it from the screen, if the field’s area was already opened, it closes, otherwise it opens for the user’s view. When the fields area is opened, if the user clicks on a field from within the fields area, if the field was already selected it is deselected by changing its status from “selected” to “normal”, removing its graphical highlighting and removing it from the list of fields from the SELECT query. If the field was not selected, it becomes selected by changing its status to “selected”, adding the field to the list of fields from the SELECT query and showing graphically that the field was selected (dot in front of its name).

E3. The user moves on the map

The functionality behind the user moving on the map is represented in the sequence diagram from Fig. E3. When the user clicks on the map to move to another position, if a store was already selected (no matter if the objects or fields within were also selected or not), it gets deselected by making its status become “normal” from “selected”, closing its details, removing its graphical highlighting and clearing the query (totally erasing it from the query area). The target position is then set as being the point on the map where the user has clicked. While the centre of the image the user gets is different from this target position, this centre is moved closer to the target position, while also repainting the background and the existing stores from the map.
Fig. E1- The sequence for the user selecting an object from a store
Fig. E2- The sequence for the user selecting a field from the store
Fig. E3- The sequence for the user moving on the map
APPENDIX F - SQL Town Screen Shots

Fig. F1 - “Homepage” with its help page

Fig. F2 - “Login” page with “The email is not valid” error

Fig. F3 - “Tutorials Configuration” with the confirmation from when adding a new tutorial
Fig. F4- “Registration” page error

Fig. F5- A training situation example
Fig. F6- The message which appears while the user didn’t complete a task in the training area

Fig. F7- The message which appears when the user completed a task in the training area

Fig. F8- Error in the learning environment for a user query due its wrong format
APPENDIX G - code from functions.as

//reset all selected elements
function computeManualQuery(input) {
  var errorMessage:String = "";
  var operationType:String = "";
  var auxVar:Array;
  var functionMessage:String;
  auxVar = input.split(" ",1);
  operationType = auxVar[0];
  if(operationType.length==0)
    errorMessage = "Incomplete query. Take a look on the syntax to see how it should look.";
  else{
    switch(operationType.toUpperCase()) {
    case "SELECT" : {
      functionMessage = computeSelectQuery(input);
      if(functionMessage != "Complete")
        errorMessage = functionMessage;
    } break;
    case "INSERT" : errorMessage = "This operation is not covered in the current version of SQL Town."; break;
    case "DELETE" : errorMessage = "This operation is not covered in the current version of SQL Town."; break;
    case "CREATE" : errorMessage = "This operation is not covered in the current version of SQL Town."; break;
    case "SHOW" : errorMessage = "This operation is not covered in the current version of SQL Town."; break;
    default : errorMessage = "Unknown operation.\nTake a look on the syntax to see how a SELECT query should look like."; break;
    }
    if(errorMessage.length>0)
      return errorMessage;
    else
      return "Complete";
  }
  function computeSelectQuery(input) {
    var errorMessage:String = "";
    var auxVar:Array = input.split(" ");
    var selectPos:Number = -1;
    var fromPos:Number = -1;
    var wherePos:Number = -1;
    var functionMessage:String;
    for(var i=0; i<auxVar.length; i++) {
      if(auxVar[i].toUpperCase() == "SELECT")
        selectPos = i;
      if(auxVar[i].toUpperCase() == "FROM")
        fromPos = i;
      if(auxVar[i].toUpperCase() == "WHERE")
        wherePos = i;
    }
    if(selectPos == -1) errorMessage = "Unknown operation.\nTake a look on the syntax to see how a SELECT query should look like.";
    else if(fromPos == -1) errorMessage = "Wrong operation syntax.\nLook for the correct syntax by using the 'Syntax' button on the left of SQL Translator";
    else if(selectPos > fromPos) errorMessage = "Wrong operation syntax.\nLook for the correct syntax by using the 'Syntax' button on the left of SQL Translator";
    else if(wherePos == -1) errorMessage = "Wrong operation syntax.\nLook for the correct syntax by using the 'Syntax' button on the left of SQL Translator";
    else if(wherePos == fromPos + 1) errorMessage = "Wrong operation syntax.\nLook for the correct syntax by using the 'Syntax' button on the left of SQL Translator";
    else return errorMessage;
  }
}
```javascript
if (fromPos == auxVar.length - 1) errorMessage = "Wrong operation syntax. Look for the correct syntax by using the 'Syntax' button on the left of SQL Translator;"
    if (errorMessage == "") return "Complete";
    else return errorMessage;
}

function replaceSubString(input: String, search: String, replace: String): String {
    var output: String = input;
    var pos: Number = -1;
    output = output.slice(0, pos);
    output = output + replace + output.slice(pos + search.length, output.length);
    return output;
}

function cleanQuery(sqlQuery) {
    for (var i = 0; i < conversions.length; i++) {
        sqlQuery = replaceSubString(sqlQuery, conversions[i][0], conversions[i][1]);
    }
    return sqlQuery;
}

function populateSelectedElem(sqlQuery) {
    var varReceiverAux: LoadVars;
    _root.manager.emptySelectedElem();
    sqlQuery = cleanQuery(sqlQuery);
    var varReceiver: LoadVars = new LoadVars();
    varReceiver.load(_root.fullURL + "getQueryResult.php?action=getTables&query=\"+sqlQuery;\";\"+this\[\"PHP_SQL_result\"
            if (this\[\"PHP_SQL_result\"] == \"yes\") {
                if (_root.trainActive == false && _root.learnActive == false) { 
                    objectIds = this.objectIds.split(\".\");
                    for (var i = 0; i < currentTable.houseObjects.length; i++) {
                        for (var j = 0; j < currentTable.houseObjects[i].houseObjects.length; j++) {
                            key = currentTable.houseObjects[i].houseObjects[j].getKeyValue();
                            if (key == objectIds[i]) {
                                // code...
                            }
                        }
                    }
                }
            } else {
                // code...
            }
        }
```
var newCondition =
new SelectCondition(currentTable.houseObjects[j].objDetails[key].detailType, currentTable.objName, objectIds[i], "", "=");

currentTable.houseObjects[j].changeStatus(currentTable.objName, newCondition);

if(this.currentI == this.lastI) {
    _root.play();
}
}

else {
    varReceiverAux = new LoadVars();
    varReceiverAux.load(_root.fullURL +
"getQueryResult.php?action=getResultValues&query="+sqlQuery);
    varReceiverAux.onLoad = function() {
        var newResponseTable:Object = new Object();
        newResponseTable.fields = this['fields'].split("...");
        newResponseTable.values = new Array();
        valueRows = this['values'].split(",,,");
        for(var i=0; i<valueRows.length; i++) {
            var valueSet:Object = new Object();
            values = valueRows[i].split("...");
            newResponseTable.values.push(values);
        }
        _root.queryResponseTable = newResponseTable;
        _root.gotoAndPlay("decisionActiveInactive");
    }
}

function setTrainingQuestionResult(sqlQuery) {
var varReceiverAux:LoadVars;

var varReceiver:LoadVars = new LoadVars();
varReceiver.load(_root.fullURL +
"getQueryResult.php?action=getResultValues&query="+sqlQuery);
varReceiver.onLoad = function() {
    if(this['PHP_SQL_result'] == "yes") {
        var newResponseTable:Object = new Object();
        newResponseTable.fields = this['fields'].split("...");
        newResponseTable.values = new Array();
        valueRows = this['values'].split(",,,");
        for(var i=0; i<valueRows.length; i++) {
            var valueSet:Object = new Object();
            values = valueRows[i].split("...");
            newResponseTable.values.push(values);
        }
        _root.expectedResponseTable = newResponseTable;
        _root.gotoAndPlay("decisionActiveInactive");
    }
}

141
APPENDIX H- The On-line Questionnaire

SQL Town Questionnaire

My Background

My name is Iain Cowie and I am a Masters student in Information Technology with Business in Heriot-Watt University, Edinburgh. As part of my Masters thesis, I considered that it is important to help high school students to easily understand the basic language for relational databases, SQL.

The Purpose of the Project

SQL Town is the project designed as part of my Masters dissertation with the purpose of being a tool for the teaching of SQL within high schools. It is useful for teachers, as they can add tutorials on it and particularly for students, who can learn SQL in a fun, relaying way by using a sunny and friendly town.

The Purpose of the Questionnaire

In order to evaluate my project and understand if it is really useful for high school pupils like yourself, I would like to present you with this questionnaire which will ask you about your opinions and suggestions about the site. The questionnaire covers the main areas of the site: the information page such as the homepage, about page and help, the SQL Basics tutorial and the Training page. If you did not look at all of these pages, please do and then return to this page. If you have, I would be very grateful if you could spare 15 minutes filling in this questionnaire.

The questionnaire is anonymous, so my database will not record any personal data about yourself apart from your age, gender and grade you are in, which will allow me to make sure you are a prospective user of the site. No personal data about yourself will thus be revealed as part of the results of the study.

I will be happy to share the conclusions drawn from all the questionnaires once my dissertation will have been submitted at the end of August, so please do not hesitate to request them from me.

If you agree with the above please press on “Continue”.

Fig. H1- The on-line questionnaire’s information page

SQL Town Questionnaire

Thank you for accepting to answer to this questionnaire. It should not take more than 15 minutes to fill in.

1. Some details about yourself

1. How old are you?

2. What is your gender?
   ○ male  ○ female

3. What grade are you in?

4. Did you know SQL before using this site?
   ○ yes  ○ no

Page: 1/6

Fig. H2- The on-line questionnaire’s first page
2. First contact

I would now like to know what you think about how the website looks like. What are your opinions about:

5. Do you like the colors used in this website?
   - [ ] yes
   - [ ] no

   If the answer was "no", why not?

6. Do you like how the stores and the objects they contain look like?
   - [ ] yes
   - [ ] no

   If the answer was "no", why not?

   Is there anything you could think of which could make them look better?

7. Do you like the environment in which the town is placed?
   - [ ] yes
   - [ ] no

   If your answer was "no", why not?

   Would you have liked something else?

8. Do you like the way information on objects is given in the "Fields" and "Details" drawers?
   - [ ] yes
   - [ ] no

   If the answer was "no", why not?
3. Getting around

I would now like to know how easy you found using the site.

<table>
<thead>
<tr>
<th>Question</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>9. Was it clear for you what are the purposes of this site?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. Have you read the information on the homepage and/or About page?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;yes&quot;, was this information clear enough for you?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11. Was it easy to create an account for yourself?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;no&quot;, where did you have difficulties?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12. Did you easily understand how to move through the town?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;no&quot;, why not?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13. Did you easily understand how to use the SQL Translator (the text area where you add your queries)?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;no&quot;, why not?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14. Was the result of you pressing on &quot;Share&quot; from the SQL Translator always clear?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;no&quot;, why not?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15. Was it always clear that you accomplished didn’t accomplish a task in training?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;no&quot;, why not?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16. Have you ever used the help?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;yes&quot;, what did you need a fear?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If the answer was &quot;yes&quot;, was it easy to find the answer from the help?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page 36</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. H4 - The on-line questionnaire’s third page

144
4. Learning the Basics of SQL

I would now like to know what your experience with learning SQL by using the Basics tutorial was like

<table>
<thead>
<tr>
<th>Question</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>17. Did you understand the lessons from the SQL Basics tutorial?</td>
<td>yes</td>
</tr>
<tr>
<td>18. How easy did you find the lessons in the tutorial?</td>
<td>very hard</td>
</tr>
<tr>
<td>If you consider they were not easy enough, is there anything you would do to change that?</td>
<td></td>
</tr>
<tr>
<td>19. Were the tasks given within the tutorial easy enough to accomplish after having read the information in the tutorial</td>
<td>very hard</td>
</tr>
<tr>
<td>20. What is your opinion about the length of the tutorial?</td>
<td>too long</td>
</tr>
<tr>
<td>21. All in all, how useful was this tutorial for you for learning SQL?</td>
<td>not useful at all</td>
</tr>
<tr>
<td>22. Do you have any suggestion for improving the Basics SQL tutorial?</td>
<td></td>
</tr>
</tbody>
</table>

Page: 4/6

Continue

Fig. H5 - The on-line questionnaire’s fourth page
5. Training in SQL

23. Did you understand the tasks which are given in the Training part of the site?
   - Yes
   - No

24. How easy were the tasks to solve after having learned the SQL Basics tutorial?
   - Very hard
   - Hard
   - So and so
   - Easy
   - Very easy

25. Was the highlighting of keywords useful for you to solve the task?
   - Yes
   - No

Could you suggest something which would help you (even) more?

26. Did you click on the stores and objects within to help you get the results?
   - Yes
   - No

27. All in all, do you think the Training part of the site was useful for you in learning SQL?
   - Not useful at all
   - Not so useful
   - So and so
   - Useful
   - Very useful

28. Do you have any suggestions for improving the Training?

Page: 5/6
Continue

---

6. Overall Opinions

29. Overall, did you like using this site?
   - Not at all
   - No
   - So and so
   - Yes
   - Very much

30. Was it useful for you?
   - Not useful at all
   - Not so useful
   - So and so
   - Useful
   - Very useful

Page: 6/6
Continue

---

Thank you!

For any other comments or suggestions, please contact:
Jesu Cuvacle
Email: nc51@lew.ac.uk

---

Fig. H6 - The on-line questionnaire’s fifth page

Fig. H7 - The on-line questionnaire’s sixth page

Fig. H8 - The on-line questionnaire’s seventh page