NTNU
Norwegian University of
Science and Technology

Adaptive personalized eLearning

Naimdjon Takhirov

Master of Science in Informatics
Submission date: May 2008
Supervisor: Ingeborg Torvik Sølvberg, IDI
Co-supervisor: Bjørn Rustberggard, Inspera AS

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

The popularity of the Internet and recent developments in multimedia technologies have created new possibilities for training and education delivered by online learning. Current learning environments do provide personalization, but this is usually limited to customization of the graphical user interface. The next generation of eLearning systems should tailor the learning experience to each individual's needs and preferences. The traditional "one-size-fits-all" mentality results in high dropout rates, and low levels of motivation and satisfaction. This thesis investigates personalization in an existing eLearning environment, called the Inspera LCMS. Inspera is a software company that produces software products for the educational sector. The objective of the thesis is to study personalization based on prior knowledge and learning style. Through developing a prototype PEDAL-NG on top of the Inspera LCMS personalization was addressed with particular attention to prior knowledge and learning style in order to deliver a tailored learning experience.

This work has found that mapping prior knowledge and learning style is important for constructing personalized learning offerings for students with different levels of knowledge and learning styles. Prior knowledge assessment and a learning style questionnaire were used to assess the knowledge level and learning style. The proposed model for automatic construction of prior knowledge assessment aims to connect questions in the assessment to specific course modules in order to identify levels on different modules, because a student may have varying levels of knowledge within different modules. We have also found that it is not easy to map students' prior knowledge with total accuracy. However, this is not required in order to achieve a tailored learning experience; an assessment of prior knowledge can still be used to decide what piece of content should be presented to a particular student.

Learning style can be simply defined as either the way people learn or an individual's preferred way of learning. The VAK learning style inventory has been found suitable to map the learning styles of students, and it is one of few learning style inventories appropriate for online learning assessment. A questionnaire consisting of 16 questions has been used to identify the learning style of students prior to commencement of the course. It is important to consider the number of questions, because the students may feel reluctant to spend too much time on the questionnaire. However, the user evaluation has shown that students willingly answer questions to allow the system to identify their learning styles.

This thesis employs metadata based models to support adaptive eLearning for personalized course offerings. In this setting, learning objects are dynamically assembled based on learner profiles and metadata associated with learning objects. The prototype PEDAL-NG is developed as a service in the Inspera LCMS and has been integrated into its service layer. Many software packages and technologies have been used in its implementation. Among others, the Lucene search engine was used for in-memory indexing and searching, XML and XSLT for presentation, and Inspera's data model for modeling learners and learning objects.

This work also presents a comprehensive overview of the state-of-the-art pertaining to learning, learning styles, Learning Management Systems, technologies related to web-based personalization and related standards and specifications. A brief comparison is also made of various schools that have tried to address personalization of content for web-based learning.

Finally, for evaluation purposes, a course on *"Designing Relational Databases"* was created, and a group of fourteen users evaluated the personalized course.

Keywords: personalization, adaptive eLearning, learning object, metadata, learning style, prior knowledge, user modeling.

Acknowledgment

This thesis is submitted to the Department of Computer and Information Science, Faculty of Information Technology, Mathematics and Electrical Engineering at Norwegian University of Science and Technology(NTNU) in partial fulfillment of the requirements for the degree of Master of Science in Information Management.

Working with this thesis has been a very interesting and valuable experience to me and I have learned a lot. I want to express my thanks to the people who have been very helpful during the time it took me to write this thesis. First and foremost, I would like to thank my supervisor Professor Ingeborg Torvik Sølvberg with whom I was very fortunate and whose guidance and constructive comments have greatly improved this work. Special thanks also go to Inspera AS and Nordic Booking Systems AS, especially CEO Bjørn Rustberggard and Chairman of the Board Torbjørn Rustberggard for financial aid during the course of my study at NTNU. I also thank Bjørn for his interesting comments and help in choosing this topic of exploration.

Professor Kjell Skogen at the Department of Special Education at the University of Oslo was very kind to dedicate time to meet with me and assist me with the pedagogical issues I encountered during this work. Bjørn Terje Bakken is thanked for his input and discussion on the use of technology to support learning and various related concepts in eLearning from both pedagogical and technical perspectives.

Also, I would like to express gratitude to my colleagues Sondre Skaug Bjørnebekk and John Arne Skjervold Pedersen for useful discussions and brainstorming meetings.

Last, I would like to thank my family for unconditional support, patience, and for simply being there.

Thank you all!

Trondheim, May 30, 2008

Naimdjon Takhirov

Table of Contents

List of Tables

List of Figures

Abbreviations

AICC	Aviation Industry Computer-Based Training Committee
AH	Adaptive Hypermedia
AHS	Adaptive Hypermedia Systems
CMS	Course Management System
DC	Dublin Core
ER	Entity Relationships
JSP	JavaServer Pages
HTML	Hyper Text Markup Language
ITS	Intelligent Tutoring Systems
iContent	Inspera Content Server
ICS	Inspera Content Server
ICT	Information and Communications Technology
IMS SS	IMS Simple Sequencing
IR	Information Retrieval
J2EE	Java 2 Platform, Enterprise Edition
LO	Learning Object
LOR	Learning Object Repository
LOM	Learning Object Metadata
LMS	Learning Management System
LCMS	Learning Content Management System
LSQ	Learning Style questionnaire
MBTI	Myers-Briggs Type Indicator
PEDAL-NG	Personalized Adaptive eLearning - Next Generation
RDBMS	Relational Database Management System
RLO	Reusable Learning Object
SCORM	Sharable Content Object Reference Model
SQL	Structured Query Language
VAK	Visual, Auditory Kinesthetic
VLE	Virtual Learning Environment
XML	Extensible Markup Language
XSLT	EXtensible Stylesheet Language

1 Introduction

In recent years, the concept of web-based learning and the use of Internet in teaching and learning have received increasing attention. People use the Internet and new technologies every day for information, communication, entertainment, obtaining goods and services, and learning. With the increasing availability of the Internet, we are now able to change what and how we deliver the learning experience to students across time or space, which has led to the evolution of eLearning. The global market for eLearning is expected to surpass $52.6 billion by 2010, according to a report by Global Industry Analysts [73].

Electronic learning or eLearning is a general term used to refer to technology-enhanced learning. Because this phrase is used in many contexts, it is important to clarify what is meant by “eLearning”. A number of terms such as e-learning, E-learning or eLearning have also been used to mean the same thing. In many respects, the term is commonly associated with the field of advanced technology to support learning, which deals with both the technologies and associated methodologies in learning to use networked and/or multimedia technologies. The simplest definition of eLearning as used here is the delivery of a learning, training, or educational program by electronic means. In mosts cases, this delivery takes place via the Internet. eLearning is often defined to utilize technology to deliver learning and training services, often in relation to the Internet and mobile learning. Some definitions are listed below:

- “eLearning is any virtual act or process used to acquire data, information, skills or knowledge. In the context of our research, eLearning is enabled learning, learning in a virtual world where technology merges with human creativity to accelerate and leverage the rapid development and application of deep knowledge” [81].
- elearning is “the delivery of a learning, training or education program by electronic means. eLearning involves the use of a computer or electronic device to provide training, educational, or learning material” [70].

Some of the advantages of eLearning over traditional classroom-based training are:

- 24/7/365 accessibility: usually, eLearning applications are accessible at all hours and from all locations and bridges both time and distance gaps;

- just-in-time learning, in which information is provided on demand;
- state-of-the-art collaboration tools;
- delivery that may be either synchronous or asynchronous;

It has been postulated that one of the main problems with eLearning platforms is their lack of personalization. Therefore, the “one-size-fits-all” approach is no longer sufficient. There is a need for a more effective and dynamic learning experience that adapts itself to students’ individual needs, requirements, and preferences. Instead of delivering the same content to all students, eLearning systems should tailor the learning process to each individual’s characteristics in order to increase the relevance and appropriateness of the learning material.

Human beings are different, and as such they learn and process information in different ways. Past experience, prior knowledge, skills, learning style and interests are among the factors that may affect the individuals’ requirements for effective learning [75]. Although the range of eLearning applications has increased substantially both in academia and industry, students in reality are often presented with the same learning style that students have been presented with for 50 years [11]. Learning is a personal and adaptive process, and thus individual student’s needs and preferences should be considered from start to finish [17].

Researchers stress the importance of “adaptation” or “personalization” in learning systems. Systems which allow the user to change certain parameters and adapt their behavior accordingly are called adaptable. Systems that adapt to the users, based on assumptions about the user’s needs are called adaptive [92]. The main idea behind personalization involves delivering a learning experience tailored to students’ individual preferences, needs, interests, and requirements. These properties are usually stored in a learner profile. Using various techniques, the underlying system should adapt itself based on the learner profile. This implies not only personalized selection of content but also adaptation of the sequence of resources and tailoring of presentation styles.

1.1 Motivation

It has long been known that differences among individuals have an effect on learning. These differences include both the past experience and the preferred learning style of learners. If these parameters are not taken into consideration when designing eLearning offerings, motivation and students' comfort levels may suffer reducing overall learning and preventing a desirable learning outcome [23].

The Knowles theory of andragogy [78] makes the following assumptions about the design of learning: (1) adults need to know why they need to learn something, (2) adults need to learn experientially, (3) adults approach learning as problem-solving, and (4) adults learn best when the topic is of immediate value. These assumptions are not easily supported by existing platforms.

The prolifiration of Internet and eLearning opens up new possibilities for learning offerings, since online learning makes it possible for learners to control the learning experience. However, the problems of the "one-size-fits-all" philosophy remain and may result in high dropout rates, low levels of motivation and satisfaction [97]. The next generation of eLearning applications should address this issue. An eLearning course should not be designed in a vacuum; rather, it should match students' needs and desires as closely as possible, and adapt during course progression.

Adapting learning material implies tailoring content based on not only prior knowledge, but also on learning style and objectives. By adapting learning material, we do not mean setting up the course interface according to preferences, like color, theme, or what sections of the page should be visible or are of more interest; these changes are referred to as static adaptation. Once stored, these settings do not require runtime reconciliation of different models. Adapting learning material implies adapting content in runtime by observing learner behavior during the progression of the course. Because this process occurs during runtime, any activity may affect the structure and sequence of the eLearning offering. This is necessary, because individuals have different preferences and goals. Learners may want learning objects to be presented based on their preferences. For example, visual learners would want content to be presented with texts and pictures, auditory learners would prefer for the content to be read by the system (using text-to-speech conversion technology or pre-recorded sound items, for example), and kinesthetic learners would learn best when material is accompanied by exercises so that learners can practice themselves [51].

1.2 Research question

The main research question this thesis attempts to answer is:

- *Is it possible to design, implement, and create an architecture for personalized eLearning on top of an existing system?*

The focus of this thesis is to outline an appropriate design model that could be used to implement and support personalization by integrating it into an existing eLearning environment. Specifically, prior knowledge and learning style are investigated. From the users' perspective it is important to receive feedback regarding what they think about the personalized course offerings. The second research question is:

- *What do users think about a personalized course?*

1.3 Objectives

The primary objectives of this thesis are to 1) investigate personalization based on prior knowledge and learning style and 2) implement personalized adaptive eLearning in a loosely coupled manner from an architectural perspective in an existing eLearning platform. The Inspera LCMS [1] is extended with an independent service component in its *service layer* to incorporate adaptation based on prior knowledge and learning style for personalized course offerings.

1.4 Approach

To answer the research questions outlined in this thesis, the following research activities were undertaken:

- The relevant literature was extensively reviewed. Personalization of eLearning offerings is of a multidisciplinary nature, combining technology, psychology, and pedagogy. Hence, the spectrum of the literature reviewed is rather wide. While this thesis does not include all aspects but instead has a limited scope, a clear understanding of current technologies and strategies related to this work is needed.

- Research was conducted on how to identify the prior knowledge of the learner. Traditionally, this information is gathered through an assessment before the course starts.

[1]LCMS is acronym both for Learning Content Management System and Learning Course Management System, but we use the former in this thesis.

- Learning styles inventories were reviewed. This involved choosing a possible inventory and applying it in the prototype.
- A prototype was developed based on these findings. The prototype is not fully functional and ready to deploy for end-users, and is currently limited to testing and evaluation.
- The prototype was evaluated by testing the prototype on a group of students.

1.5 Contribution

The most significant contributions of this thesis are:

- development of a framework for applying both prior knowledge assessment to identify the knowledge level and a questionnaire to map the learning styles of learners in LMS. The thesis proposes a simple metadata schema for modeling learning objects and users.
- proposal of an abstract model for the automatic and dynamic construction of course components.
- proposal of a model for sequencing learning objects based on learning style and prior knowledge.
- development of a prototype that delivers personalized eLearning by integrating it into an existing LCMS.

1.6 Thesis outline

Chapter 1 presents a basic introduction to the research area, along with the questions to be addressed, approach and contributions of this thesis.

Chapter 2 establishes a theoretical background by conducting a literature review. It discusses various concepts in eLearning and eLearning environments, and gives definitions and theoritecal approaches to personalization in eLearning environments.

Chapter 3 presents an overview of state-of-the-art technologies and related standards in eLearning addressing personalized eLearning. Specifically, two schools of thought are discussed in detail and outlined in terms of their similarities and differences. Various Learning Management Systems (LMSs) are also reviewed briefly in this chapter. Since Inspera's solution requires special attention, it is reviewed in more detail at the end of this chapter.

Chapter 4 summarizes the requirements set for the prototype to be developed as part of this thesis. This chapter includes both functional and non-functional requirements, as well as recommendations gathered from conversations with experts.

Chapter 5 describes the details of the design, architecture, and implementation of the prototype PEDAL-NG on top of the Inspera LCMS. A brief overview of a course utilizing PEDAL-NG is given at the end of the chapter.

Chapter 6 evaluates the prototype, and presents the major findings from a survey conducted on a group of users. A tool for evaluating performance has been used and a method to utulize this tool for load-testing the prototype is described in this chapter.

Chapter 7 concludes this work with a summary of its contributions and limitations.

Chapter 8 proposes a list of topics and areas for further exploration.

2 Background

2.1 Introduction

This chapter presents background information and definitions of important concepts related to this work.

2.2 Learning

Many attempts have been made to define learning, and many of them may leave the reader disappointed. For instance, Watkins defines learning as *"... that reflective activity which enables the learner to draw upon previous experience to understand and evaluate the present, so as to shape future action and formulate new knowledge"* [99]. He mentions that learning is an active process of relating new meaning to existing meaning, which involves the accommodation and assimilation of skills and thoughts. It makes connections between the past, present, and future and is influenced by the use to which the learning is to be put and whether the learning may be effectively retrieved in future situations.

Honey claims that people learn in two different ways [60]. Learning sometimes takes place as a result of formal activities that are structured (e.g., by attending lectures or labs, or by reading books). The second method involves learning through experiences, and it often occurs in an unconscious manner. One might suggest that the former type of learning, which is dedicated to acquisition of knowledge is more familiar and straightforward than the latter.

Merriam-Webster [4], a popular dictionary, has several definitions of learning:

- the act or experience of one that learns;
- knowledge or skill acquired by instruction or study;
- modification of a behavioral tendency by experience (as exposure to conditioning).

All of these definitions describe learning as a potential change in the mental state of human beings. Additionally, experience is often mentioned in definitions of learning. This is of great importance for this thesis, as it is directly related to the identification of the prior knowledge of students. With respect to the definition of learning, it should be mentioned that experience is unique to each individual. This uniqueness is particularly noticeable in learning paradigms [102], as shown in Chapter 2.3.

2.3 Theoretical approaches to personalization of learning paradigms

In order to proceed with a discussion pertaining to learning (especially over the Internet), it is important to mention the role that learning theories play in the process. The decision about which learning theory (behaviorism, cognitivism or constructivism) instructional designers should focus on when designing online courses has never been an easy or straightforward one. It tends to lead to a focus on technology and what it enables in terms of learning [36], and in this way the approach becomes learner- instead of content-centric. The next subchapters present a brief discussion of learning theories and their potential impact on personalized eLearning.

2.3.1 Behaviorism

Behaviourism is primarily concerned with the behaviour of human beings rather than the mental phenomena, and it looks upon learning as acquisition a new behaviour [12]. From an educational perspective, the behaviorist learner is viewed as a passive recipient of knowledge [34]. From the teaching perspective, behaviorism suggests that the the role of a teacher is to strengthen the correct behaviour. Behaviorism is a worldview that operates on a principle of "stimulus-response". Behaviorists consider learning as to involve a strengthening of the relationship between stimuli and responses. Learning is often measured by estimating the probability of a given stimulus producing the correct response. Two major types of conditioning recognized by behaviorists: *classical conditioning* (occurs when a natural reflex responds to a stimulus) [2] and *operant conditioning* (occurs when a response to a stimulus is reinforced) [3].

In applying behaviorism in adaptive learning environments, it suggests that such environments should attempt to cooperate with stimuli that elicit behavior from students related to behavioral patterns which lead to a successful learning experience [96].

[2]Ivan Pavlov is the Nobel Prize winner in Psychology, observed dogs salivate when they eat or even see food.

[3]Burrhus Frederic Skinner, a controversial behaviorist of the 20th century, used reinforcement techniques to teach pigeons to dance and bowl a ball.

2.3.2 Cognitivism

In contrast to behaviorism, cognitivism makes mental processes the primary object of study and views knowledge as symbolic and mentally constructed. People subscribing to this view feel that the focus on behavior demanded by the behaviorist school restricts the usefulness of its theories [101]. While the behaviorist view concentrates on investigating the observable behavior of humans and animals resulting from exposure to different stimuli (e.g., reinforcement, punishment, and conditioning), the cognitivist school takes a different approach that is explicitly "prohibited" in behaviorist experiments.

In cognitive theories, information is received through attention and integrated into memory. This information is then transferred into knowledge and integrated in the learner's cognitive structure for later retrieval via remembering. The process can be divided into the processes:

- receiving - information is received;
- storage - information is stored and integrated into memory;
- retrieval - information is remembered and retrieved.

The cognitivist school makes mental processes the primary object of study and tries to discover and model the learner's mental processes during the learning process. In cognitive theories, knowledge is represented by symbolic, mental constructions in the minds of individuals. Learning thus becomes the process of committing these symbolic representations to memory, where they may be processed.

The cognitive approach has emerged as a new perspective in "information-processing ideas", and it offers an alternative to the behaviorist assumptions that the learner is determined by his environments and therefore passively adapts to different situations. The cognitive approach emphasizes the learner's active mental processing. Just like in the behaviorist school, however knowledge is still viewed as given and absolute [83].

2.3.3 Constructivism

The constructivist theories take on a variety of forms. Constructivists believe that learning takes place through reflecting on experiences. People thus construct their own understanding of the world in which they live. Each person generates his own "rules" and "mental models", which are used to make sense of experiences. Learning is seen as simply the process of adjusting mental models to accommodate new experiences [96]. Knowledge is connected to previous experience in the process of gathering. The primary goal of this view is the idea that learners need to construct their own understanding of each concept, so that the primary role of teaching is not to attempt to "transfer" knowledge by lecturing and explaining but to create situations for learners that will foster their creation of the necessary mental constructions.

The distinction between this view and the other approaches discussed in previous sections is as follows. Behaviorists view knowledge as nothing more than passive, largely automatic responses to external factors in the environment, and cognitivists view knowledge as abstract symbolic representations. In constrast, the constructivist school views knowledge as a constructed entity made by each and every learner through a learning process. Therefore, knowledge cannot be easily transferred from one person to another; it will have to be reconstructed by each person. This stresses that the view of knowledge in the constructivist school differs from the *knowledge as given and absolute* views of behaviorism and cognitivism [31].

2.3.4 Summary and analysis

Behaviorism is based on behavioral changes. It focuses on the repetion of a new behavioral pattern until that pattern becomes permanent. Cognitivism, on the other hand, is based on the thought process behind the behavior. Normally, changes in behavior are observed, but only as an indicator of what is going on in the learner's mental model. Finally, constructivism is based on the premise that human beings each construct their own view of the world based on individual experiences. This view focuses on preparing the learner to solve problems in a variety of situations.

As we have seen, constructivists stress that people learn more with a teacher than from a teacher. Reeves also notes in [95] that students learn more with a computer than from a computer. Therefore, computers may simply be used as a tool to empower students and instructors. Constructivism is concerned with learners' creation of meaning and connection of new concepts to existing knowledge. Both of these processes involve a large degree of autonomy and initiative.

Constructivism emerged at the end of twentieth century [74], and its development is related to developments in educational technology [34]. Constructivism is embraced by many educational technologists [91], a fact reflected in the plethora of multimedia and computer-based software that draw from constructivist premises.

The learning theories discussed above impact on the design and use of personalized learning applications by providing insights into how individuals may learn best using such systems. Reward or punishment online seems to be the very in-frequently practiced, because it stems from the behaviorist approach. However, many of these ideas have been incorporated by cognitivism. Conlan [34] notes that cognitivism is concerned not only with a student's observable behavior, but also with his non-observable mental processes. The approach taken by constructivists is different from that taken by others, because it contends that learners construct their own view of information. This implies that constructivism may be better suited to self-motivated learners. Conlan concludes that a balance between cognitivism and constructivism is achieved as a learner moves from being a novice (prescriptive learning experience) to becoming an expert (more control over learning).

2.4 Learning styles

There are a large number of factors that can influence the extent of learning, and some of these are shown in Figure 2.1.

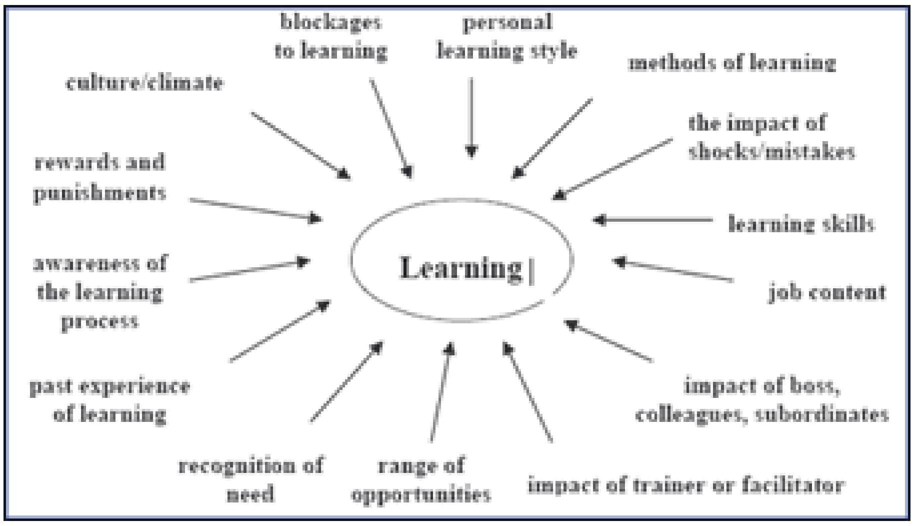
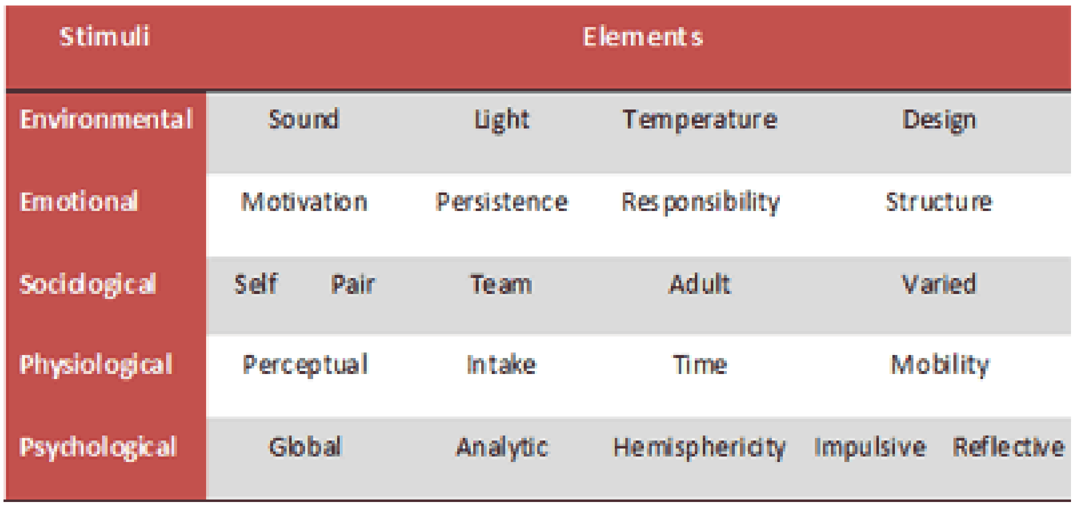


Figure 2.1: Factors that influence the learning process (Source: [29])

The study of learning style in education requires a substantial amount of research and is therefore not a trivial work. It is possible to write an entire dissertation [37] or book [43] on the subject. *Learning style* can be simply defined as the way people learn. Rita and Kenneth Dunn define learning style as the way in which individuals begin to concentrate on, process, internalize, and retain new and difficult information [45]. Some people can learn things that are easy for them without using their learning style [47], but all people learn new and difficult information better when this information is presented according to their learning style. Differences in how individuals learn explain why some people (even within the same family) do well in school while others do not. The differences also explain that it is not an easy task to find a comprehensive model that can reveal the styles of different individuals.

A recent report determined that institutions observe a 20 to 50 percent dropout rate for distance learners [53]. This study concluded that the reasons for dropout were lack of management, lack of motivation, problems with technology, and lack of student support. Frankola proposed that designing content related to a particular topic specifically address the needs of a particular learning style would allow students with this learning style to be more successful.



Stimuli	Elements			
Environmental	Sound	Light	Temperature	Design
Emotional	Motivation	Persistence	Responsibility	Structure
Sociological	Self Pair	Team	Adult	Varied
Physiological	Perceptual	Intake	Time	Mobility
Psychological	Global	Analytic	Hemisphericity	Impulsive Reflective

Figure 2.2: Dunn's learning style model

Research on learning style often attempts to find and construct guidelines on how individuals could be taught in a way that would match their learning styles. Researchers classify learning style in different ways. Dr. Rita Dunn [41] developed a model (Figure 2.2), in which an individual's learning style depends to a large extent on environmental factors (such as noise level, light, and temperature). In addition, emotional (motivation, persistence, responsibility, and structure), sociological (learning alone or in pairs), physiological (auditory, visual, kinesthetic, and learning in the evening/morning) elements also affect learning style. Dr. Dunn insists that no one is affected by all of the 22 elements. Most people are affected by somewhere between 5 and 14 elements , and many are affected by fewer (2 to 6 [41].

Learning style has received special attention in the context of personalizing educational material. A significant part of an individual's learning style is the preference of perceptual channel, which is sometimes referred to as VAK [4], since perceptual preferences are divided into visual, auditory, and kinesthetic [90]. Individuals who excel at theoretical information will normally have stronger perceptual preferences. These individuals integrate new information into their mental model easily and remember much of what they read, listen, and write.

Digital media can include pictures, audio, and video and it offers both an alternative and supplement to traditional text-based information. Thus, different types of media can be embedded into a text. Some individuals (visual learners) respond better to what they see, some (auditory learners) to what they hear, and some (kinesthetic learners) by doing. McNutt and Brennan [79] studied the relationship between a student's learning style and his success in an online module. According to the authors, students learn by:

- Reading (visual)
- Listening (auditory)
- Seeing (visual)
- Speaking (auditory)
- Doing (tactile/kinesthetic).

This is similar to the perceptive element of physiological stimuli represented in Dunn's model discussed earlier. The authors suggest that the identification and application of learning styles and learning strategies in combination with adaptive systems can help to facilitate the achievement of the goals of distance education and eLearning.

According to Boström [16], Dunn insists that perceptual preferences are the most significant entry points to the theory that learning styles produce positive results for learners. More specifically, *"Perceptual preferences for remembering new and difficult information (...) may be the most important*

[4]VAK is acronym for Visual Auditory and Kinesthetic learning style inventory.

aspects of learning style. These preferences frequently either enable or prevent individuals from achieving easily" [16]. Therefore, it is important to investigate how different perceptual preferences affect learning and cater learning adaptation to a learner's perceptual modalities.

2.4.1 Kolb's learning style model

David A. Kolb's [38] model of experiential learning can be found in many discussions of the theory and practice of adult education, informal education, and lifelong learning. He includes the cycle of learning as a central principle in his experiential learning theory. This cycle is typically expressed as a four-stage cycle of learning, in which "immediate or concrete experiences" provide the basis for "observations and reflections". Observations and reflections are assimilated into "abstract concepts" producing new implications for actions that can be "actively tested" to create new experiences[50].

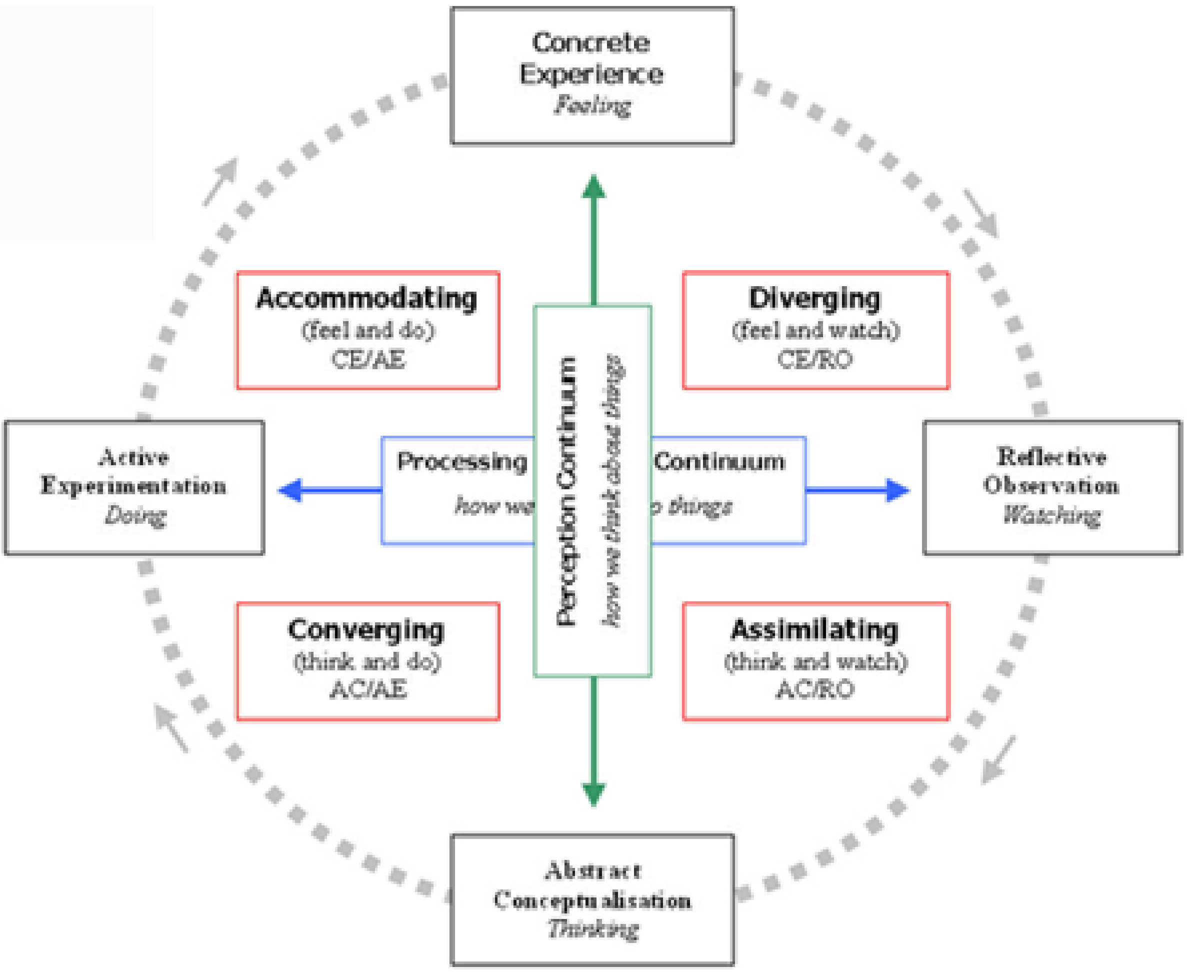


Figure 2.3: Kolb's learning style model (Source: [28])

Kolb's learning style model includes four learning styles:

1. Converger, who can be classified as someone who wants to solve a problem and relies heavily upon hypothetical-deductive reasoning to focus on specific problems.

2. Diverger, who can be classified as someone who solves problems by viewing situations from many perspectives and relies heavily upon brainstorming and the generation of ideas.

3. Assimilator, who can be classified as someone who solves problems by inductive reasoning and has the ability to create theoretical models.

4. Accommodator, who can be classified as someone who solves problems by carrying out plans and experiments, and adapting to specific immediate circumstances.

Although widely used and applied, Kolb's model has also been criticized both for its lack of consideration of the cultural backgrounds of students and because it makes too many generalizations [87].

2.4.2 MBTI

The Myers-Briggs Type Indicator (MBTI) is a well-known [88] personality inventory. The MBTI purports to identify certain consistent differences in the ways people use their minds through a self-administered questionnaire. It consists of four pairs of opposite categories (table 1) called dichotomies [5].

Table 1: Dichotomies in MBTI

ESTJ	INFP
Extraversion	Introversion
Sensing	Intuition
Thinking	Feeling
Judging	Perceiving

The main goal of the MBTI is to identify four preferences (EI[6], SN[7], TF[8]

[5]According to the theory, all categories or preference poles in MBTI are used at least some of the time by every person

[6]E-Extraversion, I-Introversion

[7]S-Sensing,N-Intuition

[8]Thinking,F-Feeling

and JP[9]). Its intent is to reflect a habitual choice between alternatives [84] [10]. The items in the MBTI offer forced choices between the poles of preference. Persons with higher total points for E than for I are classified as Extraverts. The Myers-Briggs model constructs sixteen different models of learning style via different combinations of the four types.

2.4.3 VARK learning style model

Visual, **A**uditory, **R**ead/Write, **K**inesthetic is a learning style model commonly referred to as VARK. In this prototype we use three of its dimensions and thus employ the acronym VAK.

Visual

Visual individuals remember most of what they see. They learn effectively by internalizing new and difficult concepts by reading and observing. Visual learners prefer using images, pictures, illustrations, colors, and maps to organize information and communicate with others. An estimated 40% of population [41] are considered to be visual learners.

Auditory

Auditory learners remember most of what they hear. They learn effectively by listening to information delivered orally or during oral sessions. These learners make up less than 30% of the population [41].

Adults who are auditory manage to remember three-quarters of what they hear during a 40-50 minute period. If an individual is an auditory learner, he could benefit by hearing new or difficult information before starting to read it or do the exercises. According to Dr. Dunn, it is the most difficult to learn and remember through the auditory perceptual channel [46]. Training based on lectures and discussions may be challenging for some participants and if these participants are auditory and analytic, they will learn by repeating the material to themselves [37]. Additionally, auditory individuals benefit from background music when they work [44].

[9]J-Judging, P-Perceiving

[10]This can also be compared to right-handedness or left-handedness, and some people using both hands to write always use the first to reach to the pen - is the prefered hand.

Tactile/Kinesthetic

Tactile/Kinesthetic learners are often characterized as people who "learn by doing". They remember three-quarters of what they *experience*. They learn best by using their hands (tactile sense) and body movement.

2.4.4 Learning Style questionnaire (LSQ)

One of the goals of this thesis is to investigate how students react when the presented materials are adapted to their particular learning styles. A traditional method for identifying learning style involves presenting learners with questionnaires. Fleming [52] created a questionnaire that provides users with a profile of their learning preferences. These preferences detail the ways that they want to receive and submit information. This series of questions is combined in the form of VARK learning inventory.

A learning style has more than 18 dimensions (preferences for temperature, light, food intake, biorhythms, working with others, and location). Therefore, VARK cannot be seen as a learning style by itself. VARK only expresses one preference - the preference for taking in and putting out information in a particular learning context. Although it is only a part of the learning style, Fleming considers it to be important because we can do something about it, while other dimensions are not open to change [52]. The power of VARK questionnaire that both students and teachers intuitively understand it, and it seems to correspond to practice [52].

There is a tendency among learning style preferences that the modality preference is not fixed over a long-term. However, one does not change from an auditory preference to a visual preference overnight. Therefore, multimodal learners may need to process information in more than one mode in order to effectively understand. Learners are usually encouraged to try study strategies listed under their preferences that they may not have tried before. Study strategies are suggested based on learner preference, and they allow learners to become more successful. It also may be helpful to void strategies that contradict the preference (e.g., to avoid using the popular mind mapping application if the learner does not have a visual preference).

Due to measurable factors and positive results [42], perceptual learning modalities are used as the learning style inventory for the implementation of personalized eLearning in this thesis. In the prototype, we will make use of the VARK learning style. Many course authors try to combine textual, audiovisual content and exercises in their courses without paying attention to the fact that the learning outcome of the same course may be different for learners with different learning styles (Table 2).

Learning style	Presentation
Visual	Prefer using images, pictures, colors, and maps
Auditory	Prefer sounds
Kinesthetic	Prefer “hands-on” experience

Table 2: Learners usually have different learning styles.

After each learner completes the questionnaire, information about their learning styles is stored as metadata on the learner. The questionnaire was tested by many individuals independently with possibility to evaluate it at the end on the VARK web-site, where they were asked whether their VARK profile matches their perception of their preferences for learning. The other options were “Don’t Know” and “No Match”. According to the author the percentages for those respondents aged 19 or older are:

- Match = 58%
- Don’t Know = 38%
- No match = 4%

Although self-perceptions are not always reliable, these figures support the value of the VARK questionnaire. Teachers, those who have used VARK before and older respondents have a higher figure for the “match” statistic.

There are sixteen questions in the questionnaire. Experience [52] suggests that too many questions causes some people to take the questionnaire less seriously, while others may become bored with it or provide spurious answers because of questionnaire fatigue.

2.5 Prior knowledge

Assessing the learner’s prior knowledge has been considered important in yielding better performance [39]. Prior knowledge can be defined as the knowledge, skills, or abilities that learners bring to the learning process. In other words, by assessing the prior knowledge of the learner, we can identify in which level he or she belongs. For simplicity, we only consider three levels: easy, intermediate and advanced. However, the system should be flexible enough to accommodate a more relaxed version of level specification.

2.6 Learning Object

Learning Objects (LOs) are fundamental elements of a model for content creation and distribution. Hodgin emphasized that the most significant promise of LOs is to increase and improve the effectiveness of learning and human performance [59]. One of the reasons that LOs are so difficult to define is that they can be virtually anything. Any standalone chunk of information capable of teaching something can be an LO. Examples include a book chapter, map, interactive application, multimedia resource, wiring diagram, simulation, and many others. Further, an LO can be of any size. Here is a list of several definitions employed by researchers and standard organizations:

- *A Learning Object is defined as any entity, digital or non-digital, that may be used for learning, education or training* (IEEE Learning Technology Standards Committee) [77];

- *Learning Objects are elements of a new type of computer-based instruction grounded in the object-oriented paradigm of computer science. Object-orientation highly values the creation of components (called "objects") that can be reused in multiple contexts*(David Wiley) [100];

- *Learning Objects are self-standing, reusable, discrete piece of content that meet an instructional objective. Learning Objects may be tagged with meta-data so that users can easily identify and locate specific learning objects in a web-based environment* (Academic ADL Co-Lab (AADL) at the University of Wisconsin System) [1];

- *Learning Objects are modular digital resources that are uniquely identified, meta-tagged, and used to support learning* (National Learning Infrastructure Initiative) [69].

While there are many interpretations of LOs, most educators would agree that LOs have the following characteristics [80]:

- Smaller units of learning: LOs comprise a smaller unit of learning than a course;

- Self-contained: an LO is self-contained and can be used independently of other LOs;

- Reusable: LOs are reusable, so that the same LO can be used in multiple contexts for multiple purposes;

- Aggregated: LOs can be grouped into larger collections of content to create more substantial units of learning;
- Tagged with metadata: usually, LOs are tagged with descriptive metadata that permit easy discovery and retrieval.

2.7 Learning objectives and Bloom's taxonomy

A learning objective is often defined as a set of attributes that the user should acquire over the course of the system's operation. When articulating learning objectives, authors often concentrate on what the learners should have learned when they finish the course. Depending on the type of course, objectives can focus on content, skills, or attitudes. This term is used in conjunction with learning goal. A learning goal is a statement that describes a more global learning outcome [89].

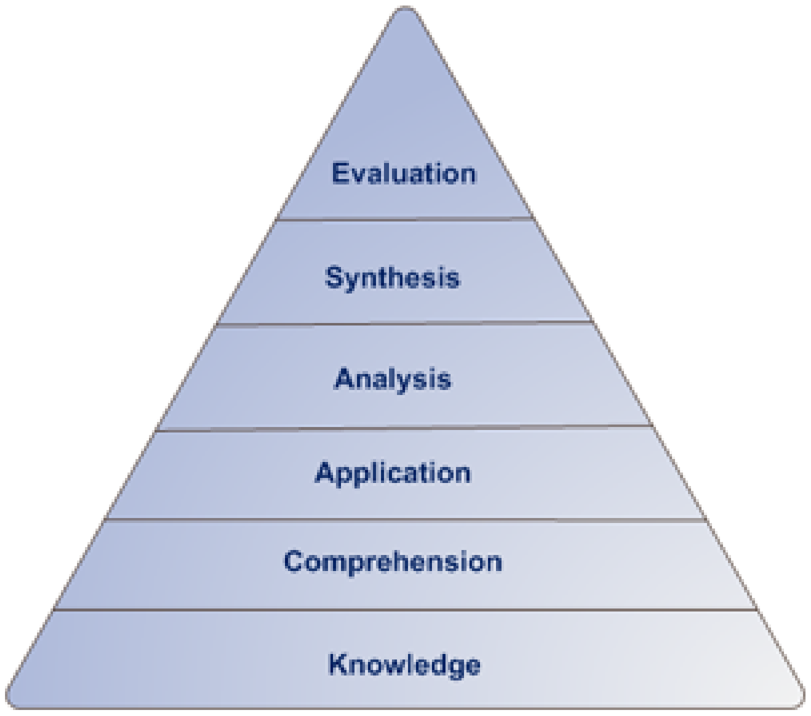


Figure 2.4: Levels of Bloom's taxonomy of educational objectives

Benjamin Bloom developed a taxonomy of educational objectives (Figure 2.4). According to Bloom and his team [15], over 95% of the test questions students encounter require them to think only at the lowest possible level (i.e. the recall of information or knowledge). Six levels are identified within the cognitive domain; the range from the simple recall of facts (lowest level) to increasingly more complex and abstract levels that culminate in evaluation (highest level). The comprehension level is defined as the ability to understand the meaning of material (e.g., explaining or summarizing material). Application, as the name suggests, refers to the ability to apply learned material in new situations (e.g., applying statistical rules to evaluate the reliability of a test). The analysis level requires the ability to break material down into components so that its structure becomes understandabe (e.g., gathering information from a department and selecting strategies for product implementation). Synthesis refers to the ability to build a structure or pattern from diverse elements (e.g., designing a machine to perform a specific task). Finally, the evaluation level is concerned with the ability to judge the value of material (e.g., judging the value of a work using external criteria).

2.8 Metadata

Metadata is an important concept in information management. No single definition of metadata is universally accepted. Many simply refer to metadata as "data about data", "any information associated with an information resource", or "value-added information that documents the administrative, descriptive, preservation, technical, and usage history and characteristics associated with resources" [61]. For LOs to be used, they must first be identified and selected. It is not easy to find anything in a large distributed online environment like the World Wide Web (WWW) or a large distributed learning environment. One approach to solving this problem involves storing not only the content itself but also descriptive information about it. The term Learning Object Metadata (LOM) is often used to denote metadata associated with LOs. LOM potentially includes information about the title, creation date, author, version, technical requirements, and educational context and intent. Researchers divide metadata into many types, but there are three general types of metadata [86]:

- Descriptive metadata describes a resource for discovery and identification purposes;
- Structural metadata describes how compound objects are put together;
- Administrative metadata provides information to help manage resources, and is divided into subsets of rights management (intellectual

property) and preservation (archive) metadata.

Metadata is used for organizing information so that we do not need the original artifact to obtain information about the LO. It is also very useful to sort the vast amount of common data in order to facilitate the retrieval process, and thus the important reason to use metadata is to facilitate the retrieval of the most relevant information.

2.8.1 Use of Metadata

The use of metadata is very popular among library practitioners. Typically, a library catalog card contains data about the contents and location of an item. The methods of using metadata can be divided into three categories [13]:

- Metadata may be embedded within the document itself. An example of this usage is META tags in an HTML file. These are used to indicate metadata information. They are assigned in the HEAD part of HTML documents, which can be used by web crawlers to harvest metadata.
- Metadata may be maintained as an independent part that is attached to the original resources.
- Metadata may be physically stored separately from the resources. Library catalog cards can be categorized into this method. The separate storage of metadata has recently become popular, since it does not require the original resource to do the work.

3 State-of-the-art

3.1 Introduction

This chapter presents short introductions to web-based learning and personalization. Additionally, a brief comparison of different approaches to personalization in web-based learning will be given. Important standards and specifications related to this work will also be described.

3.2 Web-based learning

Industry experts claim that eLearning is a cost effective and flexible alternative to traditional classroom-based education in cases where many people attend the same course [8]. There are large differences between eLearning and traditional classroom-based education from both the social and technical perspectives. A combination of multiple approaches to learning is utilized, and this is often referred to as *blended learning*. As we move towards a second wave of technology (Web 2.0), learners and instructors (teachers, trainers, or tutors) are becoming accustomed to the application of technology to the processes of teaching and learning. The demand for interactive learning technology and rich applications is increasing and there is a mood of critical judgment regarding the effectiveness of the process in delivering high-quality learning outcomes [22].

The design and development of web-based learning technology often tends to concentrate on the instructional delivery method. However, most of the pedagogical principles that apply to the traditional classroom delivery method also apply to eLearning. Pedagogical principles are theories that govern the practice of good teaching [56]. Pedagogy is often described as the art and science of educating people, and is traditionally used as a synonym for teaching and embodies teacher-focused education. In discussions of online pedagogy, the tendency has been to isolate the skills and domains of skills required by both teachers and students to work effectively online [22].

Shirley Waterhouse [98] tried to convince educators to focus first on the fundamentals of teaching and learning (i.e., on pedagogical principles) rather than on technology. Thus, pedagogy could be the driver in implementing learning strategies rather than implementing technology on its own. She defines eLearning pedagogy as *"pedagogical principles and the related instructional strategies applied to an eLearning environment... eLearning involves the application of computer technology to enhance teaching and learning"*.

In [71], the author refers to the overstatement by technology providers the role and importance of technology in the development of eLearning systems which results in expensive software implementation. The problem in these cases is that the development of an eLearning project is seen as a purely technical process. Ideally, designers should have a clear understanding of what constitutes an eLearning *ecosystem*. This systems framework will then specify a learning systems architecture for pedagogical development and systems integration.

3.3 Web-based personalization

When browsing the Internet, it is rarely a surprise to discover your own name poping up: "Welcome back, Ola!" or "Hello, Ola Normann. We have recommendations for you." [11]. Personalization has become an accepted requirement on most commercial web sites, even if it does not extend much further than a first-name greeting and the ability to remember what was purchased last time [104].

In the context of the World Wide Web, personalization is defined as the process of gathering user information during interaction with the user. This information is then used to deliver appropriate content and services that are tailored to the user's needs, with the ultimate goal of improving the user's experience of a service. Many personalization features exist, ranging from the simple display of the end-user's name on a web page, to complex catalog navigation and product customization based on in-depth models of user needs and behaviors. The objective of a web personalization system is to "provide users with the information they want or need without expecting from them to ask for it explicitly" [82].

Traditionally, there have been several types of personalization. These are outlined below [48]:

- *Content-based filtering systems* are based solely on individual user preferences. By tracking each user's behavior, the system recommends items that are similar to items the user liked in the past.
- *Collaborative filtering systems* invite users to rate objects or disclose their preferences and interests, and they then return information that

[11]Amazon uses their recommendation engine to suggest possible purchases based on browsing history and the previous orders. This is a sample text from the front page of their web site www.amazon.com

is predicted to be of interest. This is based on the assumption that users with similar behavior (e.g., users that rate objects similarity) have similar interests.

- *Rule-based filtering systems* ask users to answer a set of questions derived from a decision tree. As the user proceeds to answer the questions, he finally receives a result (e.g., a list of products) tailored to his needs.

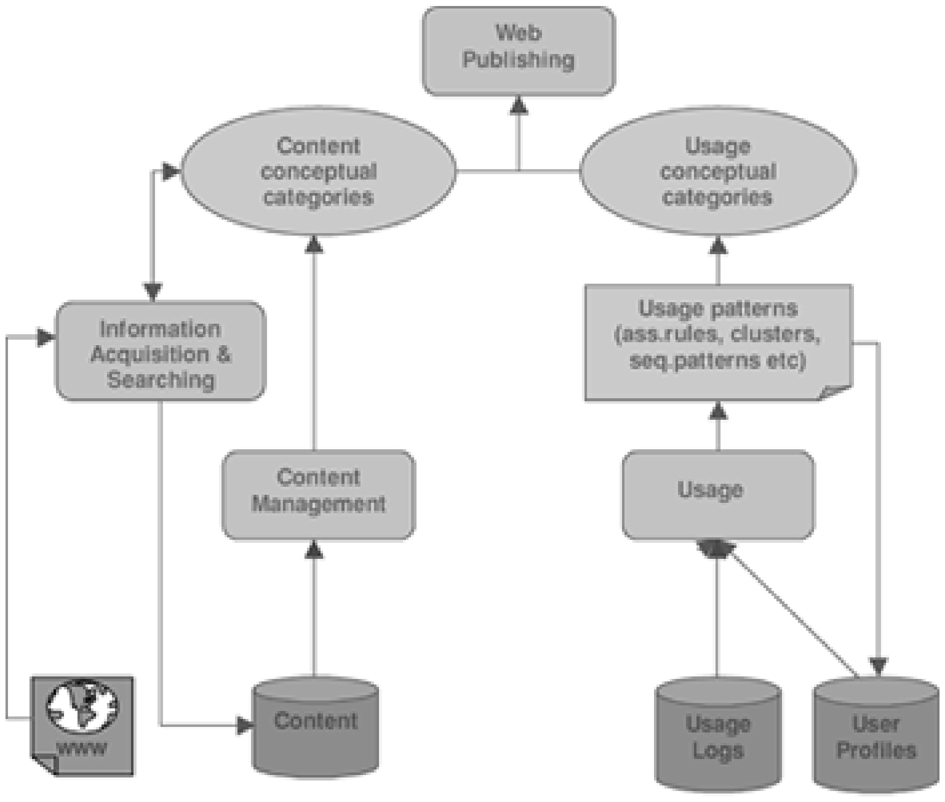

Web Publishing
Content conceptual categories
Usage conceptual categories
Information Acquisition & Searching
Usage patterns (ass.rules, clusters, seq.patterns etc)
Content Management
Usage
www
Content
Usage Logs
User Profiles

Figure 3.1: Conceptual architecture for web-based personalization (Source:[48])

All three of these approaches may also be used in combination to produce a more accurate result. A sample architecture for web-based personsalition can be seen in Figure 3.1. This figure shows how the raw content is harvested and modeled to become a candidate for browsing and retrieval by users. The modules on the right side are user specific, whereas those on the left side are the content processing modules. The content management module processes the content and classifies it into categories. The content can be enhanced with additional information acquired from other sources using advanced techniques. Given the structure and usage logs, a web usage miner provides results regarding usage patterns, user behavior, session and user clusters, clicks, and so on. Important sources of personalization include the user profiles and logs of previous usage. This information is used to distinguish between different users. The user profile and usage logs play an important role both in web-based personalization and in personalization in learning environments.

This should provide us with sufficient understanding of personalization to proceed with a discussion of learning systems and the existing approaches to personalization they employ.

3.4 Intelligent Tutoring Systems

Intelligent Tutoring System (ITS) is a broad term that describes any computer program that contains some intelligence [54]. ITSs provide direct customized instruction or feedback to users without the intervention of human beings. ITSs are regarded as the "successors" of Computer Aided Instruction (CAI) models. An ITS may itself use a number of other technologies, and is considered to be an artificial intelligence system. More specifically, ITSs are expert systems used for tutoring, and are usually based on four components: the domain model, teaching model, student model, and user interface or presentation model.

Brusilovsky states [26] that early ITSs provided little or no learning material. Their most important goal was to support a student in the process of problem solving. Soon, it became clear that hypertext or hypermedia provided the best option for organizing online learning. Theoretical issues about how to learn and teach with emerging technology remain the most challenging problem. In subsequent chapters, therefore, we will focus our discussion on a newer research field called Adaptive Hypermedia.

3.5 Adaptive Hypermedia

Peter Brusilovsky, a pioneer in Adaptive Hypermedia(AH), defines it as follows [25]:

"Adaptive Hypermedia is a relatively new direction of research on the crossroads of hypermedia and user modeling. Adaptive hypermedia systems(AHS) build a model of preferences, goals and knowledge of each in-

dividual user, and use this model throughout the interaction with the user, in order to adapt to the needs of that user".

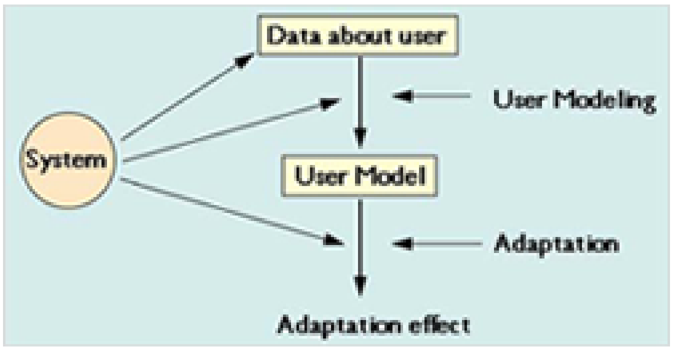


Figure 3.2: User modeling in AHS (Source:[27])

The author argues that traditional "static" hypermedia applications are limited in that they provide the same page content and the same set of links to all users. Traditional educational hypermedia systems present the same static explanation and suggest the same next page to students with widely different educational goals and knowledge of the topic[27]. This is especially true where students have different levels of prior knowledge and different learning styles.

AH offers an alternative to the traditional "one-size-fits-all" approach in the development of hypermedia systems. AH attempts to overcome the problem by using information available about a particular user that is stored in the user model to adapt the information at hand to the given user. AHSs make it possible to deliver "personalized" views of a hypermedia document without requiring any kind of programming by the content producers. Therefore, the main focus of AHSs is a "in-depth" user modeling (Figure 3.2). Although it is possible to allow users to model system settings by customizing preferences or giving out information about their preferences, an AHS can perform all of the adaptation automatically simply by observing the browsing behavior of the user [19]. Some adaptable systems allow personalized views based on user-selected stereotypes like "beginner" and "expert".

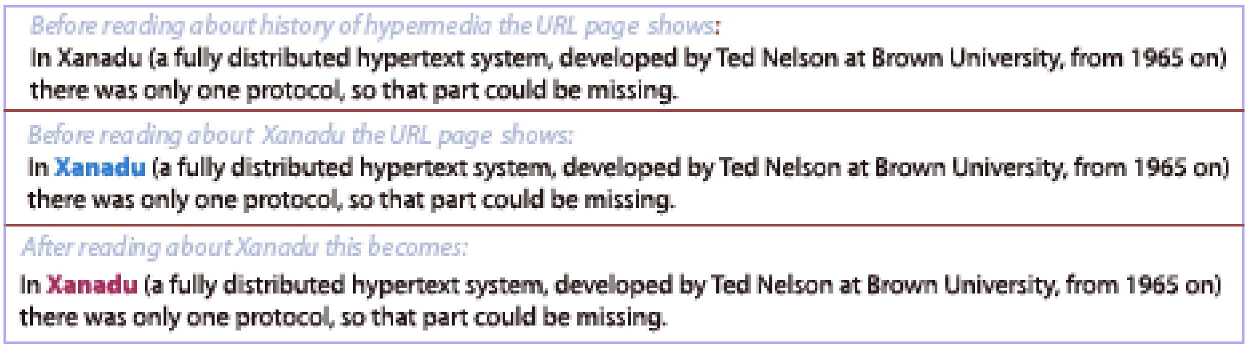
One the most popular areas for AH research is educational hypermedia. A significant user feature in educational hypermedia is user knowledge of the subject being taught [24]. Most AHSs provide the following adaptive functionalities [26]:

- direct guidance: an attempt to show the best possible link;
- link sorting: sorting links in accordance with the user model, placing more relevant links closer to the top;
- link annotation: commenting on links to tell the user about the nodes behind the annotated link;
- link hiding/disabling/removal: these prevent the user from following links that are not relevant to them at the moment.

3.5.1 Examples of AHSs

AHA!

AHA! is one of the most frequently referenced AHSs in the field of AH [20]. It is the educational AH project of the Department of Computer Science of the University of Eindhoven. It represents an open AH architecture that is suitable for many different applications. The development of the AHA! started in 1996. AHA! offers both adaptive presentation and adaptive navigation support. The textual content of pages is adapted based on the user model.


Before reading about history of hypermedia the URL page shows:
In Xanadu (a fully distributed hypertext system, developed by Ted Nelson at Brown University, from 1965 on) there was only one protocol, so that part could be missing.

Before reading about Xanadu the URL page shows:
In **Xanadu** (a fully distributed hypertext system, developed by Ted Nelson at Brown University, from 1965 on) there was only one protocol, so that part could be missing.

After reading about Xanadu this becomes:
In **Xanadu** (a fully distributed hypertext system, developed by Ted Nelson at Brown University, from 1965 on) there was only one protocol, so that part could be missing.

Figure 3.3: An example of adaptive content presentation in AHA!

Adaptive content presentation is achieved through conditional fragments by including pieces of content (virtually, a portion of HTML text) when a certain condition is met. Adaptive navigation is implemented by links with three possible states - desired, undesired, and uninteresting. The example in Figure 3.3 shows a prerequisite explanation in the hypermedia course. As shown in the figure, the standard colors of WWW browsers are used: blue links are desired and unvisited, and purple links are uninteresting (this implies that the document has been visited and does not represent new information to be learned). Links may also be hidden in the text, disabled, or removed entirely. The goal of adaptive navigation is to support learners in finding their optimal learning path through the environment.

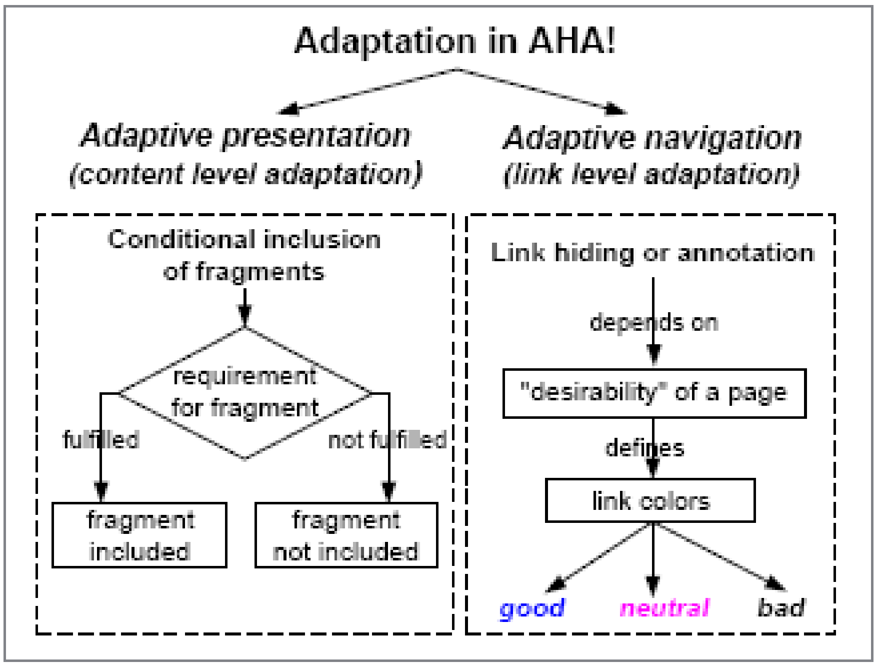


Figure 3.4: Adaptation model in AHA! (Source:[18])

AHA!'s adaptation mechanism is related to the concept overlay and is shown in Figure 3.4. Upon visiting a page, a number of concept-linked attributes within the user model are altered according to embedded values in the concept ruleset. The two most common associated attributes are the visited and knowledge attributes. Each page is composed of fragments and objects that can contain fragments requiring adaptation. This creates a mechanism for recursively creating complex pages.

One of the most important characteristics of AHA! is that it exposes two primary mechanisms of adaptive presentation. The first occurs through link hiding/annotating, and the second occurs via fragment inclusion/selection. As discussed above, the system evaluates in both processes the desirability of a page relative to the requirement set of the page and the user's knowledge vector; particular attention is paid to the visited attribute. Furthermore, AHA! offers substantial configuration options and relies heavily on this information.

InterBook

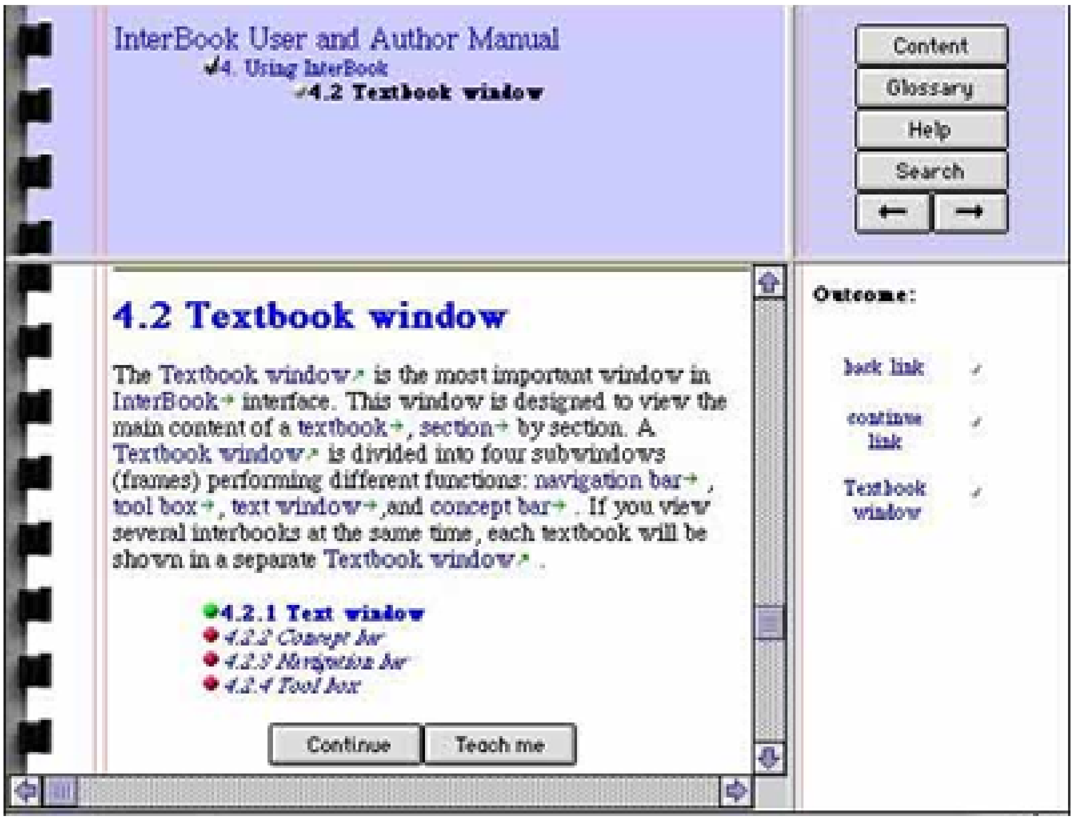
InterBook was originally designed as a tool for authoring and delivering adaptive electronic textbooks on the WWW. This system provides a technology for translating electronic textbooks from plain text to specially annotated HTML pages. InterBook also provides an HTTP server for adaptive delivery of these electronic textbooks over the Internet. For each registered user, an InterBook server maintains an individual model of the user's knowledge and applies this model to provide adaptive guidance, adaptive navigation support, and adaptive help [49].

InterBook uses the domain and textbook content knowledge to serve a well-structured hyperspace. InterBook generates contextual links between the glossary and the textbook. Links are provided from each textbook section to corresponding glossary entries for each of the involved background or outcome concepts. At the same time, it provides links from each glossary entry describing a concept to all textbook units used to learn this concept. This implies that a glossary integrates the features of an index. These links are not stored in an external format, but generated dynamically by a special module that takes into account the student's current state of knowledge as represented by the learner model.

In Interbook, adaptive navigation support is provided using colored bullets and different fonts through link annotation (Figure 3.5). At the time of this writing, four colors and three fonts were used. The meanings of the bullets are:

- green - green bullets and bold font mean "ready and recommended" (but not yet learned);
- white - white bullets mean "clear, nothing new" (all concepts presented are known to the user);
- violet - violet bullets mark nodes that have not been annotated by an author;
- red - red bullets mean that the node is not recommended, because the node contains prerequisite concepts that have not yet been met.

The learner model in InterBook is initialized from the registration page using a stereotype model, and it is modified as the user moves through the pages. There are two types of knowledge in Interbook: knowledge about the domain being taught (domain model) and knowledge about the learners (student models). The domain model serves as a basis for the creation of the



InterBook User and Author Manual
✓4. Using InterBook
✓4.2 Textbook window

4.2 Textbook window

The Textbook window is the most important window in InterBook interface. This window is designed to view the main content of a textbook, section by section. A Textbook window is divided into four subwindows (frames) performing different functions: navigation bar, tool box, text window, and concept bar. If you view several interbooks at the same time, each textbook will be shown in a separate Textbook window.

● 4.2.1 Text window
● 4.2.2 Concept bar
● 4.2.3 Navigation bar
● 4.2.4 Tool box

Figure 3.5: Adaptive link annotation in InterBook (Source: [49])

content of an adaptive electronic textbook.

As we have seen, while the rules employed in Interbook are basic, the system has a number of desirable features with regard to the user interface. A useful feature is the explicit listing of prerequisites and requirements along with a simple and direct visual representation of the adaptation state. Interbook does not seem to reflect user context [34]. However, the interface within the system does provide a useful clue for more "intelligent" adaptation.

APeLS

APeLS (Adaptive Personalized eLearning Services) is the result of a PhD thesis at Trinity College Dublin [34]. The proposed approach is based on the clear separation of content, learner and narrative models, and an adaptive engine that uses an artificial intelligence model to achieve adaptation to the learner's requirements. The adaptive engine accommodates the three models to compose a personalized course at runtime. Such dynamic construction is assessed through access to the learner model.

The multi-model approach tries to address separation concerns between learning content and adaptive linking logic. This results in new possibilities fore reuse of a piece of learning content, because LOs are no longer specific to a given implementation.

The architecture proposed in APeLS is based on at least three models: content, learners, and narratives. Specifically, content and learners need to be modeled in order to accommodate certain functionality. Modeling implies annotating learning objects (LOs) with specific metadata elements. For learners, a profile must be created and continuously updated as the learner fulfills steps. One of the main issues related to content and learner modeling involves separating elements from other logic (e.g., separating content and presentation elements). This will improve the possibilities for reusing a piece of learning resource and allows it to be repurposed as addressed in [35]. APeLS has been successfully used at Trinity College Dublin by students at the undergraduate level. The initial student trial results showed that 87% of students were satisfied or happy with how the content was structured in the personalized course. The results from this project suggest that that personalization can positively affect the learning experience.

3.6 Intelligent agent-based adaptive eLearning

Wooldridge [103] proposed the following generic definition of a software agent:

An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives.

In artificial intelligence, an agent represents an entity that is capable of perception, action, and goal-directed behavior. Such an agent might be a robot or an embedded real time software system; it is intelligent if it interacts with its environment in a manner that would normally be regarded as intelligent if that interaction were carried out by a human being.

In [30], the authors propose intelligent agent-based adaptive eLearning. Their purpose is to reduce the learning inefficiency by analyzing individuals, and guiding and reducing learner's confusion. Their hope is to improve learning through intelligent assistance and constructive feedback. An agent has autonomous (controlled inner states), interactive (communication with the environment), and adaptive (responses to particular actions based on past experience) characteristics. Intelligent agents are sometimes described as tools that can manage an information overload, and serve as academic experts. The learning process is enhanced by having many agents collaborating and competing towards achieving prescribed goals.

3.7 Semantic Web and eLearning

The Semantic Web is part of the next generation of the web where computers can understand the meaning of information being exchanged (machine-understandable information). It is seen as an extension of the current web, where the information is given a well-defined meaning or semantics. Originally proposed by the World Wide Web Consortium [14], the Semantic Web has piqued the attention of researchers in many other domains. Figure 3.6 illustrates the layered architecture of the Semantic Web.

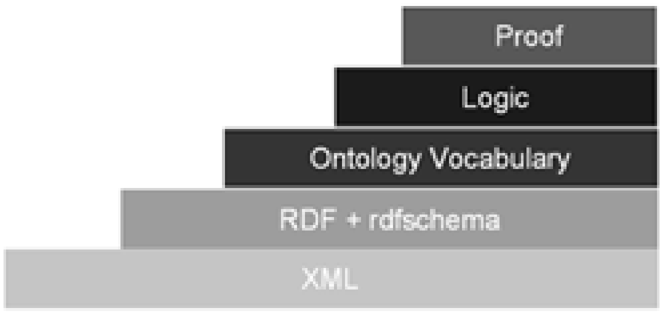


Figure 3.6: Layers in Semantic Web

One of primary characteristics of the Semantic Web is shared understanding, which utilizes ontologies (shown as layer 3 in Figure 3.6) as its key backbone. Ontologies [12] enable the organization of learning materials around small pieces of semantically annotated (enriched) LOs [93]. At least three related research areas contribute to eLearning personalization in the Semantic Web; these include open hypermedia, AH, and Semantic Web reasoning [58].

[12]The goal is not to discuss Semantic Web in detail but to give a brief description of how Semantic Web can be used in personalized eLearning environments. Ontologies are an important part of Semantic Web and here an ontology can be defined as a representation of a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of that domain, and may be used to define the domain. For a more detailed information about ontologies and Semantic Web the reader is referred to W3C Semantic Web Activity page at http://www.w3.org/2001/sw/

RDF [13](second layer from the bottom in Figure 3.6) can be utilized for the automatic generation of hypertext structures from distributed metadata which can be an approach to achieve personalized eLearning in the Semantic Web. For example, [58] uses several ontologies for describing the features of domains, users, and observations. Reasoning (first and second layers from the top in Figure 3.6) over these ontologies is enabled by the RDF query and the transformation language TRIPLE. Although Semantic Web technologies seem to be a promising research area for further exploration of personalized eLearning, more research needed regarding the reasoning rules for educational models. Since the focus of this thesis is not in evaluating technologies for eLearning personalization we will conclude the discussion of the use of the Semantic Web in eLearning here.

3.8 Different approaches to personalization

A literature review reveals two general approaches towards adapting content for learning purposes. The AH movement represents a user-centric modeling approach, and it uses a model of concepts in the learning domain in order to decide both what content and navigational features to display and how to present that content [2]. Alternatively, standards organisations like IMS [67] try to solve the problem by proposing a specification known as IMS Simple Sequencing (IMS SS) [68] . IMS SS is the specification adopted by the LO community in ADL SCORM version 1.3. It provides learners with a sequence of activities to guide them through a learning space in order to achieve a specific objective (discussed in more detail in Chapter 3.11.5). These two *schools* appear to have similar objectives. Both aim to deliver appropriate material and enable the user to achieve the learning goal as quickly as possible. Table 3 summarizes their differences.

The table illustrates that these two approaches are superficially similar but contain some important differences. AH aims to employ the intelligence and knowledge of each user to assist in achieving the learning objective. In contrast, IMS SS has no intelligence and does not differentiate between users. Instead, it simply uses a set of rules to sequence a learner towards a pre-defined learning objective. SCORM 1.3 includes IMS SS among its set of specifications and provides useful tips on how effective learning and the reusability of learning material can be achieved. The prototype PEDAL-NG lies somewhere in between these two approaches.

[13]Resource Description Framework(RDF) is a metadata model based upon the idea of making statements about resources (subject-predicate-object expressions or triples). It is a W3C specification

Criteria/Property	**AHS**	**IMS SS**
Objective	Complete all activities, avoid unneccessary ones	Assist learners with navigation
Focus	Instructor centric	Learner centric
Models	Domain Mode, User Model, Adaptation / Pedagogical model and Adaptive Engine	Sequencing Definition Model, Tracking Model and Activity State Model
Conceptual structure	Nodes representing a page	Activities in activity tree
Techniques	Link sorting, link hiding, link annotation, direct guidance etc	Uses sequencing and rollup rules
Tracking	Adaptive engine tracks learners' browsing behaviour	Controlled by its Tracking Model and Activity State Model
Re-usability	Rule definitions and links embedded within the content, hindering re-usability	Can specify meaningful sequencing behaviour externally from the learning resources
Standardisation	No existing standards; widely accepted techniques	Standardised by IMS Global Learning Consortium

Table 3: IMS SS and AHS comparison.

3.9 LMSs

This part of the thesis provides a brief overview of existing Learning Management Systems (LMSs) [14]. Both open source and commercial learning platforms have been chosen. As Inspera LCMS is used for prototype implementation, a special section is dedicated to describing Inspera LCMS.

3.9.1 LMS or LCMS?

LMSs are considered to be the cornerstone of most eLearning strategies. These systems provide a great way to deliver, track, and manage learning or training. However, another term - LCMS (Learning Content Management System) - is used frequently in the discussion of online learning. Despite the similarity between these names, the two terms have different meanings, and can sometimes be confusing. The primary goal of an LMS is to manage learners by keeping track of their progress and performance across all types of activities: it is thus administrative-centric. In contrast, an LCMS manages LOs that are served to the learner at the right time. One important difference is that most LCMSs have a built-in LMS functionality [21]. Table 4 summarizes these differences.

The content in an LCMS is reusable across courses and curricula and is not tightly bound to specific formats. Often LCMSs include a separate logic for the content itself and presentation.

3.9.2 Moodle

Moodle [15] is a course management system. It is a free, Open Source software package designed using sound pedagogical principles that intends to help educators create online learning communities. Moodle has gained widespread acceptance, and is now used worldwide as an eLearning platform closely associated with a social constructivist learning model. Moodle is also cited as one of the most user-friendly and flexible open source courseware products available. Moodle is based on LAMP - a solution stack using the Linux operating system, Apache web server, MySQL database, and PHP (sometimes Perl or Python) as scripting language. The system is not hard to understand or use. Furthermore, it can be extended and has very well documented API and programming templates [57]. Moodle's quiz module

[14]In Norway, the term "eLearning platform" (*e-læringsplattform*) is often used

[15]Originally an acronym for Object-Oriented Dynamic Learning Environment - http://moodle.org/, last checked 27.12.2007

	LMS	**LCMS**
Target users	Instructors, administrator	Instructional designers, project managers, content developers
Management	Learners	Learning Objects
Management of collaboration spaces, instructor-led training	Yes (but not always)	No
Management of collaboration spaces, instructor-led training	Yes (but not always)	No
Learner collaboration	Yes	Yes
Performance reporting of training results	Primary	Secondary
Keeping data about learner profile	Yes	No
Creation of test questions and test administration	Yes	No
Workflow tools to manage the content development process	No	Yes
Delivery of content by providing navigational controls and learner interface	No	Yes

Table 4: The differences between LMS and LCMS.

enables it to handle adaptive questions, but it does not yet have any built-in question types that make proper use of the ability of the quiz module to handle adaptive questions. It has excellent documentation, strong support for security and administration, and is evolving toward IMS/SCORM standards. The key to Moodle's success is that it is developed with both pedagogy and technology in mind.

3.9.3 Sakai

Sakai [16] is an online collaboration and learning environment. This system includes a set of software tools designed to help instructors, researchers, and students create web sites for collaboration. The description from its web site states that:

> *Many users of Sakai deploy it to support teaching and learning, ad-hoc group collaboration, support for portfolios and research collaboration. Sakai is a free and open source product that is built and maintained by the Sakai community. Sakai's development model is called "Community Source" because many of the developers creating Sakai are drawn from the "community" of organizations that have adopted and are using Sakai.*

Sakai has a long feature list, a clear interface and an easy to navigate structure. Using a web browser, users choose from a set of features to create a site that meets their needs. It is possible to set up the system in accordance with needs by adding and removing modules and tools that are used.

3.9.4 ATutor

ATutor is an Open Source web-based Learning Content Management System (LCMS). The software is cited as unique for its accessibility features (which are useful for visually-impaired and disabled learners) and suitability for educational use according to the software evaluation criteria established by the American Society for Training and Development [33]. ATutor is a promising system that provides good documentation, ease of installation, and a strong potential for development. While the user interface may not seem intuitive to many users, the overall functionality is good, wide, open/modular, and committed to standards. ATutor is one of very few LMSs that support LOR. Its support for standards is very strong, and it can import external content in an IMS/SCORM format. ATutor is also designed for adaptability to any of several teaching and learning scenarios. There

[16]Sakai web site - http://sakaiproject.org/, last checked 02.01.2008.

are four main areas that reflect this design principle: themes, privileges, tool modules, and groups. This adaptability is limited and does not address personalization based on prior knowledge or learning style.

3.9.5 it's learning

it's learning is a Virtual Learning Environment (VLE) designed specifically for schools and universities. Its defining characteristics, according to the company behind this product, are flexibility and user-friendliness. it's learning is described as a general learning platform for communication, cooperation, administration, reporting, production, management, and publication of courses and learning resources online. it's learning is an educational tool developed for use in courses and educational environments, and it support conventional as well as distance education. it's learning offers a rich set of functionality, including creating and managing curriculum lists, files, notes, messages and discussion boards along with other LOs. The system has a great deal of functionality, but it has many limitations as well.

3.9.6 Fronter

Fronter is another Norwegian company that delivers an open learning platform [17]. At the time of this writing, the Fronter LMS was being used by more than 3000 learning institutions (primary and secondary schools, colleges, universities, organizations, municipalities, and counties) across Europe. Fronter provides such tools as personal work and portfolio management, teaching and learning, setup and administration, publishing, and collaboration and communication. Fronter and it's learning are the most widespread eLearning solutions in Norway.

3.10 Inspera: a vendor of LCMS

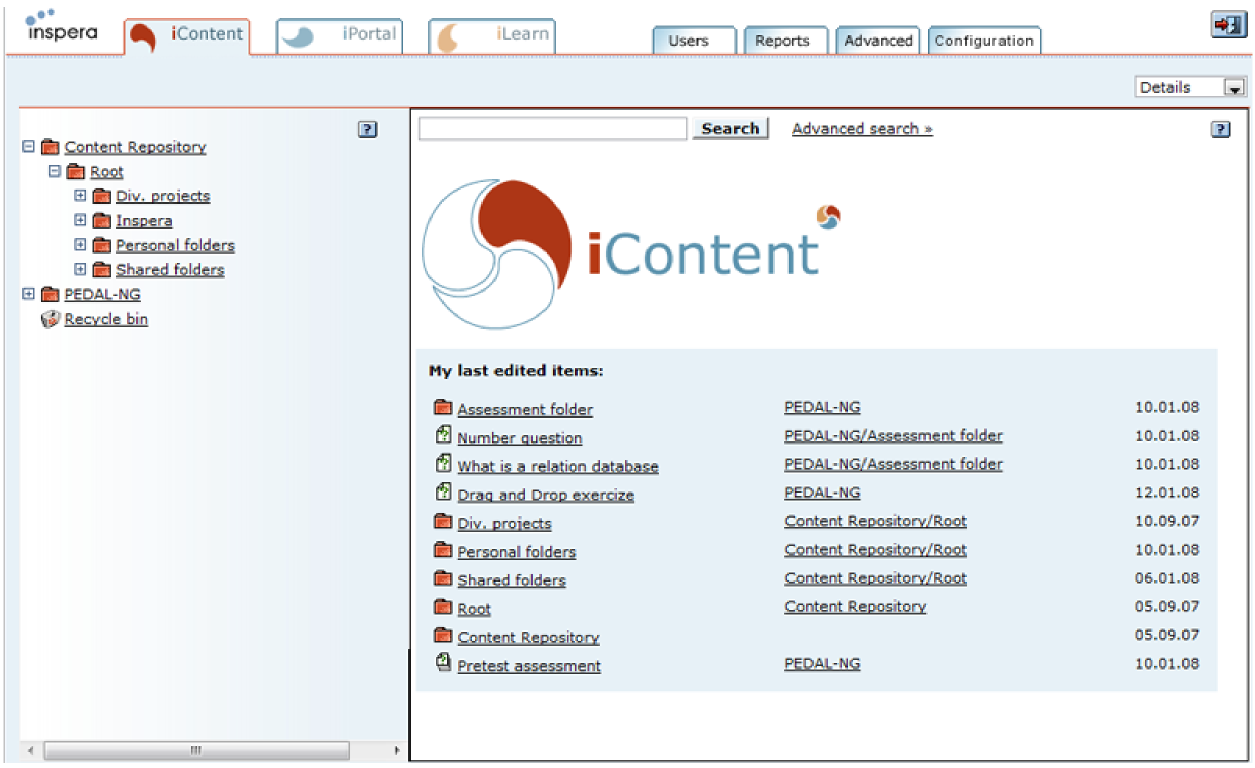
As previously mentioned, this thesis focuses on the adaptivity and personalization aspects of eLearning. Therefore, there is no need to implement a new eLearning application that requires a lot of resources. One of the prerequisites of this thesis was to use Inspera's LCMS [9] as an underlying platform for implementation in which we could concentrate solely on adaptation and personalization.

Inspera is private company that delivers products and services for electronic publishing. Founded in 1999 and based in Oslo, Norway, Inspera was

[17]The former name of the LMS was Classfronter, but it is now called Fronter.

rated in 2005 as the 17th fastest growing high-tech company in Norway (number 110 in Europe) in Deloitte's Fast50/European Fast500 rankings. Inspera had 20 employees at the time of this writing.

Inspera provides tools to create and reuse digital and interactive content across formats and channels, and is entirely web-based. Figure 3.7 illustrates the welcome screen for a user of the Content Management software known as Inspera Content Server.



My last edited items:

Assessment folder	PEDAL-NG	10.01.08
Number question	PEDAL-NG/Assessment folder	10.01.08
What is a relation database	PEDAL-NG/Assessment folder	10.01.08
Drag and Drop exercize	PEDAL-NG	12.01.08
Div. projects	Content Repository/Root	10.09.07
Personal folders	Content Repository/Root	10.01.08
Shared folders	Content Repository/Root	06.01.08
Root	Content Repository	05.09.07
Content Repository		05.09.07
Pretest assessment	PEDAL-NG	10.01.08

Figure 3.7: Welcome screen in Inspera Content Server.

Inspera's solutions have been deployed by several large customers. Among there is one of the first and largest content providers within the publishing sector, H. Aschehoug & Co, which produces content for students at Norwegian high schools. Inspera works closely with their largest customer, the publishing house Aschehoug and in autumn of 2008, a new company - Creaza [18] will

[18]There is too little information on Creaza web-site and it can be accessed at http://www.creaza.no/.

be launched (Fronter as a partner) which will provide an integrated toolbox for creative work. This partner relationship has been important for the development of the software offering. Aschehoug bought their first license for Inspera's product even before it was finished in 2001.

Inspera LCMS is web-based learning platform for the creation, publication, accomplishment and evaluation of online learning. It consists of:

- a learning portal (Inspera Portal Server) with strong support for collaborative tools;
- an LMS for course administration;
- a CMS (Inspera Content Server) that handles creation, reuse, and cross-media publishing of LOs, inclusive tests and questions;
- a set with authoring tools that support template-based content development.

Inspera is not an end-user content producer. Instead, it offers a means to achieve effective content production. Inspera Content Server supports various content types, such as the creation of hierarchical structures, documents, images, sound and video, links, mathematical objects, XML and XSL content, and interactive tests that can be organized in folder structures.

Cross-media publishing and reusability have been the underlying product philosophy for the Inspera product line since 2001. Cross-media publishing is defined as publishing content from the same source to different final formats and channels so that information objects can be used in different contexts. As an example, the same material to produce web-pages in HTML format, eLearning courses in Adobe Flash format or MAF (Multimedia Application Format) for mobile devices, and an electronic book (e-book) in PDF format. Although cross-media publishing allows information to be updated in one place, it is published to different media and formats since the same information source is used to produce the different final formats. This improves productivity, because it reduces production effort and cost and increases the quality of information since the information is consistent across different formats.

To achieve cross-media publishing, information resources should be represented independently of media. Figure 3.8 illustrates how information resources are "coded" in the Inspera Content Server so that they can be reused in different contexts.

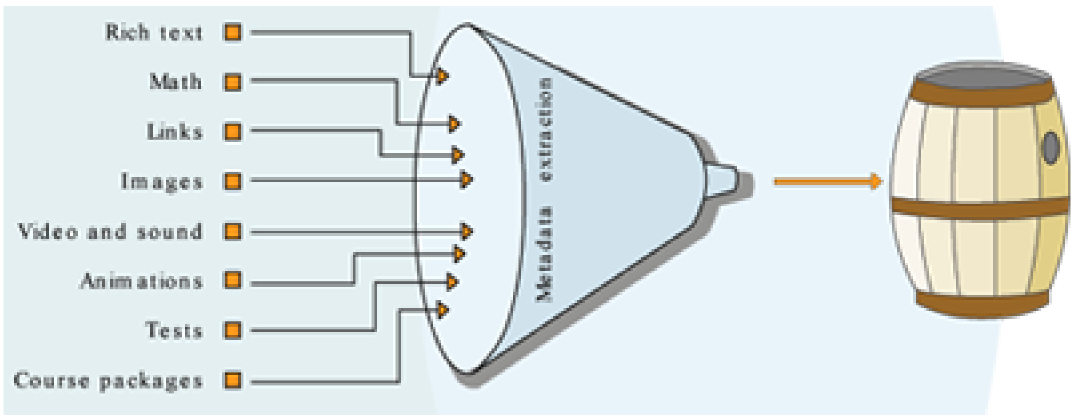


Figure 3.8: Information archival for reuse in long living storage format

Information resources are prepared (metadata extracted) and stored in a "content barrel" in a media-independent form. When these information objects are represented and stored in the barrel, they should be aggregated and transformed to an end-product in different formats as shown in Figure 3.9.

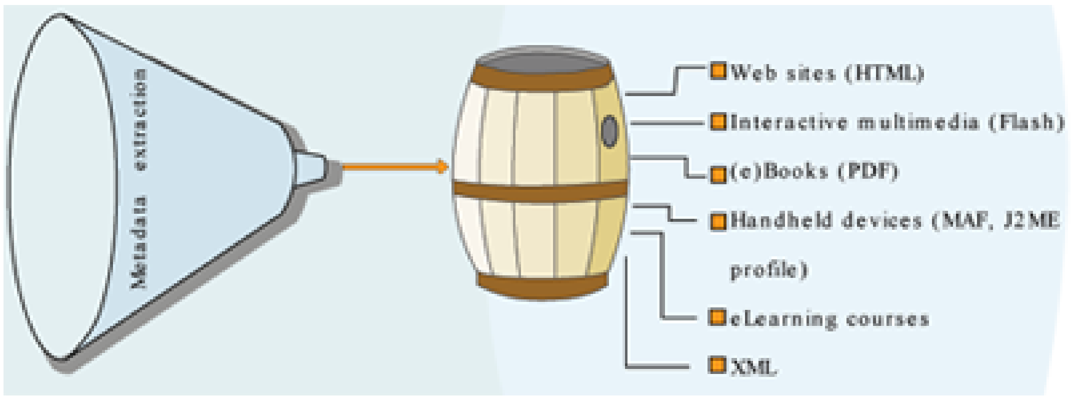


Figure 3.9: Different formats from the same information source

There are many ways to represent information in different formats. Inspera uses a standards-based approach to achieve cross-media publishing. The templating engine that is included in the transformation layer is responsible for transforming objects into different formats. This is shown in Figure 3.10 as XSLT.

The technical basis for Inspera's product relies on the Oracle Database,

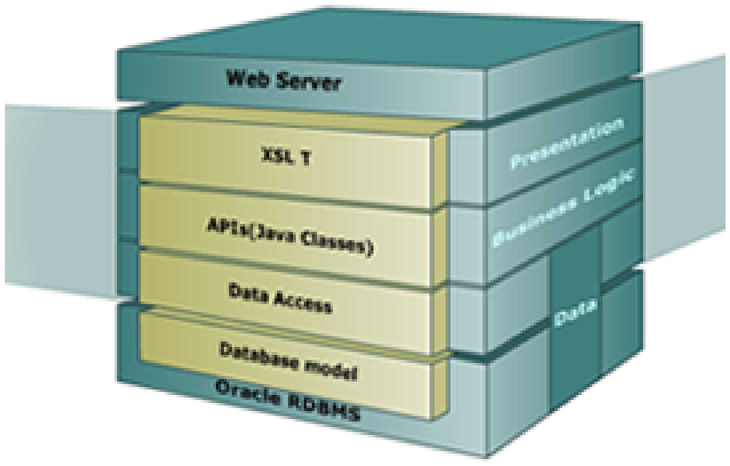


Figure 3.10: Application layers in Inspera LCMS

a Java application server, and XML technologies. Inspera thus uses a commercial database, but several open source libraries are used in their products (e.g., for indexing and search). The Inspera LCMS also utilizes object oriented software paradigm in the database layer. Thus, although there are separate tables for entities like users and LOs, all of these extend the object entity: this can be compared to Java's `java.lang.Object`. For each primary key in other tables in a Inspera LCMS database, there is exactly one entry in the object table. This makes it possible to reuse the data models in various situations. It is important to note this fact, because we will see in the prototype that the data model is reused for modeling both users and LOs.

3.11 Standards and specifications

Many standards in eLearning describe different domains within eLearning. Some standards concentrate on content description, whereas others describe learner and content packaging or the sequencing of items to be delivered. Each of these standards has various levels of complexity, generality, and notation, and each tries to solve different problems. The level of complexity is related to the amount of detail with which we want the resources to be described. The more fine-grained detail we desire, the more complex a metadata format is needed. This thesis does not evaluate standards used in eLearning, but we still must review those standards since they are important for interoperable eLearning systems.

3.11.1 IEEE LOM and IMS Learning Resource Metadata

The IEEE Learning Technology Standards Committee working group IEEE 12 (Learning Object Metadata Working Group)[63] has developed a standard conceptual model commonly referred to as the IEEE LOM. The IEEE LOM is used to describe an LO and similar digital resources used to support learning, and it aims to provide an extensive metadata description for LOs. The purpose of the IEEE LOM is to support the re-usability of LOs and facilitate their interoperability in the context of online LMSs. The standard specifies the syntax and semantics of LO metadata. The IEEE LOM consists of nine categories:

- *General* - context independent and semantic properties of the LO
- *Lifecycle* - properties related to the description of the resource's lifecycle
- *Meta-metadata* - information about the metadata itself (not the LO)
- *Technical* - technical properties
- *Educational* - learning and pedagogical properties of the LO
- *Rights* - intellectual property rights and conditions for the use of the LO
- *Relation* - the resource's relationship to other LOs
- *Annotation* - comments on the educational use of the LO
- *Classification* - description of this LO relative to a particular classification system

These main categories are divided further, and the format consists of roughly sixty fields in total. The IMS Learning Resource Meta-data Specification [66] (version 1.3 at the time of this writing) is a standard derived from the IEEE LOM Schema working document of the IEEE LOM. The IMS also provides a best practice implementation guide and XML bindings.

3.11.2 Dublin Core Metadata Initiative

The Dublin Core (DC) [40] was the result of a workshop held in Dublin, Ohio, United States in 1995. The DC metadata element set is a standard for describing information resources (primarily those on the web). It provides simple sets of elements to facilitate sharing, describing, finding, and managing information. The DC has quickly attracted broad international and interdisciplinary support and is being adopted by many communities. The DC has fifteen basic DC metadata elements: Contributor, Coverage, Creator, Date, Description, Format, Identifier, Language, Publisher, Relation, Rights, Source, Subject, Title and Type. The DC has been criticized for its simplicity [76]. Therefore, so-called qualifiers have been introduced. Qualifiers attempt to further specify existing DC elements, thereby increasing the precision of the encoded metadata. Two broad groups of qualifiers exist, element refinement and encoding scheme.

The DC has the particular purpose of aiding resource discovery and facilitating interoperability. The DC is often used, among other things, as a preferred format in the Open Archives Initiative Protocol for Metadata Harvesting [94]. Inspera uses the DC for distributed searching in combination with CQL (Contextual Query Language) which is part of the SRW/U (Search/Retrieval via Web Services/URL) protocol, which is based on Z39.50 semantics, and is endrosed by the Library of Congress.

3.11.3 IMS Learner Information Package

IMS Learner Information Packaging (LIP) [65] is another specification by IMS Global Learning Consortium and includes a collection of information about a learner or a producer of learning content (creators, providers or vendors). The IMS LIP addresses the interoperability of learner information between Internet-based systems (LMSs). The intent of the specification is to define a set of packages that can be used to exchange data in an IMS-compliant learner information server. The core structures of the IMS LIP are based on: accessibilities, activities, affiliations, competencies, goals, identifications, interests, qualifications, certifications, licences, relationship, security keys, and transcripts. IEEE has proposed a similar specification called IEEE Public and Private Information Specification.

3.11.4 IMS Content Packaging

IMS Content Packaging Specification [64] is an interoperability specification that allows content creation tools, LMSs, and runtime environments to share content in a standardized set of structures. The intent of this specification is twofold: to standardize the way learning resources are defined and to enable the organization of different components related to learning content. This specification provides a mechanism for structuring learning content into a package so that different learning environments (such as LMSs) can exchange packages of content.

3.11.5 IMS Simple Sequencing

The IMS Simple Sequencing Specification (SS) defines a method of representing the intended behavior of an authored learning experience such that any learning technology system can sequence discrete learning activities in a consistent way. This specification describes behaviors and functionality that conforming systems must implement. IMS SS includes definitions of rules that describe the branching or flow of instruction according to the outcomes of a learner's interactions with content. The sequencing is based on instructional design strategies selected by the course's instructor, and these are expressed in XML in the "manifest file". The LMS maintains a simple user model based on both the LOs the user has visited and intermediate test results. Based on this information, the LMS decides what LO should be next in the sequence. IMS SS is labeled as simple because it defines a limited number of widely used sequencing behaviors, not because the specification itself is simple. This specification does not address artificial intelligence-based sequencing, schedule-based sequencing, sequencing requiring data from closed external systems and services (e.g., sequencing of embedded simulations), collaborative learning, customized learning, or synchronization between multiple parallel learning activities. IMS SS is part of SCORM (see chapter on ADL SCORM below).

3.11.6 Aviation Industry CBT Committee

The Aviation Industry CBT Committee(AICC) [10] provides guidelines for seamless data flow between different computer-based training lessons and Computer Managed Instruction environments. It also provides data flow between different CMI systems and CBT lessons created with different authoring systems to a common data store and analysis tool. The AICC has developed several specifications. This sepcifications that have been widely adopted is AICC CMI001, AICC CMI Guidelines for Interoperability. This specificaiton describes the mechanism by which a course can be structured, launched, and tracked. According to the specification, course completion is logged and documented.

3.11.7 ADL SCORM

The SCORM (Sharable Content Object Reference Model) [19] is a set of standards and specifications for eLearning developed by the Advanced Distributed Learning Initiative [3] under the umbrella of the United States De-

[19]Original acronym was Sharable Course Object Reference Model.

partment of Defense [20]. The primary goal of the SCORM is to enable interoperability, accessibility, and reusability of learning content: this is done by packaging content in a zip file. More capabilities have been added to SCORM 2004; amoung these are sequencing, which includes rules that specify the order in which a learner may experience LOs. The ADL collaborated with AICC members and participants to develop common Launch and API specifications. The standard is based on XML, and it utilizes to a large extent work done by the AICC (CBT) [10], IMS Global Learning Consortium [67], IEEE [62], and Ariadne [7]. The current version of the SCORM is SCORM 2004 3^{rd} edition (at the time of this writing), but a new SCORM 2.0 (Web 2.0) is expected by October 2009.

3.11.8 Analysis of standards and specifications

Standards for content and learner information descriptions are becoming increasingly important in LMSs. Standards facilitate interoperability between different learning environments by exchanging content and learner information. Because they are usually described in a platform-neutral manner (mostly in XML), they can be used independently of the underlying technology of the specific environment. Most of the standards are developed for use by LMSs. Therefore, there are cases where their use in an adaptive eLearning environment is limited such that specification might not be sufficient to support adaptivity. The traditional solution for this problem has been the use of extension mechanisms. It should be noted, however, that the baseline support for standards is important and can serve as a starting point for interoperability with other systems at the basic level.

Today, dozens of LMSs are available in both the commercial market and open source community. Many institutions and organizations use more than one LMS for their training practices. Multiple LMSs means multiple “versions” of content that may not be compatible with one another. Therefore, specifications like SCORM runtime enable communication with different LMSs; this fosters the dissemination of learning material across SCORM-compliant LMSs.

3.12 Summary

This chapter discussed and analyzed the current state-of-the-art. Web-based learning and different approaches to the personalization in eLearning has

[20]Although it comes from a military organization it has nothing to do with military.

been covered. This chapter also presented a brief survey of AHSs and LMSs. A limited discussion and analysis was given of the current and emerging standards and specifications both for representing digital learning resources and learners and for packaging and managing learning content.

4 Requirements and analysis

4.1 Introduction

This chapter of the thesis discusses the requirements for the prototype PEDAL-NG that has been developed as part of this thesis. The prototype addresses personalization in an existing LMS. The Inspera LCMS was used as the underlying platform for development which is based on J2EE[21]. As an underlying persistence layer, the Oracle RDBMS is used to store data. We start by defining both functional and non-functional requirements. Then, we describe two user scenarios that can be too used as a “requirement”. The talks with experts are presented in this chapter and are taken into consideration in implementation of PEDAL-NG. Finally, a brief checklist for designing eLearning courses will be presented at the end of the chapter.

4.2 Functional requirements

Functional requirements capture the intended behavior of a system. This behavior is normally expressed in terms of services, tasks, or functions the system is required to perform. The following functional requirements are set for the prototype PEDAL-NG:

- Deliver a personalized learning experience based on a learner profile and the metadata associated with LOs. When the user logs in the system recognizes the returning user and renders the course pages appropriately. Prior knowledge and learning style are stored in learner profile.

- Keep track of assessment results, which are used for adaptation. LOs are tailored to a learner’s individual needs, which are expressed as metadata in the learner profile.

- Fetch relevant LOs. The system should be able to quickly and efficiently retrieve the most relevant LOs based on the learner profile. The notion of “quickly” here means that although the prototype may use a sophisticated algorithm to match the learner profile to LOs, this algorithm should not take a long time or reduce responsiveness. Ideally, the system should respond within a couple of seconds at maximum. The

[21]J2EE - Java Enterprise Edition, is open source programming platform from Sun to develop distributed web-based applications that includes technologies such as Java Servlets, JavaServer Pages, Enterprise JavaBeans, JavaServer Faces, WebSerivces, etc

notion of "efficiently" means that there may be several ways of retrieving LOs and the system should employ the most efficient method.

- Utilize conditional LOs. When a certain condition is met, conditional LOs should be shown to the user; these would give the user an impression of the uniqueness of the learning activities they are performing. As an example, a learner with excellent results could receive a list of related topics for further exploration.

- Render the course based on output preference. LOs should be stored in a format independent manner. Since XML is supported by the Inspera LCMS, LOs may be stored in XML format. For output, the XSLT [22] stylesheets can be used to produce content in different formats. An ideal system must support rendering the course online, in a PDF format, and version tailored for handheld devices. There should be the possibility for easy conversion to other rendering methods and formats.

- Support cross publishing. In a modern cross publishing environment, content should be stored in one place even when there is a need for internationalization and revision control. This facilitates reuse, because updates of any part are visible to the entire system at every location of publication. Furthermore, the LOs should be stored as logical units separating different language versions. The actual version can be retrieved at runtime based on the requested language and output preferences.

- Allow course authoring. The prototype must contain authoring support for content producers. Content represents the creation and management of LOs and their assembly into an eLearning course.

- Permit learner profile negotiation. Even with the best tool, it is not easy to create a course that is adaptable to every learner. In an ideal situation, the course designer creates LOs for each learning style or other personalization parameter. As the number of personalization parameters increases, more work is required for the design of a well-structured and fine-grained course. One example of this is the classic situation of a huge course with many LOs. Different versions of LOs must be added for different personalization parameters, which makes the work of the course designer very complicated. The prototype should

[22]XSLT is a language for transforming XML documents into other XML documents, http://www.w3.org/TR/xslt.

handle “missing” LOs by implementing an appropriate algorithm to choose the LO closest to the learner profile from the available LOs.

4.3 Nonfunctional requirements

The nonfunctional requirements set for the prototype are summarized as follows:

- Utilizing a loosely coupled architecture. The final product should be modularized and function as an independent unit within the platform on which it is based.

- Support extensibility. The prototype should be extensible, and adding more functionality should not be cumbersome.

- Allow component decomposition. The system should satisfy the principle of loose coupling between components that perform a unit of work. This should be facilitated by clean interfaces, simplifying the problem by dividing it into reasonably independent pieces that can be tackled separately.

- Implement in platform-neutral manner. Although this is not a strict requirement, it would be advantageous to utilize the “write once run anywhere” paradigm.

4.4 Use cases

When a new system development project begins, people usually generate real use cases to support the idea or proposed system behavior in specific situations. A scenario describes the proposed functionality of a new system under development. It represents a discrete unit of interaction between a user and the system. This interaction is a single unit of meaningful work. Scenarios are important tools for describing an architecture to gain information about a system’s fitness with respect to a set of desired functionality and quality attributes [72].

In this section, we will consider two scenarios that describe the foreseeable interactions of users and the prototype. Both scenarios outline problems and discuss how personalization can help to overcome the issues related to these problems.

Scenario 1, personalized Database Course for Ola

Ola is a student in Computer Science trying to register for the database course who has never taken a database course before. The course description mentions that there is no requirement of previous knowledge in databases, but it suggests that the knowledge of Entity Relationship modeling would be an asset. Ola is in doubt and continues to read the course description page. Suddenly, he comes across a comment on the use of a personalization technique which states that the system would take into account an individual user's preferences and needs. He finally decides to sign up for the course. After a while, the course becomes available on his personal space in an eLearning system. He is impatient and wants to see how the personalization technique mentioned in course description works.

In the welcome page, he receives a description of how this new personalization technique works. Basically, he finds out that he must first find out what learning style he has. The learning styles supported by the system are visual, auditory, and kinesthetic. The system can help him to identify his learning style. He is unsure and navigates to the page where he has to answer some questions. Some of the questions he answers are:

- *Do you understand something better when you try it out (a) or think it through (b)?*
- *Would you rather be considered realistic (a) or innovative(b)?*
- *When you think about what you did yesterday, do you mostly get a picture (a) or words (b)?*
- *Would you prefer to get new information in pictures, diagrams, graphs, or maps(a) or verbal information and written directions(b)?*

Roughly sixteen questions need to be answered. After he finishes this questionnaire, the system suggests based on the answers he has provided that he has auditory learning style. This information is stored in his profile, and the next step is to take a pre-test that will identify Ola's previous knowledge regarding databases and related concepts. This is necessary in order to tailor content to his level. Since he was not exposed to database concepts before, he gets very low scores; his level is adjusted to low and stored in his profile. Back on the course page, Ola reads that he can accomplish the course step by step online, print out the whole course in a PDF format, or download a mobile version tailored for handheld devices. He chooses the online version, as it is the most convenient way of learning for him.

In the first lesson, the eLearning system presents an introduction to databases and related simple concepts. Since his learning style was auditory he can either read (by switching off the sound) or have the text read to him by the eLearning application. After completing the introductory module he takes the test at the end of module, made up of simple questions such as:

- *What is a database?*
  - *It is simply a bunch of information (data) stored on a computer*
  - *It is a programming language mainly used for persistence*
  - *It is a Graphical User Interface application*
  - *Don't know / No answer*
- *What is SQL?*
  - *SQL(Sequence Quality Language) is a language used to check the quality of a database*
  - *SQL("sequel") is a language used to query databases*
  - *SQL is a standard based on XML to describe database domains*
  - *Don't know / No answer*

Having answered these questions correctly, Ola receives encouraging feedback, and the level of his questions is subsequently adjusted. At one point, Ola encounters a question regarding database transactions where he answers wrong. Since Ola's level and learning style are known to the system, a link appeared on the screen that directs him to additional pages on transactions.

Ola completes all modules and finishes the course by answering questions in a post-assessment. He scores low in this final assessment as well and the system suggests he should complete the course again, since he has now improved his level and will be presented with content tailored to his new level.

Scenario 2, personalized Math Course for Kari

Kari is a first year high school student and is registered for a math course. She has always liked mathematics and is looking forward to starting this course. She knows that she likes to learn with examples and by doing things herself, and the system identifies her as a kinesthetic learner. During the prior knowledge assessment module of the course, she encounters different questions about numbers and algebra, geometry, probability, and functions. Some of the questions are:

- *Write the power of* 2^3 *as an integral number*
  - ________
  - *Don't know / No answer*
- *Good A costs 600 kroners. The price went up by 20 percent. What would the new price be?*
  - ________
  - *Don't know / No answer*
- *Which of the following statement(s) is true*
  - *1 liter is the same as 1* dm^3
  - *Surface units can be* cm^2
  - $2,3m = 23cm$
  - *Don't know / No answer*
- *Find a solution for* $\frac{1}{6} + \frac{1}{3}$
  - ________
  - *Don't know / No answer*

Having scored 96% for each section of the course, Kari's level is set to advanced and stored on her profile. She feels that the course is interesting because it contains many real examples, and the system often asks her to complete an equation or find a solution for a problem. Amoung exercises, she has to solve problems by using the mouse and dragging and droping an answer option. Most of the items she encounters involve use of mouse in one or other way. The system responds in this way because her learning style was identified as kinesthetic.

After completing each section, she answers questions correctly with an average score higher than than 85%. She does particularly well on number theory and algebra and she gets a message that she does not have to complete the last repetition section for these topics. After completing the assessment item for the number theory section, her screen shows a message that states:

```
        You have shown remarkable performance, and this page is unique
to your merit. You can choose to follow this link to complete a more
challenging section on number theory and algebra. For each item in
this section, you will still receive proper feedback.
```

The course has been very successful for Kari. Instead of repeating things she had been taught before, she was presented with challenging tasks and appropriate feedback. At the end of the course, she passes the post-test with a score of 93%. She knows what her mistakes were and the system allows her to take the course again only completing sections in which she has made mistakes; it also notifies that she does not have to retake the course now, since the next time she logs into the system, she is recognized and can continue from where she stopped last time.

Scenario analysis

We have described two scenarios of how adaptivity and personalization of content and presentation can help learners to organize their learning environments and processes. The first scenario addresses the issue of tailoring the process to a learner with a lack of previous knowledge who would otherwise not be able to achieve same learning effect. The second scenario assists an advanced learner who needs challenging content and tasks.

4.5 Recommendations from experts

A literature review revealed that it was necessary to speak with people who have experience in the fields of pedagogy and eLearning. Interviews were conducted with Professor Kjell Skogen (University of Oslo) and Bjørn Terje Bakken (Inspera). The goal in these talks was to draw conclusions to take into account in the PEDAL-NG prototype. Among other things, learning *learning objective*,*prior knowledge*, *personlization* and *adaptive learning* were discussed with both experts. Many useful comments were gathered by talking to a researcher with theoretical background and a person with hands-on experience in working with eLearning; these comments are presented below.

Bjørn Terje Bakken has many years of experience (10+) as a project manager and key account manager. He currently holds a position as a pedagogical consultant at Inspera. His key area of expertise is in the use of technological tools to support learning. His main role at Inspera is to assist customers to use Inspera's products in the best possible manner to produce pedagogically sound content and learning experiences.

Kjell Skogen is a professor in special education at the University of Oslo and has years of experience as a researcher in education. He has authored a number of books on special education and adapted learning. He has experience as a teacher and consultant and currently works in the field of special education. Professor Skogen's research interests include development of distance education and interactions between students and teachers using multimedia and modern information technologies.

4.5.1 Organizing LOs and learning objectives

Organizing LOs is an important step in building a web-based course. When asked about how this is done theoretically, Professor Skogen answered that "*... there is not a golden standard to do that, unfortunately. But however, what should be taken into account is the analysis of learning materials in relation to what one is trying to achieve*". At schools in Norway, the normal procedure involves following guidelines set up by the authorities. In practice, this requires organizing content in such a way that what is presented should be connected to specific learning objectives. According to Bakken, this is not always easy either.

Organizations that choose eLearning for knowledge promotion often focus on cognitive objectives, especially in job-related training/courses. The leadership is often concerned with strengthening the knowledge of employees, or employees themselves can be interested in further education or specific courses. At Inspera, learning material is primarily authored by customers themselves. The same material is presented to everybody, so there is no personalization functionality. Bakken also noted that LOs should be connected to the learning objectives. One important factor involves having clear-cut learning objectives defined in advance. These objectives should be presented to learners prior to commencement of a course.

In the early phase of a project, customers cooperate with suppliers to prepare requirement specifications; this is a document describing the customer's potential system prior to design or development work. Customers often come up with a proposal for the content, and analysis and quality control of content are then undertaken. One challenging task still remains; and that is determining the sequence of learning material.

4.5.2 Prior knowledge

According to professor Skogen, having an idea about the level of learners' prior knowledge plays a significant role in eLearning as well as in traditional classroom-based learning. Assessing the prior knowledge and knowledge levels of learners is not usually done. A learner with no prior knowledge will have difficulty following the information presented. Bakken noted that a course is usually divided into modules, and each module is dependent on the previous module. The modules represent a logical path upwards, from easy to gradually more advanced modules. For example, a course on databases would probably start with defining concepts that were previously unknown to the user. Each user does have a certain level of prior knowledge about the subject, but some have more prior knowledge than others. Skogen noted that the selection of content should be based on given learner's prior knowledge. The prototype PEDAL-NG has a starting module for assessing prior knowledge. The goal of this module was to create a technique to identify prior knowledge and create a relative picture of the learner's level.

4.5.3 Entry point

Learners with varying prior knowledge will have different points of view about the class. Professor Skogen claimed that *... learners should have a logical progression by completing modules connected to learning objectives.* In industry, identifying the level of learning materials is not widely practiced. Bakken addressed this issue:

> *One of our largest customers, Aschehoug, has tried to categorize content into different levels of difficulty. What they found out was that it is very hard to do that, even for subject experts. A class of twenty students, will have different opinions about the difficulty level of particular content.*

As far as the starting point or starting level for a particular course is concerned, Bakken said it would be adequate to start at a suitable level. A suitable level is a level at which the learner's knowledge of the subject is not sufficient to achieve the learning goal.

If the goal is not to create fixed stereotypes (one of the simplest ways of user modeling), question formulation for prior knowledge assessment is not a trivial task. Other ways of creating this assessment involve the user self-determine whether or not he has sufficient knowledge on a particular topic, although this can be subjective. For example, the user can answer "Yes" in response to the question, "Do you know how to construct SQL select queries", while not actually possessing the required knowledge. In the prototype PEDAL-NG, there is no limitation on the types of questions that can be asked; this decision is left for the course authors. As we will see in the description of the prototype, questions in the prior knowledge assessment are connected to each course module.

4.5.4 Content

LOs are the fundamental building blocks of a course. Bakken noted that ... *an important thing to remember is the presentation of LOs. That includes, but is not limited to, the amount of information on the screen including text, images, video, and other multimedia content.* This problem is especially true today, young people, who are the future users of the technologies currently under development, have different approaches to these technologies. Inspera recommends to its customers that the time spent on any learning material should not exceed around 15-25 minutes. Thus, the amount of information on the screen must be kept to a minimum (also see the checklist in Chapter 4.6).

In the prototype PEDAL-NG, content selection is based in part on the learner's level, which is defined for each module. Therefore, the next question of interest was how to create questions in the prior knowledge assessment. Inspera follows the standards set by the IMS Global Learning Consortium. IMS Question and Test Interoperability (QTI) is a standard for describing electronic tests in XML format to enable interoperability between different LMSs. At the time of this writing, version of 2.1 of this specification had been released. In the Inspera LCMS, it is possible to define many types of questions. For the prototype, the following question types have been used:

- Fill in text; a very simple text input where the user is able to write text. This is useful for reflective answers to open questions.

- Fill in number: a very simple text input where the user is able to write numbers. This is useful for defining an answer that lies within an interval.

- Multiple-choice: a list of selectable elements presented with the possibility to choose one. There is only one correct answer.

- Multiple-response: a list of selectable elements is presented with possibility to choose one or more. There are one or more correct answers.

- Drag & drop: a user is given the possibility to interactively answer the question. Interactivity in this context usually means dragging floating "things" on screen to a particular point. The user may also receive immediate feedback (before submitting the answer), which can be useful in the practicing mode.

There are many other types of exercises such as drop text, fill in the missing words, or hot-spot. In the prototype PEDAL-NG, only the question types listed are used. Some exercises may also have hints, which can be configured in a manner that using hints leads to decrease of user scores by subtracting points. Finally, an exercise may have a solution proposal that is useful for practice.

Since the world of eLearning brings interactivity, Skogen suggested that it should be possible to check whether the learner has learned the key concepts by using a repetition technique. As an example, he used a language training program in which the user is presented and taught a concept. The system then checks whether the user possesses that concept. If the user has forgotten the concept, the system should remind and repeat the concept until the user learns it. This is challenging to achieve in a traditional classroom; with modern eLearning technologies, however, it can be accomplished relatively easily. Another technique involves the use of a glossary in which the user can look up an unknown term.

4.5.5 Summary

Speaking with these two experts with their great deal of theoretical and practical background was very helpful. Based on these two talks, the following list of items was gathered to be taken into consideration:

- The organization of LOs should be performed according to the learning objectives. Clear-cut learning objectives should be defined in advance.
- A course can be divided into modules. A learner should start at the appropriate level (i.e., the module with an appropriate difficulty level).
- It is important to take into account the amount of information on the screen and the length of LOs. A user should not spend much time on any one specific LO and the amount of information on the screen should be kept to a minimum.
- Learners should follow a logical progression during the course. Therefore, one should start at the specific module level that is most appropriate for the learner.
- Utilize practice items when possible. Practicing by going through examples is useful.

- Course authors should leverage the interactivity in eLearning by checking whether important concepts are learned by the users. The learner should not be able to continue until he has mastered the concept.

- Questions in the prior knowledge assessment should be connected to the course modules and test specific concepts. Questions like "do you know concept X" should be avoided.

4.6 Guidlines for designing an online course

The literature on how to design a successful multimedia learning course is rather rich. Dr. Ruth Colvin Clark is a recognized specialist in instructional design and technical training and holds a Ph.D. in Educational Psychology and Instructional Technology from the University of Southern California in the United States. Her award winning book on eLearning, *e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning* [32] co-authored with Professor Richard E. Mayer, is a useful resource for all those involved in designing and developing eLearning experiences. The authors provide a general checklist for designing eLearning courses covering different aspects of a course. This checklist has been taken into account for the prototype and the sample course developed as part of the prototype. The checklist is as follows:

- use relevant graphics and text to communicate content;

- integrate the text near the graphic on the screen;

- avoid covering or separating information that must be integrated for learning;

- avoid irrelevant graphics, stories, and lengthy text;

- use a conversational style;

- break content down into small topic chunks that can be accessed at the learner's preferred rate;

- do not present words as both onscreen text and narration when there are graphics on the screen;

- do not present words as both onscreen text and narration when there are graphics on the screen;

- transition from full worked examples to full practice assignments using fading;
- insert questions next to worked steps to promote self-explanations;
- add explanations to worked out steps;
- provide a worked example using realistic job tools and situations in the form of demonstrations for procedural skills;
- provide job-relevant practice questions interspersed throughout the lessons;
- provide explanatory feedback for correct and incorrect answers;
- provide the default option leading to important instructional methods such as practice;
- teach important concepts and facts prior to procedures or processes.

5 Architecture and implementation

5.1 Introduction

This chapter describes the design, architecture, and implementation of the prototype PEDAL-NG. All choices that have been made are documented and discussed. The prototype has been implemented in the Java programming language. Java servlet technology is used to serve requests from users. The final content produced is in HTML format, which is sent back to the client and rendered in a web browser. A few JSPs were used to customize the registration and login pages.

5.2 Inspera LCMS versus other LMSs

In Chapter 3.9 various LMSs were discussed. According to the documentation, most of these LMSs use plugin mechanisms to support specific features not included in the standard version. This means that if the feature is missing other developers may submit their work to some sort of plugin repository and make it available to everybody. Most of the time, these plugins work; however it requires spending time installing and configuring the plugin. In Inspera, all supported functionality is offered out of the box or is configurable by the administrator.

The Inspera development philosophy centers on cross-media publishing and radical information reuse. This offers the possibility to transform content into various formats. Since content is stored in a media independent format, it can be transformed to any standard XML format (e.g., IEEE LOM, IMS Learning Resource Metadata or IMS Content Packaging). The weakness of Inspera is its extensibility. Since it is not an Open Source product, certain feature requests must be filed by the customer. Another weaknesses of Inspera platform involves poor documentation and manual. One needs to possess details of how to use Inspera LCMS in order to utilize its power. This is related to the usability of graphical user interface and poor documentation. Inspera LCMS has a documentation and actually a special web-site, but this is not updated with the latest news in newer versions of its platform and it does not cover all set of functionlities. It should be noted, however, that the platform is easily internally extensible.

5.3 Design

PEDAL-NG is fully web-based, so users simply need to have a computer with Internet access. It delivers a personalized course for a group of users. To achieve this, the course is divided into four parts:

- Prior knowledge assessment: the learner's prior knowledge is identified based on questions from course modules.
- Learning Style Questionnaire: questions based on the VAK inventory (see chapter 2.4 for more details) are used to identify the learner's learning style.
- Runtime assembly of a personalized course: based on the results from previous assessment and questionnaire a personalized course constructed and delivered to the user.
- Post-assessment: after course completion, the users are asked to answer questions in a post-course assessment. Their knowledge is evaluated in accordance with the learning goals.

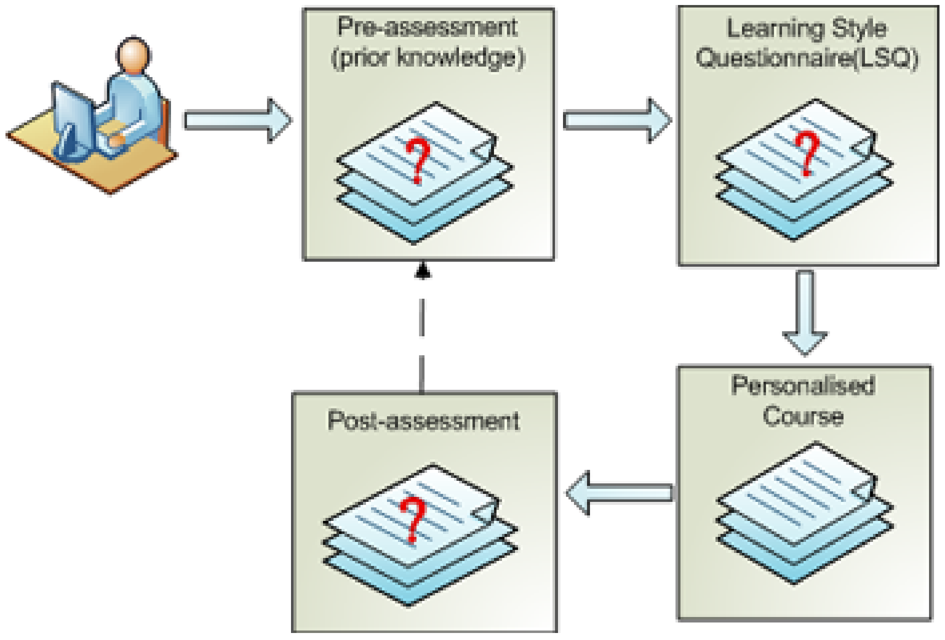

Pre-assessment (prior knowledge)
Learning Style Questionnaire(LSQ)
Personalised Course
Post-assessment

Figure 5.1: Different parts of the personalized course. PEDAL-NG uses results from prior knowledge assessment and learning style questionnaire to personalize course for different learner profiles.

Figure 5.1 illustrates how these four parts of the course are connected. The questions on prior knowledge assessment and post assessment are identical for all learners. The results are stored in the database and can be retrieved at any time using the Inspera LCMS assessment component.

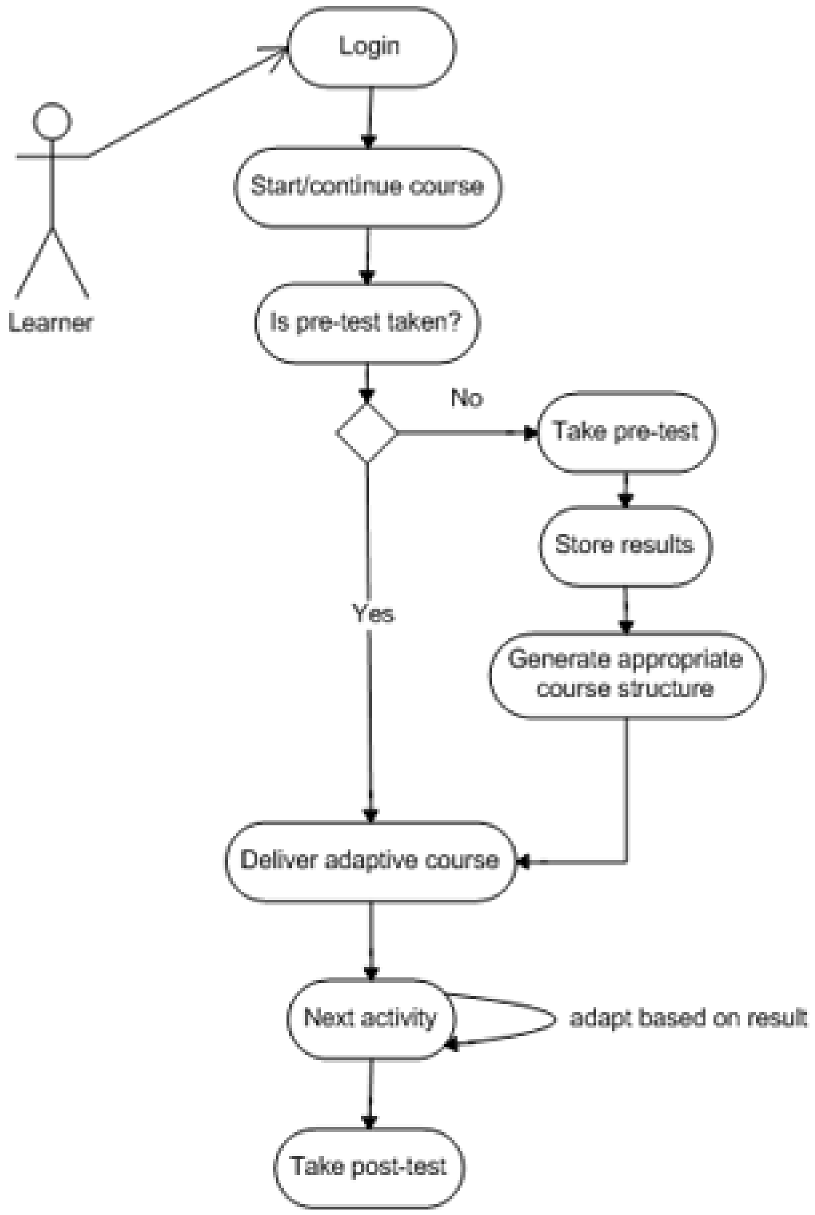


Figure 5.2: The process of personalized course construction

Process

Figure 5.2 shows the cycle PEDAL-NG follows when delivering a personalized course. The figure shows the basic flow of steps to be followed to deliver an adaptable course to a constructed learner profile. The personalized course model stored in the content management system (Inspera Content Server) does not need to be fully authored before the personalized course is delivered. The usage logs and the results from tests are used to adapt content.

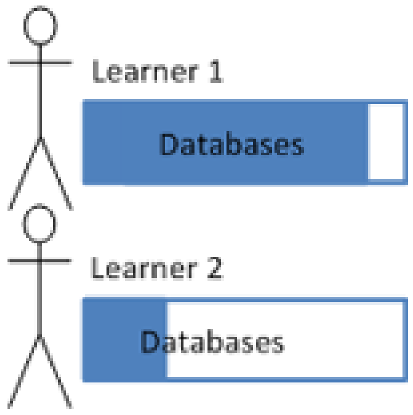


Figure 5.3: Prior knowledge of learners may vary.

Prior knowledge assessment

Using prior knowledge assessment is not the typical way to design a course to assess students' previous knowledge about the subject in existing environments. Identifying prior knowledge is an essential part of this prototype and the assessment must be completed by each learner prior to commencement of the course. Course participants may have different experiences, prior knowledge and competence as it is shown in Figure 5.3. As previously mentioned, the same activity tree is used to compose a set of questions to identify the level of the learner within certain modules. The number of questions that can be configured by setting some threshold for the number of questions in a specific level. Questions in the assessment are connected to each module in the coures. For example, a course on Java programming can be divided into syntax, control structures, collections, and server side programming modules. This enables testing the knowledge of learners on a specific subject within the course. The number of questions within the module may vary, but it should always be at least two. The results are normalized for each module. For simplicity, the normalization is performed as low when the score is between 0-55 of the total 100, intermediate when the score is between 56 and 75 and advanced for scores between 76 and 100 [23].

When the user has completed the prior knowledge assessment, the results are stored in the learner profile. This includes both points collected for each module and the overall results. Note that, the overall results are stored as metadata directly on learner, while points for each module are retrieved from *assessment results* table at runtime. Three examples of questions are shown below:

- What is an entity in the context of a relational database?
  - Entities can be thought of as categorical groupings of related information
  - Entities are any part of a database management language
  - Entities are a modeling language within the database
  - Don't know / No answer
- What does a database table consist of?
  - Compiler
  - Row
  - Column
  - Don't know / No answer
- The primary - foreign key relations are used to ...
  - Cross-reference database tables.
  - Clean-up the database.
  - Index the database.
  - Don't know / No answer

These examples show only the text based, multiple choice and multiple response (second question) questions. Some questions involve analysis before answering and may have multiple correct answers. Also note that each question has a *Don't know / No answer* option; users are instructed to choose this option if they do not know the answer. This option is included to reduce the possibility that users will guess the correct answer. A complete list of questions is included in Appendix G.

[23]These numbers represent the percentage of correct answers. For example, 5 correct answers out of 10 constitutes a 50% score.

Learning style questionnaire

The questionnaire is constructed using VAK [24] [52]. There were sixteen questions in the questionnaire. Each answer alternative resembles a profile of the learner's learning preference. Some example of questions are shown below:

- 1/16. You are helping someone who wants to go to your airport, town centre or railway station. You would:
  - go with her.
  - draw, or give her a map.
  - tell her the directions.
- 2/16. You are not sure whether a word should be spelled "dependent" or "dependant". You would:
  - see the words in your mind and choose by the way they look.
  - think about how each word sounds and choose one.
  - write both words on paper and choose one.
- 15/16. You are going to choose food at a restaurant or cafe. You would:
  - look at what others are eating or look at pictures of each dish.
  - listen to the waiter or ask friends to recommend choices.
  - choose something that you have had there before.

A complete list of questions is included in Appendix B.

5.4 Personalization support on the Inspera LCMS

The Inspera LCMS has a rich set of functionality on its platform. The following simplified list of functionality provided by the Inspera LCMS has been used in the prototype:

- user registration
- managing courses (create, change, define structure and delete)
- managing LOs
- associating metadata elements with LOs
- defining tagging structure
- defining metadata schema for learners, courses, LOs

[24]Permission is granted (see Appendix D) to use this questionnaire in this thesis. The copyright belongs to Neil Fleming.
© Copyright Version 7.0 (2006) held by Neil D. Fleming, Christchurch, New Zealand and Charles C. Bonwell, Green Mountain Falls, Colorado 80819 U.S.A.

- defining metadata elements for learners, courses and LOs
- managing pre/post-assessment

There are many other things that could be listed here, but we have focused on what is important for achieving metadata-based personalization for this thesis.

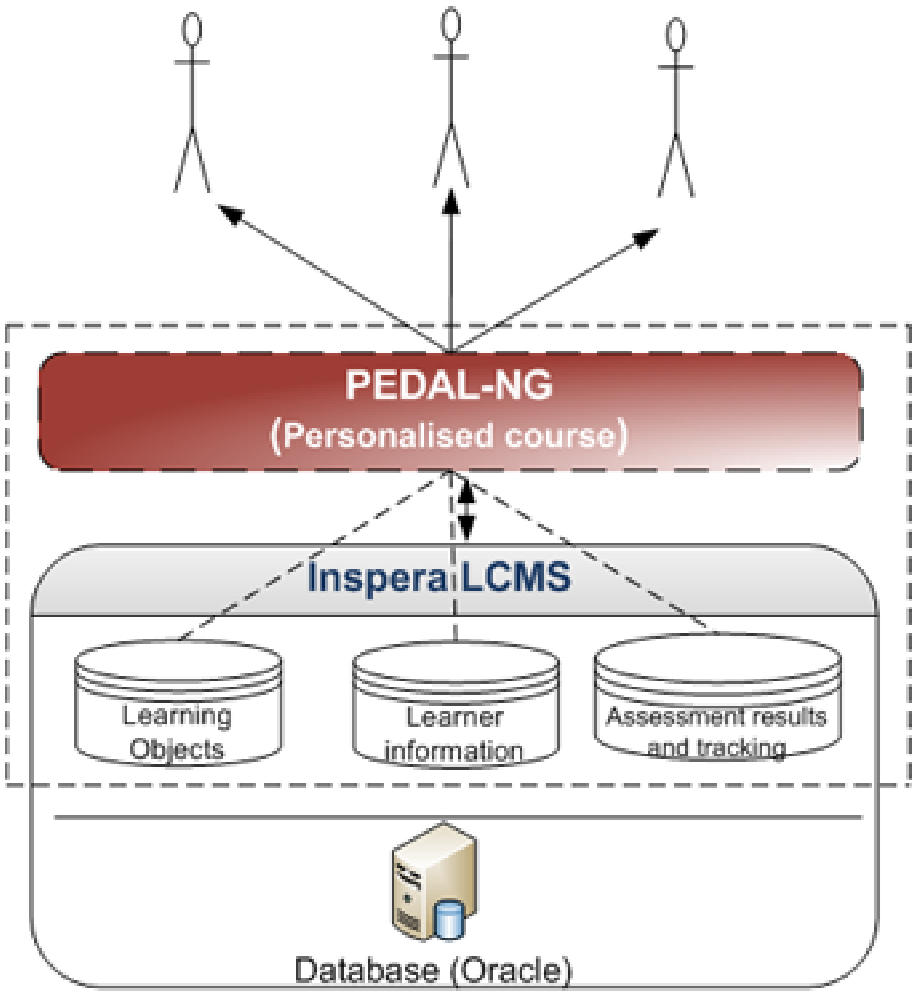


Figure 5.4: Conceptual architecture and layering PEDAL-NG on the Inspera LCMS.

Existing metadata model is flexible enough to support personalization and therefore did not require additional changes. Metadata elements that describe learning style, level, interest, and tag, however needed to be added.

In general metadata model is reused for objects. When it comes to PEDAL-NG, we needed to design an appropriate metadata schema for learners and LOs. Figure 5.4 shows both the (simplified) conceptual architecture and how PEDAL-NG utilizes the Inspera LCMS. General administrative functionality is provided by the Inspera LCMS. The Inspera LCMS has been used as well to display content, and personalization part is supported by PEDAL-NG. These two parts have been integrated so that the user would not feel that there are two separate parts in the system.

The metadata repository of the Inspera LCMS provides the capacity to describe a new metadata schema and completely customize the system to meet the user's needs. The reader is referred to chapter 5.6 for details of metadata schema used.

Figure 5.4 illustrates the conceptual architecture, depicting PEDAL-NG on top of the Inspera LCMS. The three database symbols are not meant to denote three physical databases. There is only one database, in which all information is stored. The database may however be divided conceptually into these three. PEDAL-NG tracks information about learners, such as assessment results and uses this information for further adaptation. LOs and users are physically stored in the Inspera LCMS using the standard functionality. The adaptive engine reconciles the learner information and metadata associated with LOs to compose the personalized course at runtime. Some parts of the architecture remind us of that of APeLS [34] (discussed in chapter 3.5.1). All parts of the implementation are discussed using examples subsequent chapters.

5.5 Existing functionality in the Inspera LCMS

Authoring personalized courses requires the standard functionality in the Inspera LCMS. This includes creating, organizing and managing LOs, creating courses and defining the structure of courses, associating LOs with different course modules, and associating metadata. Since these functionalities were already in place, there was no need to change the core layer (also called the kernel). The most important classes and methods are described below (see also Appendix H).

MetadataService. MetadataService is a service class used to administer metadata in the Inspera LCMS. Methods in this class are static. The class operates on objects, which are unique and have unique identifiers (Inspera-wide). By using this class, we can fetch metadata for an LO or learner (since both LOs and learners are objects, the same method is used for both). The class also has methods to define metadata on objects. Methods that have been used include:

- getMetadataForObject(): this method returns the metadata associated with an object. In the prototype, this method has been used to retrieve metadata for learners and LOs.
- getItemProperty(): retrieves the property specified by the id. The property is a metadata element.

MetadataDataAccess. This data access class stores and retrieves data from the database. The methods used include:

- storeMetaData(): Associates a set of metadata elements with an object.
- addItemValue(): Associates a single metadata element with an object.
- updateItemValue(): Updates an associated metadata element.
- newMetadataTemplate(): Creates a new metadata template when a set of metadata associated with LOs is rated by the user.

ContentPresentationDataAccess. This class is responsible for data access related to the presentation and retrieval of LOs.

- getContentItem(): retrieves a content item (LO) by a specified id and uses incremental caching.
- getLiveContentRevisionWithEffectiveLanguage(): retrieves the active version of a content item for a specified language. If a version in the requested language is not available, the second prioritized language is used until a version (in any first language) is found.

AssessmentDataAccess. This class is responsible for data access related to the retrieval of assessments. Because assessments can have a complicated structure, this class is dedicated to providing data access to assessments. The methods used include:

- getLastUserAssessment(): retrieves the last assessment completed by the user.
- getAssessment(): retrieves the assessment specified by the id.
- insertAssessmentOutcome(): inserts the outcome of the assessment to the database.

HTMLAssessmentServlet. This is a servlet that displays assessments (prior knowledge, exercises, and post assessments). Typical usage occurs when the user starts the assessment. This servlet is used until the user completes assessment. The tracking is maintained continuously until it is marked as completed. The result of the assessment is stored in the database.

ContentDisplayServlet. This servlet handles the display of LOs. It uses the built-in templating engine in Inspera LCMS. It is possible to display the same LO in different formats. For example, a text-based content item can be displayed in HTML format in a web browser, or it can be converted to PDF format. Content once can be reused in different contexts. The same content source is used fior transformation into different formats. This is achieved by storing content in XML format and using XSLT to transform it to different formats.

5.6 Architecture

The component architecture of PEDAL-NG is shown in Figure 5.5. Although services can be embedded within a specific engine, they are divided into logical units to make the model understandable for the reader.

The overall architecture gives an overview of the big picture of PEDAL-NG. The architecture is based on the existing Inspera application, and necessary functionalities were added to support adaptivity. Although these functionalities are not included in the standard Inspera product, personalized courses are shown as if they were general. A description of each component in the architecture is given below.

Learner

The learner who uses PEDAL-NG for learning purposes is the end-user of the system. Metadata describing learners are stored in Inspera LCMS. Table 5 summarizes metadata for learners with an example.

Author

The author is the person who creates and maintains the course. As stated in the requirements chapter, the authoring tool is embedded in Inspera LCMS.

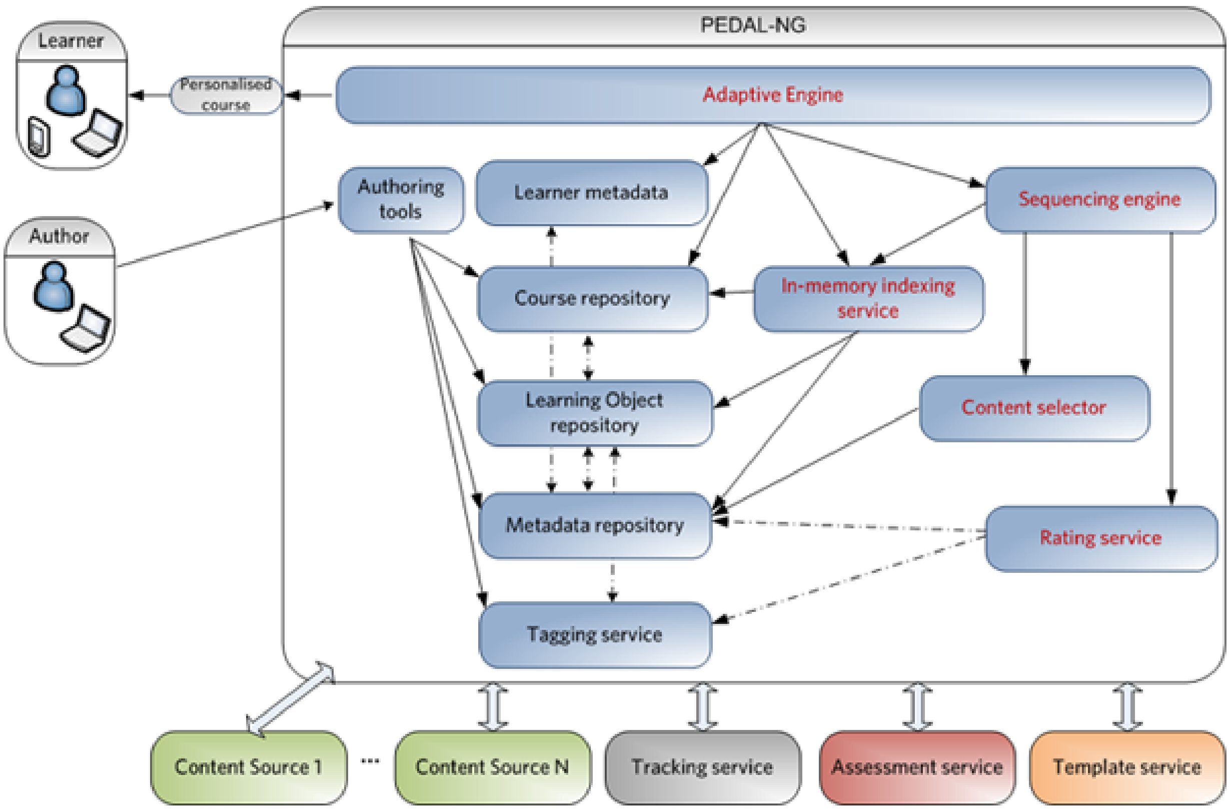


Figure 5.5: Component architecture of the system. The parts of PEDAL-NG are in brown color.

Adaptive Engine

The adaptive engine is the core of PEDAL-NG. It provides services to achieve adaptivity and makes possible the creation of the personalized course. It also uses metadata elements stored in Inspera LCMS to produce personalized courses and track activities. The adaptive engine is implemented in a modular way by applying loosely coupled architecture and using MVC pattern; this means that it should be easy to extend and add new functionality.

The adaptive Engine is responsible for ensuring that the sequencing engine has all the information it needs to select appropriate LOs. When a learner starts the course, the adaptive engine uses an in-memory indexing service to index the tree structure. Because of the dynamic nature of a personalized course, the index cannot be static and has to be generated to keep itself up-to-date with the information associated with LOs. To index the course, this component needs course information provided by the course repository. When all nodes in the tree are indexed, it calls the sequencing engine to define the "best" sequence of LOs that matches the current learner's profile.

Course repository

A course is represented, as are all other type of LOs (e.g., documents, examples, images, video, sounds) as a logical LO. This means that support to define a metadata schema and tagging is added without any further modeling or programming. The course repository holds information about course. Specifically, the structure elements are nodes that have associated LOs. There is no limit to the number of levels in the structure. A sample of course metadata is shown in table 6.

Metadata element	Description	Data type	Example
id	unique id of learner	number	85845
firstname	learner's first name	string	Naimdjon
lastname	learner's last name	string	Takhirov
email	learner's email	string	takhirov@idi.ntnu.no
address	learner's address	string	Sem Sælands vei 7-9
mobile	learner's mobile	number	98490659
birthdate	learner's date of birth	date	1995-01-20T00:00
level	learner's level based on prior knowledge assessment	string	mid(low,mid,difficult)
learning_style	learner's learning style	string	visual (visual, auditory, kinesthetic)

Table 5: Metadata for Learners

Metadata element	Description	Data type	Example
id	unique id of course	number	58546
name	name of course	string	Introduction to Databases
creator	creator of LO	string	Naimdjon Takhirov
creation_date	date when the course is created	date	2008-01-20T10:42

Table 6: Metadata for course

Learning Object Repository

Learning Object Repositories (LORs) traditionally [85] store both LOs and their metadata either physically together or separately. Since the metadata repository is used for storing metadata for all types of objects (not only LOs), data are shown as separate components in the figure. Table 7 shows the possible metadata elements for LOs.

Metadata element	Description	Data type	Example
id	unique id of LO	number	92345
type	type of LO	number	image
title	title of LO	string	Database modeling chart
creator	creator of LO	string	Naimdjon Takhirov
creation_date	date when the item is created	date	2008-01-20T10:42
level	difficulty level of an item	string	easy(easy,mid,difficult)
learning_style	learning style for which this item is best suited	string	v (visual-v, auditory-a, kinesthetic-k)

Table 7: Metadata for LOs

Metadata repository

To assemble a course, a wide range of heterogeneous LOs are used. The basic method of arranging and managing these resources is to associate them with metadata elements that will also facilitate discovery and foster information retrieval. Furthermore, we do not need the original (physical) LO to process data.

The metadata in Inspera is stored separately from its original resource. Thus, the metadata do not depend on a specific resource type. Through the reuse of the metadata modeling infrastructure, it is not necessary to reinvent the wheel; instead we can reuse the existing system. Therefore, both the course repository and LOs use the same metadata repository but different metadata schema. The Inspera LCMS supports creating new metadata schemas and associate it with specific types of objects.

Tagging service

Tagging refers to the practice of attaching descriptive labels to a resource. There are many reasons why metadata cannot be used instead. One is that the practice of tagging does not require an extra *value*. Another reason is that tagging may be hierarchical, which is a step towards more accurate information representation. Finally, tagging is a widely practiced technique by end-users and easier to perform as opposed to defining the metadata schema. Tagging has received much attention in the web community and has recently become associated with Web 2.0. The tagging support is simple and it is embedded within the MVC framework that follows PEDAL-NG. However, this functionality has not been tested.

In-memory indexing

In-memory indexing is done at the beginning of the course. It does not persist any information to the database or har disk (on the server) because of the dynamic sequencing of LOs. It is only used to filter information and rank the results. The search is performed among the available LOs within a specific topic. Lucene [6] is internally used by Inspera LCMS for text-searching. Therefore, there was no need to include this library in PEDAL-NG. In-memory indexing and searching have been chosen as the strategy for retrieving the relevant LOs for the user because they make it easy to extend queries. A sample query for a module looks like the following:

```
+parentId:653765 +learning_style:visual +level:mid interest:sports
```

This query performs a search on the index for the module with the id 653765, for a visual learner with an intermediate level and an interest in sports. The result of this query may include learning objects of theory, example, and excercise type and it is the job of adaptive engine to decide in what sequence they are presented to the user (based on learning style). This query can be easily extended if a new parameter is introduced. For example, if we also want to include a rating parameter, we can extend the query:

```
+parentId:653765 +learning_style:visual +level:mid interest:sports rating:5
```

Sequencing engine

The job of the sequencing engine is to evaluate the learner properties and set up a sequence of LOs presented during runtime. This sequence is based on assessment scores from previous modules (which adjust the learner's level), results from the prior knowledge assessment and the learning style questionnaire.

Content selector

The reason for separating content selection logic from the sequencing engine is to divide the tasks performed by these two components. The content selector is responsible for selecting LOs that are available for the current topic. For example, if the topic is SQL queries, the list of available LOs would include SQL queries for a learner at low level (less competence), intermediate level, and advanced level (learner with advanced competences). Additionally, it would also include content elements for learners with visual, auditory, and kinesthetic learning styles. This list can be large, and the content selector makes this list available to the sequencing engine. The sequencing engine must decide which content elements should be presented in what order. The content selector retrieves the content elements based on the metadata associated with the LOs.

Rating service

The rating service is used to rate metadata and tags that are associated with the LOs. When an LO is rated, it is not the actual LO that is rated, but instead the metadata associated with it. Rating information can be used to present a user with an LO that is of more interest to him. This information is then used to select appropriate content elements that would most likely be of more interest to a learner. In a database course, a learner with more interest in cars would find it interesting to model data related to cars. Another example involves setting up student working groups based on similar interests that hopefully will foster collaboration between the students in these groups.

Template service

The templating service is part of Inspera's solution. It is used to both produce an HTML page and transform content into various formats. For PEDAL-NG, only transformation into HTML format is used; for future work, however, the content can be transformed to both support learning on handheld devices and generate compendia. It can also be used for XML-based interoperability with other systems.

Assessment service

The comprehensive assessment service in Inspera supports import and export of tests according to the IMS QTI v1.1. The assessment service not only provides administrative functionality but also enables learners to take assessments. Results of the assessments are stored for further processing and then used to adjust the level of the learner within a specific topic.

Tracking service

The tracking service, as the name implies, tracks all activities performed by a learner. It stores tracking information, and this information is used by the sequencing engine. The tracking service is modularized, so it does not depend on other services. The tracking engine was in place in Inspera LCMS and did not require any additional changes.

Content source

The content source represents a source where the LOs are created. These content sources can be both internal and external. External LOs must be imported prior to commencement of a course. These LOs are imported using a standard XML format, such as IEEE LOM or IMS Content Packaging.

5.7 Class diagram

The prototype is implemented in the service layer of Inspera LCMS. Inspera's service layer combines all of the services that can be applied to the main functionality. The kernel represents a domain model, essentially the core entities that are normally stored in the databases. Examples of such entities include *User*, *ContentItem*, and *ContentRevision*. In order to understand the class diagram, a basic understanding of the other grounding classes in Inspera's solution is necessary. A description is given below [25]:

no.inspera.kernel.BaseObject. This is the main object in the Inspera platform [26]. All objects that need metadata tagging, security checking, or usage logging extend this object.

no.inspera.applications.publish.datamodel.ContentItem. This class extends BaseObject and represents a logical content item. By logical we mean that an item is independent of actual content. The content is not stored as

[25]The description only includes main classes related to the implementation of the prototype and existed in Insperas code-base before the prototype was developed.

[26]Inspera-wide, this can be compared to java.lang.Object for all objects that need metadata functionality and security check.

part of ContentItem class. One ContentItem can have many versions (e.g., versions in different languages). Each version can have content and metadata associated with it. By using ContentItem, one can define security settings in a generic way so that all versions of the same ContentItem will have the same settings. The same idea applies to usage logging.

no.inspera.applications.publish.datamodel.ContentRevision. This class represents different versions of the same information object (i.e., content item). A revision is language-dependent and extends BaseObject. This class contains the actual content (e.g., text-documents, images, audio, and video content).

no.inspera.applications.publish.datamodel.ContentFolder. This class extends ContentItem. As its name suggests, it represents a folder where items can be stored. This class reminds us of the folders in Windows Explorer and thus can have a hierarchical structure. This means that a folder is not restricted to items but can have folder children.

no.inspera.applications.publish.datamodel.ContentActivityNode. This class extends ContentFolder and is used as a node for organizing a learning activity. A single activity is a unit of instruction in hierarchically organised section of a user's course. For example, if we want to create an item that describes a theory about a topic, contains an exercise item, and offers an assessment, then we will have to create three ContentActivityNode objects for these activities. Additionally, ContentActivityNode can have sub-activities (i.e., children ContentActivityNode items).

no.inspera.applications.publish.datamodel.ContentActivityRoot. This is the root item for a course, and contains ContentActivityNode objects. A course can have only one root item represented by ContentActivityRoot.

Now that we have covered the entities from Inspera's platform used in the prototype, we can describe the most important classes in the prototype. The adaptive engine is written using a POJO [27] paradigm and MVC pattern. Model-view-controller (MVC) is an architectural pattern used in software engineering [55]. The strength of this pattern lies in the fact that it isolates business logic from user interface which results in an application where it is easier to modify either the visual appearance of the application or the underlying business rules without affecting the other. The MVC consists of:

[27]POJO is an acronym for Plain Old Java Object.

- a model that represents the business rules used to manipulate data, both in terms of data itself and the application state;
- a view that represents elements of the user interface (e.g., text, checkbox, button) and sends user inputs to the controller;
- a controller that maps user actions to data models and selects a view for the response.

A small MVC is included as part of PEDAL-NG (a complete source code of PEDAL-NG is included in Appendix H and as an attachment). The functionality provided by this MVC is minimal, but it is sufficient enough to be extendable. It provides a registration service, where new actions can be registered. A typical action is “start” which fires up the application and configures PEDAL-NG for the current user. This results in an indexing of the course structure, in which scores are configured via the fetching of assessment results. PEDAL-NG includes the following out-of-the-box actions (see Appendix H for more detailed information):

- *Start* - starts PEDAL-NG, usually when the user logsin to the course
- *Configure* - configures PEDAL-NG, set ups sequence of LOs, and calls the indexing service
- *Index* - indexes the course structure to the memory
- *View* - displays the LO in a requested module
- *Login* - authenticates user using provided credentials
- *ShowIndex* - shows the current index in the memory, and is useful for debugging
- *Rate* - rates the set with metadata elements associated with object
- *TagObject* - tags an object with the specified label found in the request
- *Empty* - empty action.

All PEDAL-NG actions implement an *Action* interface (Figure 5.6). All actions return an instance of *Result* interface. The following three classes implement the Result interface: (a) *HTML*, which means that the result of an action is the rendering of an HTML page, (b) *Forward*, which requests PEDAL-NG to forward user to a specific page, and (c) *XMLTransform*, which instructs PEDAL-NG to transform a generated XML to HTML page.

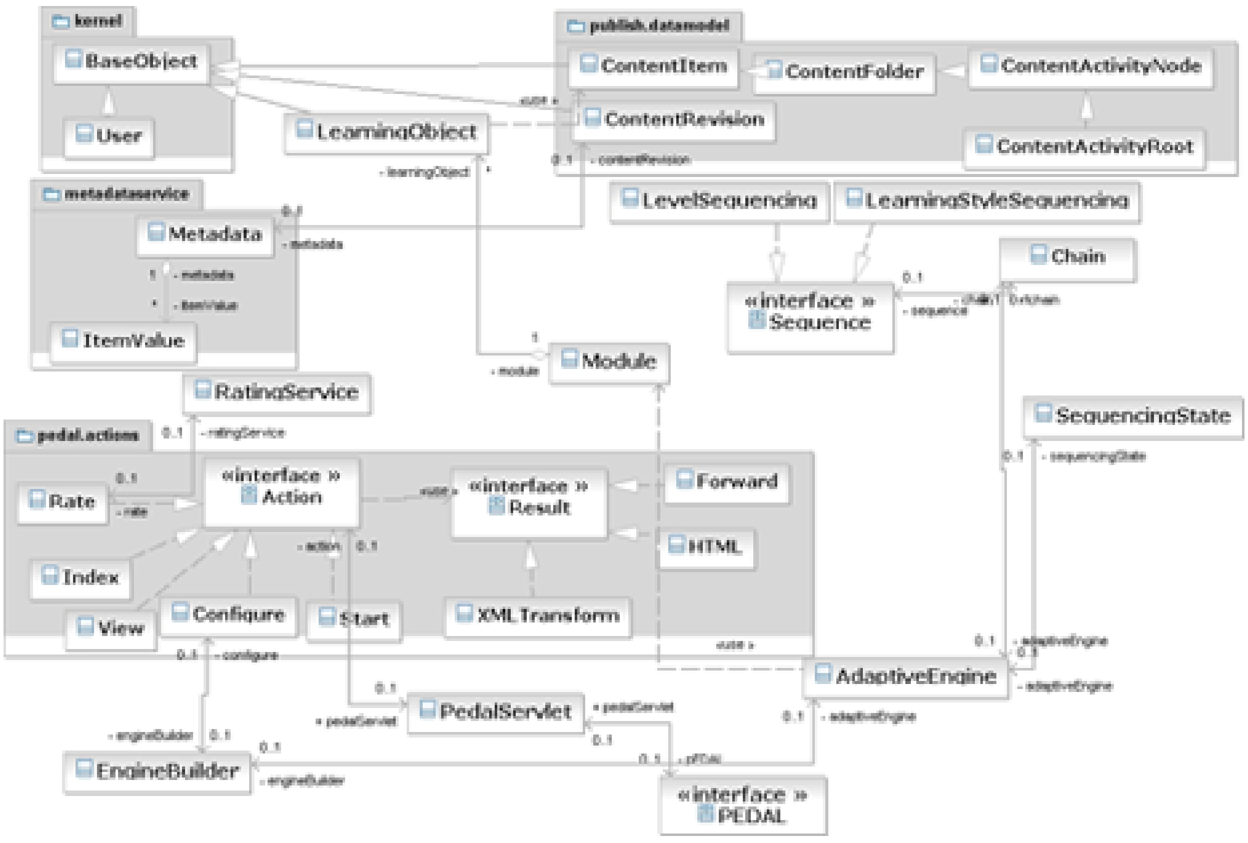


Figure 5.6: PEDAL-NG class diagram (included with higher resolution in Appendix I as well).

The MVC pattern was used because in this pattern we can decouple components (i.e., separate functionality in a modular way), which in turn makes it possible to change parts of system easily and enables easier maintainance. Furthermore, using this pattern provides possibilities to extend the prototype.

The most important classes (apart from MVC) in the prototype are presented below (see javadoc in Appendix H for a complete list) [28]:

pedal.PEDAL. This interface holds all constants used by the prototype and thus makes refactoring easier.

[28]The common package name no.inspera.services. was stripped for readability.

pedal.LearningObject. This class represents a learning object. It is used to simply wrap the ContentItem (discussed earlier) with extra properties. By introducing new properties to this class, there is no need to change ContentItem, which is organized in Inspera's kernel.

pedal.Module. This class resembles a module in a course. It can have one or more associated learning objects.

pedal.PedalServlet. This class is a servlet that processes user requests. This servlet is only responsible for locating and executing an action to be performed. The actions are those described earlier and are part of the developed MVC framework.

pedal.engine.AdaptiveEngine. This class provides the implementation of the adaptive engine. It is responsible for sequencing LOs based on the learner's profile and the metadata associated with LOs. The *Sequence* interface defines how items should be sequenced.

pedal.engine.Sequence. This class is the interface in a sequencing chain. The single doSequence() method defines the sequencing task to be done by the implementation classes. The prototype has two classes that implement this interface; these classes are described below.

pedal.engine.LevelSequencing. This class implements the *Sequence* interface. Level sequencing is responsible for sequencing LOs based on the learner's level. Currently, this job is left to the in-memory indexing, and this class simply returns the first LO. However, this class also enables to support more sophisticated sequencing logic based on level.

pedal.engine.LearningStyleSequencing. This class provides the second implementation of the *Sequence* interface. it sequences LOs based on learner's learning style. The current implementation has a very simple logic. Visual and auditory learners receive items in order of theory, example, and exercise. Kinesthetic learners are presented items in order of exercise, example, and theory.

pedal.engine.Chain. This class represents in which the sequencing logic can be described. In a sample course, it has been tested with the *LevelSequencing -> LearningStyleSequencing* order.

pedal.engine.EngineBuilder. This class uses a builder pattern to build a sequencing engine and is used by the *Configure* action in the controller.

pedal.engine.SequencingState. This class holds the state of the sequencing information, including such information as which item is currently displayed and which items have already been displayed.

5.8 Selection of learning objects

LOs are selected at runtime by the adaptive engine. This selection depends mostly on the metadata associated with LOs, metadata associated with learners, and usage logs for which the built-in tracking mechanism in the Inspera LCMS is used. The selection is performed by running an in-memory search. An important implementation choice holds the index of the course structure in the user session (when the user logs in). As we will see in the evaluation chapter (see 6), this has not had any particular effect on performance [29]. In-memory indexing was chosen because such a dynamic environment requires the construction of dynamic criteria for the selection of LOs. For example, a learner may change his/her interests and thus change the criteria for filtering non-relevant hits. Furthermore, by employing searching for selection rather than comparing objects each time, we can simply change the query used for retrieval. Using this technique of dynamic query construction gives the possibility to avoid re-deployment.

Appendix A illustrates a screenshot of the authoring tool [30]. As shown in the screenshot, each module has an associated folder where 1 - LOs are stored. The selection of LOs are performed in runtime, by the in-memory indexing component that performs the in-memory search. This is one of the distinctive features of this prototype's implementation approach. From one point of view, this may seem quite abstract for the course author, since it may seem difficult for a course author to know what LOs the user is served with at runtime. This actually depends on metadata associated with LOs and the learning profile. For example, the screenshot in Appendix A(first part, marked with **1**) shows that the first module has an associated folder called "*0.1 Entities and Tables*". On the second part (marked with **2**) of the screenshot, we see a folder structure. The folders listed here are those that are associated with course modules. When the learner starts the course and

[29] Assuming objects are cached into memory.

[30] The standard authoring tool was extended to be able to associate folders with a course module.

navigates to this module (*Entites*), the search is performed on this folder (recall the query example where we used “parentId:” as a criteria). If the learner has not been exposed to the theory, a LO of type theory is presented first. That means that the LO, which is quite simple, will be tagged as “*low*” for the example, if the learner has a low level in the specified module. If the module should be skipped by advanced learners, then the course author does not publish content a metadata element tagged as low level. Thus, 0 hits are found when the search is performed. PEDAL-NG interprets this result as a skip action.

5.9 Sample course based on PEDAL-NG

In order to test the prototype, a course on *Designing Relational Databases* was created in a tutorial like fashion. This course was not official or certified. The different portions of the course have been downloaded from the web, and its main content is based on a similar course developed by Dr. Clark [32]. Permission to use portions of that content was granted by publisher Wiley (see Appendix C). In relation to the Bloom's taxonomy of educational objectives (discussed in chapter 2.7), content of the course covers the bottom parts of the classification levels. Necessary items (such as theory, practice, and assessment) have been created by the author. The course structure is described below:

Module	Number of LOs
00. Prior knowledge Assessment	
01. LSQ (Learning Style Questionnaire)	
02. Entities	23
03. Table construction	19
04. Parent and child tables	27
05. Entity Relationship Diagrams	13
06. SQL	20
07. Post-assessment	

Table 8: Number of LOs in each module.

Each module consists of one or more LOs, and an LO can be used by other LOs. For example, an image represents an LO and can be used either as an illustration in a text document or as a video object with text around it. The Entities module (01) provides a basic introduction to entities and how to identify entities in the context of a video rental store. The entities identified include customer information , movie information and rental information. After the entities have been identified, the learner progresses to the next module. This module focuses on constructing tables (02) based on entities that have been identified. One of the strengths of using databases as opposed to traditional spreadsheets is the reduction of data redundancy. Therefore, the next logical module would be description of how to connect parent and child tables (03) using primary and foreign keys. A short introduction to entity relationships (ER) (04) is given, along with an example ER diagram based on the entities and relationships discussed earlier. Since the target group does not include technical users, only basic SQL queries are also discussed in the next module (05). In this module, users have the option to run live SQL queries against a dummy table that is hosted in the same database. The course concludes with a post-assessment.

In the sample, LOs are described using metadata in a new schema created by the author. The schema for LOs is very simple. Apart from the title, description, creation date, and other such properties, they also have *learning_style*, and *level* metadata elements. The latter elements describe the learning style and difficulty level for which each LO is suitable. These are the metadata elements used by the adaptive engine to sequence LOs and personalize the learning experience.

5.9.1 The structure of the course

The course structure and contents are depicted visually in Figure 5.7. Each module is comprised of a group of LOs used to teach that specific module. Theory, example, exercises, and assessment were included in modules. A module represents a *node* as described in chapter 5.6.

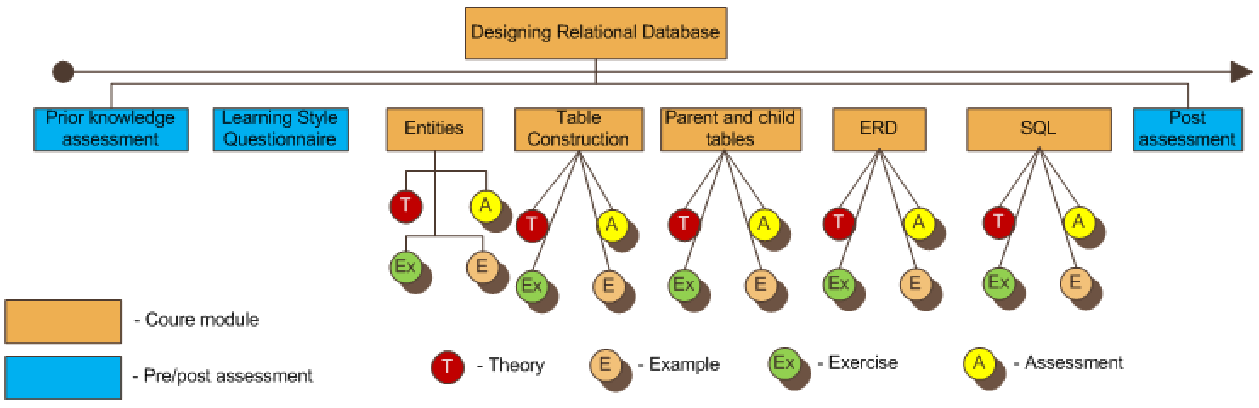


Figure 5.7: The structure of the course on Designing Relational Database

Participants are asked to log in when navigating to the course web-page for the first time. It is possible to register and ask the system to resend a forgotten password. When the user logs into the system, the system checks whether the user has already taken the prior knowledge assessment. If the user has not, he is redirected to the assessment page. When the assessment is completed, the results are stored in the database and can be retrieved during subsequent logins. The learner has the option to retake the prior knowledge assessment to adjust his competence level or learning style. Course pages are designed without too many graphics, which would unnecessarily draw the learner's attention away from important things. This is discussed in more detail in [32].

Questions in the prior knowledge assessment were categorized based on five modules. There were a total of 21 questions: Entities (3), Table construction (2), Parent and child tables (4), ERD (2), and SQL(10). The number of questions in each module varied depending on the depth of the module. By clicking next, the user is brought to the learning style questionnaire. A complete list of questions is included in Appendix B . It was not possible to browse modules if these steps were not completed. When the two modules have been completed, the user can browse course modules. The course and its structure were defined by the author of this thesis. The most difficult parts were the organization of the content, metadata-tagging with level (low, intermediate and advanced) and creating different material formats to address different learning styles.

The structure of the course may change for lower level (not in the module level). A learner with high competence does not need to go through introductory material; however review may offer a good strategy to remember those things previously learned. On the other hand, the sequence of items may be changed depending on the learning style. A visual learner is presented with texts and visual material before exercises and assessments, while a kinesthetic learner may prefer to do exercises examples first to get hands-on experience.

5.9.2 Personalization

Most of the modules have LOs of the exercise and example types. Different methods for showing LOs for users with different learning styles are shown in Figure 5.8 for a visual user, Figure 5.9 for an auditory user and Figure 5.10 for a kinesthetic user.

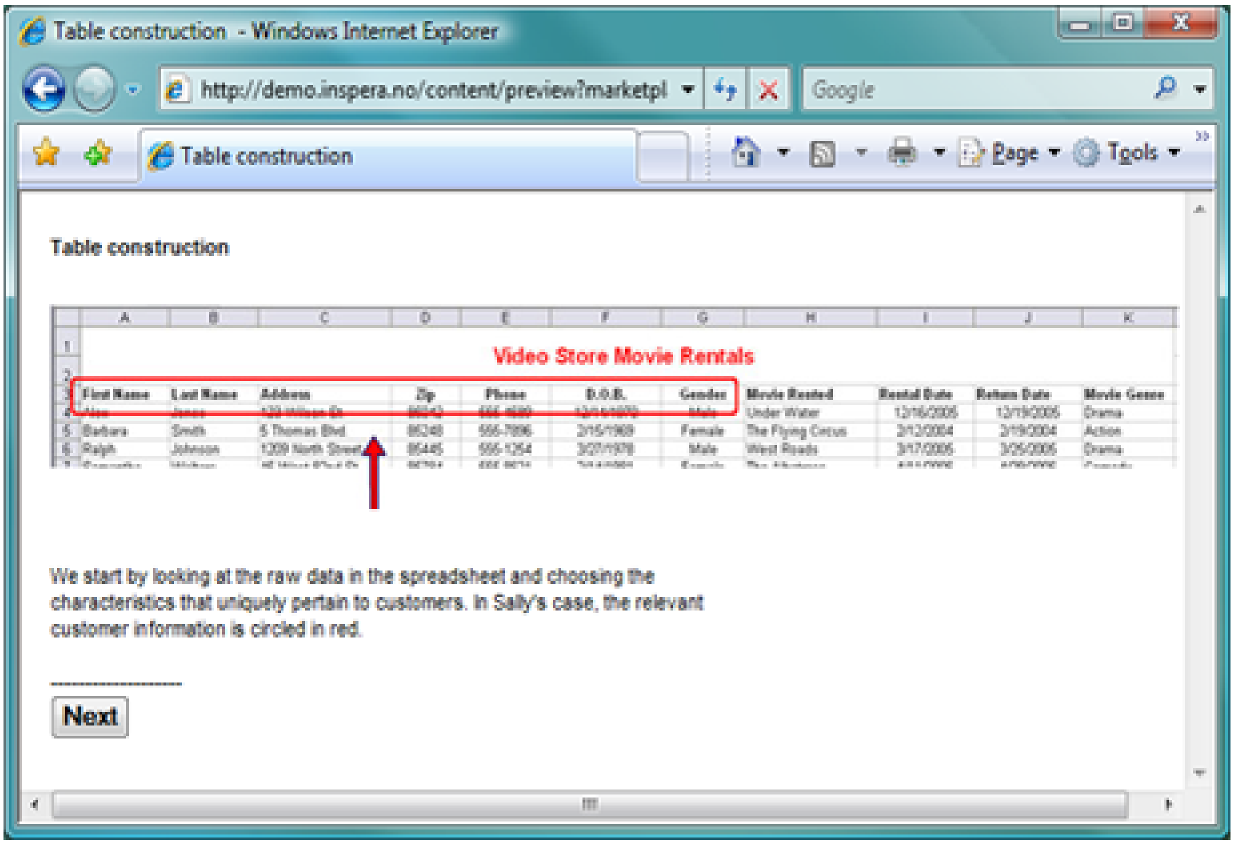


Table construction - Windows Internet Explorer
http://demo.inspera.no/content/preview?marketpl
Google
Table construction
Page
Tools

Table construction

	A	B	C	D	E	F	G	H	I	J	K
1						Video Store Movie Rentals					
2											
3	First Name	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	Movie Genre
4	Alex	Jones	123 Wilson Dr	86542	555-4589	12/14/1970	Male	Under Water	12/16/2005	12/19/2005	Drama
5	Barbara	Smith	5 Thomas Blvd	86248	555-7896	2/15/1969	Female	The Flying Circus	2/12/2004	2/19/2004	Action
6	Ralph	Johnson	1209 North Street	85445	555-1254	3/27/1978	Male	West Roads	3/17/2005	3/26/2005	Drama
7	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Female	[illegible]	[illegible]	[illegible]	[illegible]

We start by looking at the raw data in the spreadsheet and choosing the characteristics that uniquely pertain to customers. In Sally's case, the relevant customer information is circled in red.

Next

Figure 5.8: A sample view of content for a visual learner.

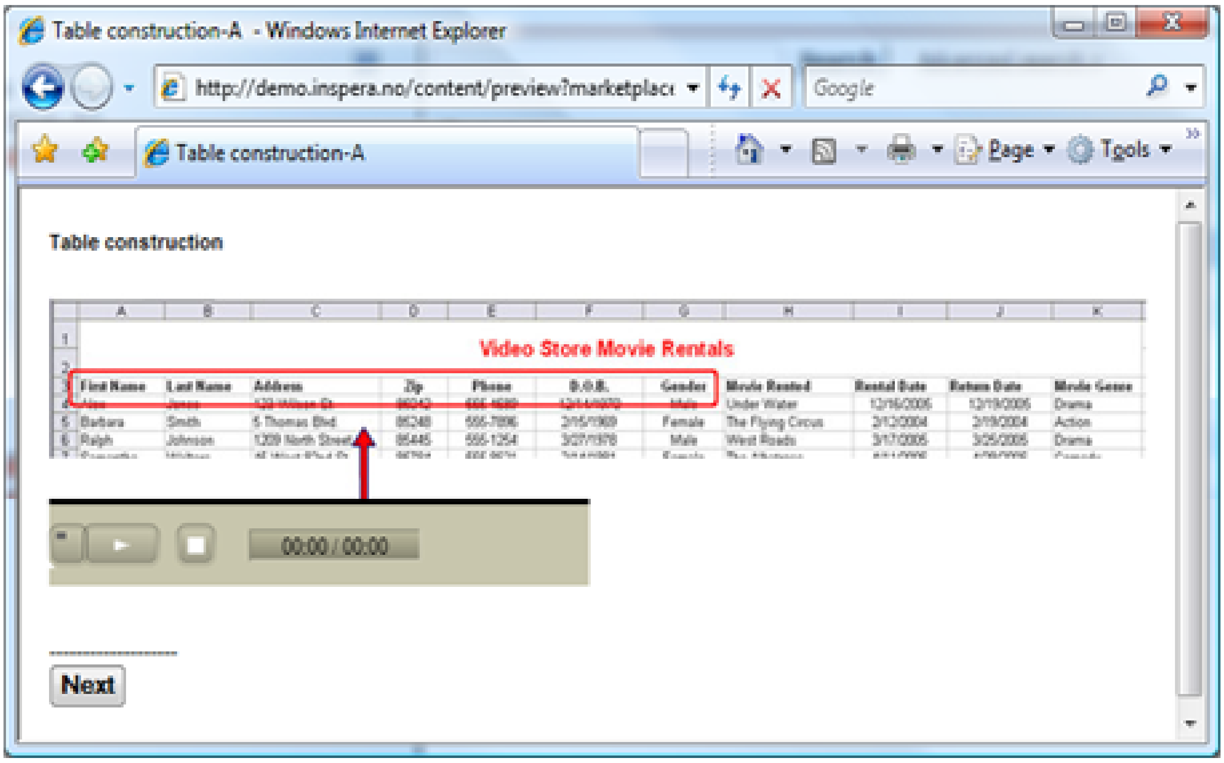


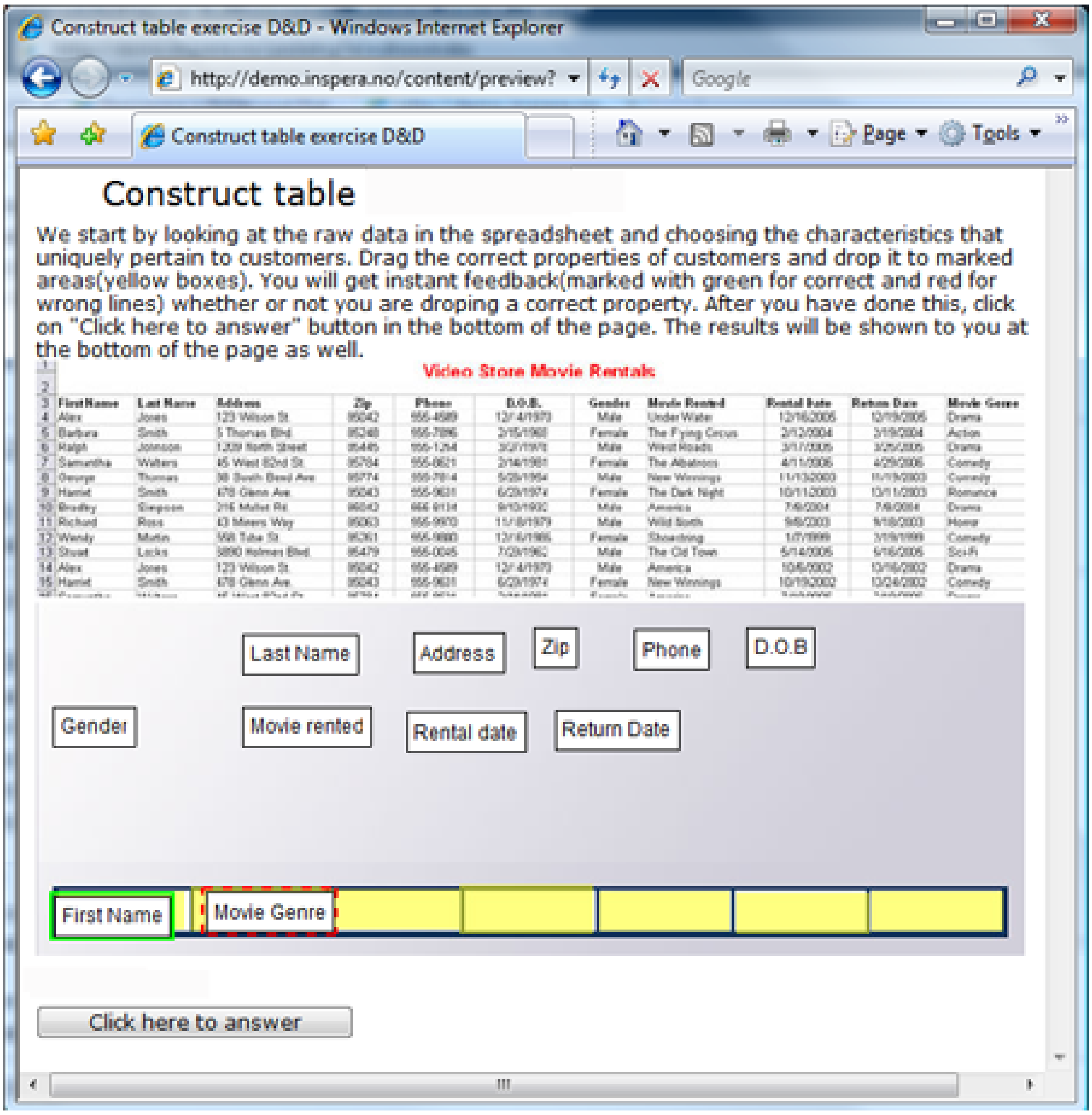
Table construction-A - Windows Internet Explorer
http://demo.inspera.no/content/preview?marketplac
Google
Table construction-A
Page
Tools
00:00 / 00:00
Next

Table construction

Video Store Movie Rentals

First Name	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	Movie Genre
Alex	Jones	123 Wilson St	86042	555-4689	12/14/1970	Male	Under Water	12/16/2006	12/19/2006	Drama
Barbara	Smith	5 Thomas Blvd	85248	555-7896	2/15/1969	Female	The Flying Circus	2/12/2004	2/19/2004	Action
Ralph	Johnson	1209 North Street	85445	555-1254	3/27/1978	Male	West Roads	3/17/2005	3/25/2005	Drama
Samantha	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	Female	[illegible]	[illegible]	[illegible]	[illegible]

Figure 5.9: A sample view of content for an auditory learner. The content is presented in a form of sound.



Construct table

We start by looking at the raw data in the spreadsheet and choosing the characteristics that uniquely pertain to customers. Drag the correct properties of customers and drop it to marked areas(yellow boxes). You will get instant feedback(marked with green for correct and red for wrong lines) whether or not you are droping a correct property. After you have done this, click on "Click here to answer" button in the bottom of the page. The results will be shown to you at the bottom of the page as well.

Video Store Movie Rentals

3	FirstName	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	Movie Genre
4	Alex	Jones	123 Wilson St	95042	555-4589	12/ 4/1970	Male	Under Water	12/16/2005	12/19/2005	Drama
5	Barbara	Smith	5 Thomas Blvd	95248	555-7896	2/15/1960	Female	The Flying Circus	2/12/2004	2/19/2004	Action
6	Ralph	Johnson	1209 North Street	95445	555-1254	3/27/1970	Male	Weird Roads	3/17/2005	3/25/2005	Drama
7	Samantha	Walters	45 West 82nd St	95784	555-0821	2/14/1981	Female	The Albatross	4/11/2006	4/29/2006	Comedy
8	George	Thomas	98 South Bend Ave	95774	555-7814	5/26/1954	Male	New Winnings	11/13/2003	11/19/2003	Comedy
9	Harriet	Smith	478 Glenn Ave	95043	555-9631	6/29/1974	Female	The Dark Night	10/11/2003	10/11/2003	Romance
10	Bradley	Simpson	316 Muller Rd	96043	444-9114	9/10/1932	Male	America	7/6/2004	7/6/2004	Drama
11	Richard	Ross	43 Meers Way	95063	555-9970	11/18/1979	Male	Wild Sloth	9/8/2003	9/18/2003	Horror
12	Wendy	Martin	568 Tulsa St	95361	444-9880	12/16/1986	Female	Showdown	1/7/1999	3/19/1999	Comedy
13	Stuart	Locks	5890 Holmes Blvd	95479	555-0045	7/29/1962	Male	The Old Town	5/14/2006	5/16/2006	Sci-Fi
14	Alex	Jones	123 Wilson St	95042	555-4589	12/ 4/1970	Male	America	10/6/2002	10/16/2002	Drama
15	Harriet	Smith	478 Glenn Ave	95043	555-9631	6/29/1974	Female	New Winnings	10/19/2002	10/24/2002	Comedy

Figure 5.10: Presentation of sample content for a kinesthetic user. The content presented in a form that gives the user opportunity to perform actions. In this example, by placing the field onto the marked areas the user receives an immediate feedback from the system. In this illustration, the user places genre as a field for customer entity and the system marks the border with the red color (i.e., the system signals an “incorrect” action).

5.10 Limitations

The purpose of developing this prototype was to demonstrate a proof of concept, not to deploy it in production [31]. Although is not limited in terms of the types of courses, the prototype has been tested on a group of users with a sample course on *Designing Relational Databases* (see chapter 5.9). In the prototype the following types of content were used:

- Text items. Various kinds of text werestored in XML format.
- Multimedia. Multimedia elements were represented as standalone items or embedded in the text to make eLearning lessons more interesting.
- Assessment items. Assessment items included a set of questions related to the curriculum or specific topics.
- Question. A question item may include various forms of audiovisual content and text.
- Exercises. Exercises are used within the module to repeat concepts that have been discussed in specific modules.
- Examples. Examples were used to illustrate concepts. For example, kinesthetic learners prefer to do things; therefore, the use of interactive examples could be leveraged to tailor content to this learning style.

The prototype supports derscriptions of learners' interests [32], but this feature has not been tested with real users and data due to the limited time frame and lack of pedagogical competence. The idea, however, is that learners should be served with content matching their interests. A learner with an interest in cars will most likely be interested in course examples that deal with cars.

In this thesis, personalization of a course on a PC was investigated. Although, personalized courses delivered to PDAs or as eBooks are out of the scope of this thesis, their development is not far in the future. Delivering courses as eBooks, however, would require exercises to be completed offline and results to be checked during subsequent Internet connection.

In the functional requirements chapter, the retrieval of conditional LOs was noted; however, this has not yet been implemented. Learner profile

[31]During evaluation by users, several bugs have been encountered.

[32]Dummy content elements have been created to test functionality.

negotiation and course-rendering based on output preference were also not implemented. The limited time frame forced us to focus only on the main areas in this thesis.

6 Evaluation

6.1 Introduction

The evaluation part of this work is divided into two parts: performance and user experience with the prototype. Below we present a description of the results and a summary of findings.

6.2 Disclaimer

As the quantity of data being analyzed in the evaluation is not sufficient to accurately represent the real world, the results shown below are not statistically significant. The goal of the evaluation was rather to simply find a way to evaluate the system and propose a possible method for performance evaluation.

6.3 Performance

Performance testing has recently become very important due to the vast amount of information produced by people daily and the need to access relevant information quickly and efficiently. Performance testing is primarily related to determining how fast some aspect of a system performs under a particular workload.

For the prototype PEDAL-NG, JMeter [5] was used to test performance. JMeter is an open source Apache Jakarta project for load testing, analysis, and performance measurement of a variety of services with a focus on web applications. Its functionality is not limited to load testing, and JMeter supports assessment (assertions) to ensure that the data received is correct, enables configuration of the variables, and offers a variety of reports.

The tests were run on a laptop that was set up as a server. The laptop had an Intel Pentium Dual Core T5600 / 1.83 GHz with 2 GB of RAM running Windows Vista Business. PEDAL-NG was deployed on a Resin application server [33] (version 3.0.14 standard edition). To obtain more objective results, the JMeter tests were executed by a separate machine so that the server only served the incoming requests. The database was run on a separate server [34].

[33] Resin is application server from Caucho. See www.caucho.com/resin for more information.

[34] Although much depends on database, querying database has not been tested. This is due to the established infrastructure at Inspera and that the prototype only developed on

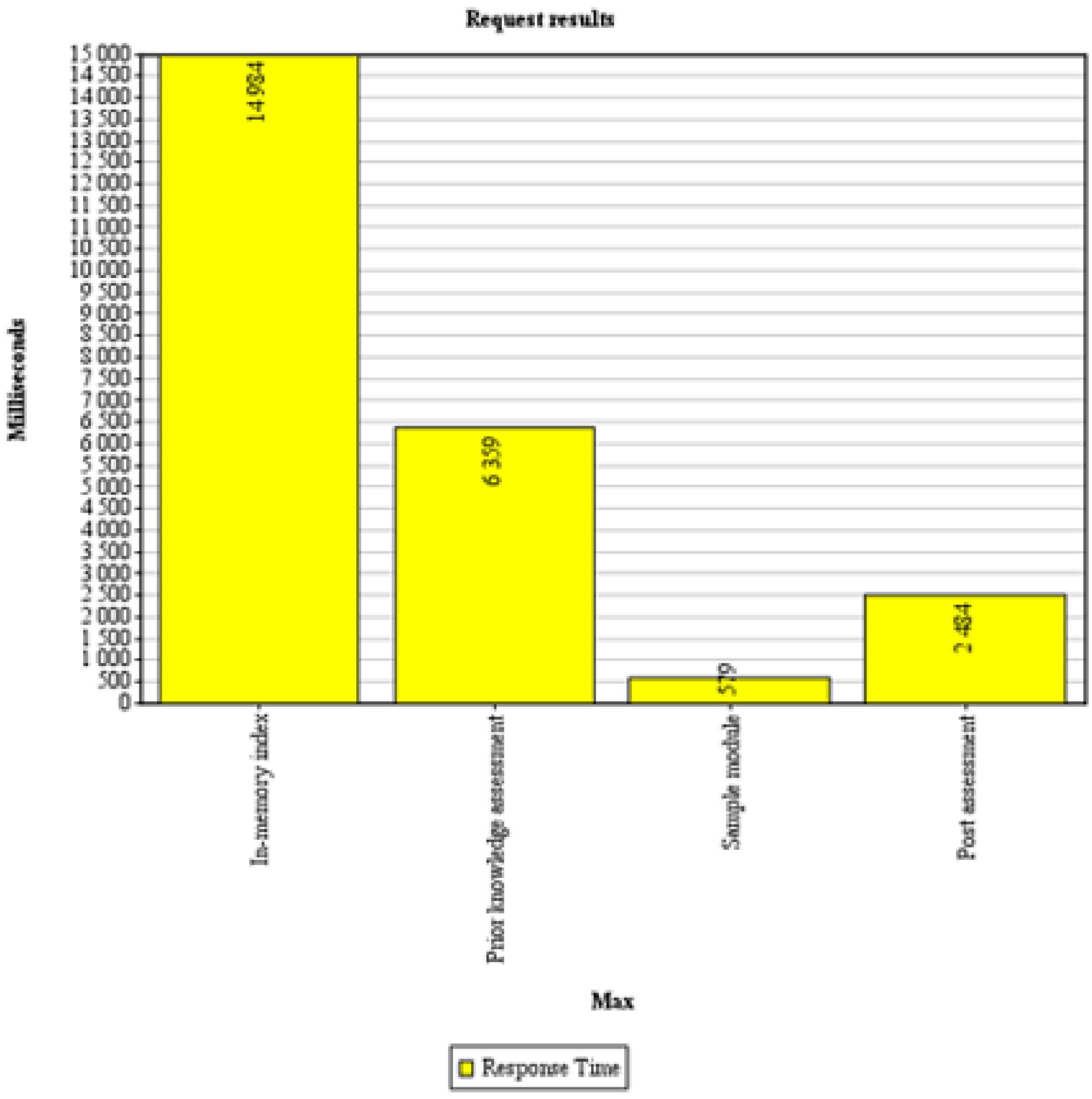
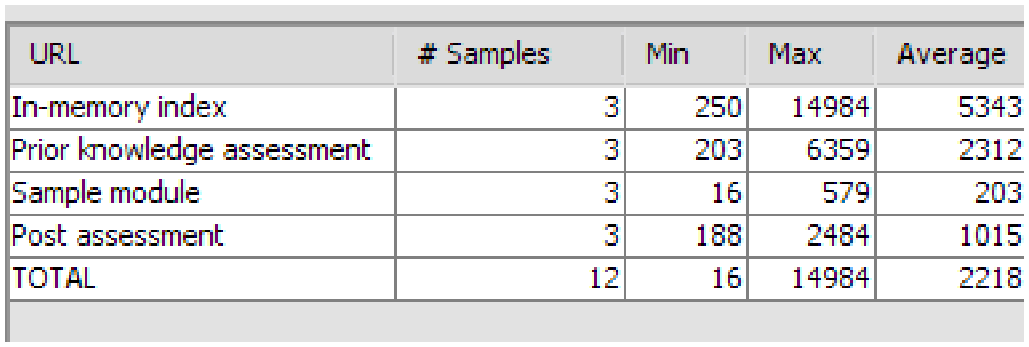


Figure 6.1: JMeter maximum time in milliseconds for each request.



URL	# Samples	Min	Max	Average
In-memory index	3	250	14984	5343
Prior knowledge assessment	3	203	6359	2312
Sample module	3	16	579	203
Post assessment	3	188	2484	1015
TOTAL	12	16	14984	2218

Figure 6.2: JMeter results

The entire test plan for PEDAL-NG is included in Appendix I.

6.3.1 In-memory indexing

Performance evaluation is not the core of this work, but is useful for providing a benchmark performance. When the user logs into the course, the whole course tree is indexed into the memory in application server on the server-side. This information is stored in the user's session. This strategy was chosen to permit the quick accessing of data and efficient searching. In-memory searching facilitates the dynamic configuration and building of queries. A good example of its use occurs when we want to have an extra adaptivity parameter, for example context; it can be added to the query without querying the database. Therefore, it is important that the in-memory indexing function performs well. As shown in Figure 6.1 in-memory indexing requires the most processing time during its initial run [35]. The figure shows the maximum time spent by the server during the serving of requests. However, it is interesting that subsequent requests take roughly 250 milliseconds ("Min" column in Figure 6.2). This discrepancy is explained by caching. The first request takes a long time, and there are many round trips to the database. Once these items are fetched, however, they are cached into the memory; thus no additional requests are made for subsequent requests.

top of it.

[35]First time means when the application server has been restarted and the first request is made by the first user

6.3.2 Assessments

Assessments also resemble tree-like structures and were objects of study, since retrieving assessment items may take a long time. Figure 6.2 reveals that the first request takes roughly 6359 milliseconds. The server can handle subsequent requests in roughly 203 milliseconds; this value is more or less acceptable. As above, subsequent requests take less time to process due to caching. There is a slight difference between the prior knowledge assessment and the post assessment. This difference arises because the post assessment has fewer elements.

6.3.3 Modules

Browsing a module is of primary importance, because this action is performed by the user many times during a session. When the user clicks on the “Next” button, PEDAL-NG displays the next LO in the sequencing chain. The test demonstrates that the performance of displaying the LO in the sequence is acceptable which takes at most half a second.

6.3.4 Scalability

The performance evaluation did not take scalability into account, an important factor in performance evaluation. If PEDAL-NG is to be deployed to a production environment, this evaluation is necessary in order to generate realistic numbers and predict system performance. One method of testing scalability would involve setting up JMeter in a server mode and arranging more machines as slaves. This way we could simulate a large number of simultaneous users submitting requests to the server.

6.3.5 Summary

Performance tests have shown that initial requests take long time to process for the sample course. The processing time for subsequent requests is reduced dramatically. This is due to the incremental caching technique employed by Inspera LCMS. Since PEDAL-NG performs searches in-memory, it is vital that objects are cached into the memory so that there is no need to access the database. However, room for improvement remains. Initial requests should not take so much time for processing, because users may think that the server has failed to respond and hit the refresh button in the browser.

6.4 User experience

6.4.1 Introduction

This chapter discusses users' experiences with the prototype. The evaluation of PEDAL-NG within a course on Designing Relational Databases was carried out by users who completed the course and learned about relational databases. We conclude with a summary of the key findings of the evaluation. The primary goal of the experiment was to obtain information regarding participant attitudes towards different aspects of the adapted content. Further, we wished to know whether the prior knowledge assessment and learning style questionnaire were useful in mapping the learners' prior knowledge and learning styles.

6.4.2 Participants

A total of fourteen users participated in the experiment and represented a demographically diverse test group. Some participants were friends, and others were invited by friends of the author. Participant age ranged from 23 to 42, with six males and eight females. Participants were sent an email with a link to the web site where they could register (the invitation email is included in Appendix E). Participants had different backgrounds and were from different cultures. All users were fluent in English (the language of the course) and had basic experience with using computers. There was no requirement for previous experience with databases. Table 9 below summarizes the participants' levels, learning styles, and prior knowledge and post assessment scores. Note that levels shown are the levels for the whole course.

6.4.3 Evaluation survey

Each participant was presented with an identical online survey (using SurveyMonkey [36]). A complete list of survey questions is included in Appendix F, and the email for participants is included in Appendix E. Figure 6.3 presents the evaluation setup.

The evaluation survey was comprised of thirty-one, principally multiple choice questions. At the end of each section, there was a space for open comments. The questions were divided into six different areas:

[36]SurveyMonkey is an online survey tool, that enables creation of online surveys easily; http://www.surveymonkey.com/

1. Information regarding the *participant* was obtained first. Questions were demographic in nature and addressed age, sex, and contact information.

2. *General* questions about the participants' background and their attitudes toward online learning were presented in the next section.

3. Next, questions related to the *prior knowledge assessment* (i.e., its appropriateness and participants' view of its usefulness) were given.

4. The fourth section was a survey regarding the *learning style questionnaire.*

5. Participant satisfaction with the *course content* and their views of the personalized course were addressed next.

6. Finally, satisfaction and participant views regarding the *post assessment* were obtained.

The survey was divided into these parts in an attempt to examine learner satisfaction with different parts of the course. We present some of the key issues related to each part below.

Participant ID	Level	Learning style	Prior knowledge assessment results	Post assessment results
635596	low	kinesthetic	45%(10/22)	96% (25/26)
633764	low	visual	41% (9/22)	34% (9/26)
633761	low	visual	50% (11/22)	73% (19/26)
633758	intermediate	auditory	63% (14/22)	80% (21/26)
633752	low	kinesthetic	54% (12/22)	46% (12/26)
633605	low	kinesthetic	18% (4/22)	73% (20/26)
633596	intermediate	visual	68% (15/22)	85% (22/26)
633593	low	auditory	0% (0/22)	54% (14/26)
632637	low	visual	0% (0/22)	69% (18/26)
631652	advanced	auditory	77% (17/22)	88% (23/26)
632634	low	auditory	32% (7/22)	46% (12/26)
632594	low	auditory	9% (2/22)	61% (16/26)
631673	low	kinesthetic	0% (0/22)	61% (16/26)
631673	advanced	visual	91% (20/22)	65% (17/26)

Table 9: Information about each participant.

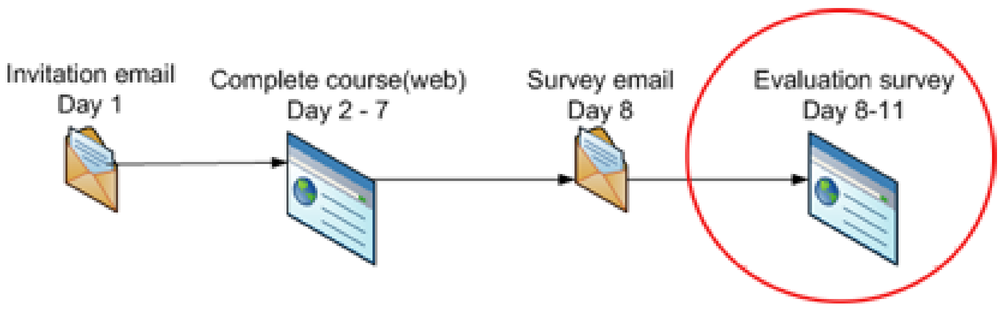


Figure 6.3: Experiment setup where the current evaluation survey being described is marked with red oval.

General questions

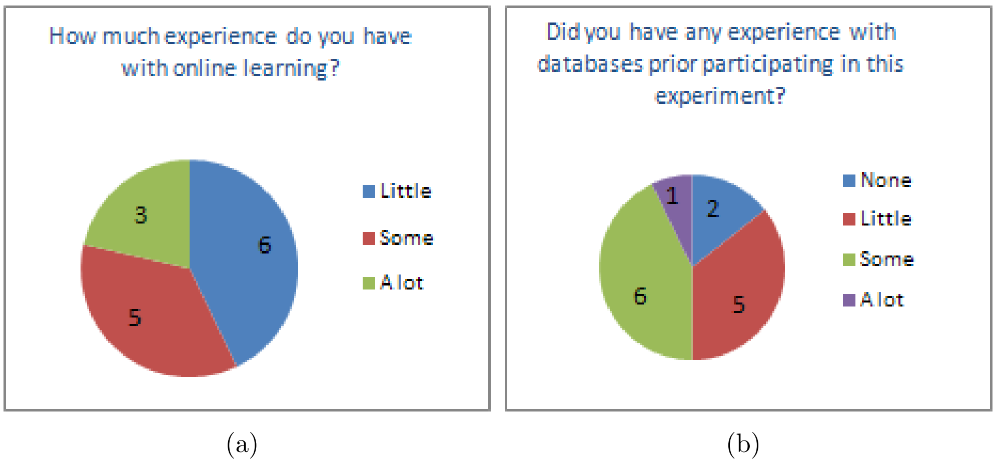
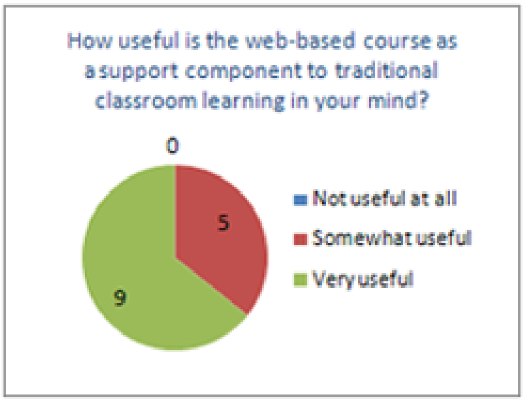
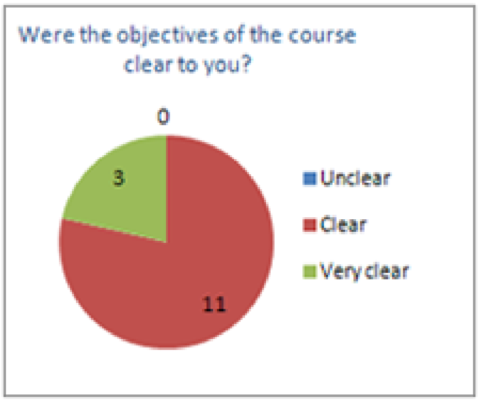


Figure 6.4: Participants' previous experiences.

The general questions were designed assess the level of participants' utilization of the personalized course offering. This can be important and may be used to evaluate and elaborate on the results of personalization. If a learner is not comfortable with the use of technology and the web for learning, then this information can be used as context in which to view the answers given regarding the personalized course. The figures below are self-explanatory. Figure 6.4 shows that participants had different backgrounds with varying degrees of experience with online learning.



(a)



(b)

Figure 6.5: Partcipants' view of online learning and their understanding of course objectives.

Most of the participants had little or some database experience. This diverse level of experience is interesting, because the persionalized course delivered was also diverse. Past experience has mainly an affect on personalization based on prior knowledge. The effectiveness of the personalized course would have been reduced (in terms of prior knowledge-based personalization) if all participants had the same level of experience.

Today, information technologies are becoming ubiquitous, and also play an increasingly larger role in the context of education. The participants' attitudes towards online learning supports this idea (Figure 6.5). None of the participants had a negative view of using web for online learning, and most of participants were eager to learn new material on the web. This finding impacts any results relating to learner satisfaction in using personalized courses.

In Chapter 4.5, we had a recomendation that the objectives of the course be very clear from the beginning. As shown in Figure 5(b), participants have understood the course objectives; no participant failed to understand the objectives at all. The three participants that gave *Very clear* as an answer to the question in Figure 5(b) were advanced (two of them advanced and one intermediate with 68% correct answers) users, or who were recognized by the system as having much prior knowledge about the databases. This sounds reasonable, since participants with less prior knowledge may not know what the records or fields are[37]. Therefore, it might be useful to use appropriate "language" to describe objectives so all learners are able to understand.

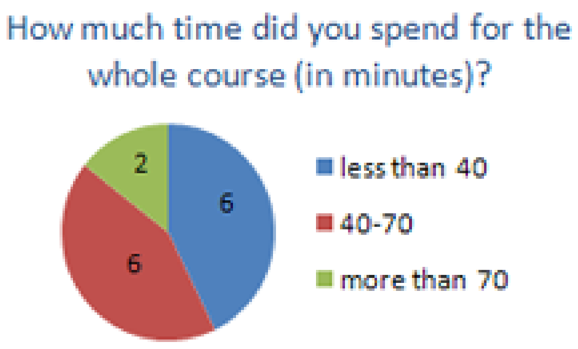


Figure 6.6: Time spent by participants.

Another recommendation was a course of action with regard to length of time spent on each LO (see Chapter 4.5.4). The time spent by each participant on the course is shown in Figure 6.6. Two participants completed the course in less than 40 minutes. One of these learners had a lot of experience with databases and the second one had some. This result indicates that participants with more prior knowledge received a limited amount of LO, since their level of experience required that introductory material to be skipped.

Prior knowledge assessment

There were six questions about the prior knowledge assessment in the survey. Since the use of a prior knowledge assessment is not widespread in existing LMSs, these questions were meant to uncover participants' views toward this kind of test.

Figure 6.7 illustrates that the learners would like prior knowledge to be assessed in two ways. They want the system to gather information about their prior knowledge (by asking questions in the course on designing a relational database), and they also want to submit this information themselves. In Chapter 4.5, we argue that questions should be connected to the course modules to avoid statements like "*do you know topic X*". The reason that

[37]These words were used to describe course objectives, see Appendix G (Welcome screen) for more information on course objectives.

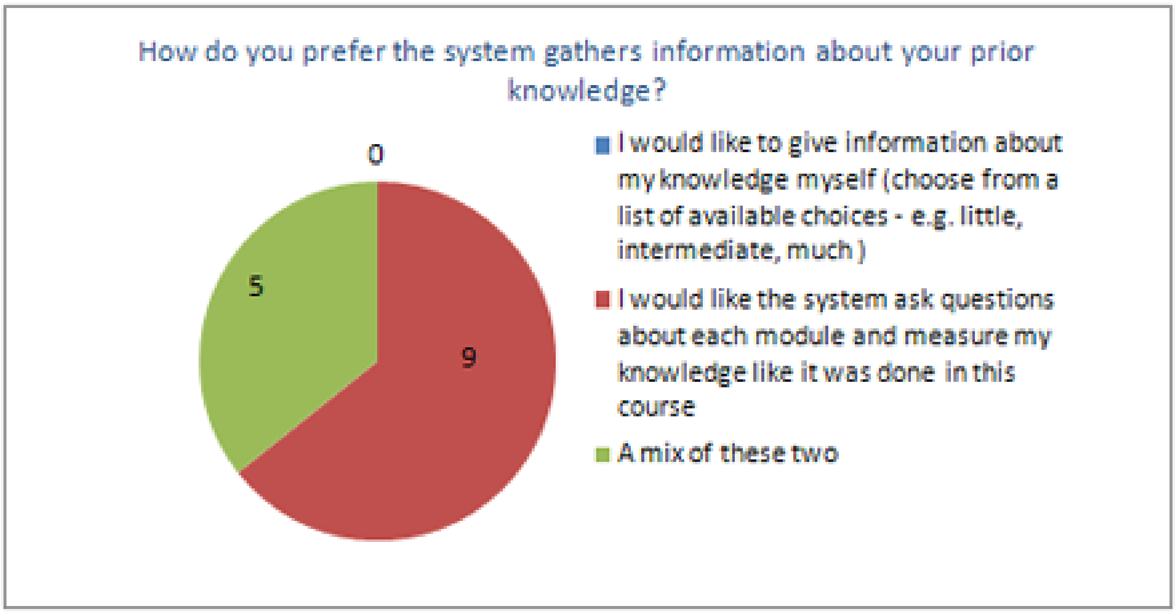
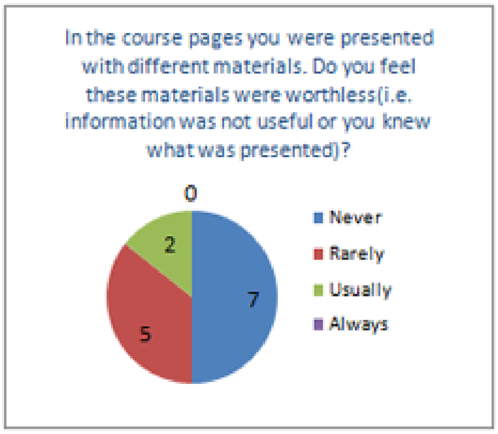


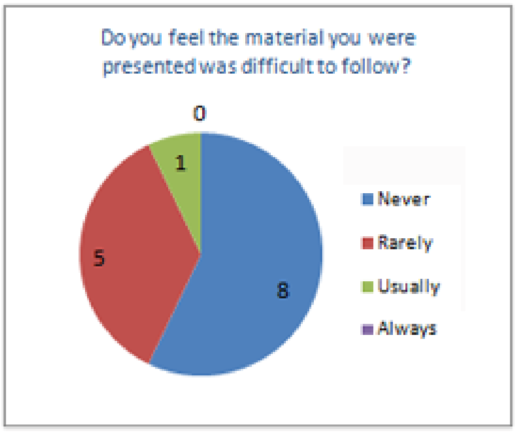
Figure 6.7: The participants' preference regarding collection of the prior knowledge.

users would like to use both of the techniques (Figure 6.7) may be rooted in the fact that some questions may have appeared too challenging for some participants. However, this is exactly the point of a prior knowledge assessment, in which the prior knowledge of users is mapped. The lesson learned from these answers is that the course authors should carefully design questions with the actual content in mind.

The level of prior knowledge was used to personalize content based on the difficulty level; advanced learners were not presented with simple introductory content. Figure 6.8 shows that none of the users thought that the material was always useless or that they always knew what was presented. Only two of the participants felt that the content was usually not new or worthless; these participants scored low on post-assessment (34% and 46%). This is explained by lack of motivation or simply by the fact that the course was not interesting enough for these participants. Five users rarely experienced it as such, and the remaining seven participants said that they never saw uninteresting or useless content. Roughly the same results were obtained for questions regarding content difficulty, and only one participant thought the content was usually difficult to follow. The learning style of participants who answered these two questions varies, but the two participants who stated



(a)



(b)

Figure 6.8: Participants' view regarding course material.

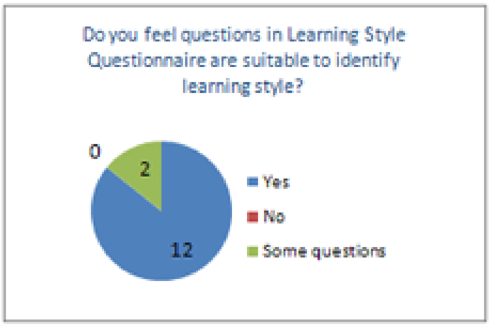
that they did not like material were kinesthetic learners. This indicates that there were not so many content modeled for this learning style and techniques (e.g., sequencing objects based on learning style) may not be reasonable and there is more research to be conducted in this area.

Learners stated (not shown in figures; a complete list of questions is available in Appendix F) that they would like to spend an average of 20-30% of the total time on the prior knowledge assessment. Further. they all wanted similar prior knowledge assessment techniques to be used in the next web-based course in which they participate.

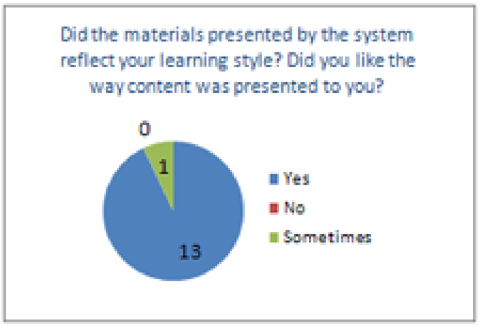
Discussion

The answers to the questions related to the prior knowledge assessment indicate that participants are not reluctant to spend time on an assessment that evaluates their competence before starting the course. The majority of participants stated that they would want to both provide information regarding their prior knowledge by themselves and have the system test their competence. Participants with kinesthetic learning style require special attention, because they answered that they did not like the material or there were not many learning content. This was due to the number of learning objects prepared for kinesthetic learning style. The important suggestion regarding this issue is that course authors should ensure there is no gap that creates a room for incorrect personalization.

Learning style questionnaire



(a)



(b)

Figure 6.9: Participants' view on learning style questionnaire.

The questions regarding the learning style questionnaire were used to determine whether the questions were appropriate for 1) identifying users' learning style and 2) whether personalized content tailored to different learning styles met the users' expectations. Twelve participants stated that the questions were suitable to identify their learning styles (Figure 9(a)). Two participants thought that *some questions* were suitable to identify learning style. This response may arise if users are not tight to one particular learning style. Being a kinesthetic user does not necessarily mean a person prefers only the kinesthetic channel. In fact, many people may have more than one preferred way of learning [41] (discussed in more detail in Chapter 2.4). Figure 9(b) shows that thirteen participants said they liked the content; one respondent stated that content was sometimes favorable. This again may be accounted for by the need to process information in more than one modality in order for effective learning to occur. The respondent who felt that the content was tailored well in accordance with his learning style was kinesthetic. The reason might be rooted to the fact that the participant did not like the sequencing of LOs. The personalized course included supplementary features, such as a live database that could be queried via a web browser interface and show results in real-time. A comment from a kinesthetic learner regarding this feature was positive:

Practicing the commands in SQL was very useful (participant id: 635596).

The participants were also asked if they would like to have more control over the sequence of different types of content (examples, theory, and practice items). Six respondents said that they would like more control, whereas eight said they would not like to have more control over the sequence of learning content. Those participants who would have liked more control had some prior experience with online learning and databases. Users with more experience with technology and especially databases may already know how to use web-based learning resources. They may thus feel confident about selecting and using the preferred learning material.

Discussion
The results of the survey assessing the learning style questionnaire show that participants do not disregard or appear unwilling to complete the questionnaire. Despite this positive view, room for improvement remains. Users with more than one preference channel should be presented with content that is tailored to multiple channels. For example, a user who has both visual and kinesthetic preferences would likely prefer to practice visual tasks. This can be an area for further exploration and is included as a recommendation for further work.

Personalized content

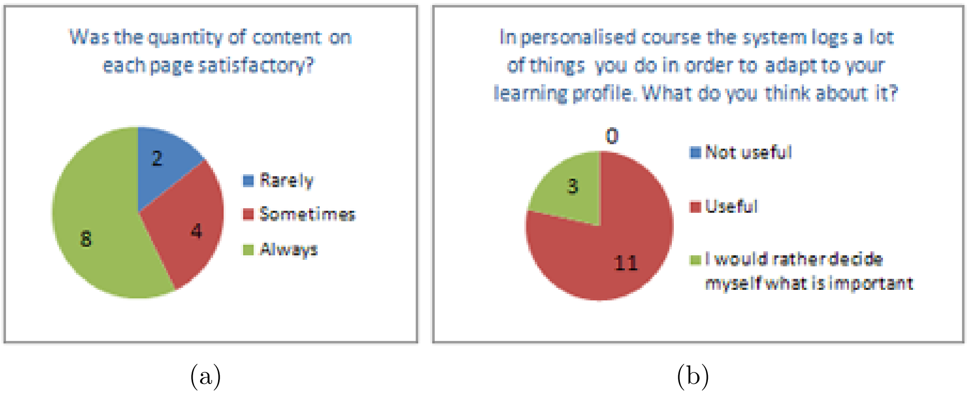
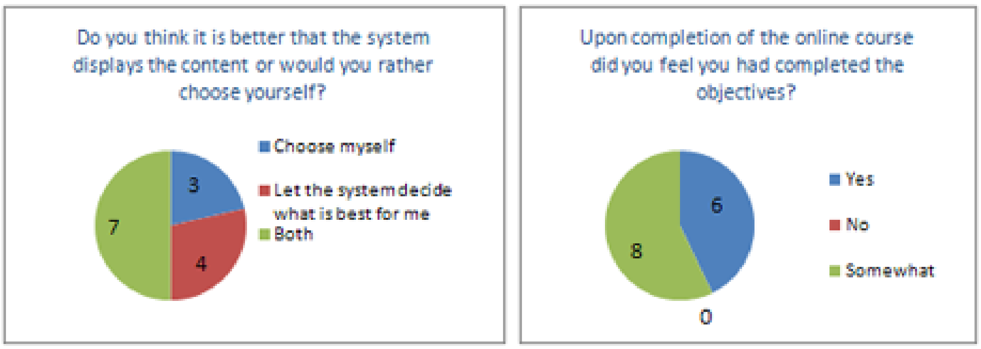


Figure 6.10: Participants' view of personalized content and usage of logs.

The presentation and construction of personalized content were mainly based on the prior knowledge assessment and learning style questionnaire. These two instruments were used primarily to construct a learner profile for participants. As noted in the learning style questionnaire and prior knowledge assessment discussions, participants appeared to feel positive toward the personalized course. Figure 6.10 shows participants' views on the personalized course and the usage of logs for personalization. Two participants (kinesthetic and visual learners) stated that the amount of content was rarely satisfactory, four said it was *sometimes* satisfactory, and the remaining eight stated they were *always* presented with satisfactory content (Figure 10(a)). The fact that kinesthetic learners did not think the quantity of content was satisfactory indicates that the system should have been modeled in a more fine-grained fashion addressing the needs of this type of users. Another issue was the sequencing of objects the kinesthetic participants did not like. Room for improvement remains here as well; one method of trying to solve this problem is to apply more coherent pedagogical principles in cooperation with experts in the field. Those who answered sometimes or rarely had more prior knowledge and experience using online learning and would have preferred to decide what is important (Figure 10(b)).



(a) (b)

Figure 6.11: Learning objectives and control over personalization.

When asked whether the course led to the completion of its objectives, participants were unsure: eight answered *yes* and the remaining six answered *somewhat* (Figure 11(b)). This result is not as satisfactory as one would desire. Experts emphasize that the clarity of course objectives is an important factor in eLearning (see Chapter 4.5). This result indicates that the objectives set before the course might not have been understood or that the course itself was not satisfactory for participants' own objectives.

The course modules had a static sequence, but the actual learning content was adaptively selected as the learner browsed the content. The survey included a question about the adaptive change in the structure of the content. Six participants noted they did not like this feature, but eight said it was a good functionality. This result indicates that not all techniques used for personalization are beneficial. The reason for this most likely lies in the construction of the learner profile and the logic used to continuously adapt. Although, we hoped to achieve a better learning experience by constructing a realistic learner profile [38], it is simply impossible to map prior knowledge with 100% precision for all users. At the end of this section, survey participants were given a choice to provide overall comments on the use of the personalized course. One participant commented:

> *When I clicked on "Back" button, I did not know where I was. I took a pause and when I came back the web page showed me content that I had no clue about* (participant id: 632634).

This means that the adaptivity did not have a favorable result for this participant. A possible remedy for this problem may require a clear navigation information bar available as the learner progresses through the course pages.

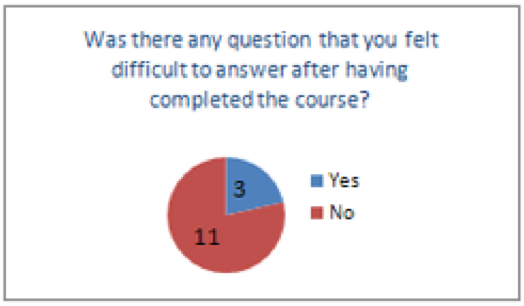
Discussion

The survey results seem to indicate that participants found the personalized course suitable. However, some participants did not have a clear understanding of course objectives. Those users who did not understand course objectives were users with little or no experience with databases. The clarity of the objectives may be attributed to the fact that the course was not authored by a subject expert. The majority of learners would like to both choose the sequence of LOs themselves and let the system decide by adapting to their learning profile. Further work should examine the results of personalization for users by letting the user decide if the sequence of LOs proposed as optimal by the system is appropriate.

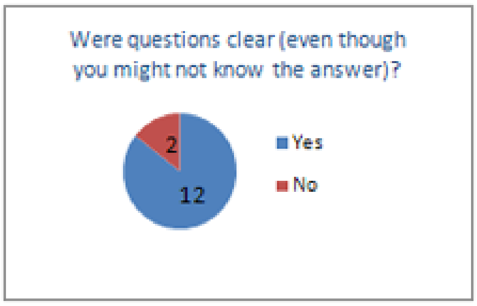
Post assessment

The final part of survey contained questions about post assessment. When asked about the clarity of questions in the post assessment, only two participants stated that the questions were unclear. The same participants also said that they would prefer to have more control over the selection of content. This

[38]By realistic profile it is ment that the level of learner matches all the learning objects that have been presented to a particular learner or content presentation was according to learner's learning style.



(a)



(b)

Figure 6.12: Questions regarding the post assessment.

statement is not surprising, as we have seen from the previous discussion. In addition, there were three participants who felt that it was difficult to answer question(s). A high number of participants felt that the questions were not difficult, indicating that the level of difficulty could be increased. Those participants who made this statement were kinesthetic and auditory users, and had little prior knowledge. Additionally, they obtained low scores in the post-assessment. This statement requires a more appropriate techniques for tailoring content for these learning styles and a more fine-grained description of the course. A final question was a comment regarding the post-assessment.

Discussion

We have seen that not all participants were happy with the personalized course. Some of the respondents noted that the questions in the post-assessment were difficult to answer after completion of the course, and some also noted that questions were unclear. Those users who were less happy appeared to have low scores in the final assessment. These participants did not have much prior knowledge either and had kinesthetic and auditory learning styles. This is explained by the fact that the content may not have been authored to be satisfactory and interesting for these users. Another point is that if a user does not know about the subject at all, this may reduce further interest and motivation. Most of the participants thought that the questions were not difficult. This does not necessarily mean that the personalization technique employed worked by all means. The data set and number of LOs were very limited, and therefore it is not easy to draw constructive conclusions. Obtaining more objective results requires evaluation with larger data sets and more participants. We can say, however, that the technical concept may have worked to some extent.

6.4.4 Limitations

This experiment had several limitations. First, the course could be larger and include more LOs. A total of 102 LOs were included in the course; many LOs were embedded within other objects (for example, an image is LO itself could be in a text document). As we have seen in Chapter 5.9, the number of questions in prior knowledge assessment depends on the depth of a course and how fine-grained the course is. Second, far more users could participate to obtain more close numbers to real situations. Despite this limitation of number of participants, users with different learning styles and prior knowledge participated in the experiment. Finally, the course was not authored by a subject expert.

6.5 Summary

This chapter discussed the evaluation of the prototype PEDAL-NG. Due to the small data set and low number of users, the evaluation did not lead to representative results that mimic those in real world. Despite this fact, we argue that the performance can be evaluate in a way that was demonstrated in performance evaluation chapter.

The number of users is significant issue. Results from a group of fourteen users may not be significant; although we have seen that many users have a positive attitude toward the personalized course, not all of personalization is beneficial.

7 Conclusions

Constructing a dynamic personalized course is not a common task in existing eLearning environments. The goal of this research project was to develop a prototype that addresses personalization, within an existing system, based mainly on prior knowledge and learning style. The Inspera LCMS was used as an underlying platform for implementation. This thesis has discussed the design, architecture and implementation of PEDAL-NG. Two primary parameters for personalization were used to construct the sample course. A prior knowledge assessment and learning style questionnaire were utilized in order to create a learner profile. Subsequent usage logs and results from assessments have been used to continuously adapt and personalize learning experience. The personalized course offering was based on both this learner profile and metadata associated with learning objects. These methods have been implemented in the prototype PEDAL-NG. The evaluation of the prototype was carried out via a sample course, *Designing a Relational Database*. The primary focus of this thesis was on the technical and architectural parts of the system. The main research question was:

Is it possible to design, implement and create an architecture for personalized eLearning on top of an existing system?

The results indicate that it is possible to implement a personalized eLearning architecture on top of an existing LMS; however this greatly depends on the environment. Using Inspera LCMS as an underlying platform dramatically reduced the amount of coding necessary for the development of a functional prototype. These functionalities include the assessment module, functionality for defining a course structure, and built-in metadata support for both learning objects and users. If all of these components had to be implemented, it would have taken far longer to actually implement the prototype. This does not mean that the Inspera LCMS is a brilliant solution. Many other systems may have this kind of support. Evaluation of different systems was out of the scope of this thesis, since the choice of environment was made beforehand. Various implementation scenarios, as well as the use of diverse instruments and technologies may not be suitable for all systems in all contexts.

The integration of (the VAK) learning style questionnaire in Inspera LCMS was useful to determine the learning style of users. However, application of different learning style models will depend on the environment as well as the target group (i.e., students) who will actually be using the system. The choice of the learning style model in this thesis was simple but sufficient to demonstrate the concept of learning style-based personalization by applying this concept in Inspera LMS. In-memory indexing was chosen as a strategy to select learning objects and a generic framework for sequencing learning objects has been proposed (see Chapter 5). This approach has worked well in combination with the incremental caching technique employed in Inspera LCMS.

The evaluation of the system by users has shown that the diversity of the user group makes it difficult to draw specific conclusions regarding these premises. However, it is interesting that users generally have positive feelings regarding the personalized course. This was true despite the facts that 1) some of the course participants felt that they would like to have more control over the sequencing of learning objects and 2) they did not complete the learning objectives. Understanding learning objectives is an important factor and may affect the learning experience.

Although the evaluation did not reveal statistically significant results (which neither was the goal), we can at least suggest that the prototype is worth to further develop. In order to support personalization, eLearning environments should contain an automatic prior knowledge assessment and learning style questionnaire and use the results to construct a learner profile. Discussions and recommendations regarding different aspects of personalization related to this thesis are summarized below.

Prior knowledge

Through the prototype development and user evaluation, we have seen that an assessment offers one way to map the prior knowledge of learners. It is feasible to connect questions in this assessment to modules and identify the learner's prior knowledge within that module. However, this does not suggest that we should disregard the prior knowledge regarding the entire course. Furthermore, these questions may be selected automatically from each module. The sample course consisted of file modules ("Entities", "Table construction ", "Parent and child tables", "Entity Releationship Diagrams", and "SQL"), and automating the selection of questions was beneficial. This choice removed the burden of manual assessment construction.

Learning style

VAK (see chapter 2.4.3) is one of the few inventories of learning styles that can be used for personalization based on learning style in eLearning environments.

Visual learners should be presented with items that have more graphics, auditory learners should be given the opportunity to listen to the material, and kinesthetic learners should be given the option to practice things (i.e., by placing more weight on interactivity). It is not surprising that the evaluation survey also supported this view. The results from the evaluation survey of users experience indicate that the questionnaire used to identify learning style seems to be effective. User evaluation suggests that the sequencing model proposed needs improvement. Kinesthetic participants in the experiment were less satisfied with the sequence of learning objects proposed as optimal by the system.

Learning objects

To enable personalization and adaptivity, both learners and learning objects have been modeled using metadata. Learning objects should be described in detail to enable automated personalization. In the prototype PEDAL-NG, we have looked at level, learning_style as metadata elements. The system must know and be able to retrieve learning objects based on metadata elements that match learner's profile. The employed in-memory indexing proved to be easy and ensures that the system met requirements that had been set before the prototype.

Users

As we have seen from the literature review, user modeling is very popular in Adaptive Hypermedia Systems. To personalize an eLearning course, a clear picture of learner properties must be available to the system. This includes the elements of personalization (such as based on learning style or prior knowledge) to which the system tries to adapt.

Pedagogical aspects

There is a convergence in the research community that pedagogy is important and should be represented in a consistent way. The empowerment of educators to create pedagogically sound personalized courses is therefore of vital importance. The power of PEDAL-NG is being a great tool for educators but its disadvantage might be a lack of guidelines on how to apply coherent pedagogical principles. However, this was not the goal of this thesis and we leave this issue for pedagogical experts to address.

7.1 Limitations

Due to the multidisciplinary nature of eLearning, the pedagogical quality of the sample course was omitted as a variable from this thesis. Rather, the focus was placed on the technical aspects of implementation by discussing the design, architecture, and implementation of a personalized course in existing environment. The evaluation of the group of users in our sample course is not representative, and very few learning objects were used in the evaluation. These topics provide fodder for future work in which subject experts from diverse fields cooperate to obtain statistically significant results. Additionally, they should also evaluate the pedagogical quality of course offerings.

8 Future work

Many areas for further exploration exist. Some of these areas have been roughly noted in various chapters of this thesis. A good starting point for future work involves extending the number of learning objects in the repository and testing them with more users. The in-memory index can be evaluated using Information Retrieval (IR) metrics such as recall, precision, and F-Measure. Below is a short list of topics for future work.

Different learning style inventories

Learners may have more than one preferred learning style mode. In the prototype, learning style was used as an exclusive mode. Future work should ensure that if the learner has both kinesthetic and visual preferences, the system adapts to consider both preferences. Also, other learning style inventories, such as the Honey and Mumford or Kolb's learning style models, provide a subject for further study and implementation. Methods of content presentation and sequencing changes based on different learning style inventories could be an interesting area to explore.

Collaborative rating and user interests

The prototype supports rating functionality and personalization based on student interests. Due to the limited time frame, however, this was not tested in the prototype. Future work should evaluate this functionality by tagging the content, which requires a substrantial number of learning objects. The idea behind this process is that students with similar interests may learn better. Collaboration between students is becoming more prevalent in educational institutions [39] due to the fact that technology is used more frequently in the context of education. Students with interest in soccer may find it more interesting to learn by learning objects that are about soccer. For example, groups of students at NTNU are often formed on a random basis. Basing such groups on similar interests would probably create a more desirable situation for students. Collaborative rating refers to the students' ratings of the *metadata* associated with learning objects. Students who rate a specific set of metadata elements associated with learning objects may find similarly rated learning objects more suitable to their preferences.

[39] At NTNU many subjects allow or even require students to complete exercises in groups.

Output formats

Transforming the personalized course into different formats would be an interesting area of exploration. Mobile devices are becoming ubiquitous, and supporting learning everywhere could reveal interesting results. It is also important to discover new methods for presentation since the screens in which the learning objects are shown are constrained, and thus might generate problems with usability. An alternative output channel is the print format. It would be useful for a user to perform the actions needed for personalization and then receive a print version of the whole course to take with himself for later material review. Challenges connected to this include adaptivity. Since the system continuously adapts based on the user's behavior it would be difficult to support adaptivity in print formats.

Ontologies

The word "ontology", often described as a specification of a conceptualization, seems to generate a lot of controversy in discussions about both Artificial Intelligence and the Semantic Web. This can be an area for future work and further exploration of the relationship between adaptive knowledge-based environments and ontology languages. Ontologies may potentially provide a wealth of information to support the process of adaptation. Ontologies may help to generate an explicit specification of models that are able to exchange information. As was noted in implementation chapter, several learning object repositories can be used in an eLearning environment, and thus may require a common ontology to support interoperability. Challenges regarding this include a choice of appropriate metadata formats for both content description and specification of adaptive funtionalities and delivering a personalized experience.

References

[1] Academic ADL Co-Lab (AADL). What are learning objects? *The University of Wisconsin System Online Resources*, 2001.

[2] Nor Aniza Abdullah and Hugh Davis. Is Simple Sequencing Simple Adaptive Hypermedia? In *Proceedings of the fourteenth ACM conference on Hypertext and hypermedia*, Nottingham, UK, 2003.

[3] Advanced Distributed Learning Initiative. Sharable Content Object Reference Model. *http://www.adlnet.gov/*, 2007.

[4] Michael E. Agnes. *Webster's New World College Dictionary*. Wiley Publishing, Cleveland, OH, USA, 4 edition, 2004.

[5] Apache Software Foundation. Apache JMeter. *http://jakarta.apache.org/jmeter/*, 2008.

[6] Apache Software Foundation. Apache Lucene. *http://lucene.apache.org/*, 2008.

[7] Ariadne Foundation. ARIADNE, Foundation for the European Knowledge Pool. *http://www.ariadne-eu.org/*, 2006.

[8] Inspera AS. What is elearning? *http://www.inspera.no/servlets/dispatcher?languageId=2&siteNodeId=406347*, 2003.

[9] Inspera AS. *http://www.inspera.no*, 2008.

[10] Aviation Industry CBT Committee Computer. The Aviation Industry CBT (Computer-Based Training) Committee Guidelines. *http://www.aicc.org/*, 2001.

[11] Alex Büchner and David Patterson. Personalised e-learning opportunities - call for a pedagogical domain knowledge model. In *15th International Workshop on Database and Expert Systems Applications*, Zaragoza, Spain, 2004.

[12] William Bechtel and George Graham. *A Companion to Cognitive Science*. Blackwell companions to philosophy 14. Blackwell Publishing, Malden, MA, USA, 1998.

[13] Tim Berners-Lee. Metadata architecture. *http://www.w3.org/DesignIssues/Metadata.html*, 2007.

[14] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 283(5), 2001.

[15] Benjamin S. Bloom. *Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain.* Addison Wesley Publishing Company, Readin, MA, USA, 1956.

[16] Lena Boström. Lärande & metod. *Lärstilsanpassad undervisning jämfört med traditionell undervisning i svensk grammatik.*, 2004.

[17] Jesús G. Boticario, Olga Santos, and Peter van Rosmalen. Issues in developing standardbased adaptive learning management systems. *Paper for the EADTU 2005 Working Conference: Towards Lisbon 2010: Collaboration for Innovative Content in Lifelong Open and Flexible Learning*, 2005.

[18] Paul De Bra, Ad Aerts, Bart Berden, Barend de Lange, Brendan Rousseau, Tomi Santic, David Smits, and Natalia Stash. AHA! The Adaptive Hypermedia Architecture. In *HT'03*, Nottingham, United Kingdom, 2003.

[19] Paul De Bra, Peter Brusilovsky, and Geert-Jan Houben. Adaptive Hypermedia: From Systems to Framework. *ACM Computing Surveys*, 31(4), 1999.

[20] Paul De Bra and Licia Calvi. AHA: a Generic Adaptive Hypermedia System. In *2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT'98,*, Pittsburgh, PA, USA, 1998.

[21] Brandon Hall Research Group. LMS Selection Guide: LMS and LCMS Demystified. *ELearning! Magazine.*

[22] Roslin Brennan. *One size doesnt́ fit all.* National Centre for Vocational Education Research Ltd, Leabrook, SA, Australia, 1999.

[23] Catherine Bruen and Owen Conlan. Dynamic Adaptive ICT Support for learning Styles - A Development Framework for re-useable learning resources for different learning styles & requirements. In *Annual Conference of the Association of Information Technology for Teacher Education, ITTE 2002*, 2002.

[24] Peter Brusilovsky. Methods and techniques of adaptive hypermedia. In *User Modeling and User Adapted Interaction*, pages 87–129, 1996.

[25] Peter Brusilovsky. Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 2(2), 2001.

[26] Peter Brusilovsky. Adaptive Hypermedia: From Intelligent Tutoring Systems to Web-Based Education. In *Proceedings of the 5th International Conference, ITS 2000*, Montreal, Canada, 2002.

[27] Peter Brusilovsky and Mark T. Maybury. From Adaptive Hypermedia to the Adaptive Web. *Communications of the ACM*, 45(5), 2002.

[28] businessballs. David kolb's learning styles model and experiential learning theory (elt). *http://www.businessballs.com/kolblearningstyles.htm*, 2007.

[29] John Canavan. Personalized E-Learning Through Learning Style Aware Adaptive Systems. *Trinity College Dublin*, 2004.

[30] Yi-Hsing Chang, Tsung-Yi Lo, and Rong-Jyue Fang. An adaptive e-learning system based on intelligent agents. In *Proceedings of the 6th WSEAS International Conference on Applied Computer Science*, Hangzhou, China, 2007.

[31] Dennis W. Cheek. *Thinking Constructively About Science, Technology, and Society Education*. State Univ of New York Pr, Albany, NY, USA, 1992.

[32] Ruth Clark and Richard E. Mayer. *e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning*. Pfeiffer, San Francisco, CA, 2 edition, 2007.

[33] Sharon Clark and Jon Baggaley. Assistive software for disabled learners. *The International Review of Research in Open and Distance Learning*, 5(3), 2004.

[34] Owen Conlan. *The Multi-Model, Metadata Driven Approach to Personalised eLearning Services*. PhD thesis, Trinity College Dublin, Dublin, Ireland, 2005.

[35] Owen Conlan, Vincent Wade, Catherine Bruen, and Mark Gargan. Multi-model, metadata driven approach to adaptive hypermedia services for personalized elearning. In *2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems*, Malaga, Spain, 2002.

[36] Morris Cooze and Michael Barbour. Learning styles: A focus upon e-learning practices and their implications for successful instructional design. *Journal of Applied Educational Technology*, 4(1), 2007.

[37] Kristin Dahle. *Lærende skoler : en studie av individuelle læringsstiler og organisasjonslæring.* University of Oslo, Norway, 2006.

[38] James M. McIntyre David A. Kolb and Irwin M. Rubin. *Organizational psychology : readings on human behavior in organizations.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1984.

[39] Filip Dochy, Mien Segers, and Miehelle M. Buehl. The Relation Between Assessment Practices and Outcomes of Studies: The Case of Research on Prior Knowledge. *Review of Educational Research*, 69(2), 1999.

[40] Dublin Core Metadata Initiative. Dublin core metadata element set. *http://dublincore.org/documents/dces/*, 2007.

[41] Kenneth Dunn and Rita Stafford Dunn. *Practical Approaches to Individualizing Staff Development for Adults.* Greenwood Publishing Group, United Kingdom, 1998.

[42] Rita Dunn. Understanding the Dunn and Dunn Learning Styles Model and the need for individual diagnosis and prescription. *Reading & Writing Quarterly*, 1990.

[43] Rita Dunn and Shirley Griggs. *Læringsstiler.* Universitetsforlaget, Norway, 2004.

[44] Rita Stafford Dunn. *Nå skjønner jeg det! Finn din læringsstil og lær deg selv å lære!* Kommuneforlaget, Oslo, Norway, 2005.

[45] Rita Stafford Dunn and Kenneth Dunn. *Teaching elementary students through their individual learning styles.* Allyn & Bacon, Boston, 1992.

[46] Rita Stafford Dunn and Shirley Griggs. *Bringing Out the Giftedness in Your Child: Nurturing Every Child's Unique Strengths, Talents, and Potential.* John Wiley & Sons Inc, Hoboken, NJ, USA, 1992.

[47] Rita Stafford Dunn and Shirley Griggs. *Practical approaches to using learning styles in higher education.* Bergin & Garvey, Westport, CT, USA, 2000.

[48] Magdalini Eirinaki and Michalis Vazirgiannis. Web mining for web personalization. *ACM Transactions on Internet Technology (TOIT)*, 3(1), 2003.

[49] John Eklund and Peter Brusilovsky. InterBook: an Adaptive Tutoring System. *UniServe Science News*, 12, 1999.

[50] Richard M. Felder. Matters of style. *ASEE Prism*, 6(4), 1996.

[51] Richard M. Felder and Linda K.Silverman. Learning and teaching styles. *Engineering Education*, 78(7), 1988.

[52] Neil Fleming. VARK. A guide to learning styles. *http://www.vark-learn.com*, 2008.

[53] Karen Frankola. Why online learners drop out. *Workforce Management Research Center*, 2001.

[54] Reva Freedman. What is an intelligent tutoring system? *Intelligence*, 11(3), 2000.

[55] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design patterns : elements of reusable object-oriented software.* Addison-Wesley, Reading, MA, USA, 1995.

[56] Thavamalar Govindasamy. Successful implementation of e-learning pedagogical considerations. *The Internet and Higher Education*, 4(3), 2001.

[57] Sabine Graf and Beate List. An evaluation of open source e-learning platforms stressing adaptation issues. In *Fifth IEEE International Conference on Advanced Learning Technologies, 2005. ICALT 2005.*, 2005.

[58] Nicola Henze, Peter Dolog, and Wolfgang Nejdl. Reasoning and Ontologies for Personalized E-Learning in the Semantic Web. *Educational Technology & Society*, 7(4), 2004.

[59] Wayne Hodgins. The future of learning objects. In *2002 eTEE Conference on e-Technologies in Engineering Education: Learning Outcomes Providing Future Possibilities*, Davos, Switzerland, 2002.

[60] Peter Honey and Alan Mumford. *The manual of Learning Styles.* Peter Honey Publications, 1992.

[61] Jane Hunter. Working towards metautopia - a survey of current metadata research. *http://www.itee.uq.edu.au/ eresearch/papers/2003/LibTrends_paper.pdf*, Last checked, 13.01.2008.

[62] IEEE. *http://www.ieee.org/*, 2008.

[63] IEEE Learning Technology Standard Committee. Learning Object Metadata (LOM) Standard. *http://www.ieeeltsc.org*, 2002.

[64] IMS Global Learning Consortium. IMS Content Packaging Specification, Version 1.1.4 - Final Specification. *http://www.imsglobal.org/content/packaging/*, 2006.

[65] IMS Global Learning Consortium. IMS Learner Information Package Specification, Version 1.0.1 - Final Specification. *http://www.imsglobal.org/profiles/*, 2006.

[66] IMS Global Learning Consortium. Learning Resource Meta-data Specification, Version 1.3 - Final Specification. *http://www.imsglobal.org/metadata/*, 2006.

[67] IMS Global Learning Consortium, Inc. IMS Global Learning Consortium, Inc. *http://www.imsglobal.org/*, 2003.

[68] IMS Global Learning Consortium, Inc. IMS Simple Sequencing Specification. *http://www.imsglobal.org/simplesequencing/*, 2003.

[69] National Learning Infrastructure Initiative. Learning objects. *EDUCAUSE Connect*, 1998.

[70] Intelera. Training solutions for software. glossary. *http://www.intelera.com/glossary.htm*, March 2008.

[71] Johan Ismail. The design of an e-learning system beyond the hype. *The Internet and Higher Education*, 4(3), 2001.

[72] Rick Kazman, Gregory Abowd, Len Bass, and Paul Clements. Scenario-Based Analysis of Software Architecture. *http://www.sei.cmu.edu/architecture/scenario_paper/*, Last checked, 12.12.2007.

[73] K.C. Jones. Worldwide E-Learning To Draw Nearly $53 Billion By 2010. *InformationWeek*, 2007.

[74] Timothy Koschmann. *CSCL:theory and practice of an emerging paradigm.* Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1996.

[75] Kyung-Sun Kim and Joi L. Moore. Web-based learning:Factors affecting students' satisfaction and learning experience. *PEER-REVIEWED JOURNAL ON THE INTERNET*, 10(11), 2005.

[76] Carl Lagoze. Keeping dublin core simple. *D-Lib Magazine*, 7(1), 2001.

[77] LTSC. Ieee learning technology standards committee. *http://ieeeltsc.org/*, 2008.

[78] Knowles M. *The modern practice of adult education: from pedagogy to andragogy.* Cambridge Adult Education, Cambridge, 1980.

[79] Larry McNutt and Marie Brennan. Work in progress - learning styles and elearning,what is the connection? *35th ASEE/IEEE Frontiers in Education Conference*, 2001.

[80] Glenn Millar. Learning objects: A primer for neophytes. *BCIT Learning and Teaching Centre*, 2003.

[81] Mountain Quest Institute. Glossary, definitions. *http://www.mountainquestinstitute.com/definitions.htm*, March 2008.

[82] Maurice D. Mulvenna, Sarabjot S. Anand, and Alex G. Büchner. Personalization on the net using web mining: introduction. *Communications of the ACM*, 43(8), 2000.

[83] John Murray. Pedagogical implications of cognitive science research. *Independent Teacher. The eJournal for Independent School Educators*, 2006.

[84] Isabel Briggs Myers and Mary H. McCaulley. *A Guide to the Development and Use of the Myers-Briggs Type Indicator.* Consulting Psychologists Press, Palo Alto, CA, USA, 1985.

[85] Filip Neven and Erik Duval. Reusable learning objects: a survey of lom-based repositories. In *Proceedings of the tenth ACM international conference on Multimedia*, Juan-les-Pins, France, 2002.

[86] National Information Standards Organization. Understanding metadata. *NISO Press, National Information Standards Organization*, 2004.

[87] Judith Poole. E-learning and learning styles: students' reactions to web-based language and style at blackpool and the fylde college. *Language and Literature*, 15(3), 2006.

[88] Naomi L. Quenk. *Essentials of Myers-Briggs Type Indicator Assessment*. John Wiley & Sons, Inc, New York, 2000.

[89] Raoul A. Arreola. Writing Learning Objectives. *The University of Tennessee, Memphis*, 1998.

[90] Patricia Murphy Raupers. Research on perceptual strengths: I see what you mean; i hear what you say; are you staying in touch? are you moving my way? part 1: Perceptual strengths of adults. i: Dunn, rita & griggs, shirley a. *Synthesis of the Dunn and Dunn Learning Style Model Research: Who, What, When, Where and So What?*, 2004.

[91] Maria Roussos. Issues in the Design and Evaluation of a Virtual Reality Learning Environment. Master's thesis, Graduate College, University of Illinois at Chicago, 1997.

[92] Mohammad Issack Santally and Alain Senteni. A learning object approach to personalized web-based instruction. *European Journal of Open, Distance and E-Learning*, 2005.

[93] Ljiljana Stojanovic, Steffen Staab, and Rudi Studer. elearning based on the semanticweb. In *WebNet2001 - World Conference on the WWW and Internet*, Orlando, FL, USA, 2001.

[94] The Open Archives Initiative. Protocol for metadata harvesting. *http://www.openarchives.org/*.

[95] Reeves C. Thomas. Answering critics of media and technology in education. In *ED-MEDIA/ED-TELECOM 98 World Conference on Educational Multimedia and Hypermedia & World Conference on Educational Telecommunications*, Freiburg, Germany, 1998.

[96] Catherine Tuckey. *Uses of New Technology in Higher Education - Guiding Principles*. Institute for Computer Based Learning, Heriot-Watt University, Edinburgh, UK, 2 edition, 1992.

[97] Keith Tyler-Smith. Early attrition among first time elearners: A review of factors that contribute to drop-out, withdrawal and non-completion rates of adult learners undertaking elearning programmes. *Journal of Online Learning and Teaching*, 2(2), 2006.

[98] Shirley Waterhouse. *The Power of eLearning: The Essential Guide for Teaching in the Digital Age.* Allyn & Bacon, 2005.

[99] Chris Watkins. *Learning about Learning : Resources for Supporting Effective Learning.* Routledge, Florence, KY, USA, 2 edition, 2000.

[100] David A. Wiley. *Instructional Use of Learning Objects.* Agency for Instructional Technology, Bloomington, IN, USA, 2002.

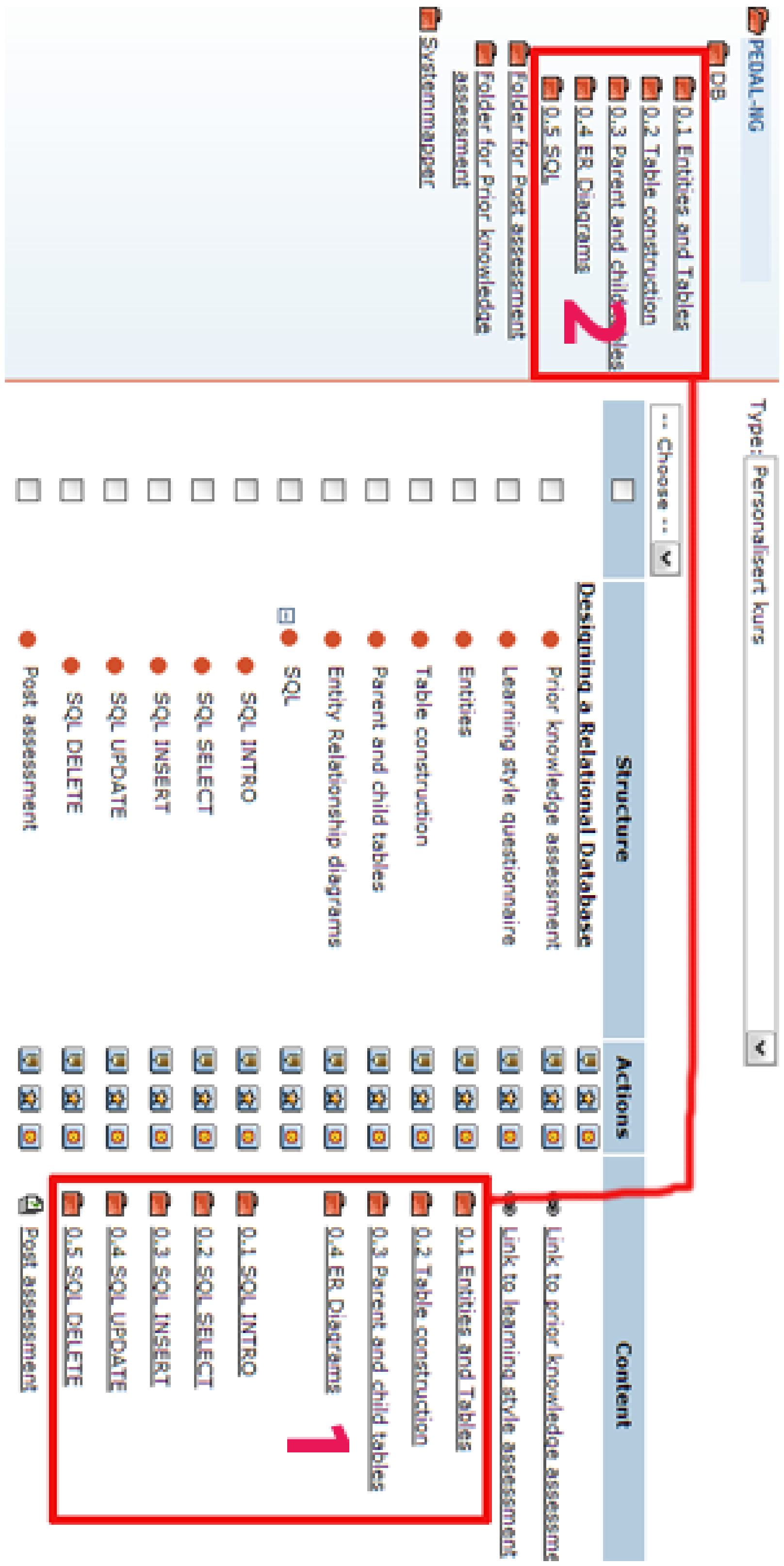
[101] Sonja Wilhelmsen, Stein Inge Åsmul, and Øyvind Meistad. Psychological theories; a brief survey of the changing views of learning. *Publication by Dept. of Information Science, University of Bergen, Norway*, 1998.

[102] Christian Wolf. *Construction of an Adaptive E-learning Environment to Address Learning Styles and an Investigation of the Effect of Media Choice.* PhD thesis, School of Education, RMIT University, 2007.

[103] Michael Wooldridge. *An Introduction to MultiAgent Systems.* John Wiley & Sons, Hoboken, NJ, USA, 2002.

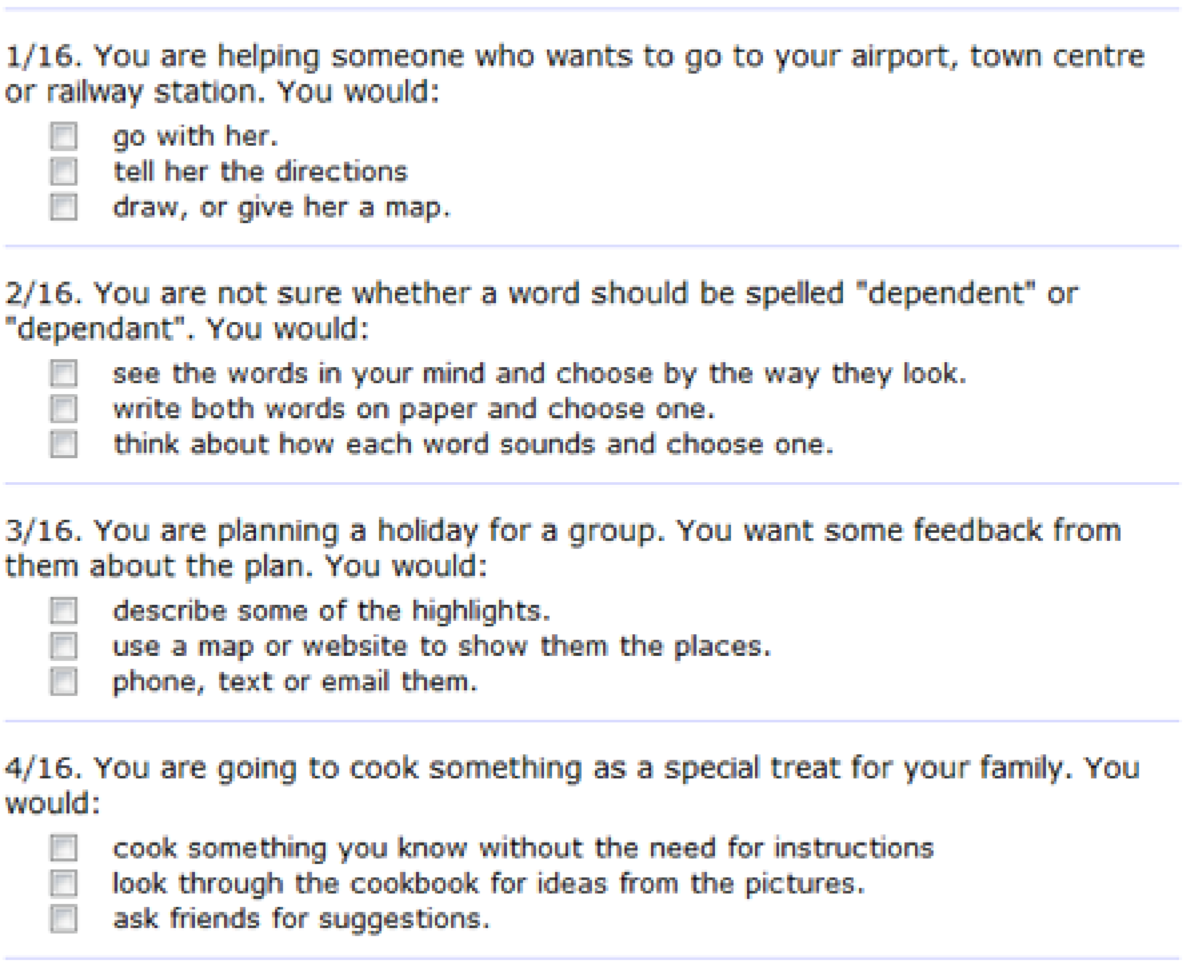
[104] ZDNet Australia. Don't take it personal. *Australian Technology & Business magazine*, 2003.

A Authoring tool



	Structure	Actions	Content
	Designing a Relational Database		
	Prior knowledge assessment		Link to prior knowledge assessme
	Learning style questionnaire		Link to learning style assessment
	Entities		0.1 Entities and Tables
	Table construction		0.2 Table construction
	Parent and child tables		0.3 Parent and child tables
	Entity Relationship diagrams		0.4 ER Diagrams
	SQL		
	SQL INTRO		0.1 SQL INTRO
	SQL SELECT		0.2 SQL SELECT
	SQL INSERT		0.3 SQL INSERT
	SQL UPDATE		0.4 SQL UPDATE
	SQL DELETE		0.5 SQL DELETE
	Post assessment		Post assessment

B Learning Style Questionnaire



1/16. You are helping someone who wants to go to your airport, town centre or railway station. You would:

- ☐ go with her.
- ☐ tell her the directions
- ☐ draw, or give her a map.

2/16. You are not sure whether a word should be spelled "dependent" or "dependant". You would:

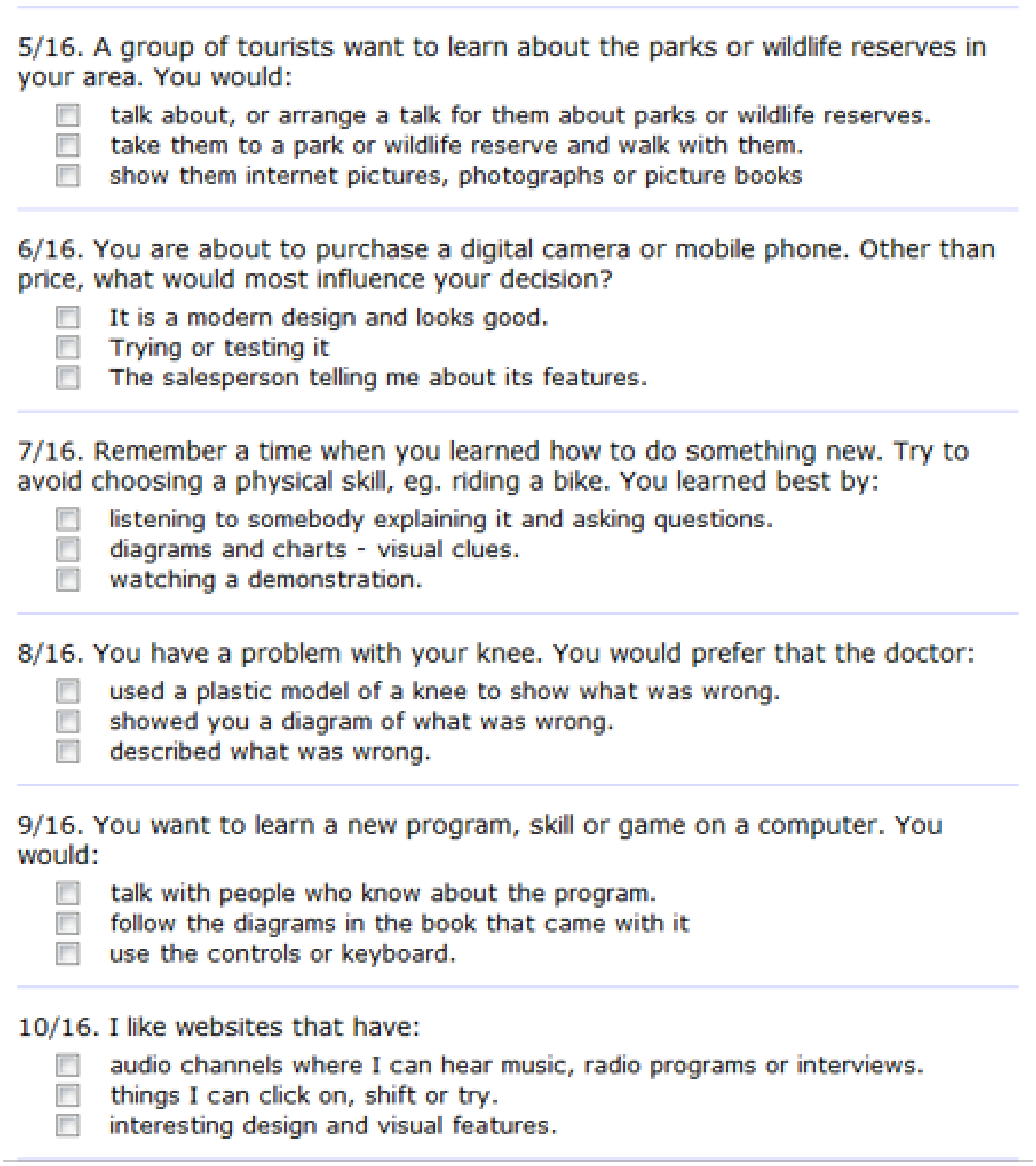
- ☐ see the words in your mind and choose by the way they look.
- ☐ write both words on paper and choose one.
- ☐ think about how each word sounds and choose one.

3/16. You are planning a holiday for a group. You want some feedback from them about the plan. You would:

- ☐ describe some of the highlights.
- ☐ use a map or website to show them the places.
- ☐ phone, text or email them.

4/16. You are going to cook something as a special treat for your family. You would:

- ☐ cook something you know without the need for instructions
- ☐ look through the cookbook for ideas from the pictures.
- ☐ ask friends for suggestions.



5/16. A group of tourists want to learn about the parks or wildlife reserves in your area. You would:

- ☐ talk about, or arrange a talk for them about parks or wildlife reserves.
- ☐ take them to a park or wildlife reserve and walk with them.
- ☐ show them internet pictures, photographs or picture books

6/16. You are about to purchase a digital camera or mobile phone. Other than price, what would most influence your decision?

- ☐ It is a modern design and looks good.
- ☐ Trying or testing it
- ☐ The salesperson telling me about its features.

7/16. Remember a time when you learned how to do something new. Try to avoid choosing a physical skill, eg. riding a bike. You learned best by:

- ☐ listening to somebody explaining it and asking questions.
- ☐ diagrams and charts - visual clues.
- ☐ watching a demonstration.

8/16. You have a problem with your knee. You would prefer that the doctor:

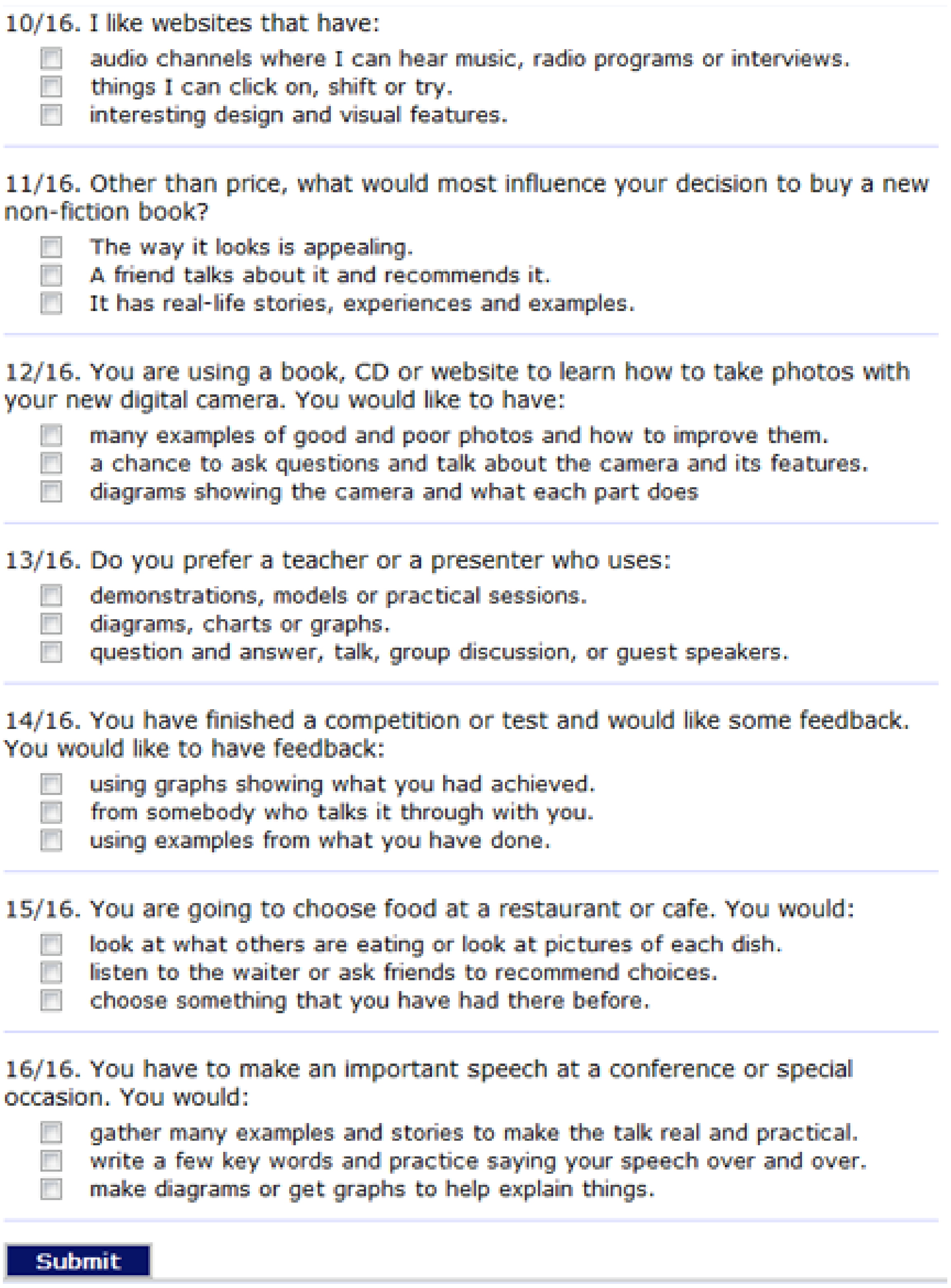
- ☐ used a plastic model of a knee to show what was wrong.
- ☐ showed you a diagram of what was wrong.
- ☐ described what was wrong.

9/16. You want to learn a new program, skill or game on a computer. You would:

- ☐ talk with people who know about the program.
- ☐ follow the diagrams in the book that came with it
- ☐ use the controls or keyboard.

10/16. I like websites that have:

- ☐ audio channels where I can hear music, radio programs or interviews.
- ☐ things I can click on, shift or try.
- ☐ interesting design and visual features.



10/16. I like websites that have:

- ☐ audio channels where I can hear music, radio programs or interviews.
- ☐ things I can click on, shift or try.
- ☐ interesting design and visual features.

11/16. Other than price, what would most influence your decision to buy a new non-fiction book?

- ☐ The way it looks is appealing.
- ☐ A friend talks about it and recommends it.
- ☐ It has real-life stories, experiences and examples.

12/16. You are using a book, CD or website to learn how to take photos with your new digital camera. You would like to have:

- ☐ many examples of good and poor photos and how to improve them.
- ☐ a chance to ask questions and talk about the camera and its features.
- ☐ diagrams showing the camera and what each part does

13/16. Do you prefer a teacher or a presenter who uses:

- ☐ demonstrations, models or practical sessions.
- ☐ diagrams, charts or graphs.
- ☐ question and answer, talk, group discussion, or guest speakers.

14/16. You have finished a competition or test and would like some feedback. You would like to have feedback:

- ☐ using graphs showing what you had achieved.
- ☐ from somebody who talks it through with you.
- ☐ using examples from what you have done.

15/16. You are going to choose food at a restaurant or cafe. You would:

- ☐ look at what others are eating or look at pictures of each dish.
- ☐ listen to the waiter or ask friends to recommend choices.
- ☐ choose something that you have had there before.

16/16. You have to make an important speech at a conference or special occasion. You would:

- ☐ gather many examples and stories to make the talk real and practical.
- ☐ write a few key words and practice saying your speech over and over.
- ☐ make diagrams or get graphs to help explain things.

C Permission to use course material

Permission to use a portion of course on designing a relational database by Ruth C. Clark and Richard E. Mayer is granted by the publisher Wiley. A CD is shipped with their award winning book on eLearning *e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning* [32]. This CD contains materials on sample course that uses authors' guidelines for designing a multimedia learning.

See the next page for the permission received from the publisher.

<republication@wiley.com>
03/23/2008 06:45 PM
Please respond to <republication@wiley.com>

To <republication@wiley.com>
cc
bcc
Subject pfeiffer.com Republication/Electronic Permission Request

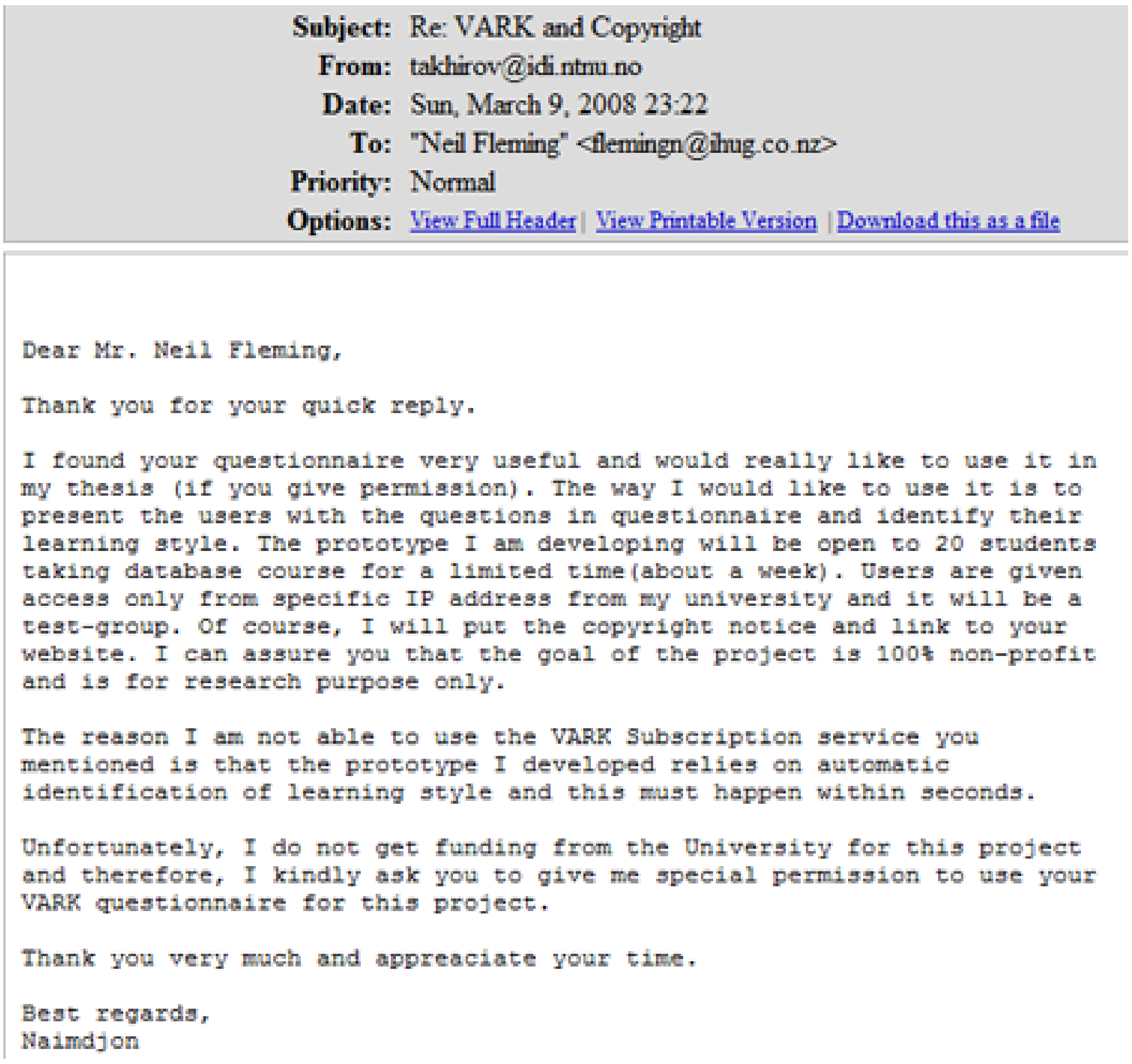
```
FIRST_NAME: Naimdjon
mailfieldfromName: Takhirov
COMPANY_NAME: Student
ADDRESS: Sognsveien 102 E, H0201
CITY: Oslo
STATE: Oslo
ZIP: 0857
COUNTRY: Norway
PHONE: +47 98490659
FAX:
mailfieldfromAddress: takhirov@idi.ntnu.no
REFERENCE:
BOOK_TITLE: CD from ISBN-13: 978-0787986834
BOOK_OR_JOURNAL: Book
BOOK_AUTHOR: Ruth Clark
ISBN: 978-0787986834
JOURNAL_MONTH:
JOURNAL_YEAR:
JOURNAL_VOLUME:
JOURNAL_ISSUE_NUMBER:
COPY_PAGES: Some of the texts from the sample course on CD - for research
purpose only.
MAXIUM_COPIES:
YOUR_PUBLISHER: Myself, electronic version
YOUR_TITLE: Currently completing thesis on eLearning and this is part of my
research work
PUBLICATION_DATE: will be available for 10 users in a period of two weeks.
FORMAT: Software Program
PRINT_RUN_SIZE:
EBOOK_READER_TYPE:
IF_WWW_URL: Internal address at the university. Trafic from outside of local
network will not be accepted.
IF_WWW_ADOPTED_FROM_BOOK: No
IF_WWW_PASSWORD_ACCESS: Yes
WWW_PROFESSOR_NAME: Naimdjon Takhirov
WWW_COURSE_NAME_NUMBER: No specific course
WWW_USERS: 10
IF_WWW_MATERIAL_POSTED_FROM: 10.04.2008
IF_WWW_MATERIAL_POSTED_TO:
IF_INTRANET_URL: http://stud3097.idi.ntnu.no/pedal
IF_INTRANET_FROM_ADOPTED_BOOK: No
IF_INTRANET_PASSWORD_ACCESS: Yes
INTRANET_PROFESSOR_NAME: Naimdjon Takhirov
INTRANET_COURSE_NAME_NUMBER: No specific course
INTRANET_USERS: 10
IF_INTRANET_MATERIAL_POSTED_FROM: 10/04/2008
IF_INTRANET_MATERIAL_POSTED_TO: 17/04/2008
IF_SOFTWARE_PRINT_RUN: self developed software program.
COMMENTS_FOR_REQUEST: I would like to use some of the texts and images from
the CD that was attached to the book I bought. This material will be used for
research purposes only and there is no profit from the project. The material
is to be accessible only to 10 users for a period of one week and from
```

PERMISSION GRANTED
BY: [illegible] 03/24/08.
Global Rights Dept., John Wiley & Sons, Inc.

NOTE: No rights are granted to use content that appears in the work with credit to another source specific network addresses.

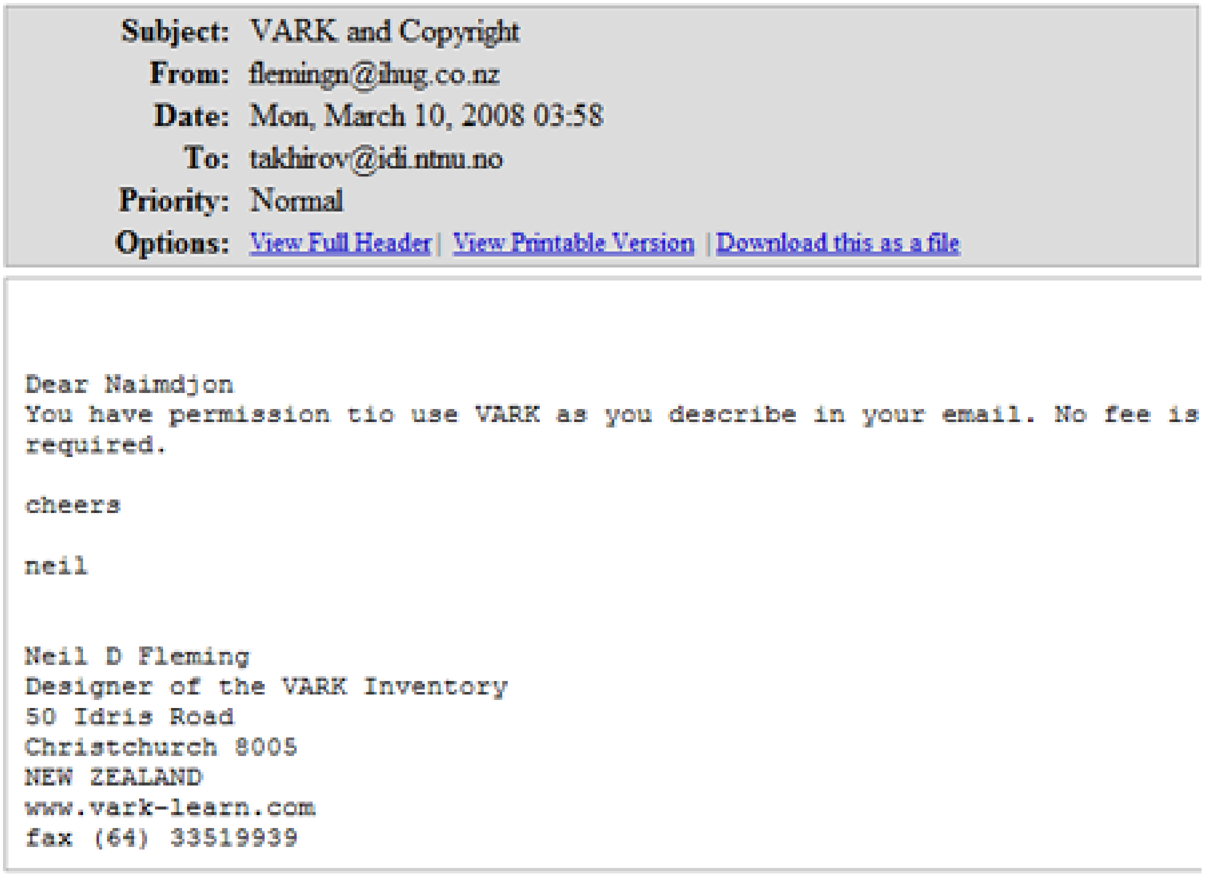
I would only like to copy the texts as it is explained in an easy to
understand language.
Thank you.

D Permission to use Learning Style Questionnaire VAK



Subject: Re: VARK and Copyright
From: takhirov@idi.ntnu.no
Date: Sun, March 9, 2008 23:22
To: "Neil Fleming" <flemingn@ihug.co.nz>
Priority: Normal
Options: View Full Header | View Printable Version | Download this as a file

```
Dear Mr. Neil Fleming,

Thank you for your quick reply.

I found your questionnaire very useful and would really like to use it in
my thesis (if you give permission). The way I would like to use it is to
present the users with the questions in questionnaire and identify their
learning style. The prototype I am developing will be open to 20 students
taking database course for a limited time(about a week). Users are given
access only from specific IP address from my university and it will be a
test-group. Of course, I will put the copyright notice and link to your
website. I can assure you that the goal of the project is 100% non-profit
and is for research purpose only.

The reason I am not able to use the VARK Subscription service you
mentioned is that the prototype I developed relies on automatic
identification of learning style and this must happen within seconds.

Unfortunately, I do not get funding from the University for this project
and therefore, I kindly ask you to give me special permission to use your
VARK questionnaire for this project.

Thank you very much and appreaciate your time.

Best regards,
Naimdjon
```



Subject: VARK and Copyright
From: flemingn@ihug.co.nz
Date: Mon, March 10, 2008 03:58
To: takhirov@idi.ntnu.no
Priority: Normal
Options: View Full Header | View Printable Version | Download this as a file

```
Dear Naimdjon
You have permission tio use VARK as you describe in your email. No fee is
required.

cheers

neil


Neil D Fleming
Designer of the VARK Inventory
50 Idris Road
Christchurch 8005
NEW ZEALAND
www.vark-learn.com
fax (64) 33519939
```

E Survey email to participants

Invitation email to participate in the experiment

Thank you for your interest in participating in the experiment on personalized course on *Designing Relational Databases*. This is part of my master thesis at NTNU. Through participating in this course you will be able to learn about Relational Databases. Do not worry, this course is not too technical and you are not *required* to have experience in programming. After you finish the course, approximately 8 days later you will receive a link to a survey where you can express your opinion about the course. You will receive a separate email on this.

Structure of the course
The course is fully web based. So you do not need anything but a computer with Internet connection.

1. Register for the course. Click on "New user" link. After you register, you can either login or come back later and use your username and password to login.

2. Prior knowledge assessment. In order to personalize course, you will need to answer some simple questions about databases and related concepts before actually starting the course. If you do not know the answer to the questions simply choose "Don't know / No answer" option. This is only needed to identify how much you know about the databases from before, so when you start the actual course you are not presented with the material that can be too simple or boring for you.

3. Learning style questionnaire. People learn in different ways. The second step in the course is to answer questions in a questionnaire consisting of sixteen questions in total. This questionnaire is about finding out your learning preference. Please, choose the option that best suits you or you recognize yourself in particular situation.

4. The actual course content. The web page has been created with simply designed. All you need to do is to click on the "Next" button located on the top of the page. Alternatively, you can navigate directly to a module of particular interest or if you want to repeat the concept. The course is interactive and you will also get a chance to practice different things.

5. Post assessment. After you complete the course there will be final assessment that will assess your knowledge about designing relational databases.

Your personal data will be treated confidentially and no course results will be made public. Please, be honest and treat the course seriously as the results are very important for evaluation. Should you have any doubts or assistance, you can always drop me an email at takhirov@idi.ntnu.no or you can reach me at 98 49 06 59.

Here is the link to the course web page: http://demo.inspera.no/

Finally, I hope you will enjoy the course and thank you once again for you interest.

Good luck!

Regards, Naimdjon

Final email about participation in the evaluation survey

First, I would like to thank you for you interest and participation in the experiment. This project is part of a master thesis at NTNU and your contribution is greatly appreciated.

We have managed to gather enough users to participate in this course. Now that all users have finished their sessions we conduct a survey on what users think about the personalized course on *Designing Relational Databases.*

Here is a link to the survey: `http://www.surveymonkey.com/s.aspx?sm=Tf_2bn5ok_2bJzF4RghEfcZzJg_3d_3d`

It is very important to me and I kindly ask you to complete it ASAP and before monday(28.april 2008).

As a sign of appreciation, I will randomly choose two participants and will give each two movie tickets to any movie.

Have a great day and hope everybody responds to the survey.

With best regards,

Naimdjon Takhirov

F Survey results

Course survey (Designing Relational Databases)

Participant

Q1. Age:

Answer Options	Response Count
25	2
24	2
42	1
29	2
31	1
32	1
34	1
27	2
23	1
26	1
answered question	14
skipped question	0

Participant

Q2. Sex

Answer Options	Response Percent	Response Count
Male	42,9%	6
Female	57,1%	8
	answered question	14
	skipped question	0

Participant

Q3. Email(the same as you used for course)

Answer Options	Response Count
	14
answered question	14
skipped question	0

General

Q1. How much experience do you have with online learning?

Answer Options	Response Percent	Response Count
Little	42,9%	6
Some	35,7%	5
A lot	21,4%	3
	answered question	14
	skipped question	0

General

Q2. Did you have any experience with databases prior participating in this experiment?

Answer Options	Response Percent	Response Count

None	14,3%	2
Little	35,7%	5
Some	42,9%	6
A lot	7,1%	1
	answered question	**14**
	skipped question	**0**

General

Q3. How useful is the web-based course as a support component to traditional classroom learning in your mind?

Answer Options	**Response Percent**	**Response Count**
Not useful at all	0,0%	0
Somewhat useful	35,7%	5
Very useful	64,3%	9
	answered question	**14**
	skipped question	**0**

General

Q4. How useful is the web-based course as a stand-alone learning component in your mind?

Answer Options	**Response Percent**	**Response Count**
Not useful at all	0,0%	0
Somewhat useful	35,7%	5
Very useful	64,3%	9
	answered question	**14**
	skipped question	**0**

General

Q5. How much time did you spend for the whole course (in minutes)?

Answer Options	**Response Count**
300	1
20	1
50	1
40	3
90	1
15	1
35	4
60	2
answered question	**14**
skipped question	**0**

General

Q6. Were the objectives of the course clear to you?

Answer Options	**Response Percent**	**Response Count**
Unclear	0,0%	0
Clear	78,6%	11

Very clear	21,4%	3
	answered question	**14**
	skipped question	**0**

Prior knowledge assessment

Q1. How do you prefer the system gathers information about your prior knowledge?

Answer Options	**Response Percent**	**Response Count**
I would like to give information about my knowledge myself (choose from a list of available choices - e.g. little, intermediate, much)	0,0%	0
I would like the system ask questions about each module and measure my knowledge like it was done in this course	64,3%	9
A mix of these two	35,7%	5
	Comments:	0
	answered question	**14**
	skipped question	**0**

Prior knowledge assessment

Q2. In the course pages you were presented with different materials. Do you feel these materials were worthless(i.e. information was not useful or you knew what was presented)?

Answer Options	**Response Percent**	**Response Count**
Never	50,0%	7
Rarely	35,7%	5
Usually	14,3%	2
Always	0,0%	0
	answered question	**14**
	skipped question	**0**

Prior knowledge assessment

Q3. Do you feel the material you were presented was difficult to follow?

Answer Options	**Response Percent**	**Response Count**
Never	57,1%	8
Rarely	35,7%	5
Usually	7,1%	1
Always	0,0%	0
	answered question	**14**
	skipped question	**0**

Prior knowledge assessment

Q4. In general, how much time (in % of total time you plan to spend on the course) would you like to spend on prior knowledge assessment in future courses?

Answer Options	**Response Count**
	14
answered question	**14**

skipped question	**0**

Prior knowledge assessment

Q5. Would you like the use of a prior knowledge assessment for the next course you ever take?

Answer Options	Response Percent	Response Count
Yes	100,0%	10
No	0,0%	0
	answered question	**10**
	skipped question	**4**

Prior knowledge assessment

Q6. Any comments regarding prior knowledge assessment?

Answer Options	Response Count
This was useful, but as far as I am attending the course it means that I do not know a subject, or would like to learn. So I think that my prior knowledge is not important.	1
answered question	**1**
skipped question	**13**

Learning Style Questionnaire

Q1. Do you feel questions in Learning Style Questionnaire are suitable to identify learning style?

Answer Options	Response Percent	Response Count
Yes	85,7%	12
No	0,0%	0
Some questions	14,3%	2
	answered question	**14**
	skipped question	**0**

Learning Style Questionnaire

Q2. In general, how much time (in %, of total time you plan to spend on the course) would you like to spend on learning style questionnaire?

Answer Options	Response Count
	14
answered question	**14**
skipped question	**0**

Learning Style Questionnaire

Q3. Did the materials presented by the system reflect your learning style? Did you like the way content was presented to you?

Answer Options	Response Percent	Response Count
Yes	92,9%	13
No	0,0%	0
Sometimes	7,1%	1

Comments	
Practicing the comands in SQL was very useful.	1
answered question	**14**
skipped question	**0**

Learning Style Questionnaire

Q4. Based on you learning style, different types of content(examples, theory and practice) were presented in specific sequence. Would you like to have more control of the sequence?

Answer Options	Response Percent	Response Count
Yes	42,9%	6
No	57,1%	8
	Comments:	0
	answered question	**14**
	skipped question	**0**

Personalised Course

Q1. Was the quantity of content on each page satisfactory?

Answer Options	Response Percent	Response Count
Rarely	14,3%	2
Sometimes	28,6%	4
Always	57,1%	8
	answered question	**14**
	skipped question	**0**

Personalised Course

Q2. Would you have liked more control on the material included in the course?

Answer Options	Response Percent	Response Count
Yes	42,9%	6
No	21,4%	3
Sometimes	35,7%	5
	answered question	**14**
	skipped question	**0**

Personalised Course

Q3. Please rate the following modules of the course(1-5; 1 = least interesting, 5 = most

Answer Options	1	2	3	4	5	Rating Average	Response Count
Entities	0	0	2	4	4	4,2	10
Table Construction	0	0	2	2	6	4,4	10
Parent and Child tables	0	0	3	2	5	4,2	10
Entity Relationship modeling	0	0	2	5	3	4,1	10
SQL Intro	0	0	4	2	4	4	10
SQL SELECT	0	0	2	3	5	4,3	10
SQL INSERT	0	0	3	3	4	4,1	10
SQL UPDATE	0	0	2	4	4	4,2	10
SQL DELETE	0	0	3	3	4	4,1	10

answered question	10
skipped question	4

Personalised Course

Q4. In personalised course the system logs a lot of things you do in order to adapt to your learning profile. What do you think about it?

Answer Options	Response Percent	Response Count
Not useful	0,0%	0
Useful	78,6%	11
I would rather decide myself what is important	21,4%	3
answered question		14
skipped question		0

Personalised Course

Q5. Do you think it is better that the system displays the content or would you rather choose yourself?

Answer Options	Response Percent	Response Count
Choose myself	21,4%	3
Let the system decide what is best for me	28,6%	4
Both	50,0%	7
answered question		14
skipped question		0

Personalised Course

Q6. Upon completion of the online course did you feel you had completed the objectives?

Answer Options	Response Percent	Response Count
Yes	42,9%	6
No	0,0%	0
Somewhat	57,1%	8
answered question		14
skipped question		0

Personalised Course

Q7. Would you have liked a greater level of control as to how the content was structured?

Answer Options	Response Percent	Response Count
Yes	42,9%	6
No	57,1%	8
answered question		14
skipped question		0

Personalised Course

Q8. Were there anything that made the use of course pages difficult?

Answer Options	Response Count
No	2
Somewhat	1
When I clicked on "Back" button, I did not know where I was. I took a pause and when I came back the web page showed me content that I had no clue about.	1
answered question	**4**
skipped question	**10**

Personalised Course

Q9. Is there any functionality you were missing in the course?

Answer Options	Response Count
It is difficult to say now, but I think no.	1
No	1
answered question	**2**
skipped question	**12**

Post assessment

Q1. Was there any question that you felt difficult to answer after having completed the course?

Answer Options	Response Percent	Response Count
Yes	21,4%	3
No	78,6%	11
	answered question	**14**
	skipped question	**0**

Post assessment

Q2. Were questions clear (even though you might not know the answer)?

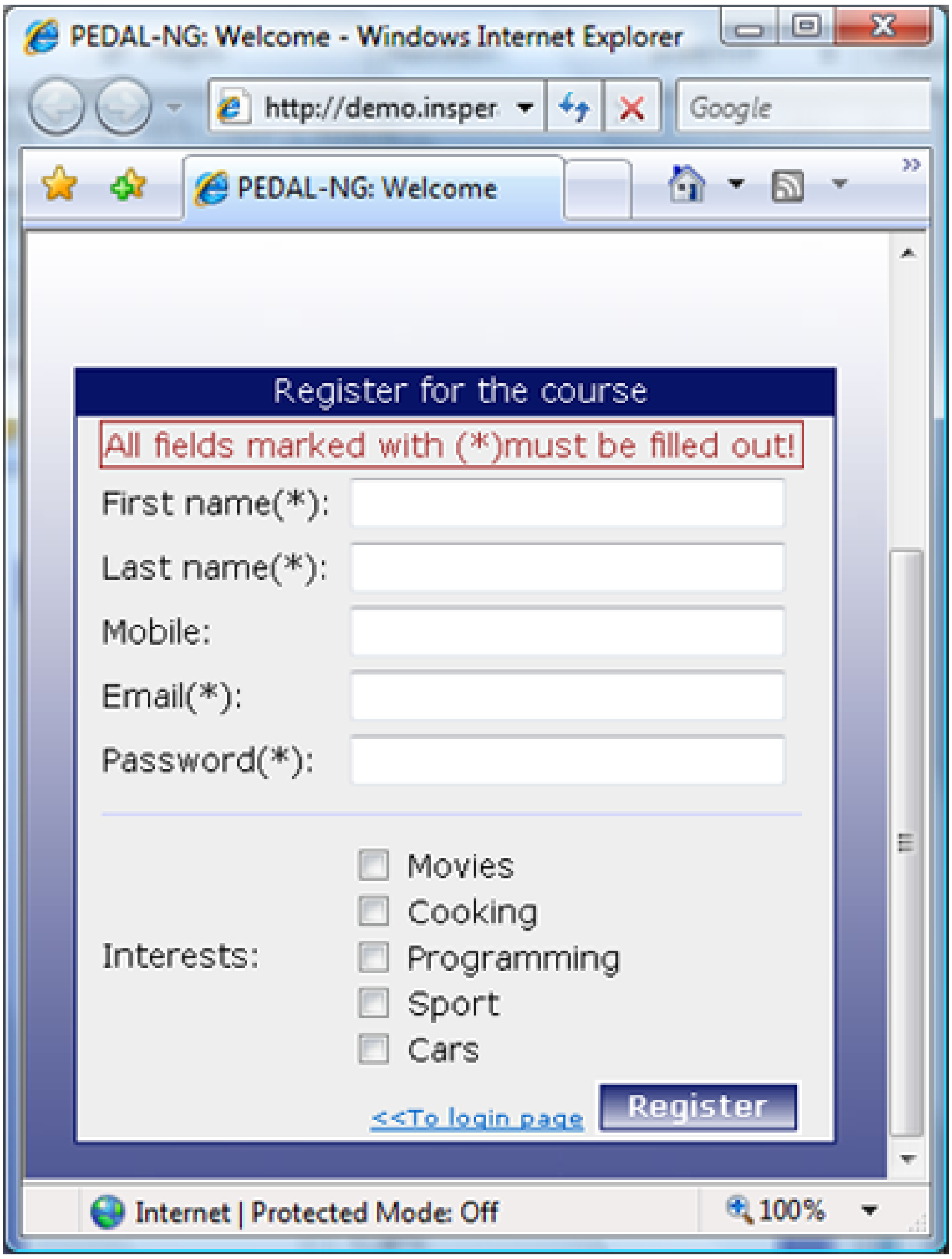
Answer Options	Response Percent	Response Count
Yes	85,7%	12
No	14,3%	2
	answered question	**14**
	skipped question	**0**

Post assessment

Q3. Any other comments?

Answer Options	Response Count
perfect!	1
Some sentences/text on the guidence through web-based system was a bit difficult for me to understand (not the questions on the course itself).	1
answered question	**2**
skipped question	**12**

G Screenshots of a PEDAL-NG-enabled course


PEDAL-NG: Welcome - Windows Internet Explorer
http://demo.insper
Google
PEDAL-NG: Welcome
Internet | Protected Mode: Off
100%

Register for the course

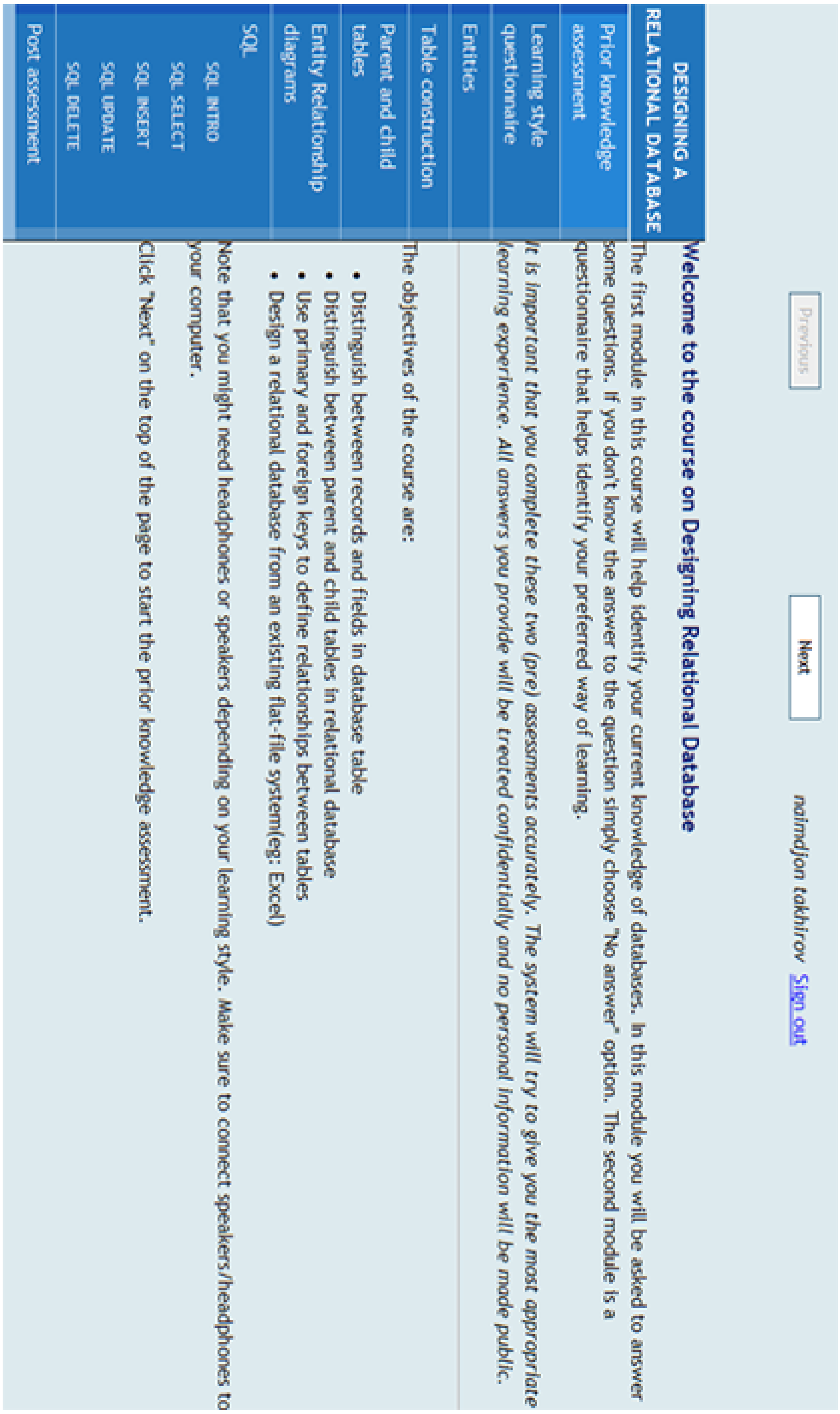
All fields marked with (*)must be filled out!

First name(*):	
Last name(*):	
Mobile:	
Email(*):	
Password(*):	

Interests:
- ☐ Movies
- ☐ Cooking
- ☐ Programming
- ☐ Sport
- ☐ Cars

<<To login page Register

Figure 7.1: Registration page



Welcome to the course on Designing Relational Database

The first module in this course will help identify your current knowledge of databases. In this module you will be asked to answer some questions. If you don't know the answer to the question simply choose "No answer" option. The second module is a questionnaire that helps identify your preferred way of learning.

It is important that you complete these two (pre) assessments accurately. The system will try to give you the most appropriate learning experience. All answers you provide will be treated confidentially and no personal information will be made public.

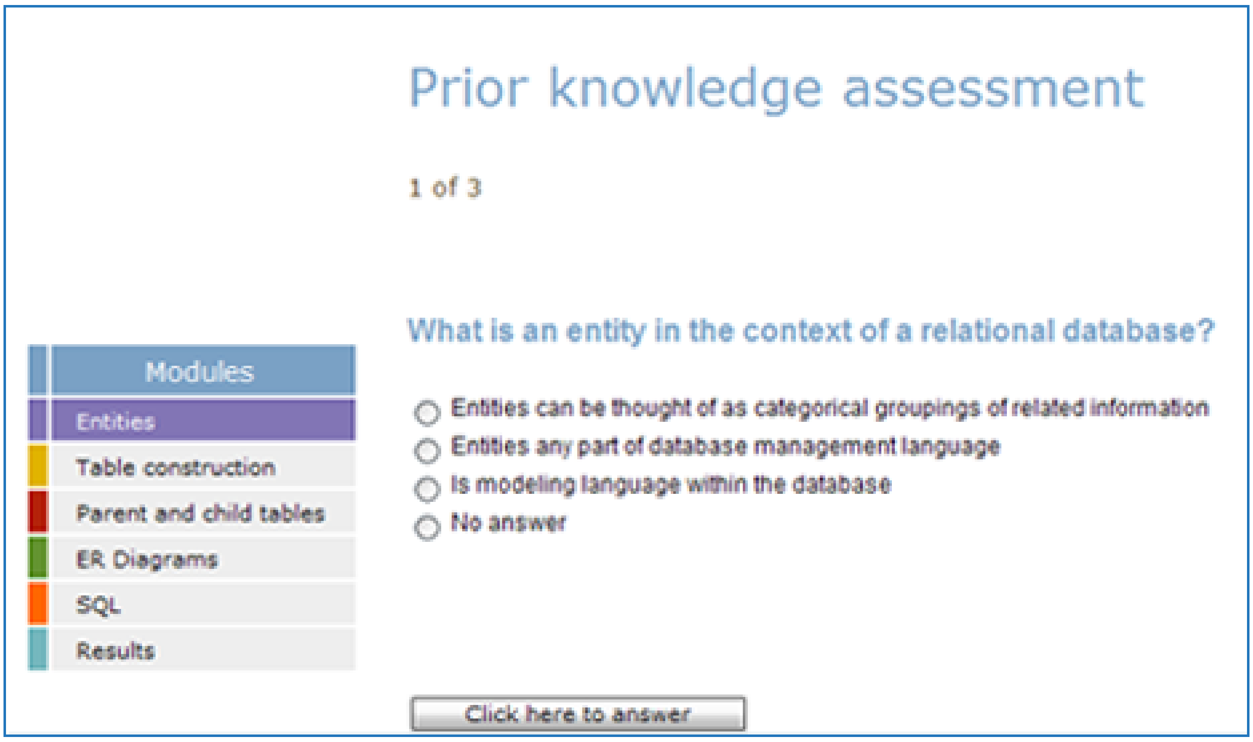
The objectives of the course are:

- Distinguish between records and fields in database table
- Distinguish between parent and child tables in relational database
- Use primary and foreign keys to define relationships between tables
- Design a relational database from an existing flat-file system(eg: Excel)

Note that you might need headphones or speakers depending on your learning style. *Make* sure to connect speakers/headphones to your computer.

Click "Next" on the top of the page to start the prior knowledge assessment.

Prior knowledge assessment



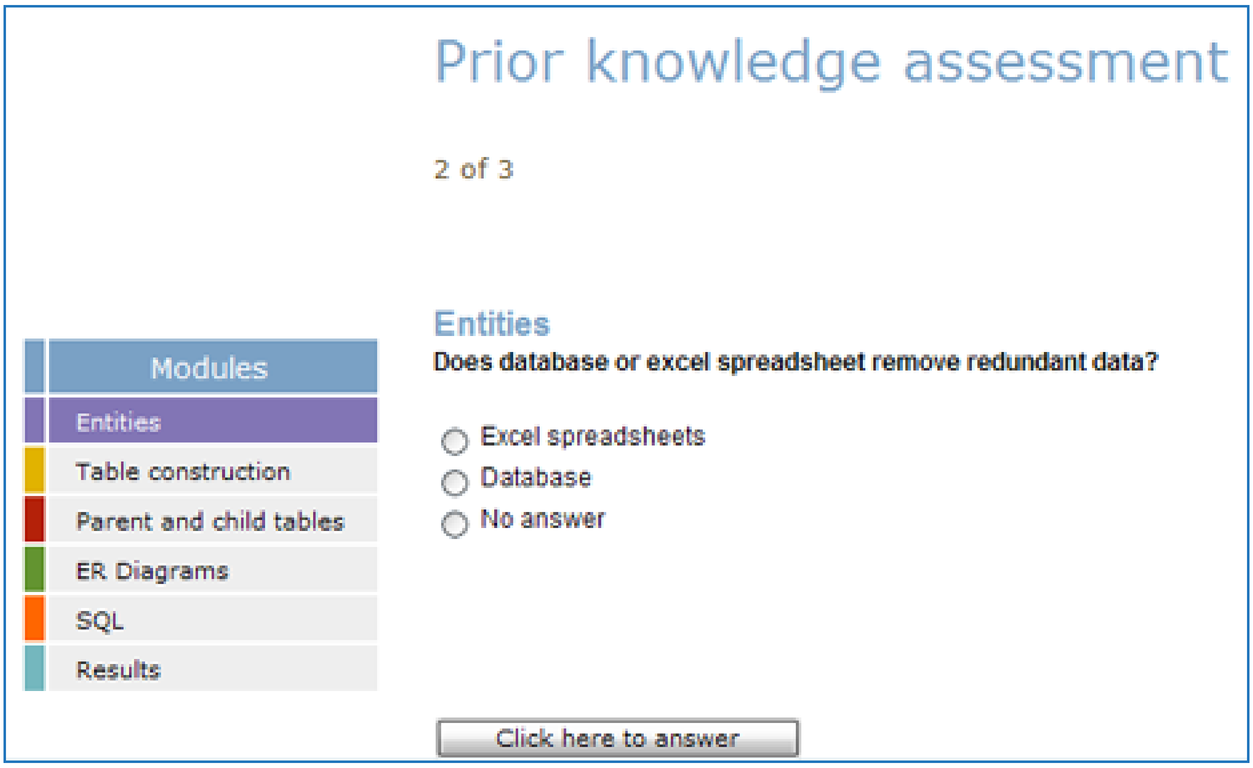
Prior knowledge assessment

1 of 3

What is an entity in the context of a relational database?

○ Entities can be thought of as categorical groupings of related information
○ Entities any part of database management language
○ Is modeling language within the database
○ No answer

Figure 7.3: (1) Question on Entities



Prior knowledge assessment

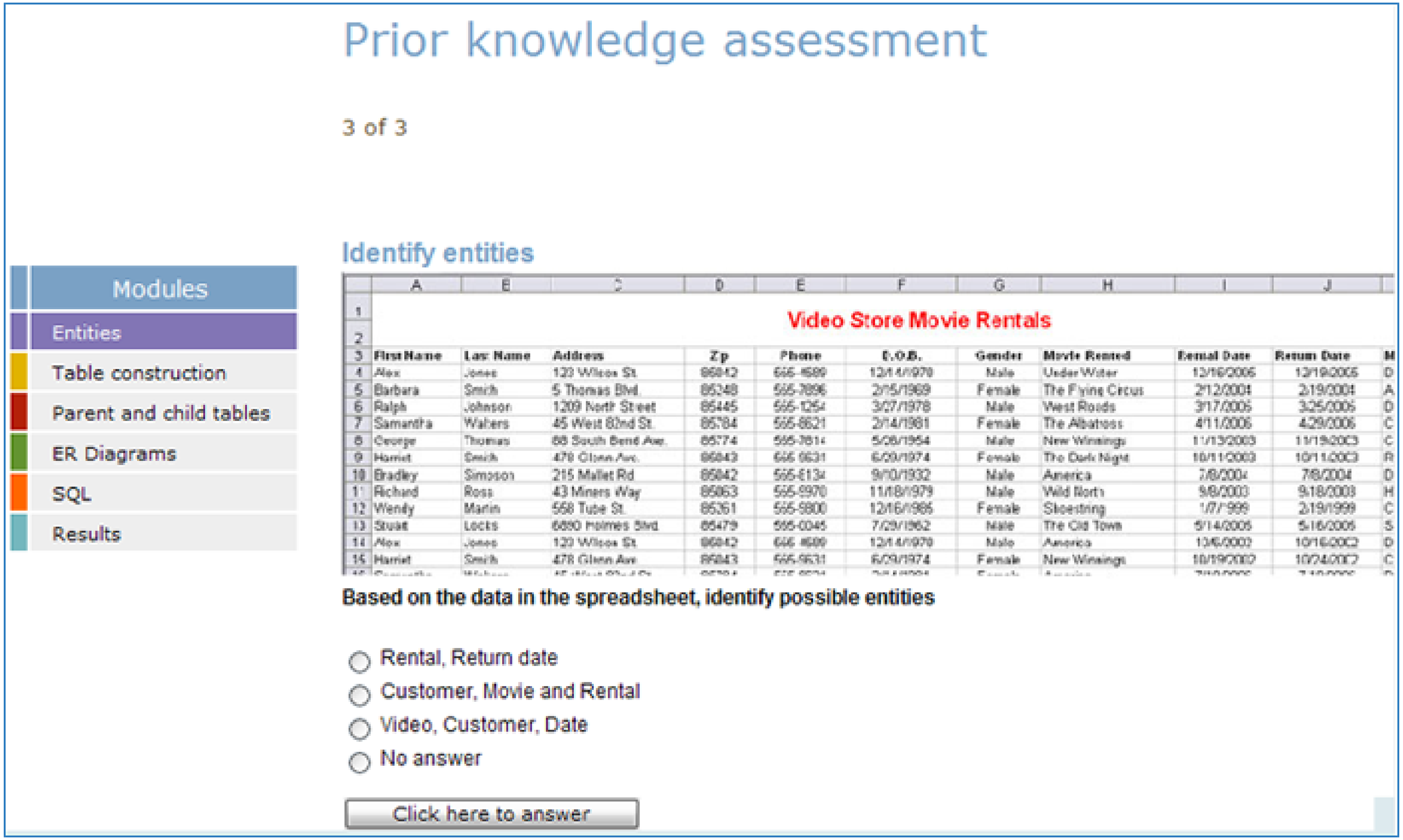
2 of 3

Entities

Does database or excel spreadsheet remove redundant data?

○ Excel spreadsheets
○ Database
○ No answer

Figure 7.4: (2) Question on Entities



Prior knowledge assessment

3 of 3

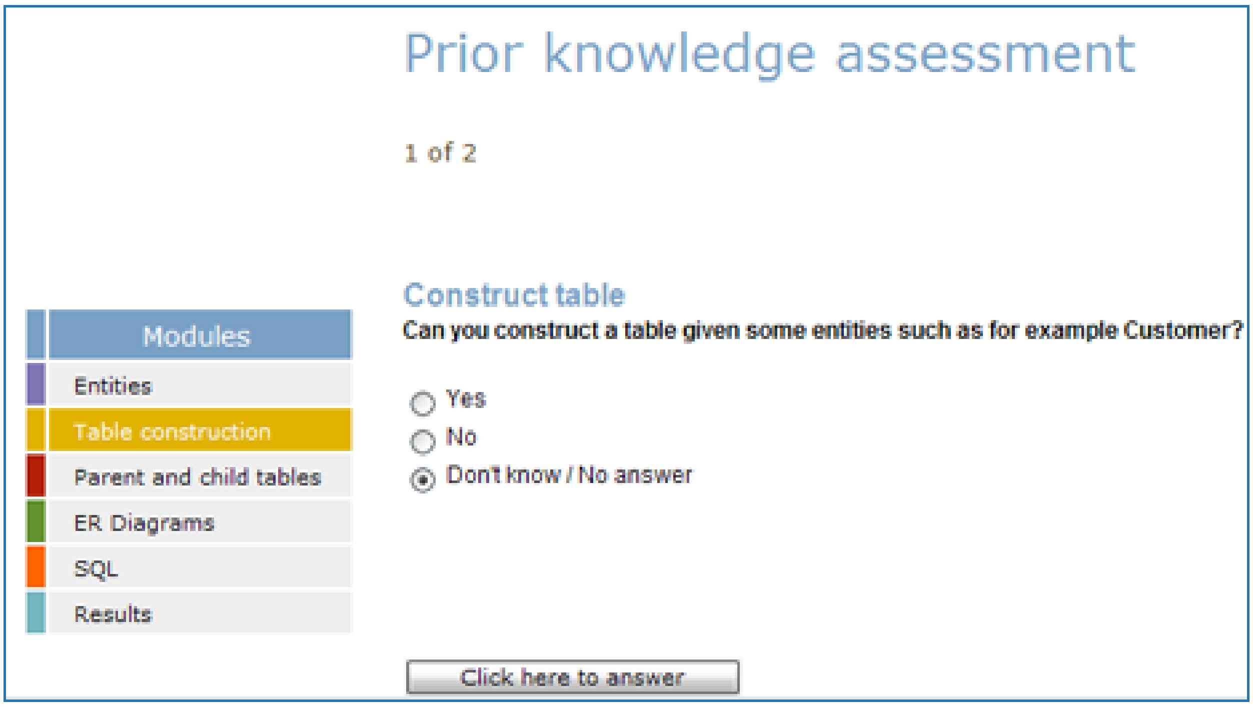
Identify entities

Video Store Movie Rentals

	First Name	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	M
4	Alex	Jones	120 Wilson St.	86842	666-4689	12/14/1979	Male	Under Water	12/16/2006	12/19/2006	D
5	Barbara	Smith	5 Thomas Blvd.	85248	555-7896	2/15/1969	Female	The Flying Circus	2/12/2004	2/19/2004	A
6	Ralph	Johnson	1209 North Street	85445	555-1254	3/27/1978	Male	West Roads	3/17/2006	3/25/2006	D
7	Samantha	Walters	45 West 82nd St.	85784	555-8521	2/14/1981	Female	The Albatross	4/11/2006	4/29/2006	C
8	George	Thomas	88 South Bend Ave.	85774	555-7814	5/26/1954	Male	New Winnings	11/13/2003	11/19/2003	C
9	Harriet	Smith	478 Glenn Ave.	86843	666-9631	6/29/1974	Female	The Dark Night	10/11/2003	10/11/2003	R
10	Bradley	Simpson	215 Mallet Rd	86842	555-6134	9/10/1932	Male	America	7/8/2004	7/8/2004	D
11	Richard	Ross	43 Miners Way	86863	555-5970	11/18/1979	Male	Wild North	9/8/2003	9/18/2003	H
12	Wendy	Martin	568 Tube St.	85261	555-9800	12/16/1986	Female	Shoestring	1/7/1999	2/19/1999	C
13	Stuart	Locks	6890 Holmes Blvd.	85479	555-0045	7/29/1962	Male	The Old Town	5/14/2006	5/16/2006	S
14	Alex	Jones	120 Wilson St.	86842	666-4689	12/14/1979	Male	America	10/6/2002	10/16/2002	D
15	Harriet	Smith	478 Glenn Ave.	86843	666-9631	6/29/1974	Female	New Winnings	10/19/2002	10/24/2002	C

Based on the data in the spreadsheet, identify possible entities

- ◯ Rental, Return date
- ◯ Customer, Movie and Rental
- ◯ Video, Customer, Date
- ◯ No answer



Prior knowledge assessment

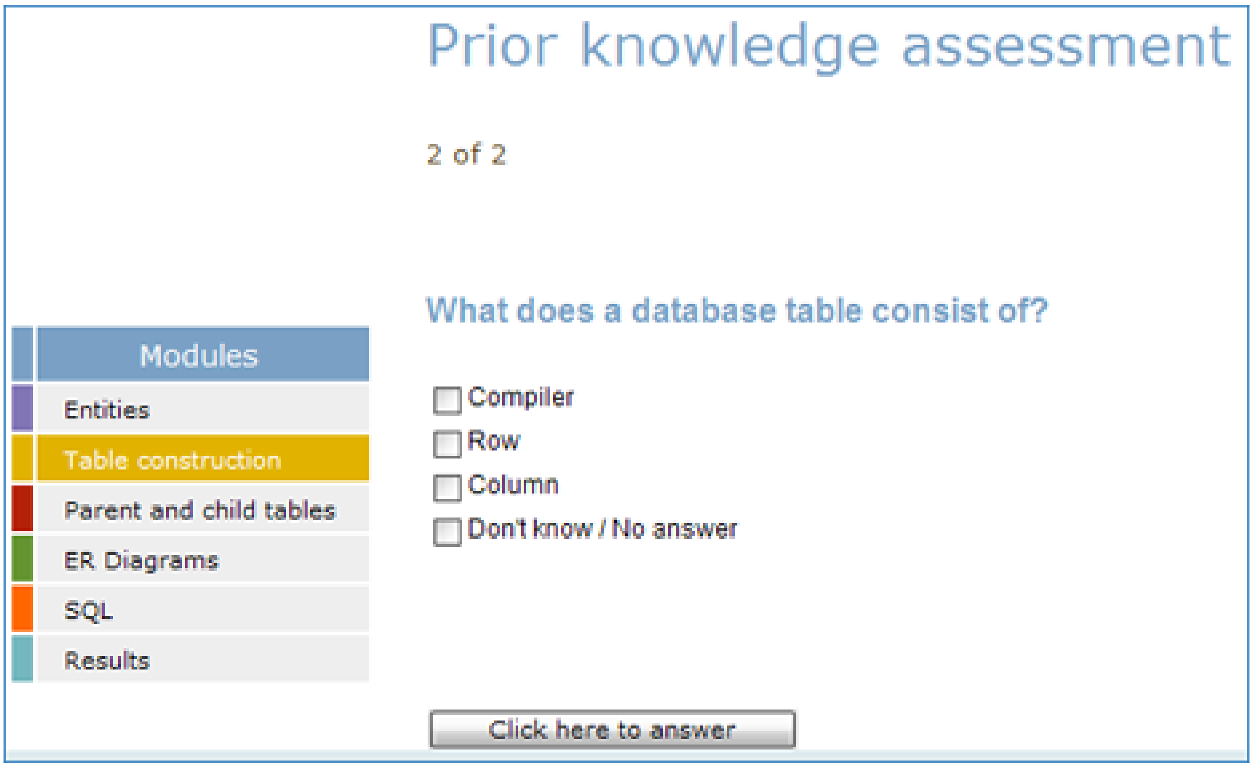
1 of 2

Construct table

Can you construct a table given some entities such as for example Customer?

- Yes
- No
- [✓ Don't know / No answer]

Figure 7.6: (1) Question on table construction



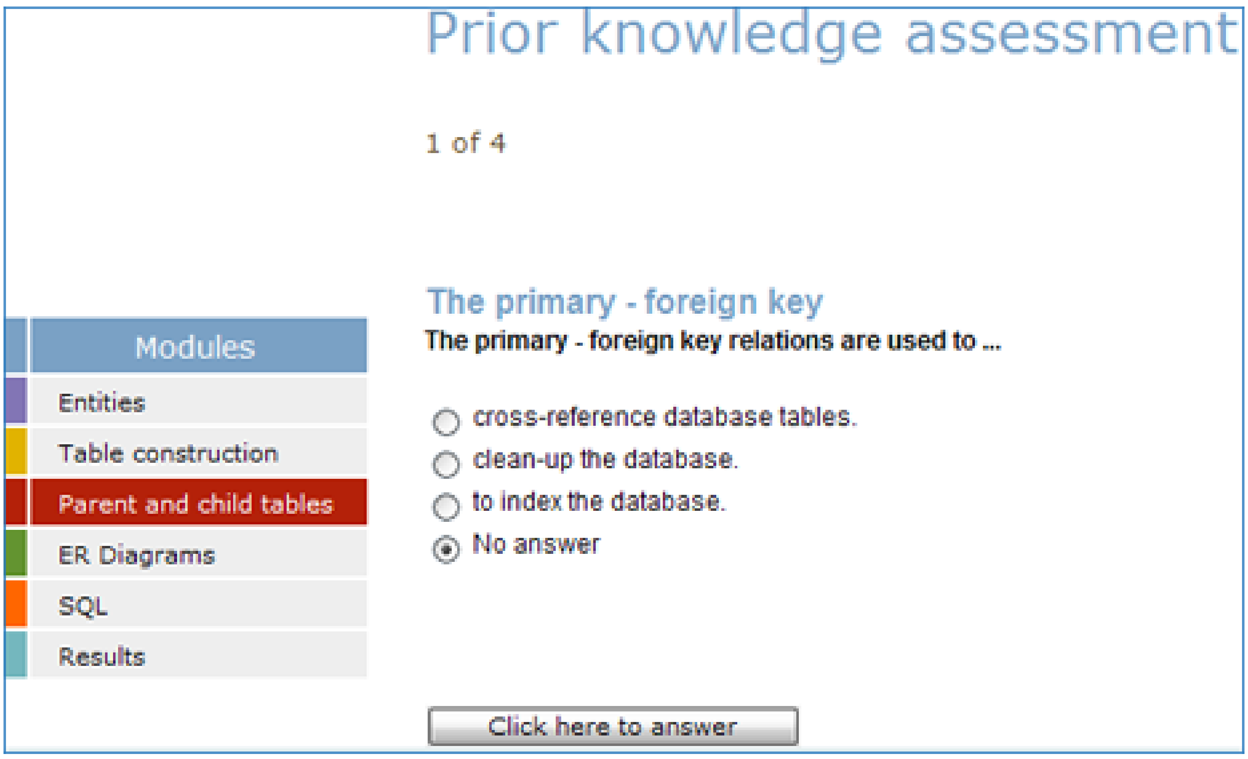
Prior knowledge assessment

2 of 2

What does a database table consist of?

- ☐ Compiler
- ☐ Row
- ☐ Column
- ☐ Don't know / No answer

Figure 7.7: (2) Question on table construction



Prior knowledge assessment

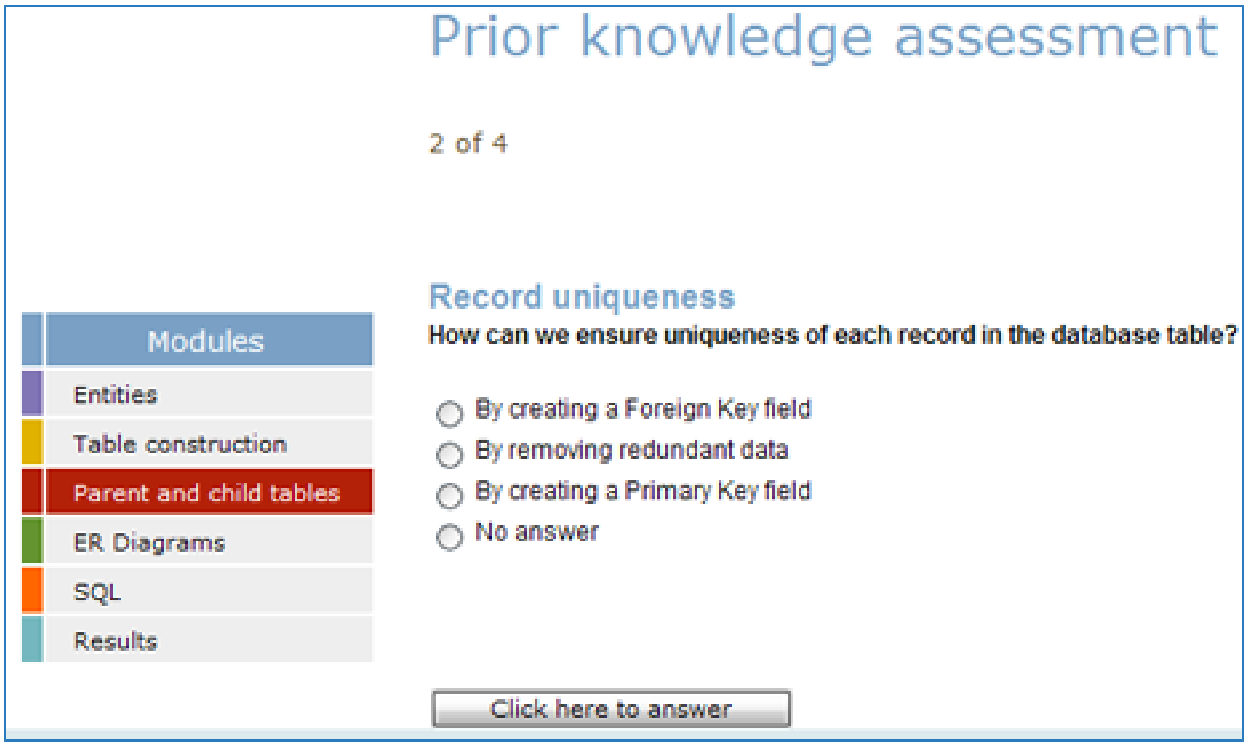
1 of 4

The primary - foreign key

The primary - foreign key relations are used to ...

- ○ cross-reference database tables.
- ○ clean-up the database.
- ○ to index the database.
- ◉ No answer

Figure 7.8: (1) Question on parent and child tables



Prior knowledge assessment

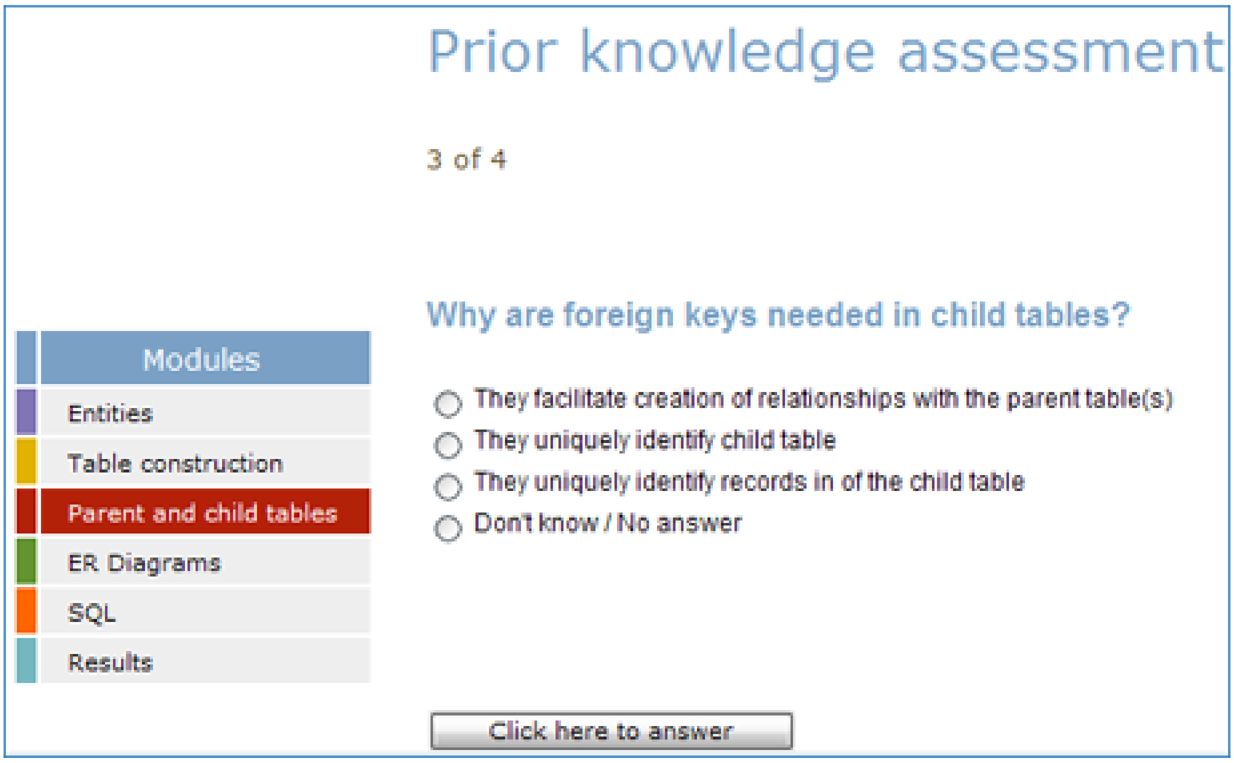
2 of 4

Record uniqueness

How can we ensure uniqueness of each record in the database table?

- By creating a Foreign Key field
- By removing redundant data
- By creating a Primary Key field
- No answer

Figure 7.9: (2) Question on parent and child tables



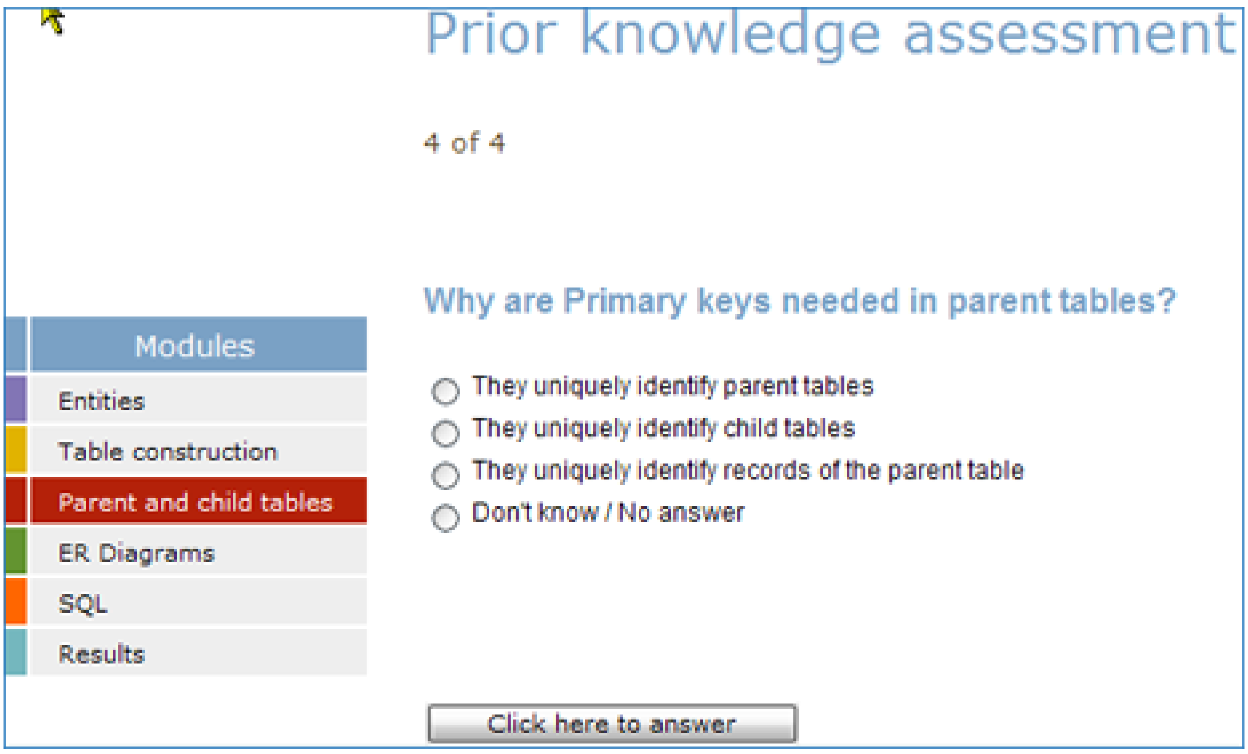
Prior knowledge assessment

3 of 4

Why are foreign keys needed in child tables?

- ○ They facilitate creation of relationships with the parent table(s)
- ○ They uniquely identify child table
- ○ They uniquely identify records in of the child table
- ○ Don't know / No answer

Figure 7.10: (3) Question on parent and child tables

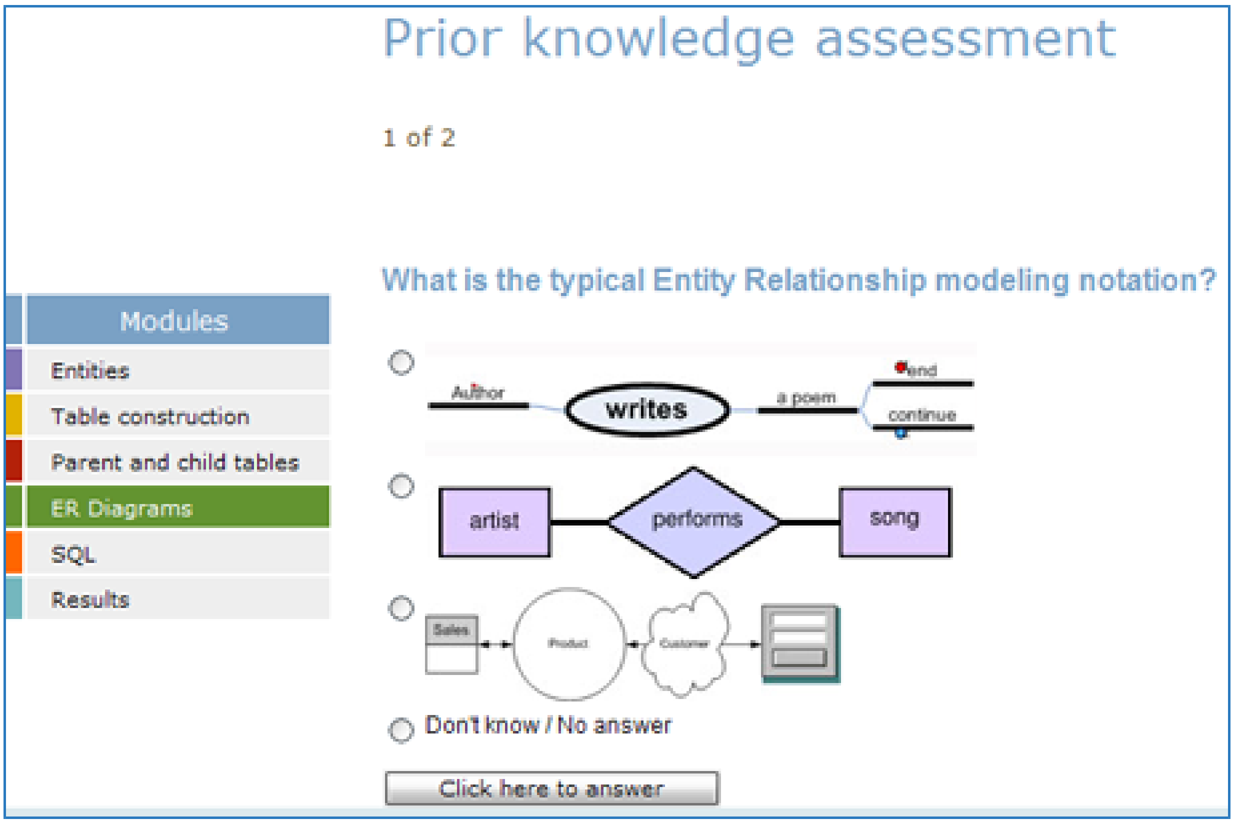

Prior knowledge assessment

4 of 4

Why are Primary keys needed in parent tables?

- ○ They uniquely identify parent tables
- ○ They uniquely identify child tables
- ○ They uniquely identify records of the parent table
- ○ Don't know / No answer

Figure 7.11: (4) Question on parent and child tables

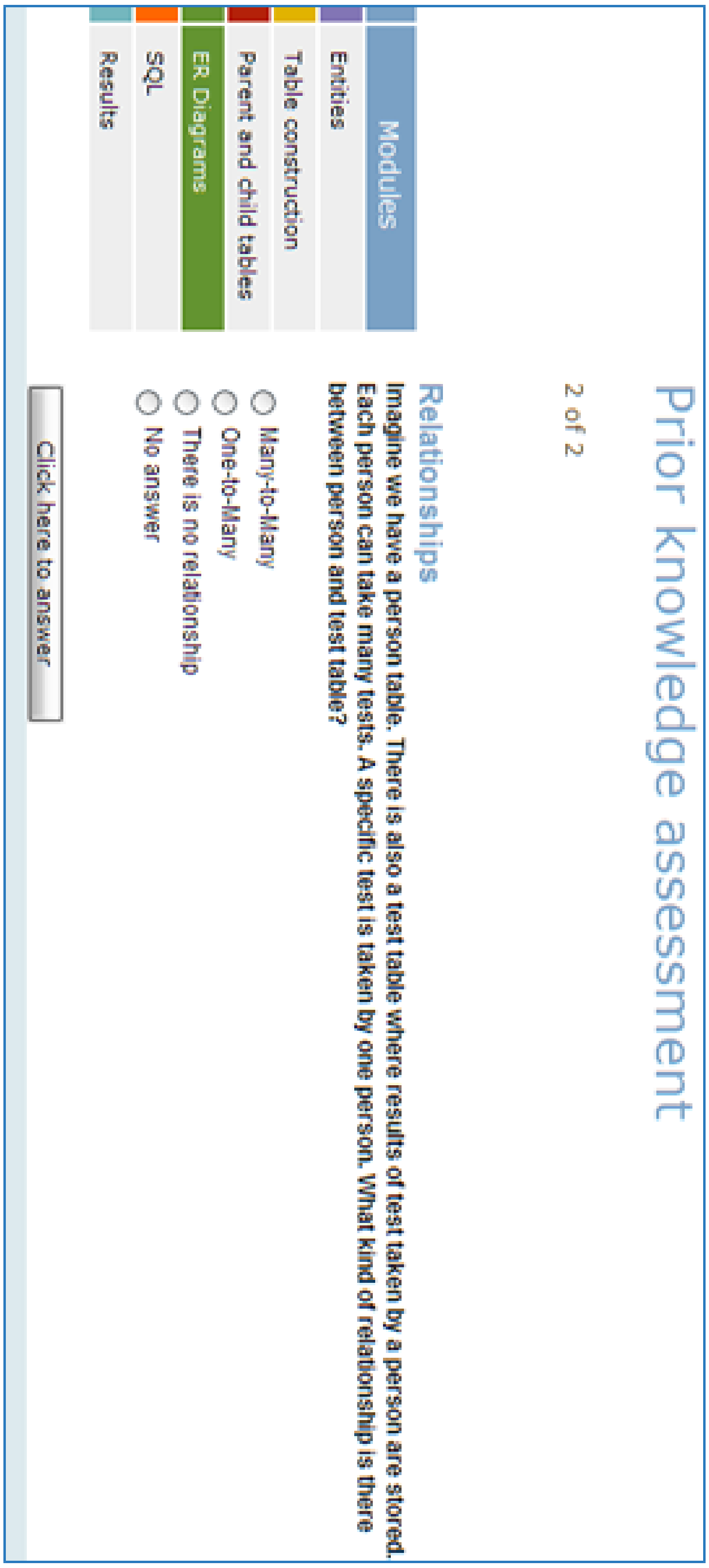


Prior knowledge assessment

1 of 2

What is the typical Entity Relationship modeling notation?

Figure 7.12: (1) Question on ER diagrams



Prior knowledge assessment

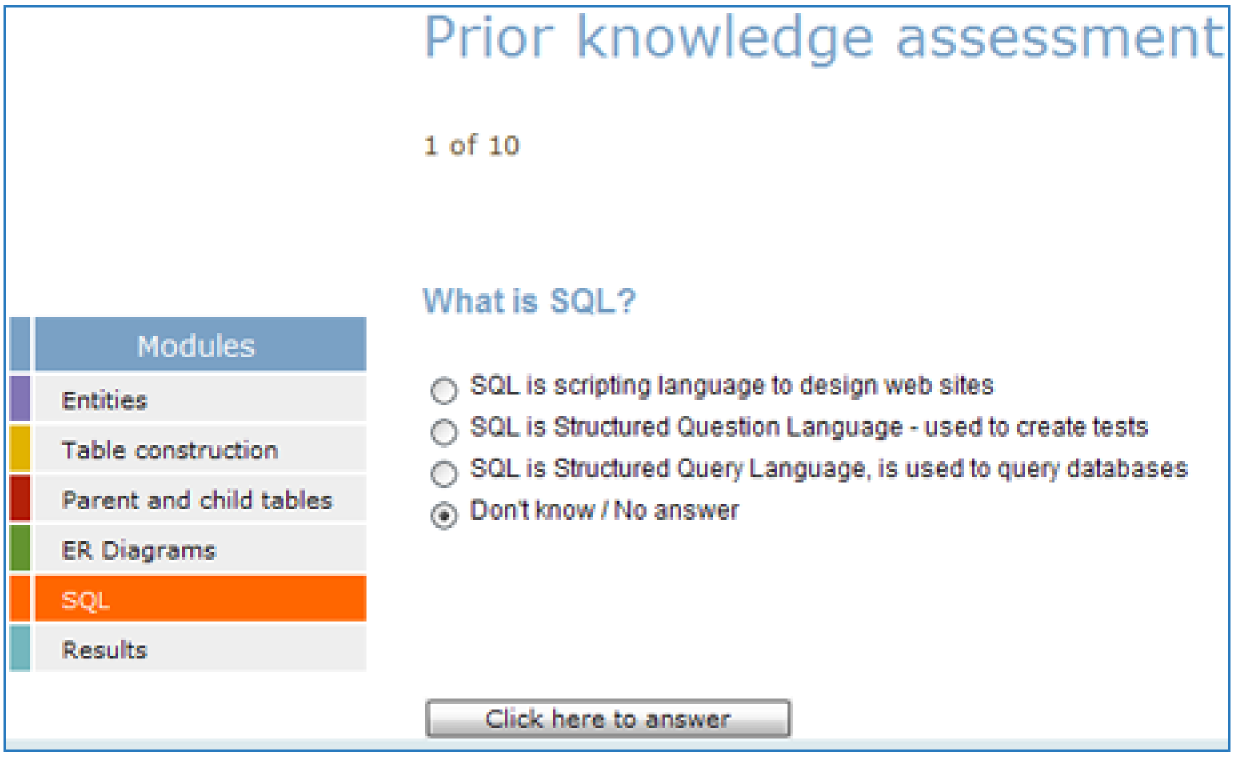
2 of 2

Relationships

Imagine we have a person table. There is also a test table where results of test taken by a person are stored. Each person can take many tests. A specific test is taken by one person. What kind of relationship is there between person and test table?

- Many-to-Many
- One-to-Many
- There is no relationship
- No answer

Figure 7.13: (2) Question on ER diagrams



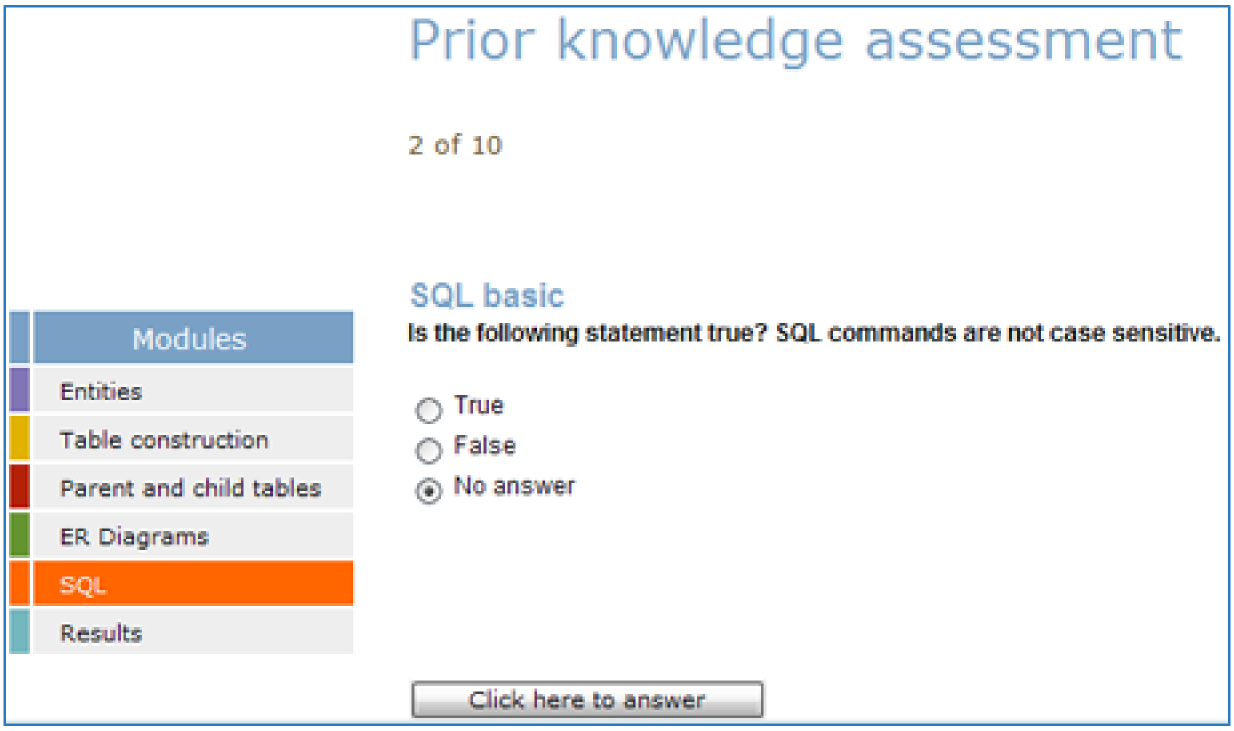
Prior knowledge assessment

1 of 10

What is SQL?

- ○ SQL is scripting language to design web sites
- ○ SQL is Structured Question Language - used to create tests
- ○ SQL is Structured Query Language, is used to query databases
- [✓ Don't know / No answer]

Figure 7.14: (1) Question on SQL


Prior knowledge assessment

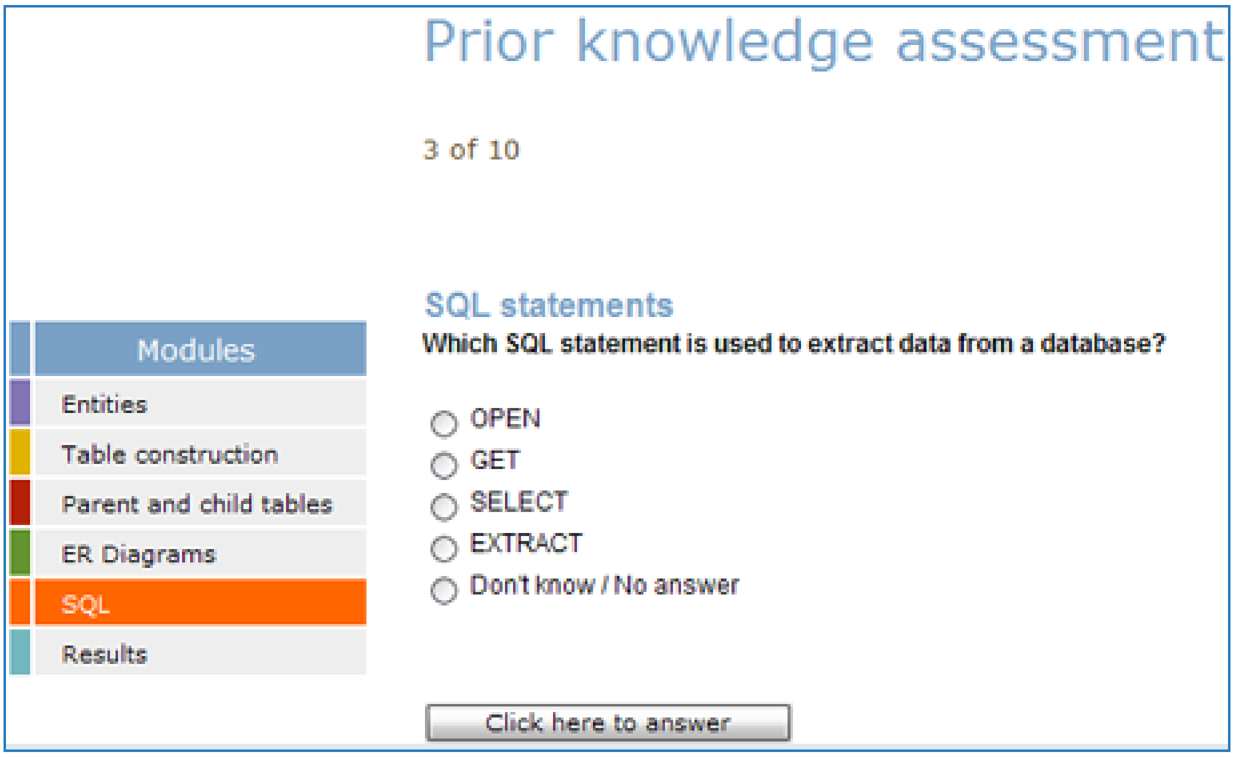
2 of 10

SQL basic

Is the following statement true? SQL commands are not case sensitive.

- ○ True
- ○ False
- [✓ No answer]

Figure 7.15: (2) Question on SQL



Prior knowledge assessment

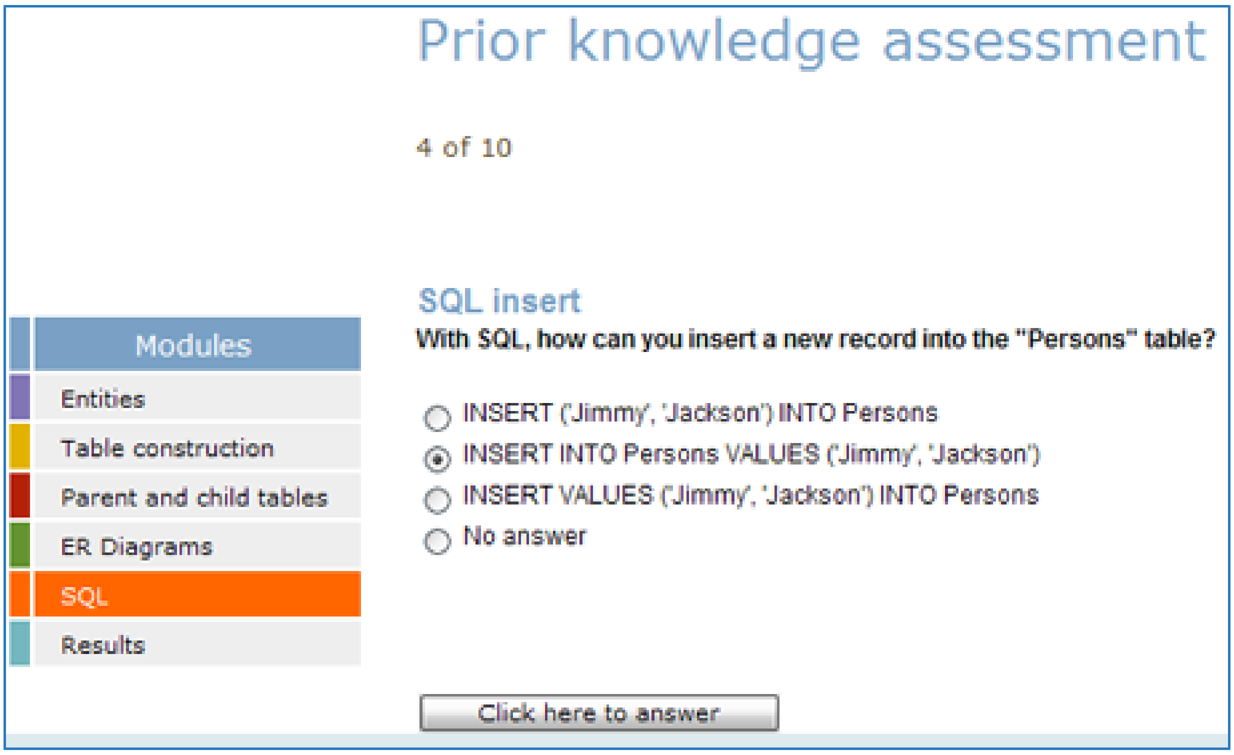
3 of 10

SQL statements

Which SQL statement is used to extract data from a database?

- OPEN
- GET
- SELECT
- EXTRACT
- Don't know / No answer

Figure 7.16: (3) Question on SQL



Prior knowledge assessment

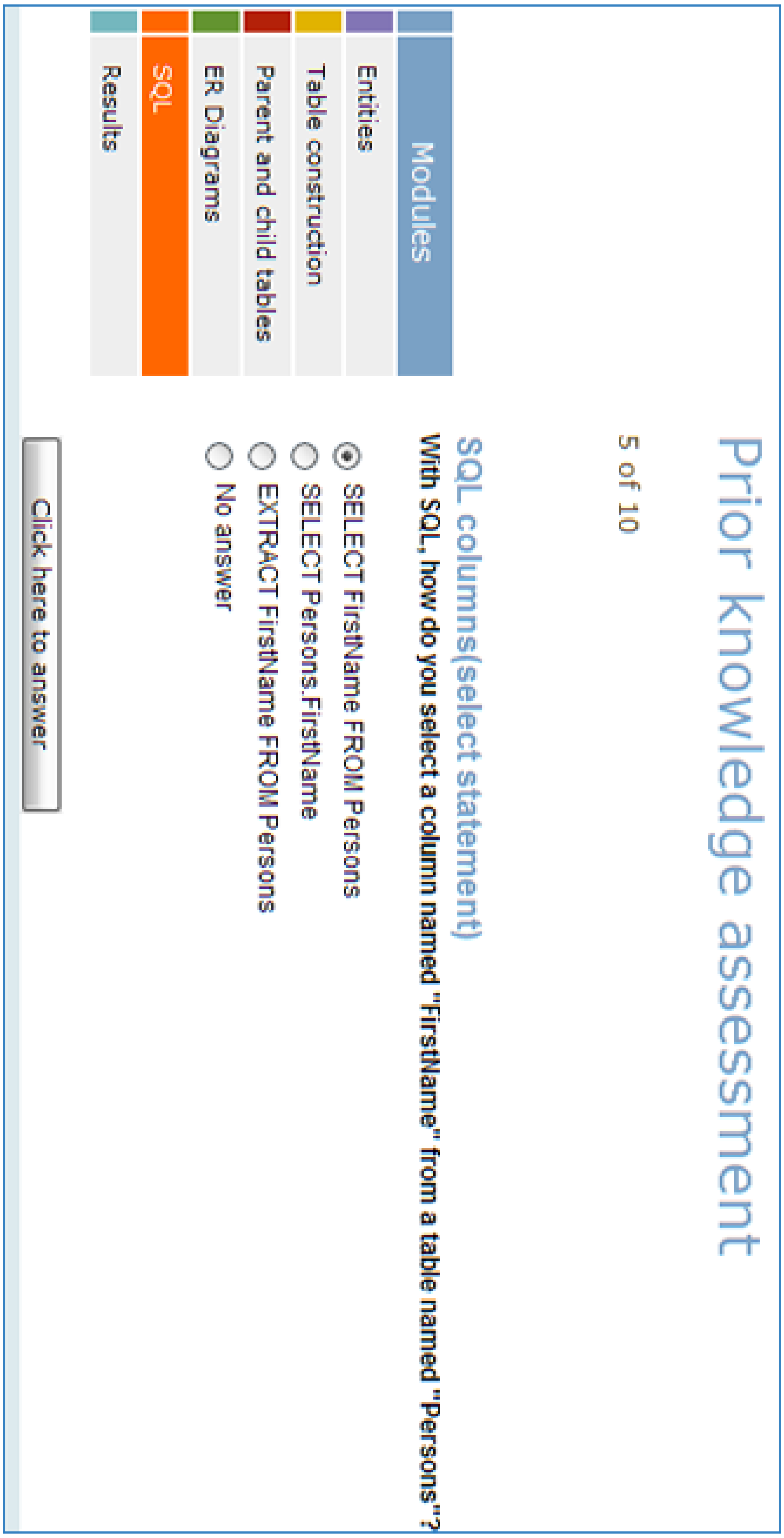
4 of 10

SQL insert

With SQL, how can you insert a new record into the "Persons" table?

- ○ INSERT ('Jimmy', 'Jackson') INTO Persons
- [✓ INSERT INTO Persons VALUES ('Jimmy', 'Jackson')]
- ○ INSERT VALUES ('Jimmy', 'Jackson') INTO Persons
- ○ No answer

Figure 7.17: (4) Question on SQL



Prior knowledge assessment

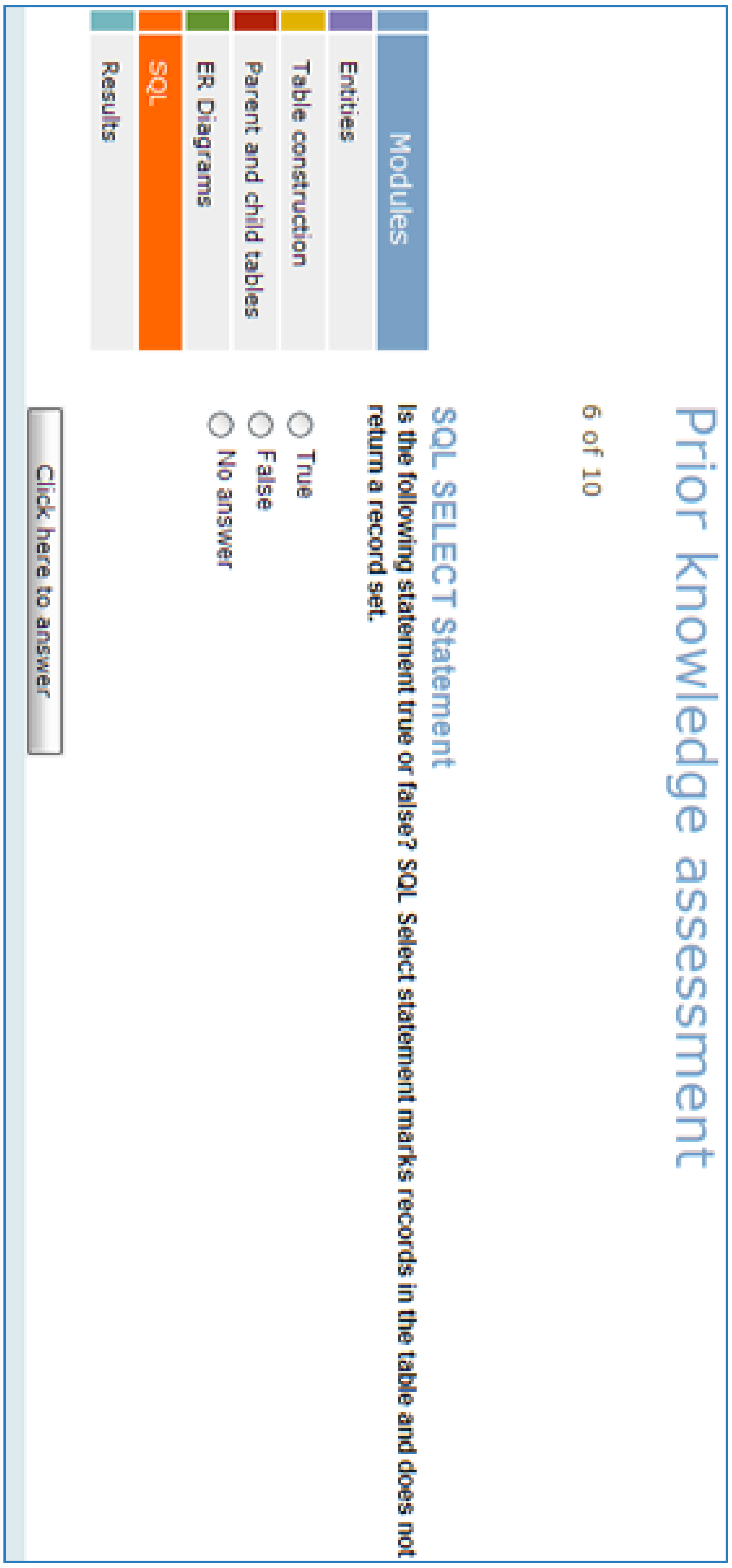
5 of 10

SQL columns(select statement)

With SQL, how do you select a column named "FirstName" from a table named "Persons"?

- [✓ SELECT FirstName FROM Persons]
- SELECT Persons.FirstName
- EXTRACT FirstName FROM Persons
- No answer

Figure 7.18: (5) Question on SQL



Prior knowledge assessment

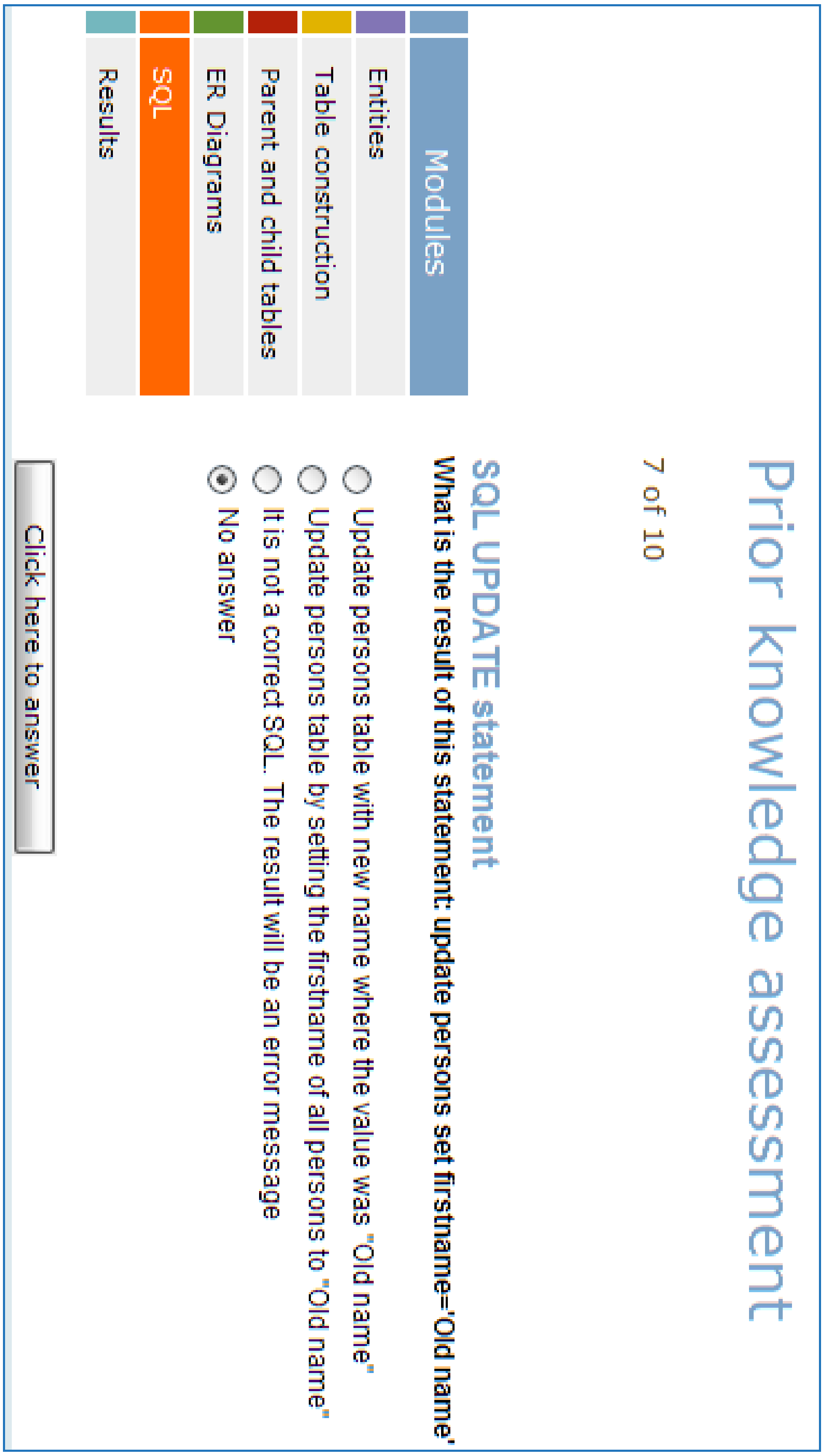
6 of 10

SQL SELECT Statement

Is the following statement true or false? SQL Select statement marks records in the table and does not return a record set.

- True
- False
- No answer

Figure 7.19: (6) Question on SQL



Prior knowledge assessment

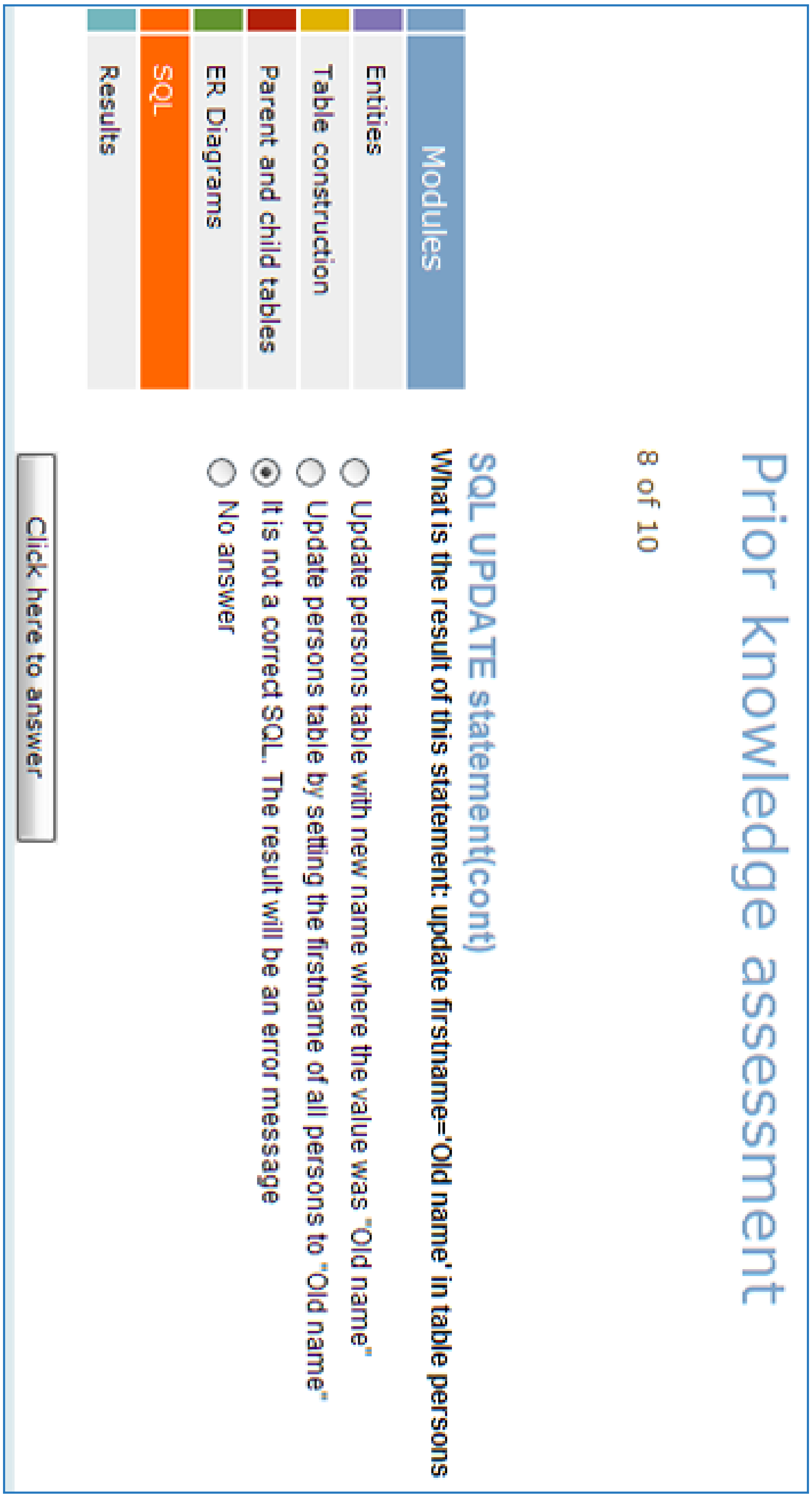
7 of 10

SQL UPDATE statement

What is the result of this statement: update persons set firstname='Old name'

- Update persons table with new name where the value was "Old name"
- Update persons table by setting the firstname of all persons to "Old name"
- It is not a correct SQL. The result will be an error message
- [✓ No answer]

Figure 7.20: (7) Question on SQL



Prior knowledge assessment

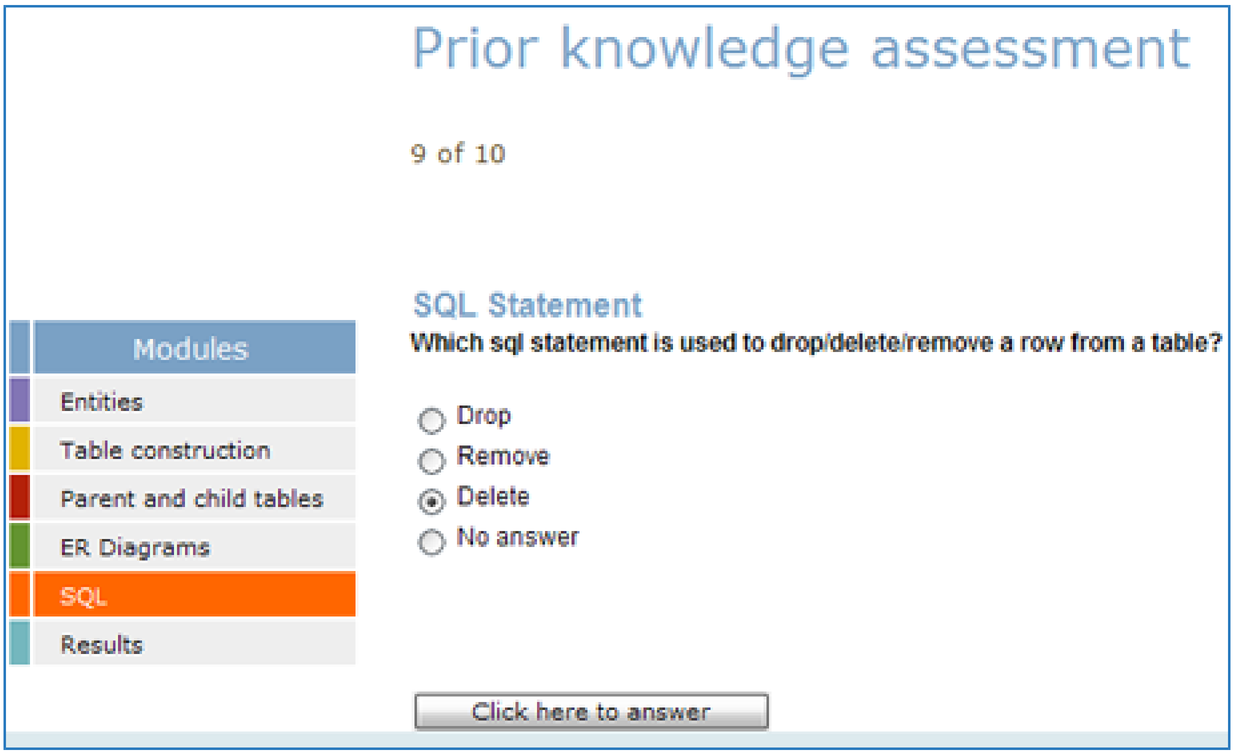
8 of 10

SQL UPDATE statement(cont)

What is the result of this statement: update firstname='Old name' in table persons

- Update persons table with new name where the value was "Old name"
- Update persons table by setting the firstname of all persons to "Old name"
- [✓ It is not a correct SQL. The result will be an error message]
- No answer

Figure 7.21: (8) Question on SQL



Prior knowledge assessment

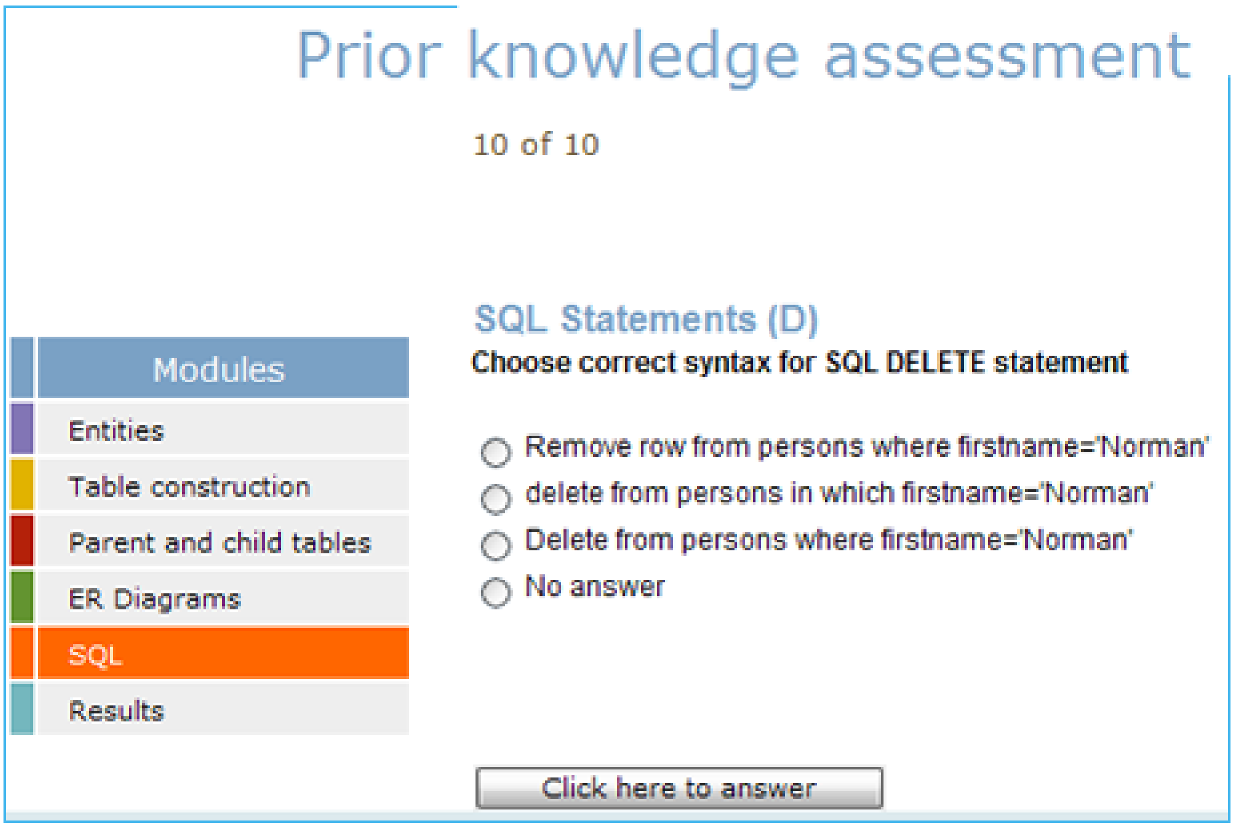
9 of 10

SQL Statement

Which sql statement is used to drop/delete/remove a row from a table?

- Drop
- Remove
- [✓ Delete]
- No answer

Figure 7.22: (9) Question on SQL



Prior knowledge assessment

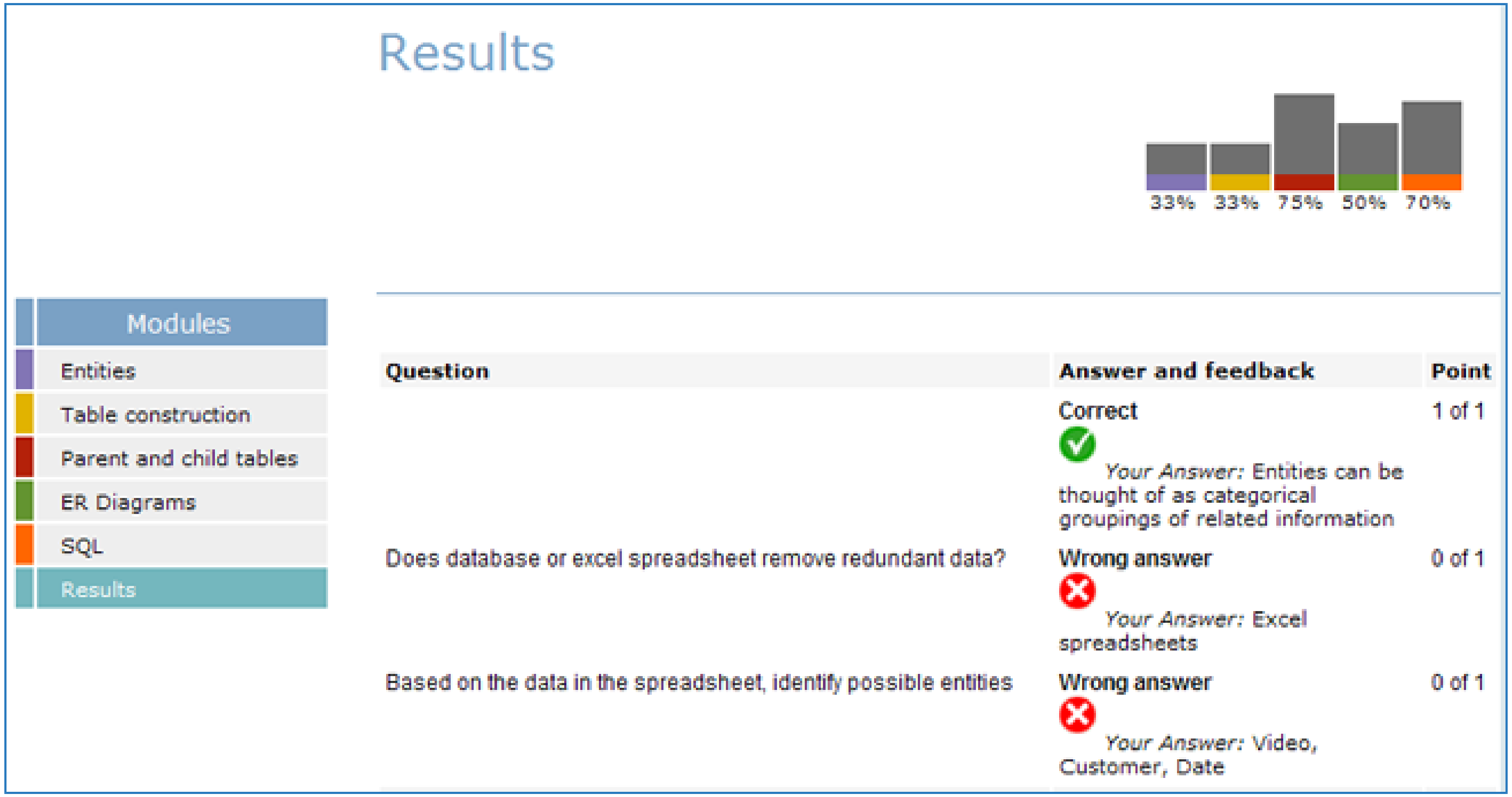
10 of 10

SQL Statements (D)

Choose correct syntax for SQL DELETE statement

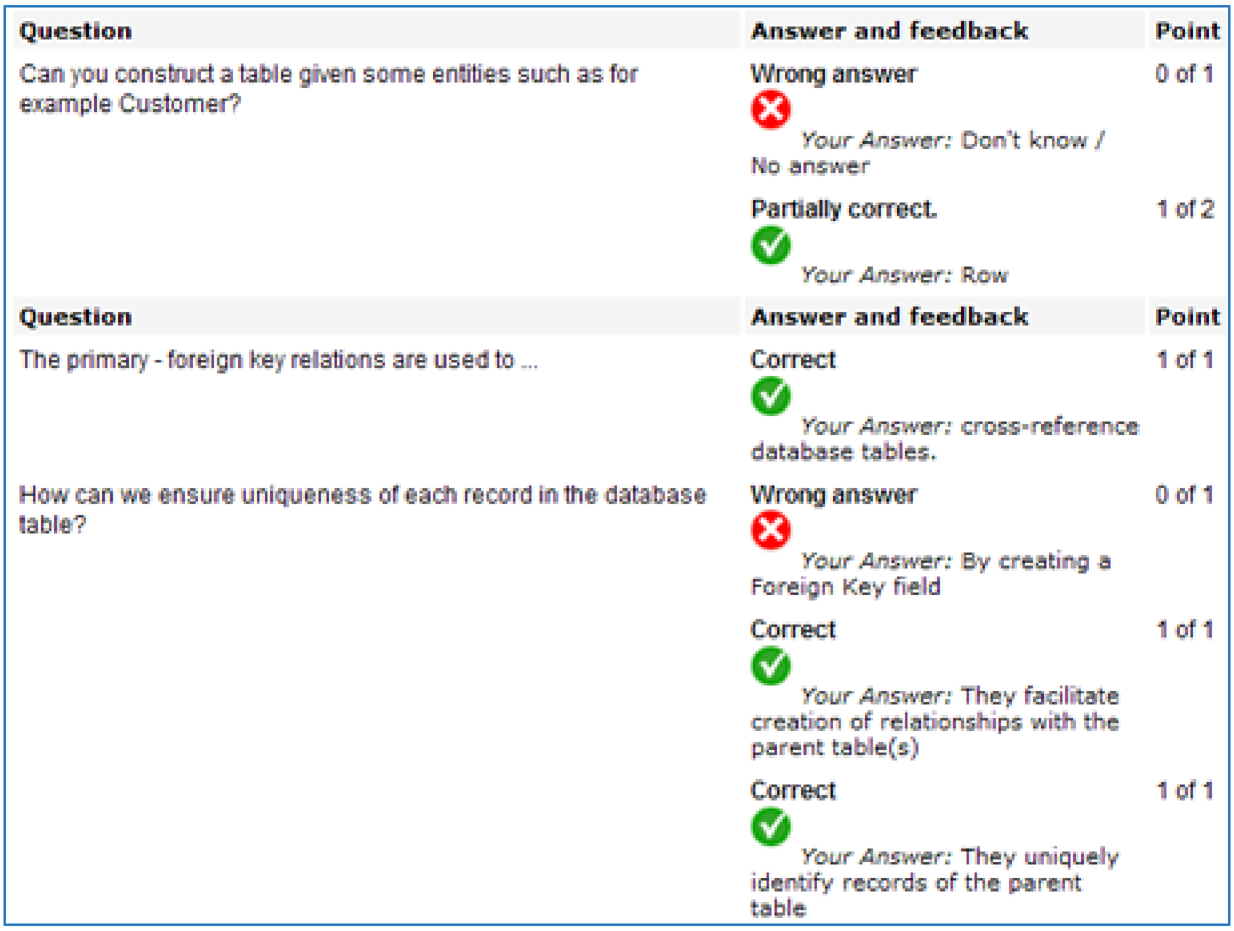
- ○ Remove row from persons where firstname='Norman'
- ○ delete from persons in which firstname='Norman'
- ○ Delete from persons where firstname='Norman'
- ○ No answer

Figure 7.23: (10) Question on SQL



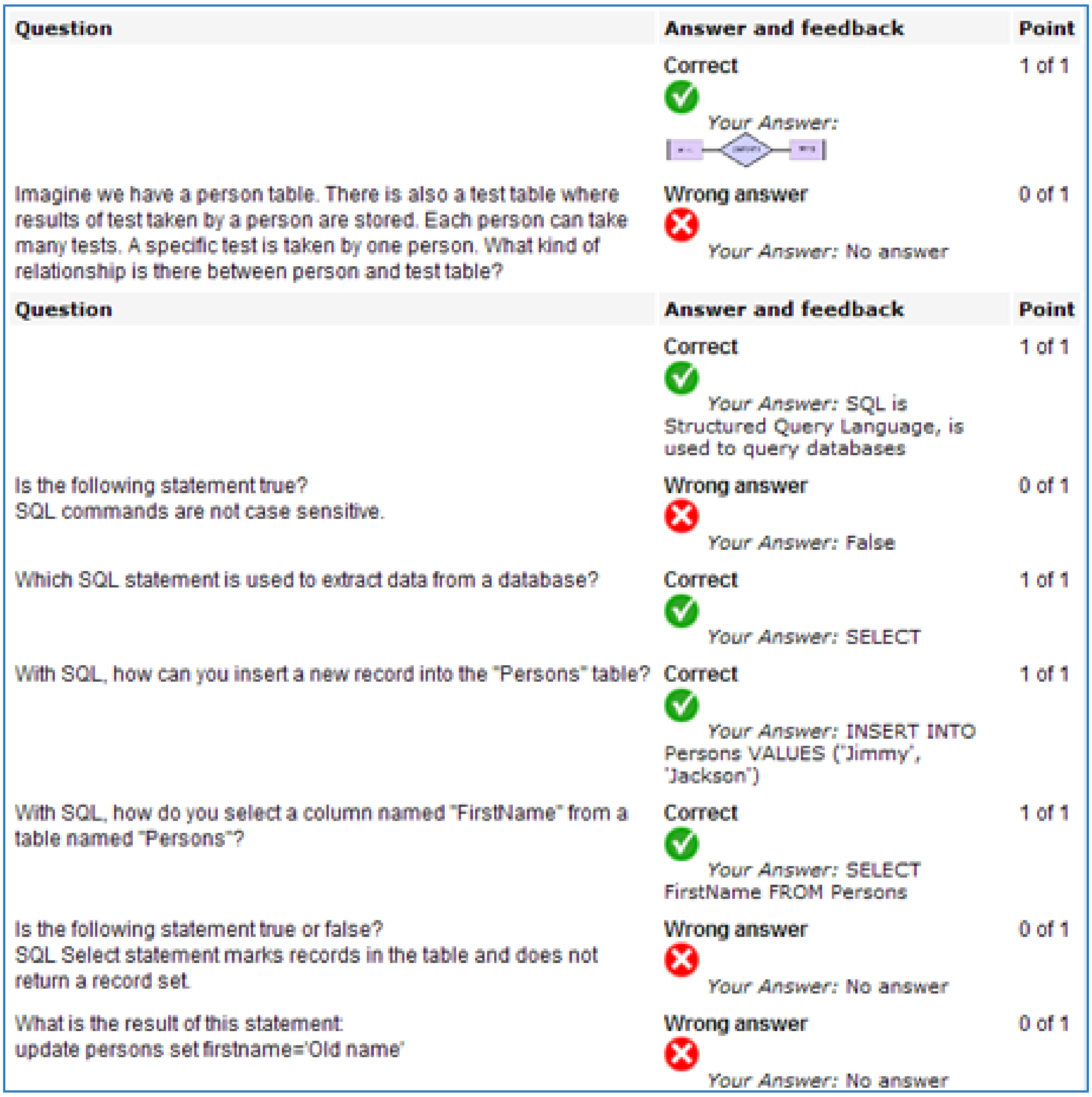
Question	Answer and feedback	Point
	Correct ✓ *Your Answer:* Entities can be thought of as categorical groupings of related information	1 of 1
Does database or excel spreadsheet remove redundant data?	Wrong answer ✗ *Your Answer:* Excel spreadsheets	0 of 1
Based on the data in the spreadsheet, identify possible entities	Wrong answer ✗ *Your Answer:* Video, Customer, Date	0 of 1

Figure 7.24: Results page (1 of 4)



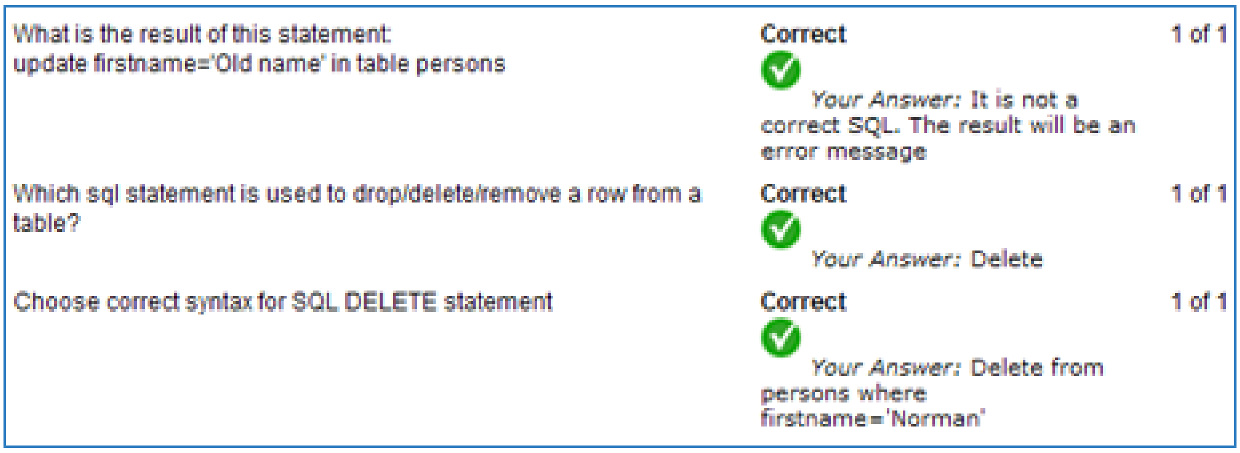
Question	Answer and feedback	Point
Can you construct a table given some entities such as for example Customer?	Wrong answer ✗ *Your Answer:* Don't know / No answer	0 of 1
	Partially correct. ✓ *Your Answer:* Row	1 of 2
Question	**Answer and feedback**	**Point**
The primary - foreign key relations are used to ...	Correct ✓ *Your Answer:* cross-reference database tables.	1 of 1
How can we ensure uniqueness of each record in the database table?	Wrong answer ✗ *Your Answer:* By creating a Foreign Key field	0 of 1
	Correct ✓ *Your Answer:* They facilitate creation of relationships with the parent table(s)	1 of 1
	Correct ✓ *Your Answer:* They uniquely identify records of the parent table	1 of 1

Figure 7.25: Results page (2 of 4)



Question	Answer and feedback	Point
	Correct ✓ *Your Answer:*	1 of 1
Imagine we have a person table. There is also a test table where results of test taken by a person are stored. Each person can take many tests. A specific test is taken by one person. What kind of relationship is there between person and test table?	Wrong answer ✗ *Your Answer:* No answer	0 of 1
Question	**Answer and feedback**	**Point**
	Correct ✓ *Your Answer:* SQL is Structured Query Language, is used to query databases	1 of 1
Is the following statement true? SQL commands are not case sensitive.	Wrong answer ✗ *Your Answer:* False	0 of 1
Which SQL statement is used to extract data from a database?	Correct ✓ *Your Answer:* SELECT	1 of 1
With SQL, how can you insert a new record into the "Persons" table?	Correct ✓ *Your Answer:* INSERT INTO Persons VALUES ('Jimmy', 'Jackson')	1 of 1
With SQL, how do you select a column named "FirstName" from a table named "Persons"?	Correct ✓ *Your Answer:* SELECT FirstName FROM Persons	1 of 1
Is the following statement true or false? SQL Select statement marks records in the table and does not return a record set.	Wrong answer ✗ *Your Answer:* No answer	0 of 1
What is the result of this statement: update persons set firstname='Old name'	Wrong answer ✗ *Your Answer:* No answer	0 of 1

Figure 7.26: Results page (3 of 4)

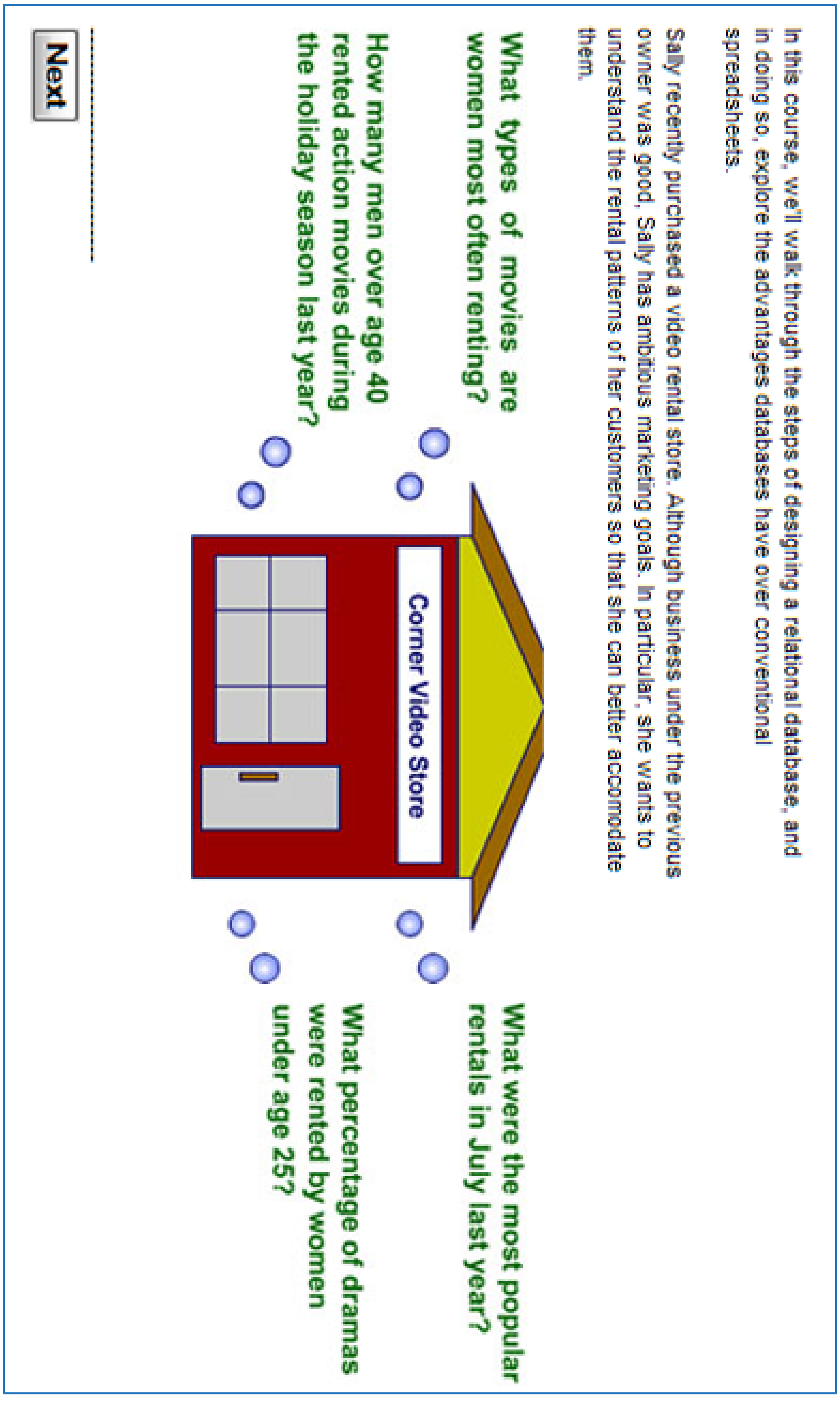


What is the result of this statement: update firstname='Old name' in table persons	**Correct** ✓ *Your Answer:* It is not a correct SQL. The result will be an error message	1 of 1
Which sql statement is used to drop/delete/remove a row from a table?	**Correct** ✓ *Your Answer:* Delete	1 of 1
Choose correct syntax for SQL DELETE statement	**Correct** ✓ *Your Answer:* Delete from persons where firstname='Norman'	1 of 1

Figure 7.27: Results page (4 of 4)

Course pages

The audio and flash content are uploaded to the server and inserted in documents. Sound and flash items require Internet connection, and therefore, these items can only be demonstrated using a PC.



In this course, we'll walk through the steps of designing a relational database, and in doing so, explore the advantages databases have over conventional spreadsheets.

Sally recently purchased a video rental store. Although business under the previous owner was good, Sally has ambitious marketing goals. In particular, she wants to understand the rental patterns of her customers so that she can better accomodate them.

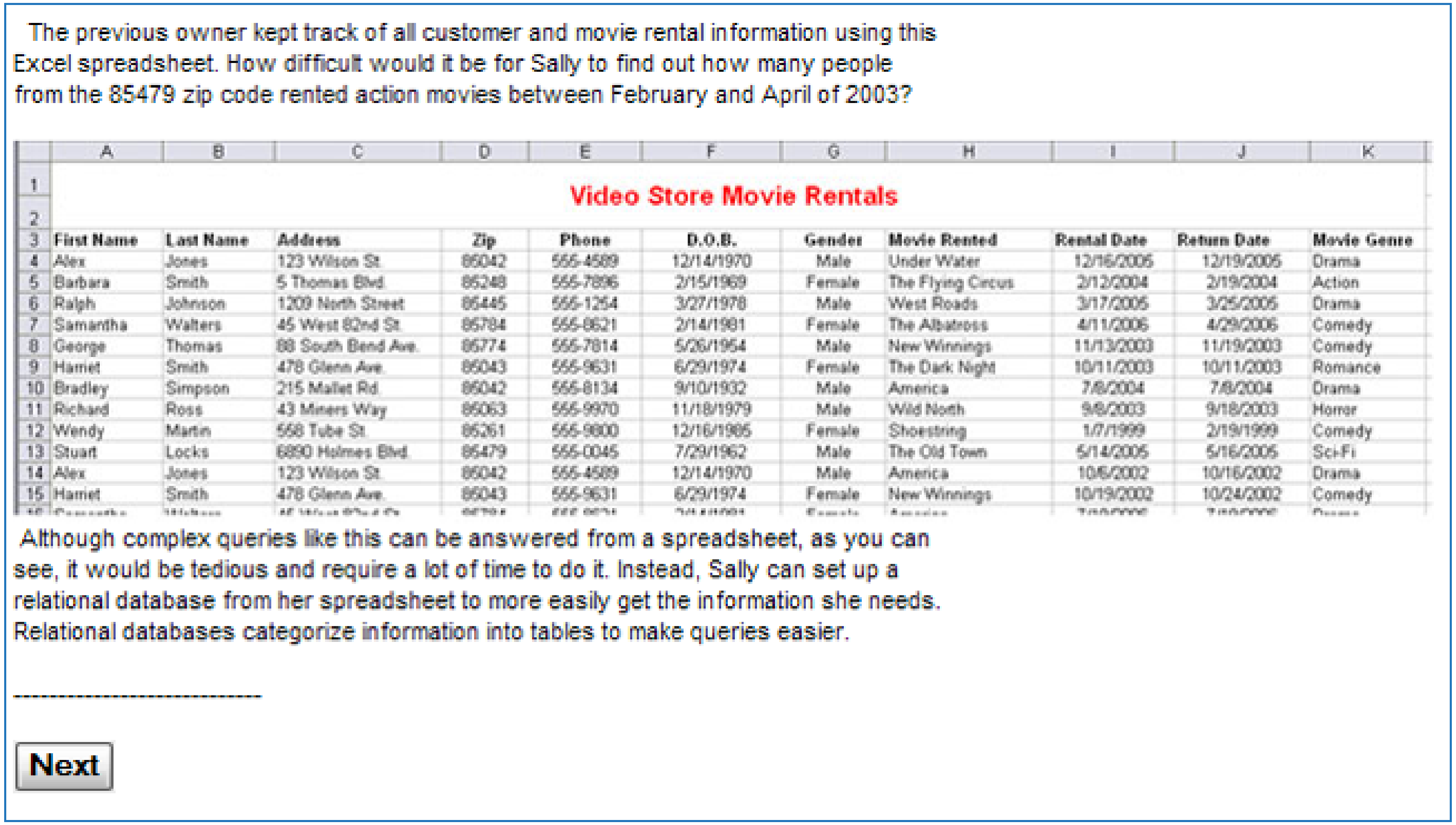
What types of movies are women most often renting?

How many men over age 40 rented action movies during the holiday season last year?

What were the most popular rentals in July last year?

What percentage of dramas were rented by women under age 25?

Figure 7.28: (1) Entities module for a visual user.



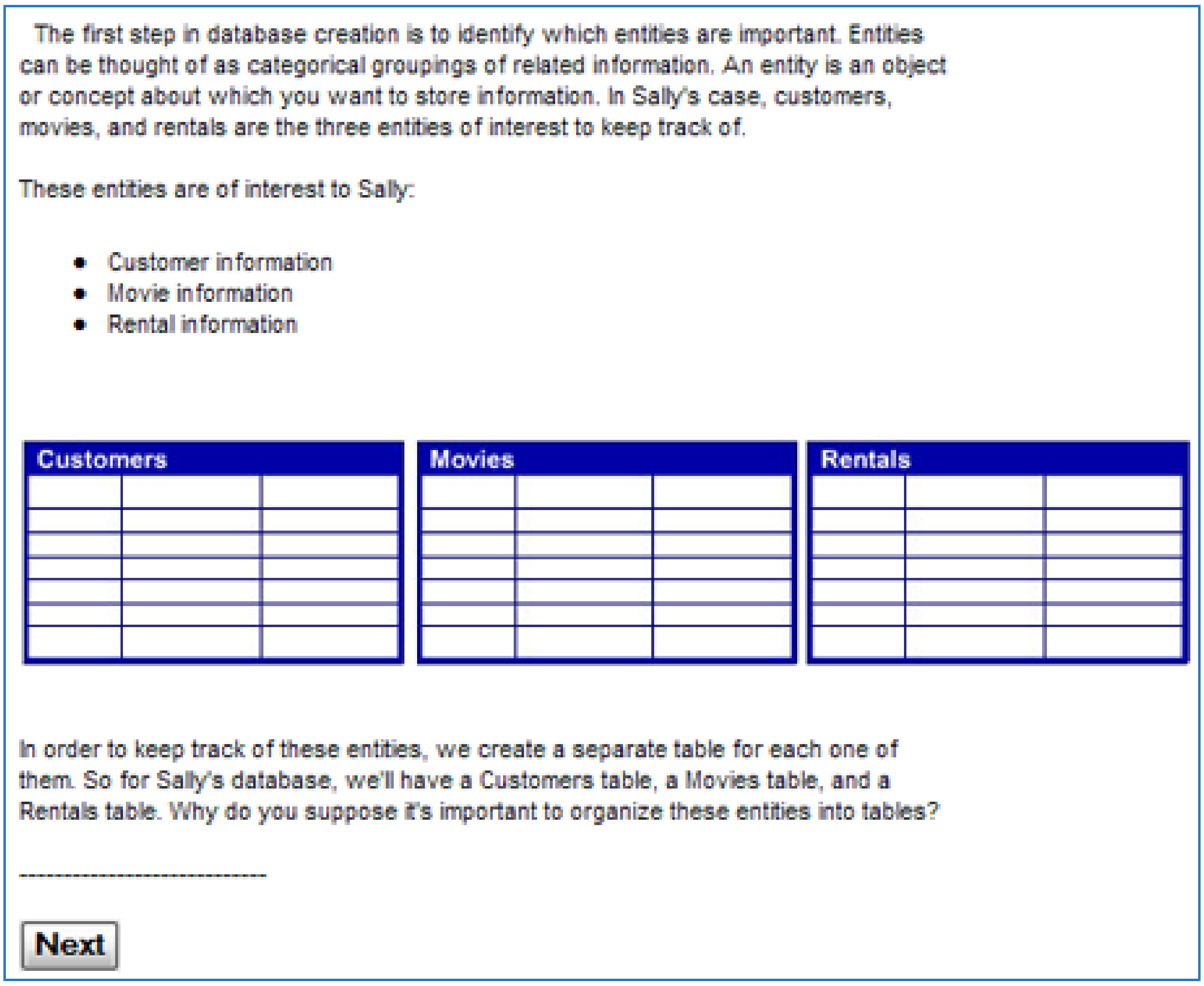
The previous owner kept track of all customer and movie rental information using this Excel spreadsheet. How difficult would it be for Sally to find out how many people from the 85479 zip code rented action movies between February and April of 2003?

Video Store Movie Rentals

	A	B	C	D	E	F	G	H	I	J	K
1											
2											
3	First Name	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	Movie Genre
4	Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male	Under Water	12/16/2005	12/19/2005	Drama
5	Barbara	Smith	5 Thomas Blvd.	85248	555-7896	2/15/1969	Female	The Flying Circus	2/12/2004	2/19/2004	Action
6	Ralph	Johnson	1209 North Street	85445	555-1254	3/27/1978	Male	West Roads	3/17/2005	3/25/2005	Drama
7	Samantha	Walters	45 West 82nd St.	85784	555-8621	2/14/1981	Female	The Albatross	4/11/2006	4/29/2006	Comedy
8	George	Thomas	88 South Bend Ave.	85774	555-7814	5/26/1954	Male	New Winnings	11/13/2003	11/19/2003	Comedy
9	Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female	The Dark Night	10/11/2003	10/11/2003	Romance
10	Bradley	Simpson	215 Mallet Rd.	85042	555-8134	9/10/1932	Male	America	7/8/2004	7/8/2004	Drama
11	Richard	Ross	43 Miners Way	85063	555-9970	11/18/1979	Male	Wild North	9/8/2003	9/18/2003	Horror
12	Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female	Shoestring	1/7/1999	2/19/1999	Comedy
13	Stuart	Locks	6890 Holmes Blvd.	85479	555-0045	7/29/1962	Male	The Old Town	5/14/2005	5/16/2005	Sci-Fi
14	Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male	America	10/6/2002	10/16/2002	Drama
15	Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female	New Winnings	10/19/2002	10/24/2002	Comedy

Although complex queries like this can be answered from a spreadsheet, as you can see, it would be tedious and require a lot of time to do it. Instead, Sally can set up a relational database from her spreadsheet to more easily get the information she needs. Relational databases categorize information into tables to make queries easier.

Figure 7.29: (2) Entities module for a visual user.



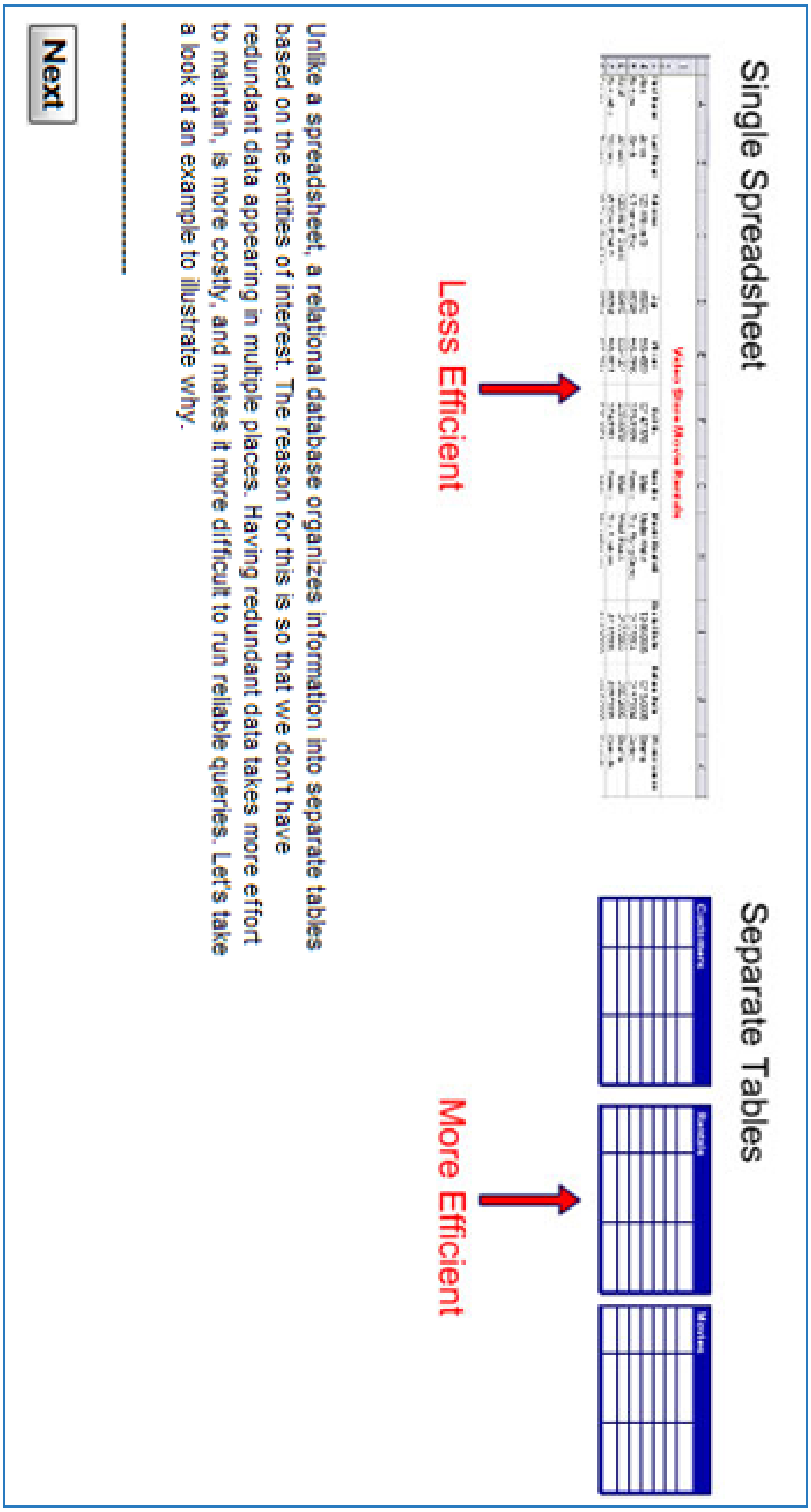
The first step in database creation is to identify which entities are important. Entities can be thought of as categorical groupings of related information. An entity is an object or concept about which you want to store information. In Sally's case, customers, movies, and rentals are the three entities of interest to keep track of.

These entities are of interest to Sally:

- Customer information
- Movie information
- Rental information

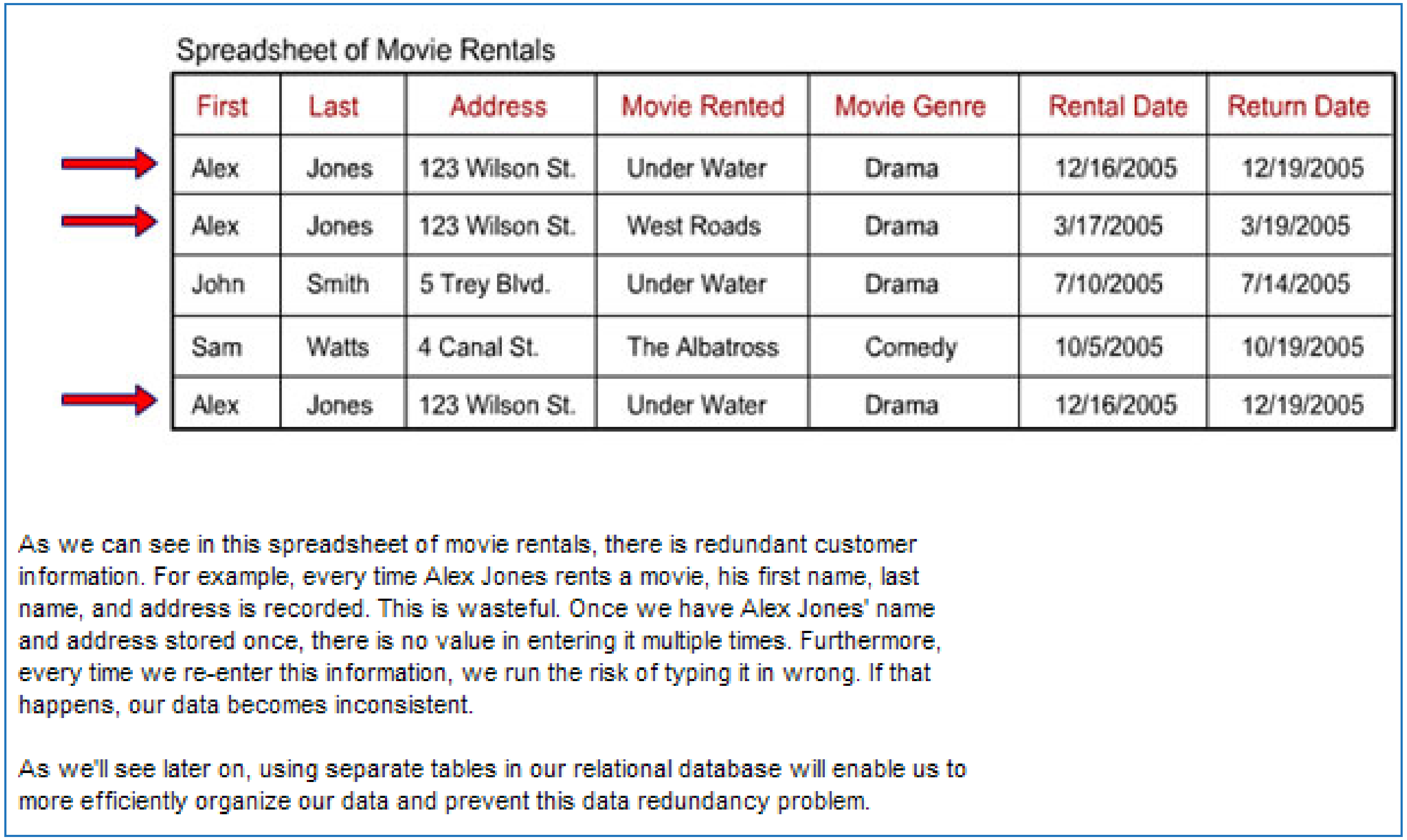
In order to keep track of these entities, we create a separate table for each one of them. So for Sally's database, we'll have a Customers table, a Movies table, and a Rentals table. Why do you suppose it's important to organize these entities into tables?

Figure 7.30: (3) Entities module for a visual user.



Unlike a spreadsheet, a relational database organizes information into separate tables based on the entities of interest. The reason for this is so that we don't have redundant data appearing in multiple places. Having redundant data takes more effort to maintain, is more costly, and makes it more difficult to run reliable queries. Let's take a look at an example to illustrate why.

Figure 7.31: (4) Entities module for a visual user.



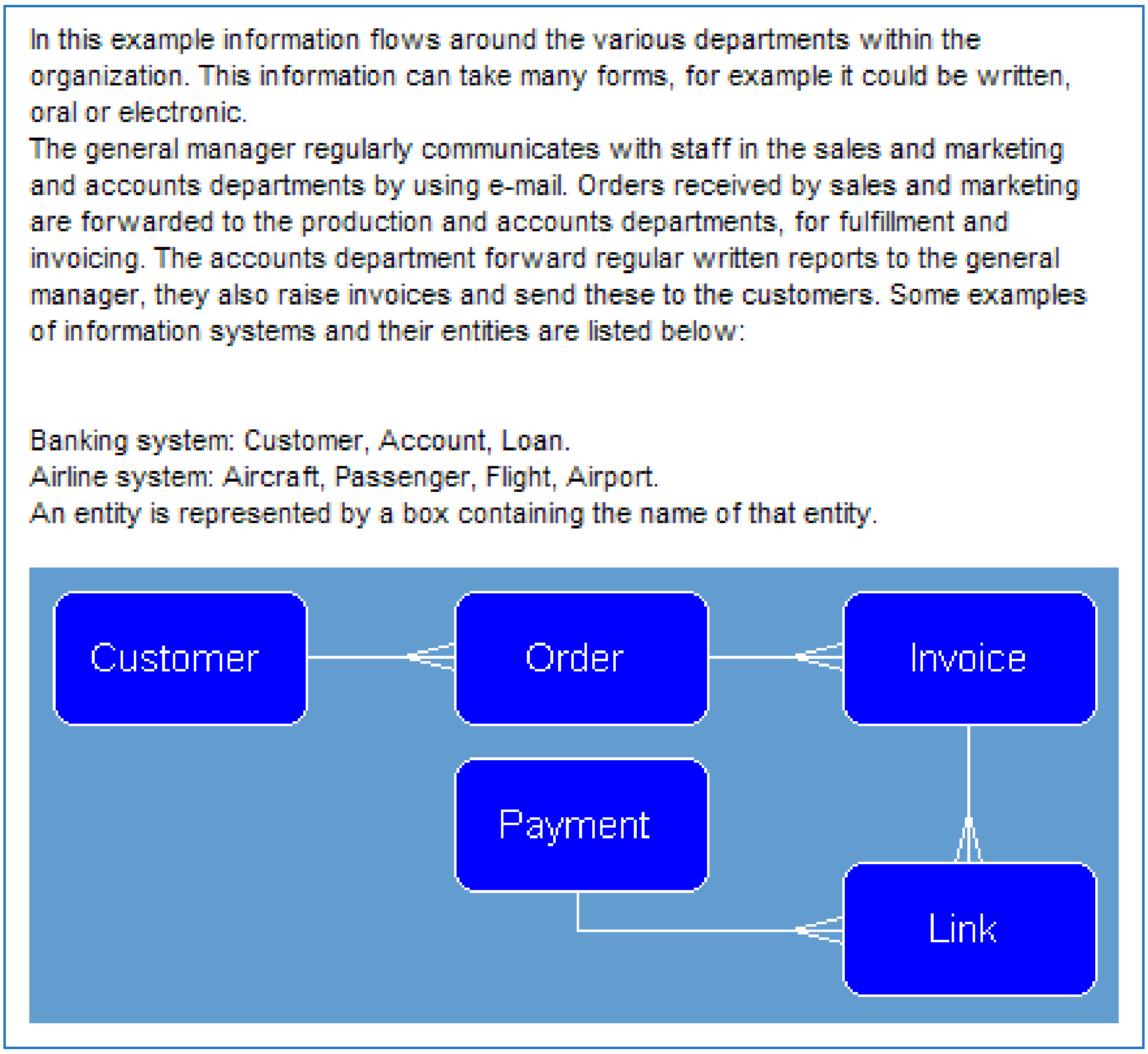
Spreadsheet of Movie Rentals

First	Last	Address	Movie Rented	Movie Genre	Rental Date	Return Date
Alex	Jones	123 Wilson St.	Under Water	Drama	12/16/2005	12/19/2005
Alex	Jones	123 Wilson St.	West Roads	Drama	3/17/2005	3/19/2005
John	Smith	5 Trey Blvd.	Under Water	Drama	7/10/2005	7/14/2005
Sam	Watts	4 Canal St.	The Albatross	Comedy	10/5/2005	10/19/2005
Alex	Jones	123 Wilson St.	Under Water	Drama	12/16/2005	12/19/2005

As we can see in this spreadsheet of movie rentals, there is redundant customer information. For example, every time Alex Jones rents a movie, his first name, last name, and address is recorded. This is wasteful. Once we have Alex Jones' name and address stored once, there is no value in entering it multiple times. Furthermore, every time we re-enter this information, we run the risk of typing it in wrong. If that happens, our data becomes inconsistent.

As we'll see later on, using separate tables in our relational database will enable us to more efficiently organize our data and prevent this data redundancy problem.

Figure 7.32: (5) Entities module for a visual user.


In this example information flows around the various departments within the organization. This information can take many forms, for example it could be written, oral or electronic.
The general manager regularly communicates with staff in the sales and marketing and accounts departments by using e-mail. Orders received by sales and marketing are forwarded to the production and accounts departments, for fulfillment and invoicing. The accounts department forward regular written reports to the general manager, they also raise invoices and send these to the customers. Some examples of information systems and their entities are listed below:

Banking system: Customer, Account, Loan.
Airline system: Aircraft, Passenger, Flight, Airport.
An entity is represented by a box containing the name of that entity.

Figure 7.33: Document describing example of entities for a visual user.

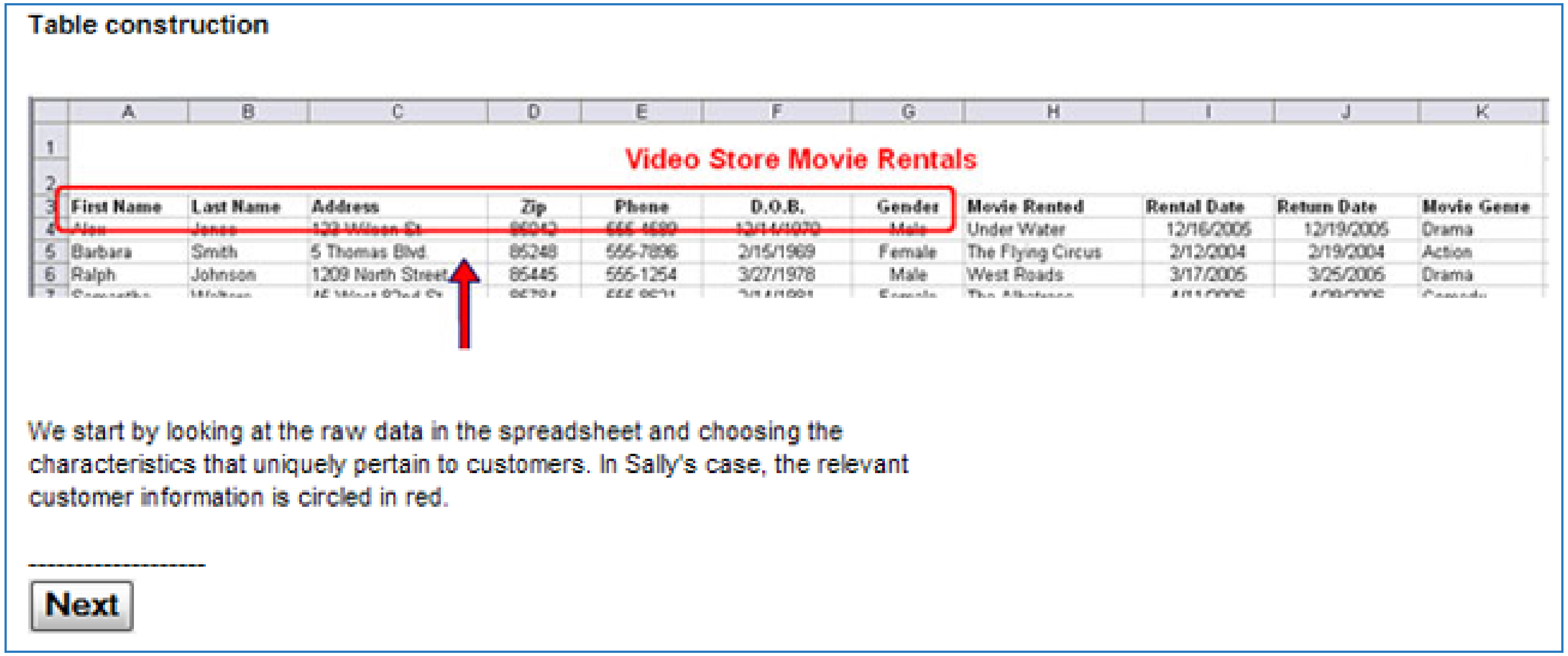
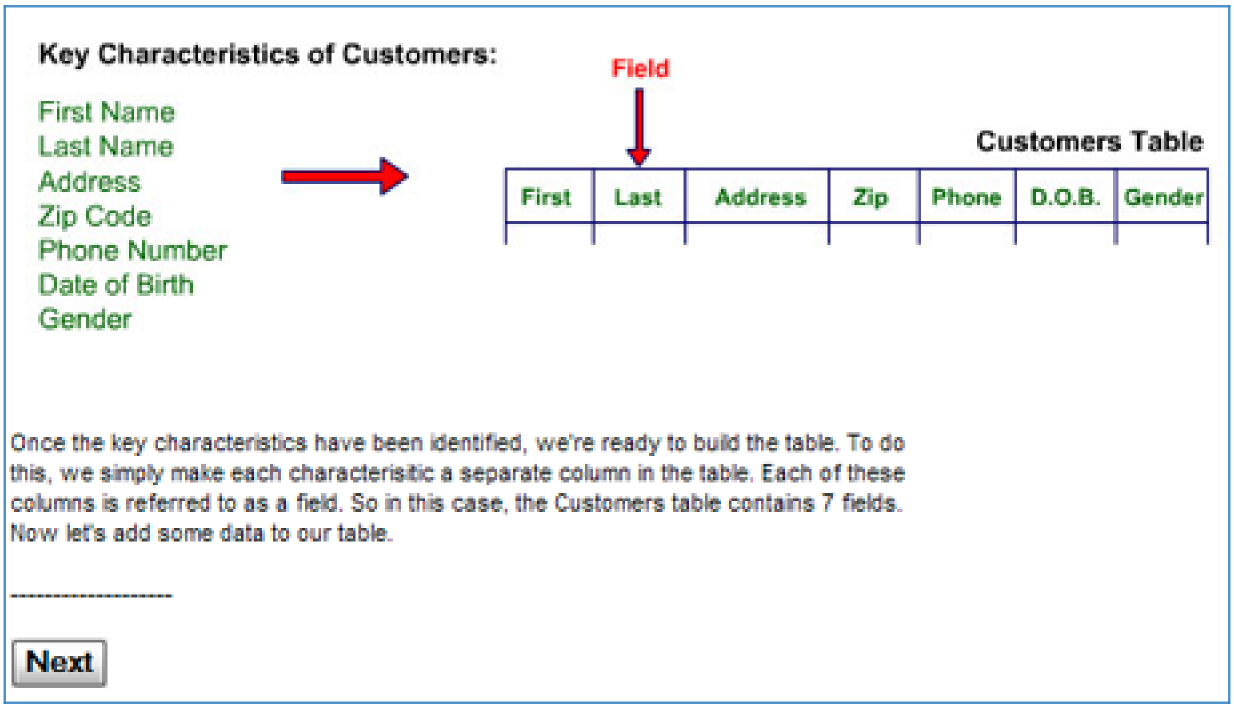


Table construction

	A	B	C	D	E	F	G	H	I	J	K
1						Video Store Movie Rentals					
2											
3	First Name	Last Name	Address	Zip	Phone	D.O.B.	Gender	Movie Rented	Rental Date	Return Date	Movie Genre
4	Alex	Jones	123 Wilson St	85042	555-4589	12/14/1970	Male	Under Water	12/16/2005	12/19/2005	Drama
5	Barbara	Smith	5 Thomas Blvd.	85248	555-7896	2/15/1969	Female	The Flying Circus	2/12/2004	2/19/2004	Action
6	Ralph	Johnson	1209 North Street	85445	555-1254	3/27/1978	Male	West Roads	3/17/2005	3/25/2005	Drama
7	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

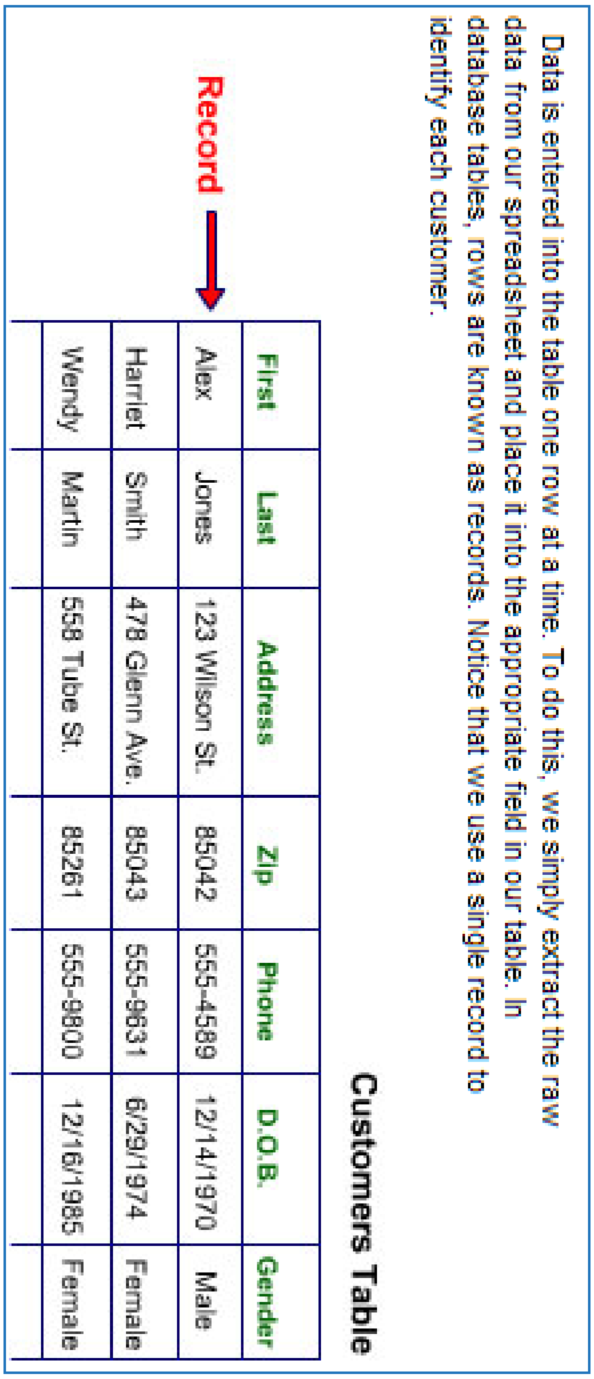
We start by looking at the raw data in the spreadsheet and choosing the characteristics that uniquely pertain to customers. In Sally's case, the relevant customer information is circled in red.

Figure 7.34: (1) Table construction module for a visual user.



Once the key characteristics have been identified, we're ready to build the table. To do this, we simply make each characterisitic a separate column in the table. Each of these columns is referred to as a field. So in this case, the Customers table contains 7 fields. Now let's add some data to our table.

Figure 7.35: (2) Table construction module for a visual user.



Customers Table

Data is entered into the table one row at a time. To do this, we simply extract the raw data from our spreadsheet and place it into the appropriate field in our table. In database tables, rows are known as records. Notice that we use a single record to identify each customer.

First	Last	Address	Zip	Phone	D.O.B.	Gender
Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male
Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female
Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female

Figure 7.36: (3) Table construction module for a visual user.

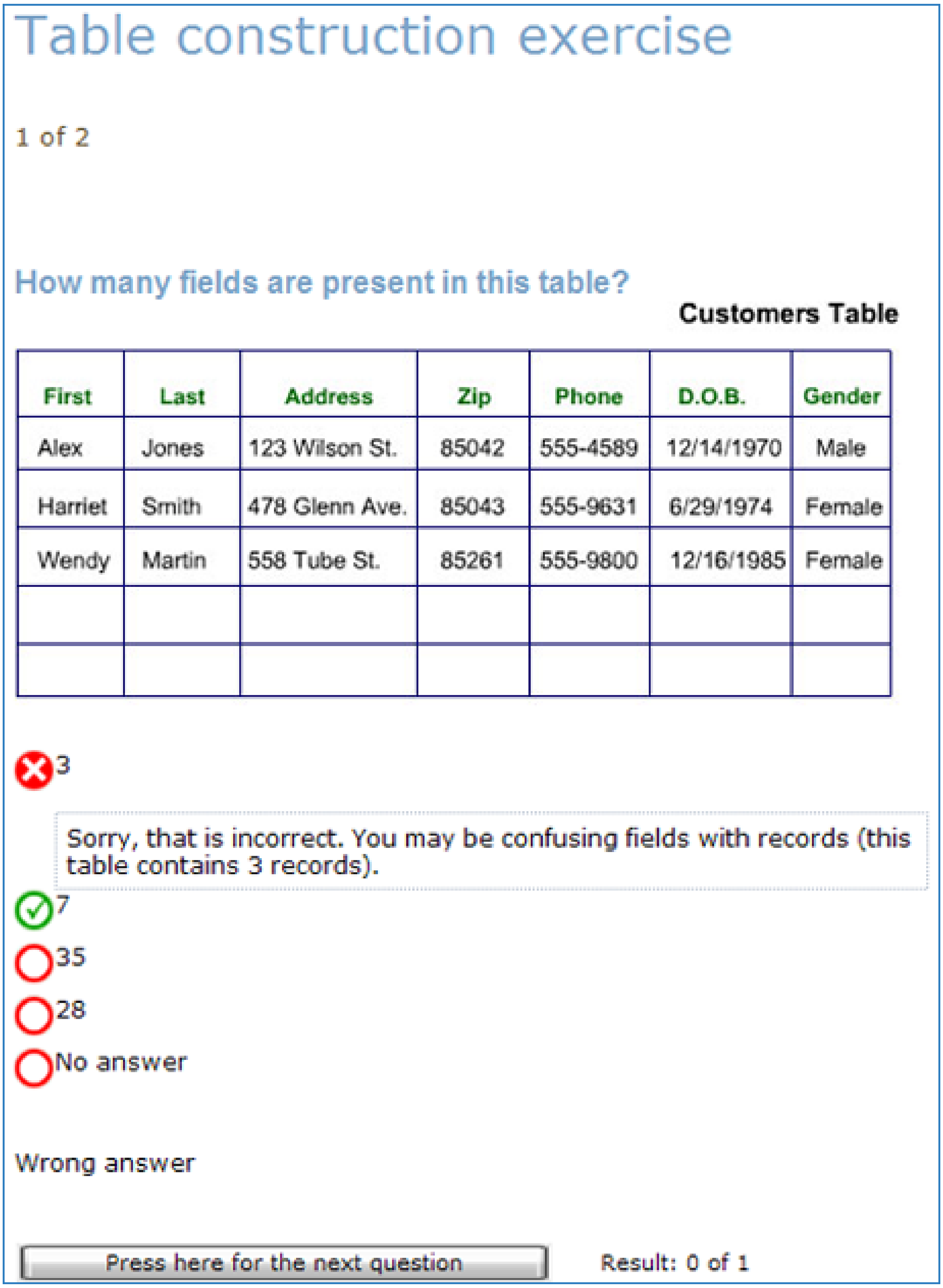


Table construction exercise

1 of 2

How many fields are present in this table?

Customers Table

First	Last	Address	Zip	Phone	D.O.B.	Gender
Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male
Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female
Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female

✗ 3

Sorry, that is incorrect. You may be confusing fields with records (this table contains 3 records).

✓ 7

○ 35

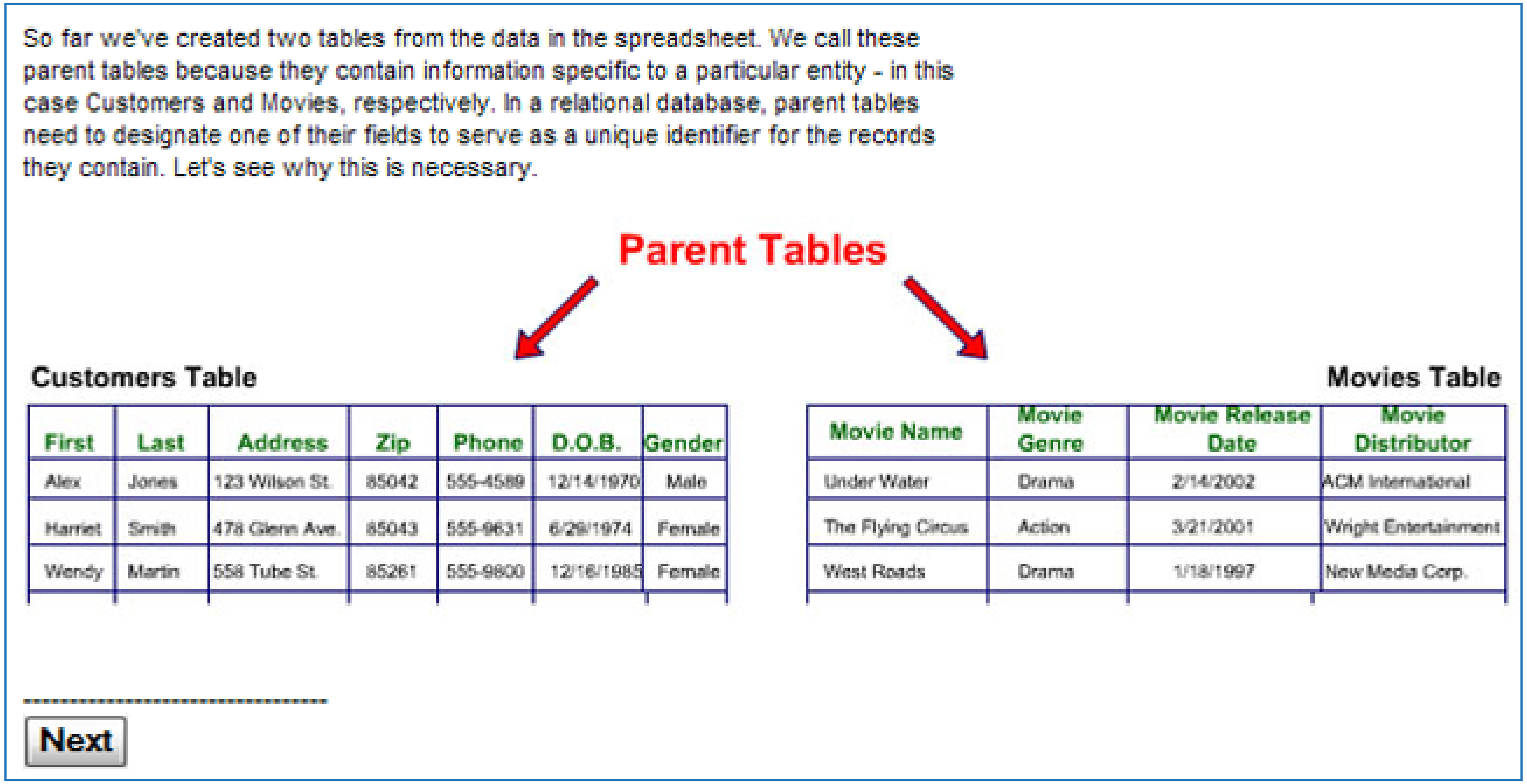
○ 28

○ No answer

Wrong answer

Result: 0 of 1

Figure 7.37: (4) An exercise in table construction module for a visual user. This example shows that the user has given a wrong answer and the system shows the correct answer and gives feedback.



So far we've created two tables from the data in the spreadsheet. We call these parent tables because they contain information specific to a particular entity - in this case Customers and Movies, respectively. In a relational database, parent tables need to designate one of their fields to serve as a unique identifier for the records they contain. Let's see why this is necessary.

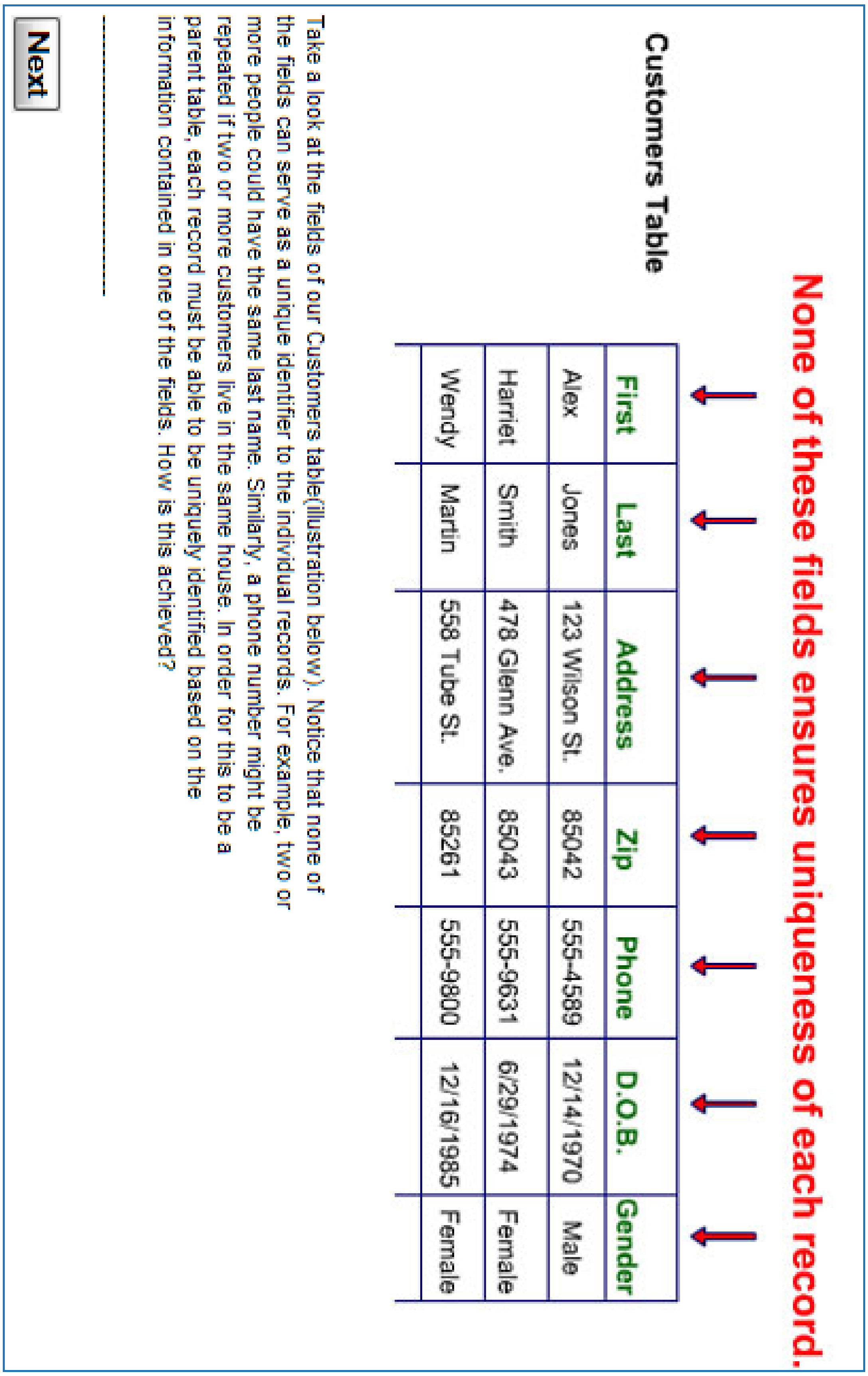
Customers Table

First	Last	Address	Zip	Phone	D.O.B.	Gender
Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male
Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female
Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female

Movies Table

Movie Name	Movie Genre	Movie Release Date	Movie Distributor
Under Water	Drama	2/14/2002	ACM International
The Flying Circus	Action	3/21/2001	Wright Entertainment
West Roads	Drama	1/18/1997	New Media Corp.

Figure 7.38: (1) “Parent and child tables” module for a visual user.



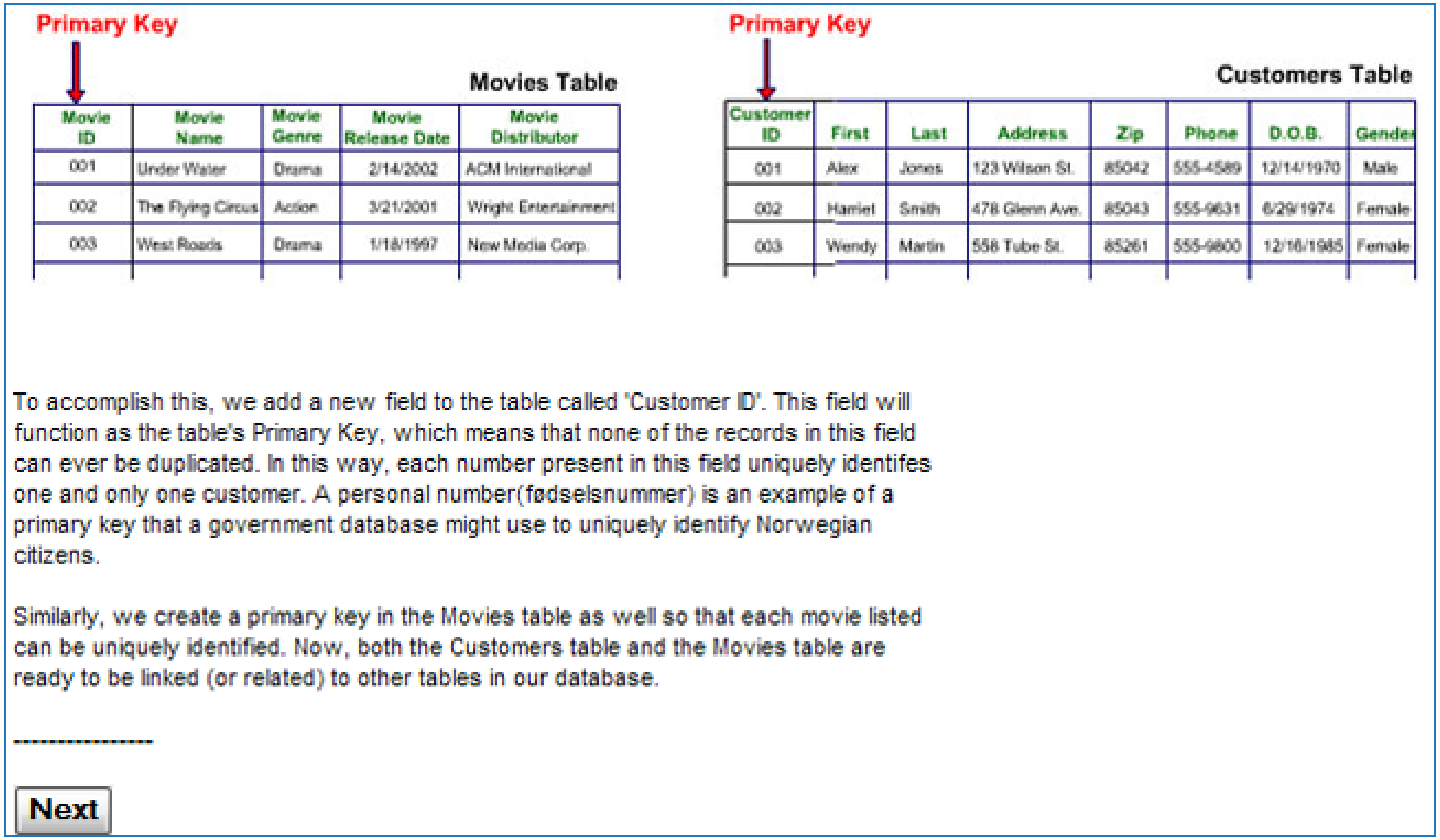
None of these fields ensures uniqueness of each record.

Customers Table

First	Last	Address	Zip	Phone	D.O.B.	Gender
Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male
Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female
Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female

Take a look at the fields of our Customers table(illustration below). Notice that none of the fields can serve as a unique identifier to the individual records. For example, two or more people could have the same last name. Similarly, a phone number might be repeated if two or more customers live in the same house. In order for this to be a parent table, each record must be able to be uniquely identified based on the information contained in one of the fields. How is this achieved?

Figure 7.39: (2) “Parent and child tables” module for a visual user.



Movies Table

Movie ID	Movie Name	Movie Genre	Movie Release Date	Movie Distributor
001	Under Water	Drama	2/14/2002	ACM International
002	The Flying Circus	Action	3/21/2001	Wright Entertainment
003	West Roads	Drama	1/18/1997	New Media Corp.

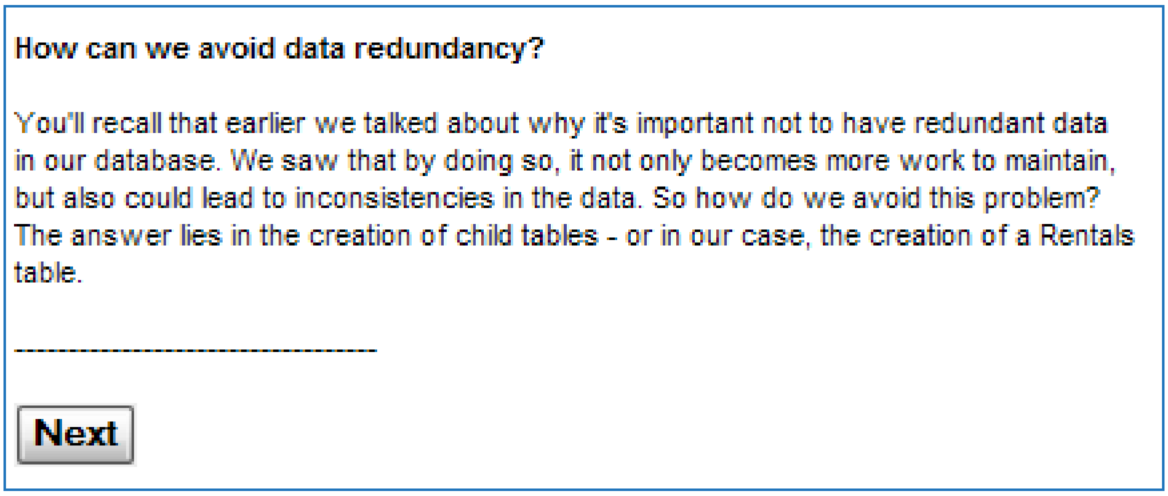
Customers Table

Customer ID	First	Last	Address	Zip	Phone	D.O.B.	Gender
001	Alex	Jones	123 Wilson St.	85042	555-4589	12/14/1970	Male
002	Harriet	Smith	478 Glenn Ave.	85043	555-9631	6/29/1974	Female
003	Wendy	Martin	558 Tube St.	85261	555-9800	12/16/1985	Female

To accomplish this, we add a new field to the table called 'Customer ID'. This field will function as the table's Primary Key, which means that none of the records in this field can ever be duplicated. In this way, each number present in this field uniquely identifes one and only one customer. A personal number(fødselsnummer) is an example of a primary key that a government database might use to uniquely identify Norwegian citizens.

Similarly, we create a primary key in the Movies table as well so that each movie listed can be uniquely identified. Now, both the Customers table and the Movies table are ready to be linked (or related) to other tables in our database.

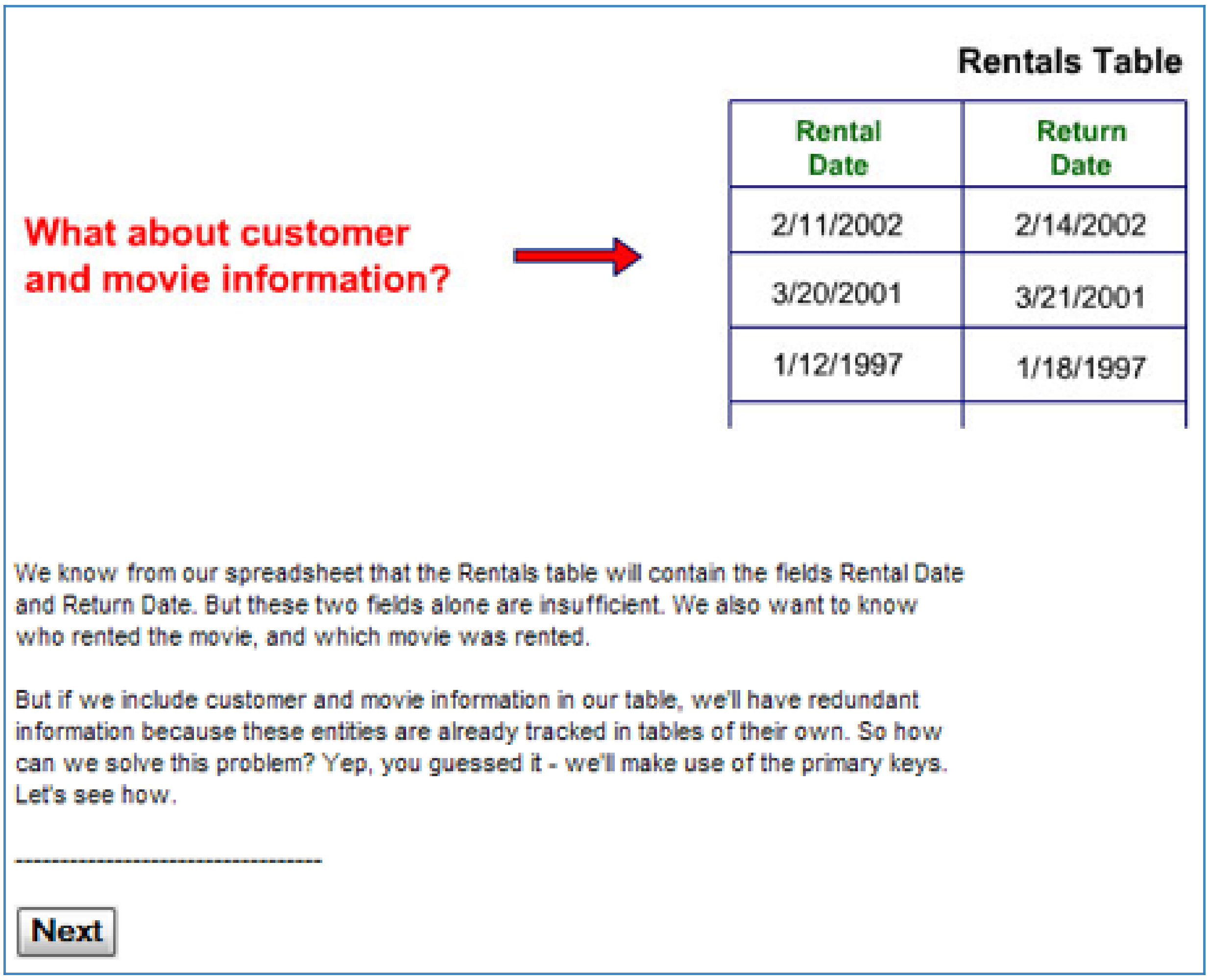
Figure 7.40: (3) “Parent and child tables” module for a visual user.



How can we avoid data redundancy?

You'll recall that earlier we talked about why it's important not to have redundant data in our database. We saw that by doing so, it not only becomes more work to maintain, but also could lead to inconsistencies in the data. So how do we avoid this problem? The answer lies in the creation of child tables - or in our case, the creation of a Rentals table.

Figure 7.41: (4) “Parent and child tables” module for a visual user.



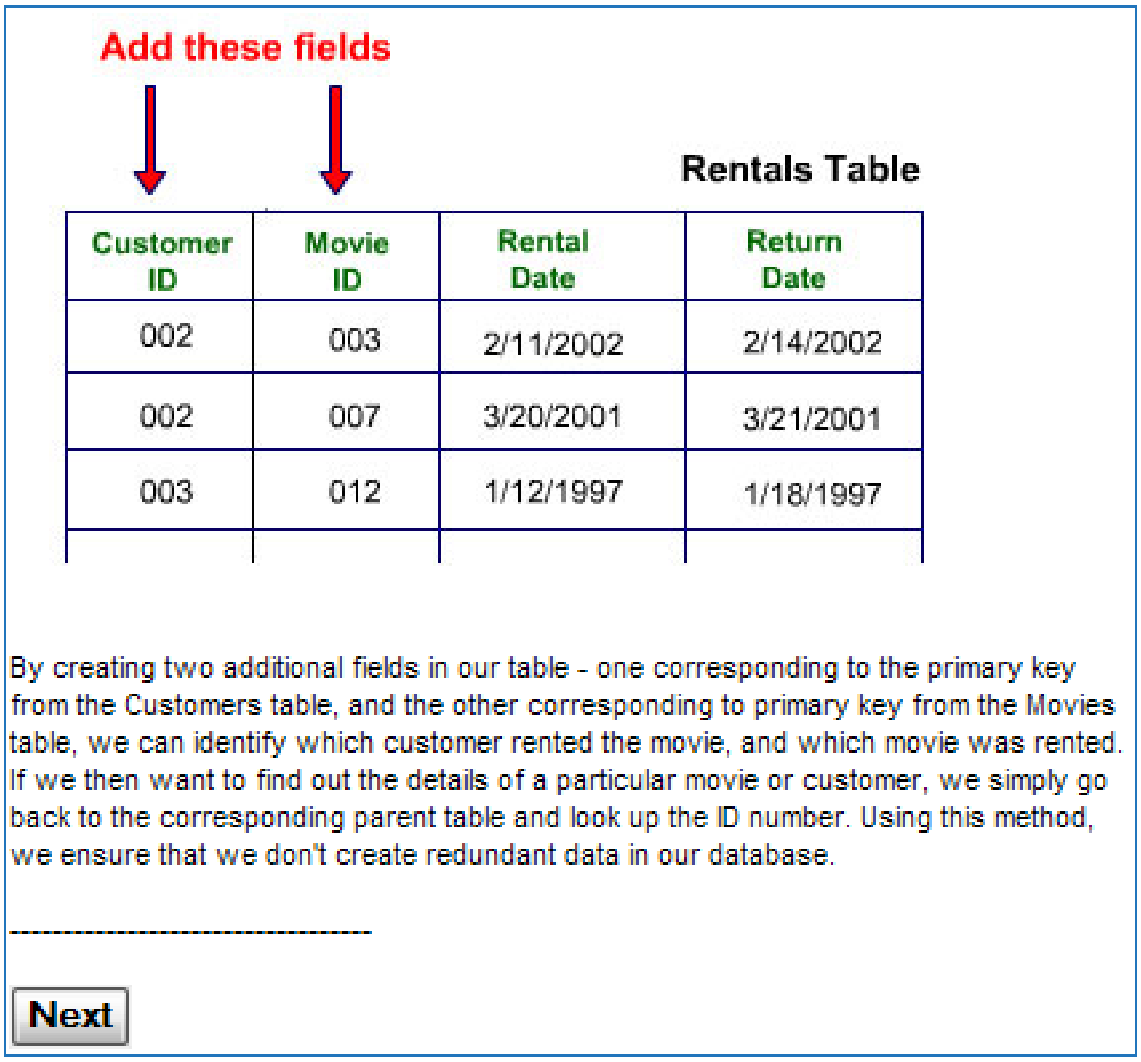
Rentals Table

Rental Date	Return Date
2/11/2002	2/14/2002
3/20/2001	3/21/2001
1/12/1997	1/18/1997

We know from our spreadsheet that the Rentals table will contain the fields Rental Date and Return Date. But these two fields alone are insufficient. We also want to know who rented the movie, and which movie was rented.

But if we include customer and movie information in our table, we'll have redundant information because these entities are already tracked in tables of their own. So how can we solve this problem? Yep, you guessed it - we'll make use of the primary keys. Let's see how.

Figure 7.42: (5) “Parent and child tables” module for a visual user.

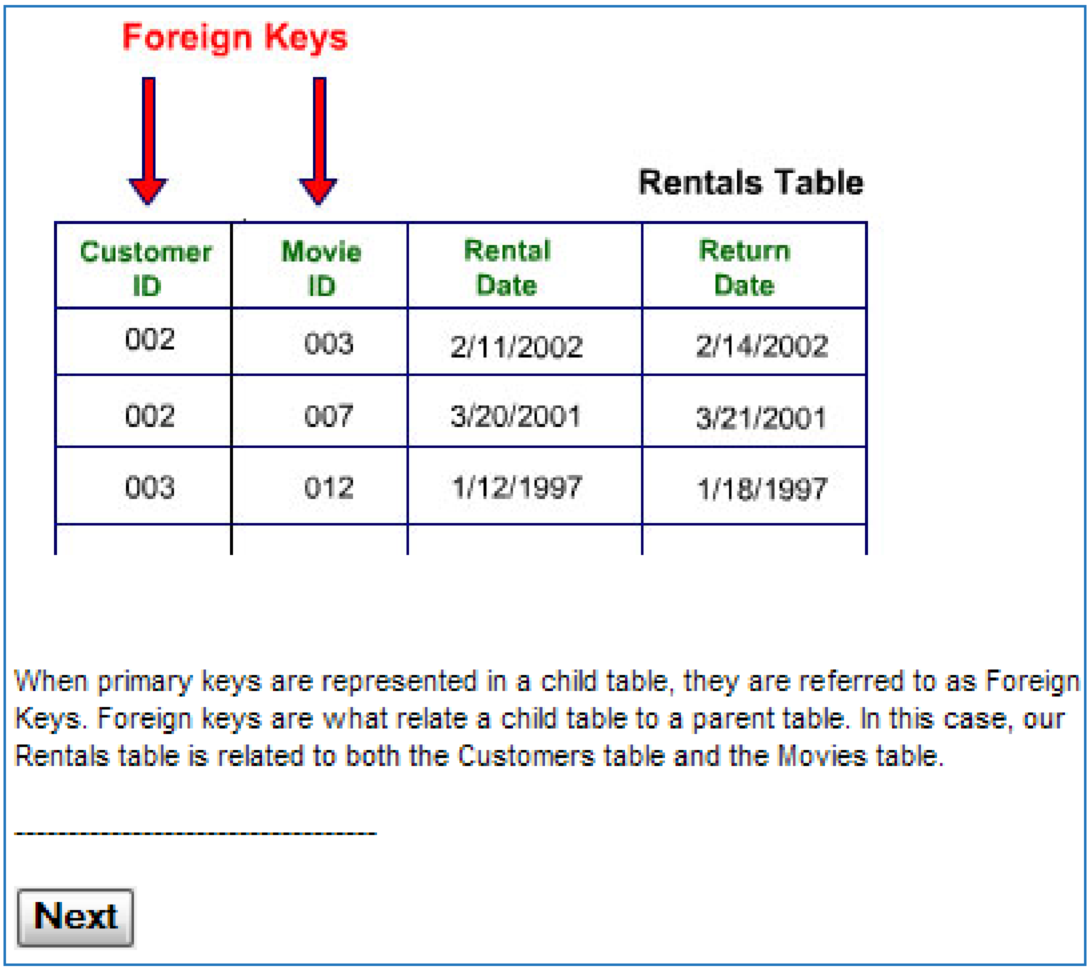


Rentals Table

Customer ID	Movie ID	Rental Date	Return Date
002	003	2/11/2002	2/14/2002
002	007	3/20/2001	3/21/2001
003	012	1/12/1997	1/18/1997

By creating two additional fields in our table - one corresponding to the primary key from the Customers table, and the other corresponding to primary key from the Movies table, we can identify which customer rented the movie, and which movie was rented. If we then want to find out the details of a particular movie or customer, we simply go back to the corresponding parent table and look up the ID number. Using this method, we ensure that we don't create redundant data in our database.

Figure 7.43: (6) "Parent and child tables" module for a visual user.

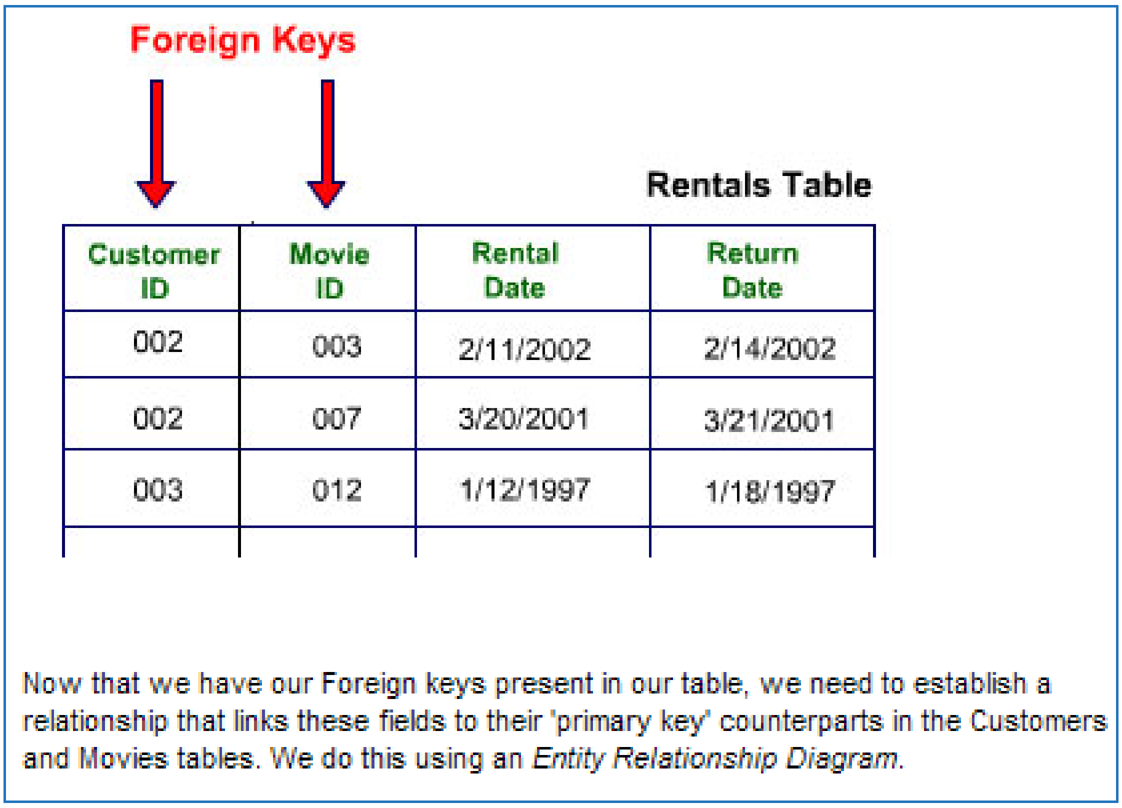


Rentals Table

Customer ID	Movie ID	Rental Date	Return Date
002	003	2/11/2002	2/14/2002
002	007	3/20/2001	3/21/2001
003	012	1/12/1997	1/18/1997

When primary keys are represented in a child table, they are referred to as Foreign Keys. Foreign keys are what relate a child table to a parent table. In this case, our Rentals table is related to both the Customers table and the Movies table.

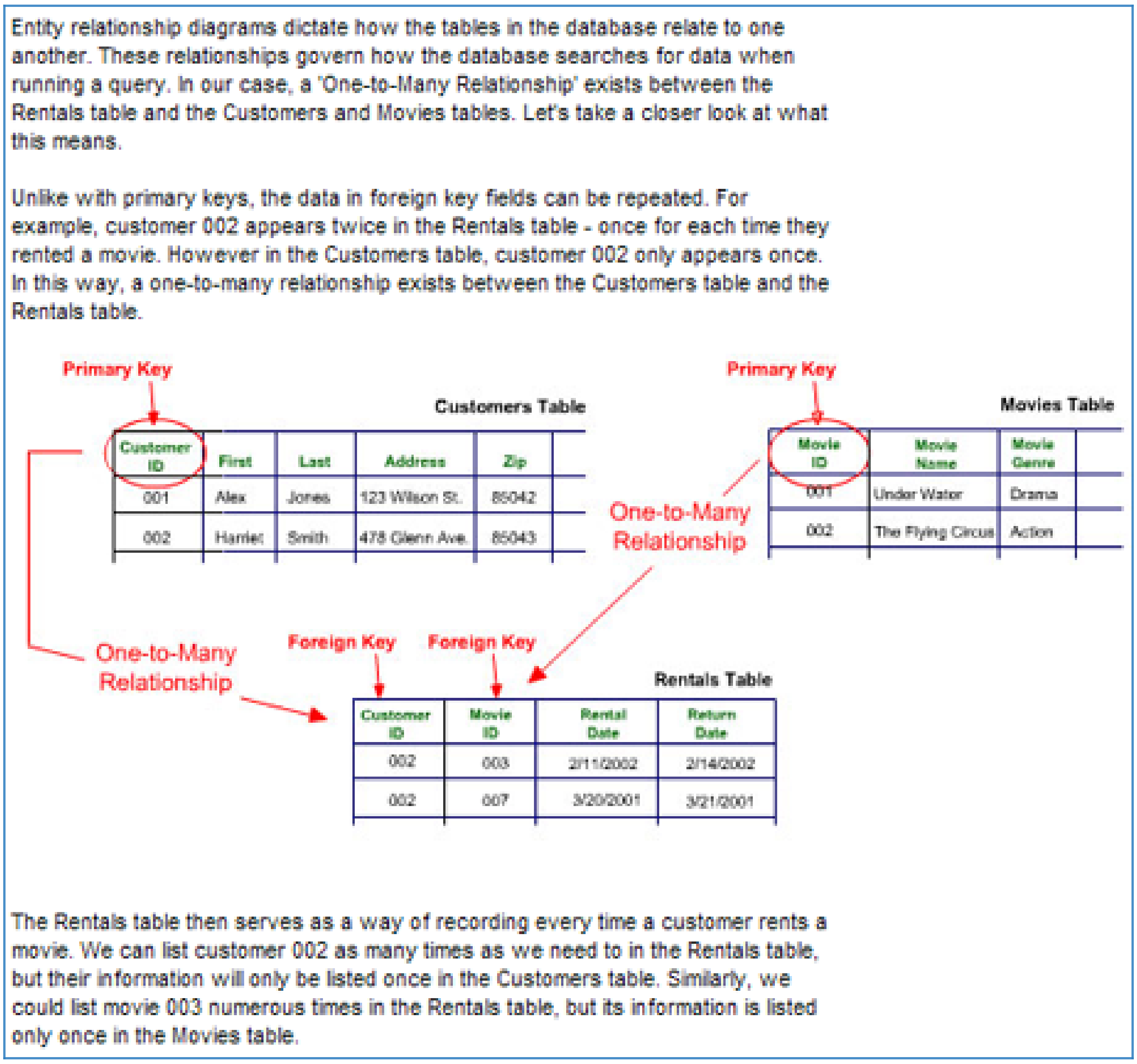
Figure 7.44: (7) "Parent and child tables" module for a visual user.



Customer ID	Movie ID	Rental Date	Return Date
002	003	2/11/2002	2/14/2002
002	007	3/20/2001	3/21/2001
003	012	1/12/1997	1/18/1997

Now that we have our Foreign keys present in our table, we need to establish a relationship that links these fields to their 'primary key' counterparts in the Customers and Movies tables. We do this using an *Entity Relationship Diagram*.

Figure 7.45: (8) “Parent and child tables” module for a visual user.



Entity relationship diagrams dictate how the tables in the database relate to one another. These relationships govern how the database searches for data when running a query. In our case, a 'One-to-Many Relationship' exists between the Rentals table and the Customers and Movies tables. Let's take a closer look at what this means.

Unlike with primary keys, the data in foreign key fields can be repeated. For example, customer 002 appears twice in the Rentals table - once for each time they rented a movie. However in the Customers table, customer 002 only appears once. In this way, a one-to-many relationship exists between the Customers table and the Rentals table.

Customers Table

Customer ID	First	Last	Address	Zip
001	Alex	Jones	123 Wilson St.	85042
002	Harriet	Smith	478 Glenn Ave.	85043

Movies Table

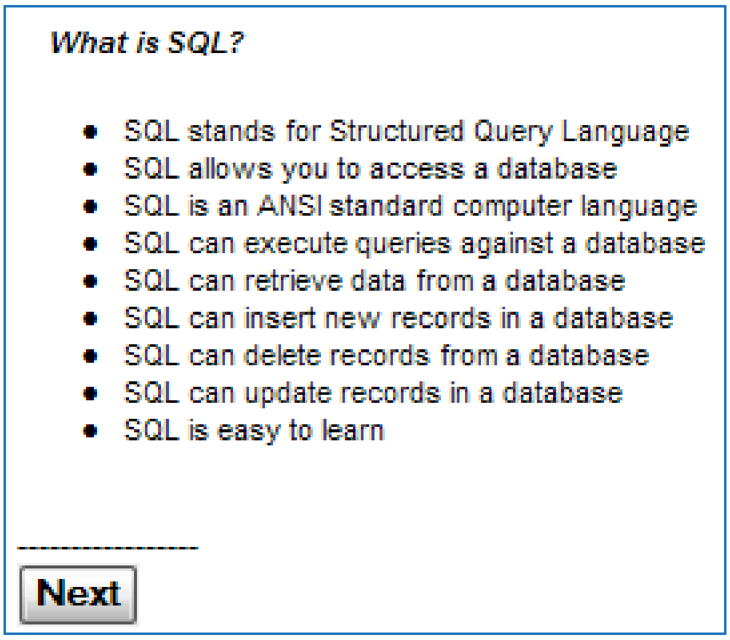
Movie ID	Movie Name	Movie Genre
001	Under Water	Drama
002	The Flying Circus	Action

Rentals Table

Customer ID	Movie ID	Rental Date	Return Date
002	003	2/11/2002	2/14/2002
002	007	3/20/2001	3/21/2001

The Rentals table then serves as a way of recording every time a customer rents a movie. We can list customer 002 as many times as we need to in the Rentals table, but their information will only be listed once in the Customers table. Similarly, we could list movie 003 numerous times in the Rentals table, but its information is listed only once in the Movies table.

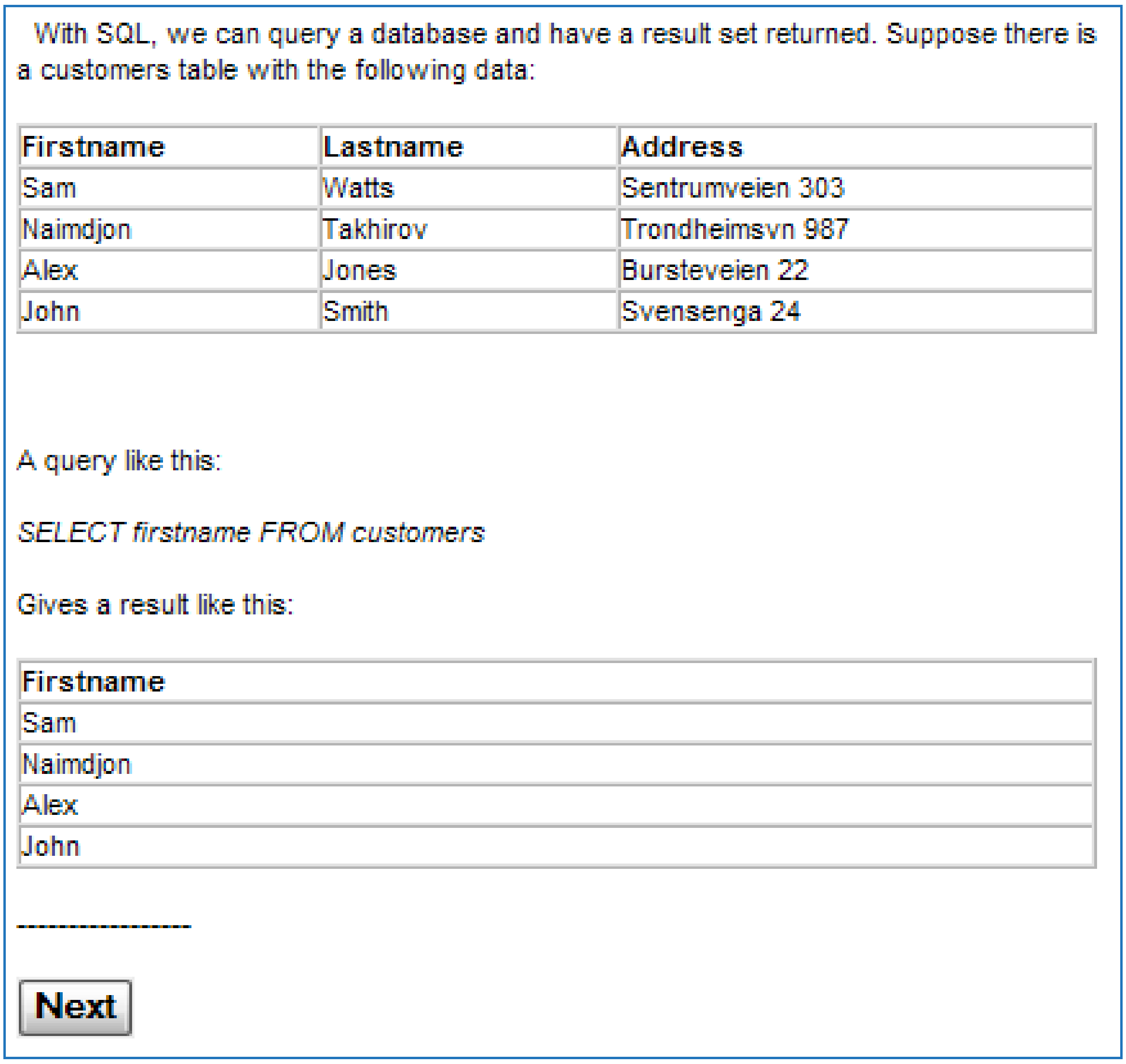
Figure 7.46: (1) “Entity Relationship diagrams” module for a visual user.



What is SQL?

- SQL stands for Structured Query Language
- SQL allows you to access a database
- SQL is an ANSI standard computer language
- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert new records in a database
- SQL can delete records from a database
- SQL can update records in a database
- SQL is easy to learn

Figure 7.47: (1) “SQL Intro” module for a visual user.



With SQL, we can query a database and have a result set returned. Suppose there is a customers table with the following data:

Firstname	Lastname	Address
Sam	Watts	Sentrumveien 303
Naimdjon	Takhirov	Trondheimsvn 987
Alex	Jones	Bursteveien 22
John	Smith	Svensenga 24

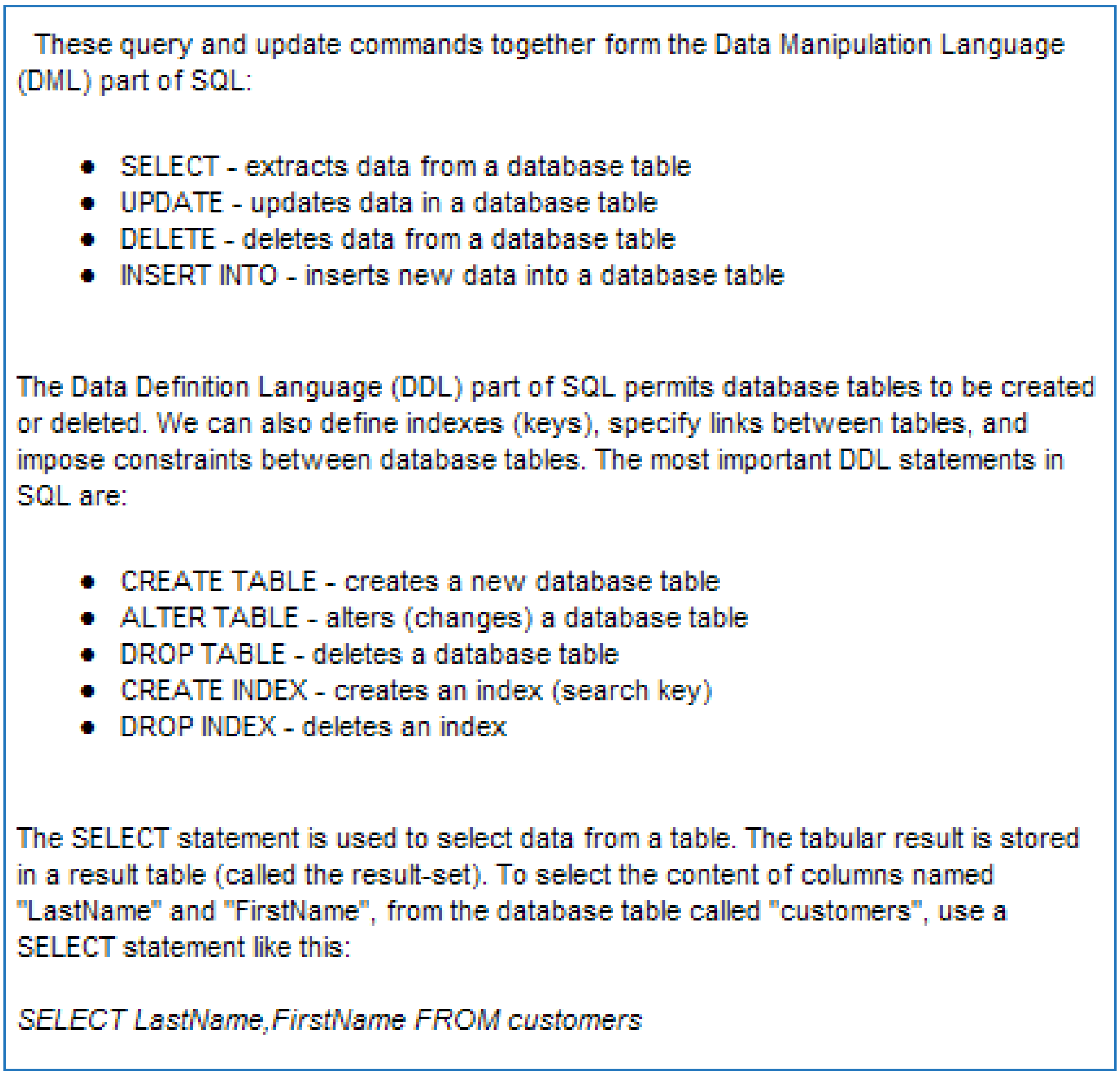
A query like this:

SELECT firstname FROM customers

Gives a result like this:

Firstname
Sam
Naimdjon
Alex
John

Figure 7.48: (2) "SQL Intro" module for a visual user.


These query and update commands together form the Data Manipulation Language (DML) part of SQL:

- SELECT - extracts data from a database table
- UPDATE - updates data in a database table
- DELETE - deletes data from a database table
- INSERT INTO - inserts new data into a database table

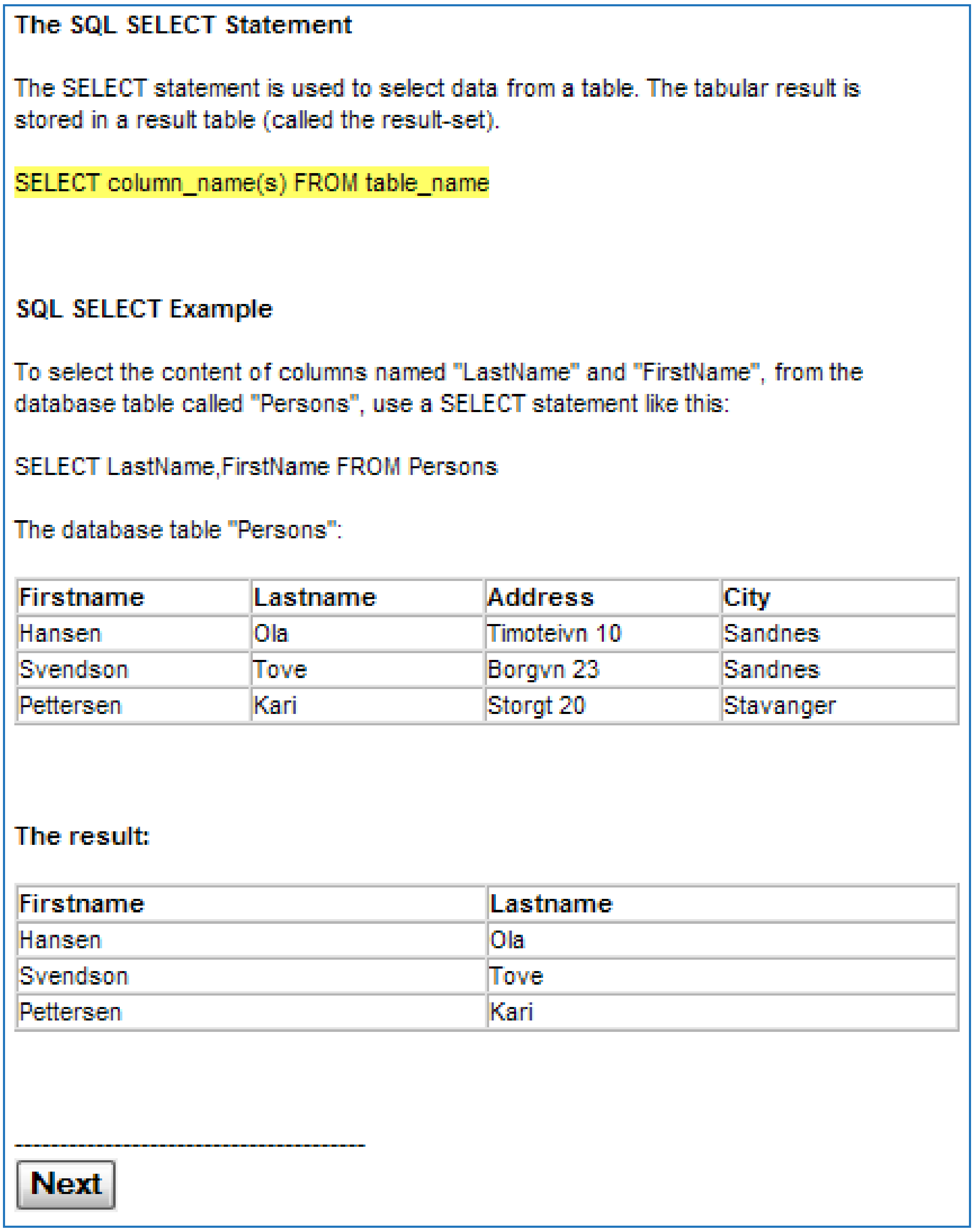
The Data Definition Language (DDL) part of SQL permits database tables to be created or deleted. We can also define indexes (keys), specify links between tables, and impose constraints between database tables. The most important DDL statements in SQL are:

- CREATE TABLE - creates a new database table
- ALTER TABLE - alters (changes) a database table
- DROP TABLE - deletes a database table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

The SELECT statement is used to select data from a table. The tabular result is stored in a result table (called the result-set). To select the content of columns named "LastName" and "FirstName", from the database table called "customers", use a SELECT statement like this:

SELECT LastName,FirstName FROM customers

Figure 7.49: (3) “SQL Intro” module for a visual user.



The SQL SELECT Statement

The SELECT statement is used to select data from a table. The tabular result is stored in a result table (called the result-set).

SELECT column_name(s) FROM table_name

SQL SELECT Example

To select the content of columns named "LastName" and "FirstName", from the database table called "Persons", use a SELECT statement like this:

SELECT LastName,FirstName FROM Persons

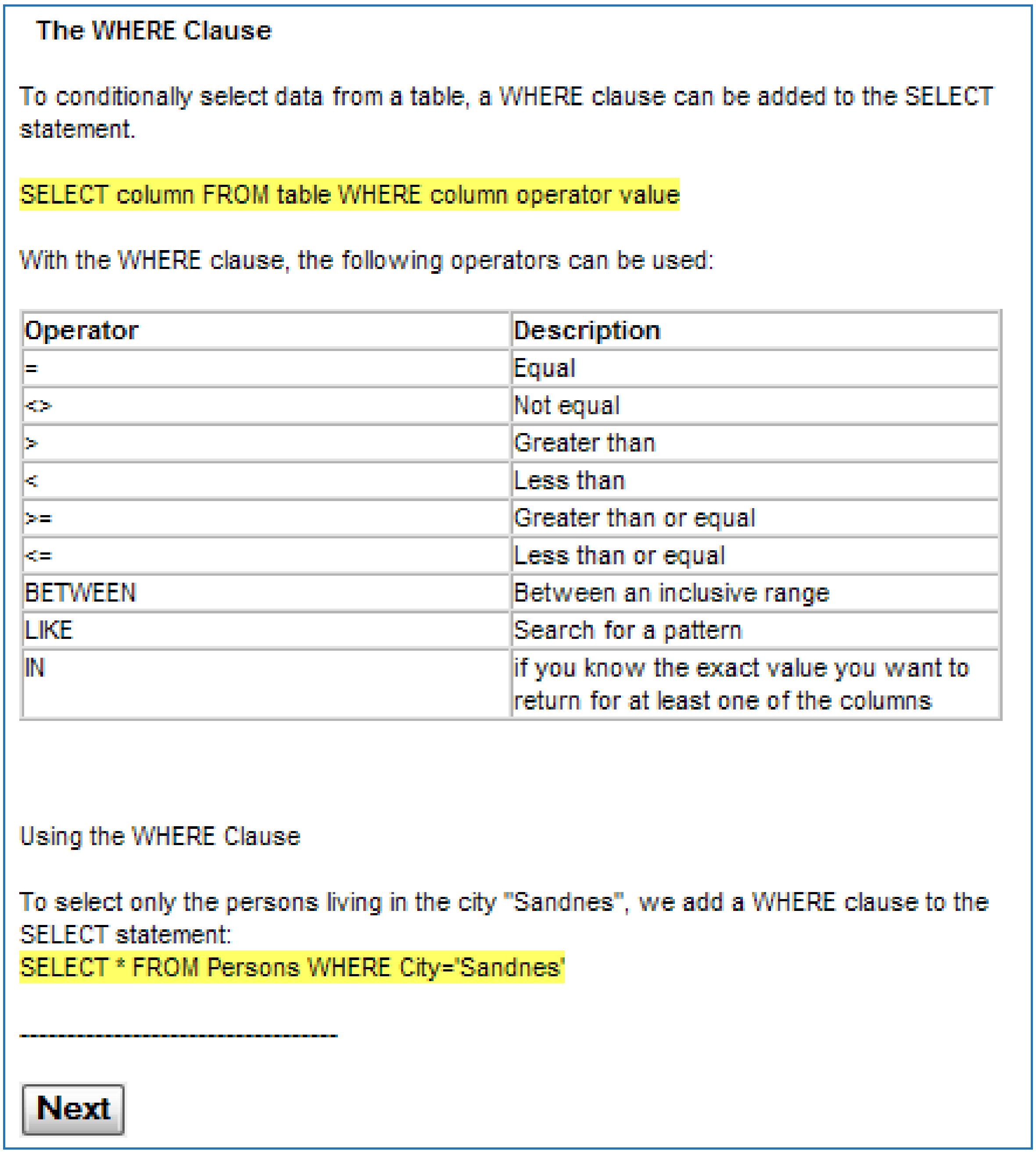
The database table "Persons":

Firstname	Lastname	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Svendson	Tove	Borgvn 23	Sandnes
Pettersen	Kari	Storgt 20	Stavanger

The result:

Firstname	Lastname
Hansen	Ola
Svendson	Tove
Pettersen	Kari

Figure 7.50: (1) “SQL Select” module for a visual user.



The WHERE Clause

To conditionally select data from a table, a WHERE clause can be added to the SELECT statement.

SELECT column FROM table WHERE column operator value

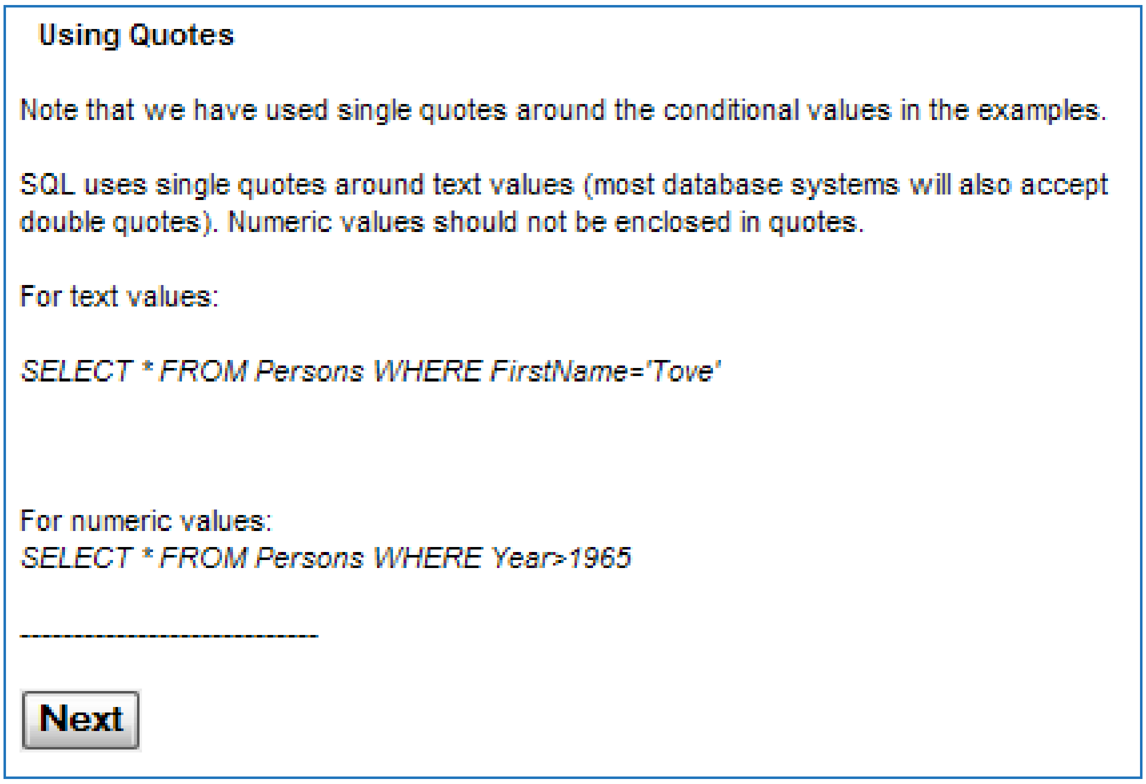
With the WHERE clause, the following operators can be used:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	if you know the exact value you want to return for at least one of the columns

Using the WHERE Clause

To select only the persons living in the city "Sandnes", we add a WHERE clause to the SELECT statement:
SELECT * FROM Persons WHERE City='Sandnes'

Figure 7.51: (2) “SQL Select” module for a visual user.


Using Quotes

Note that we have used single quotes around the conditional values in the examples.

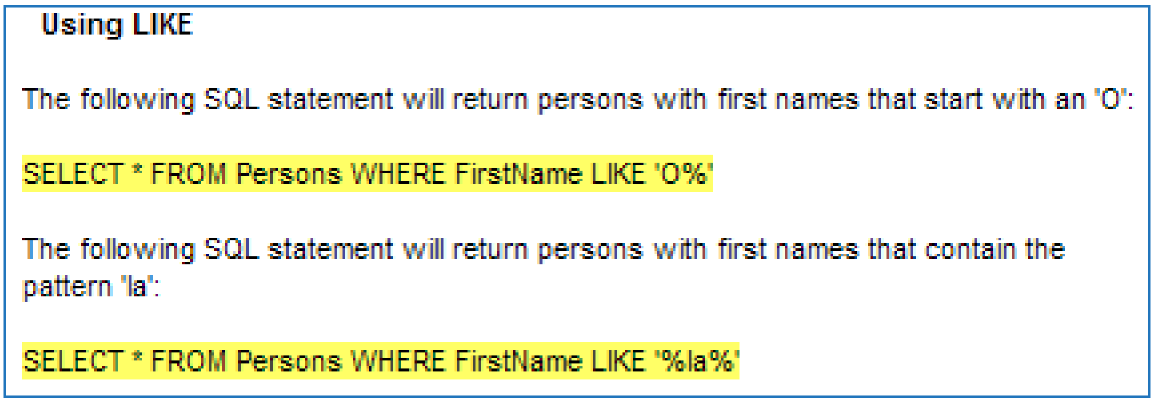
SQL uses single quotes around text values (most database systems will also accept double quotes). Numeric values should not be enclosed in quotes.

For text values:

*SELECT * FROM Persons WHERE FirstName='Tove'*

For numeric values:
*SELECT * FROM Persons WHERE Year>1965*

Figure 7.52: (3) “SQL Select” module for a visual user.


Using LIKE

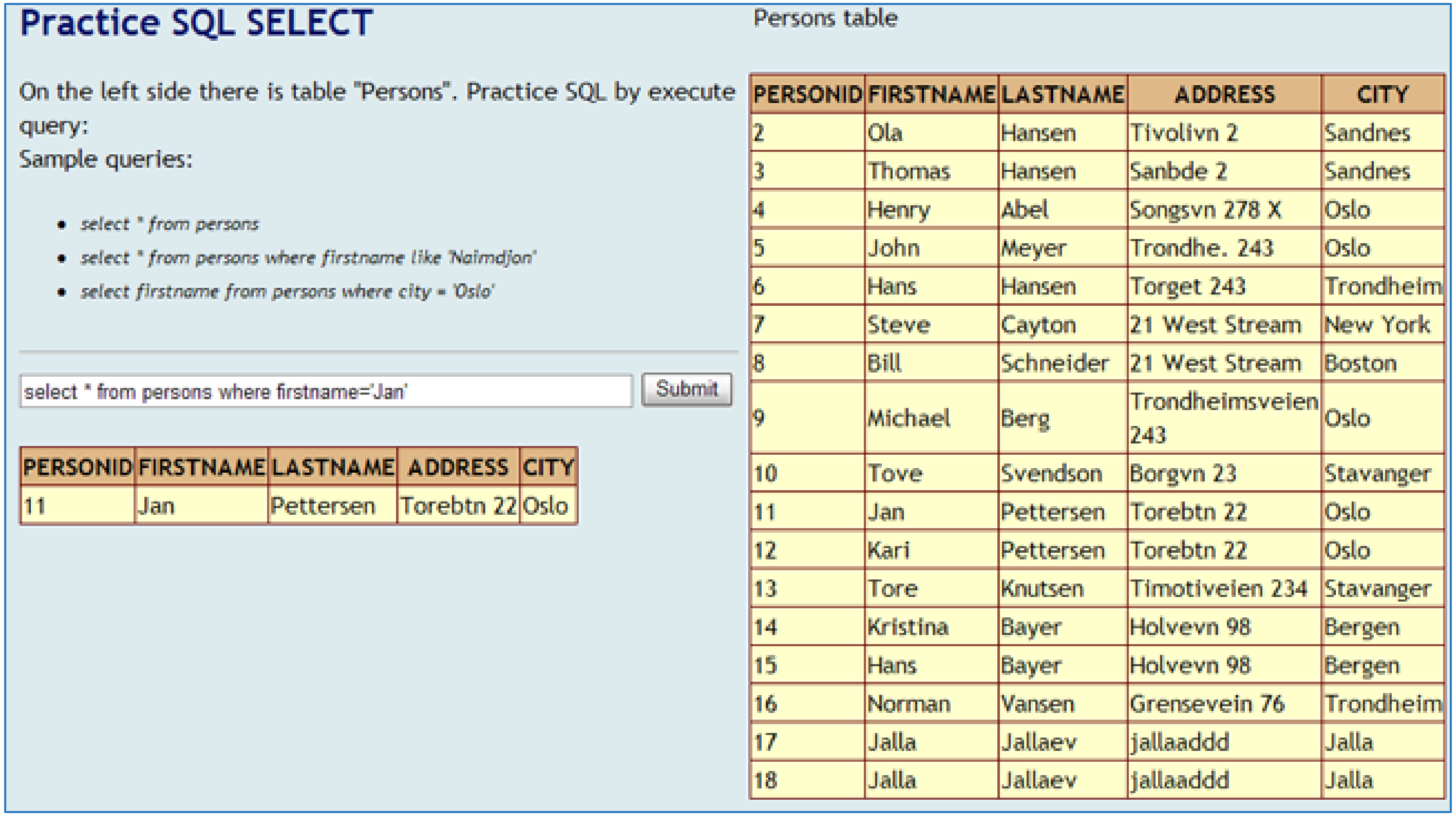
The following SQL statement will return persons with first names that start with an 'O':

SELECT * FROM Persons WHERE FirstName LIKE 'O%'

The following SQL statement will return persons with first names that contain the pattern 'la':

SELECT * FROM Persons WHERE FirstName LIKE '%la%'

Figure 7.53: (4) “SQL Select” module for a visual user.



Practice SQL SELECT

On the left side there is table "Persons". Practice SQL by execute query:
Sample queries:

- *select * from persons*
- *select * from persons where firstname like 'Naimdjon'*
- *select firstname from persons where city = 'Oslo'*

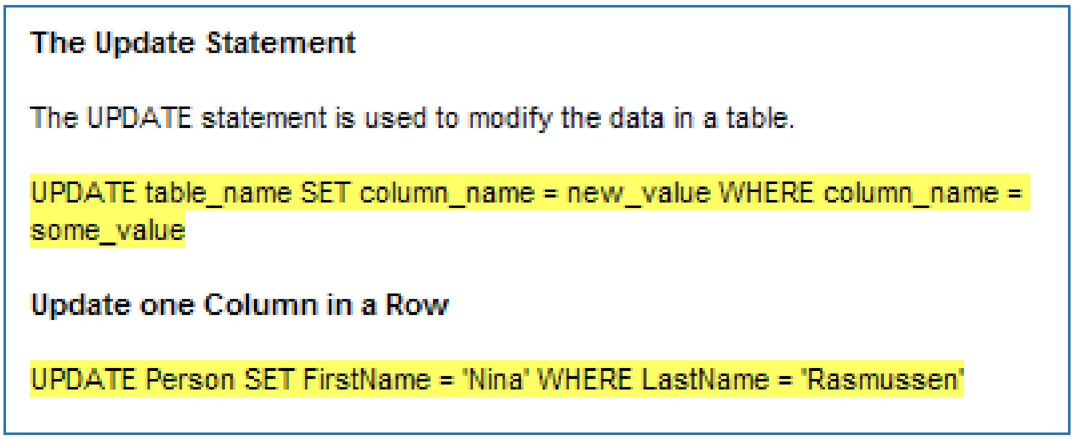
```
select * from persons where firstname='Jan'
```

PERSONID	FIRSTNAME	LASTNAME	ADDRESS	CITY
11	Jan	Pettersen	Torebtn 22	Oslo

Persons table

PERSONID	FIRSTNAME	LASTNAME	ADDRESS	CITY
2	Ola	Hansen	Tivolivn 2	Sandnes
3	Thomas	Hansen	Sanbde 2	Sandnes
4	Henry	Abel	Songsvn 278 X	Oslo
5	John	Meyer	Trondhe. 243	Oslo
6	Hans	Hansen	Torget 243	Trondheim
7	Steve	Cayton	21 West Stream	New York
8	Bill	Schneider	21 West Stream	Boston
9	Michael	Berg	Trondheimsveien 243	Oslo
10	Tove	Svendson	Borgvn 23	Stavanger
11	Jan	Pettersen	Torebtn 22	Oslo
12	Kari	Pettersen	Torebtn 22	Oslo
13	Tore	Knutsen	Timotiveien 234	Stavanger
14	Kristina	Bayer	Holvevn 98	Bergen
15	Hans	Bayer	Holvevn 98	Bergen
16	Norman	Vansen	Grensevein 76	Trondheim
17	Jalla	Jallaev	jallaaddd	Jalla
18	Jalla	Jallaev	jallaaddd	Jalla

Figure 7.54: (5) “SQL Select” practice item for a kinesthetic user featuring a live database.


The Update Statement

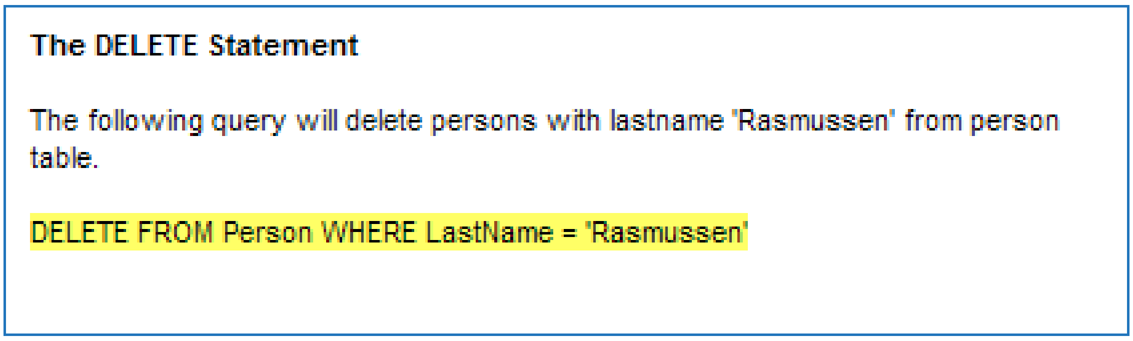
The UPDATE statement is used to modify the data in a table.

```
UPDATE table_name SET column_name = new_value WHERE column_name =
some_value
```

Update one Column in a Row

```
UPDATE Person SET FirstName = 'Nina' WHERE LastName = 'Rasmussen'
```

Figure 7.55: (1) “SQL Update” module for a visual user.

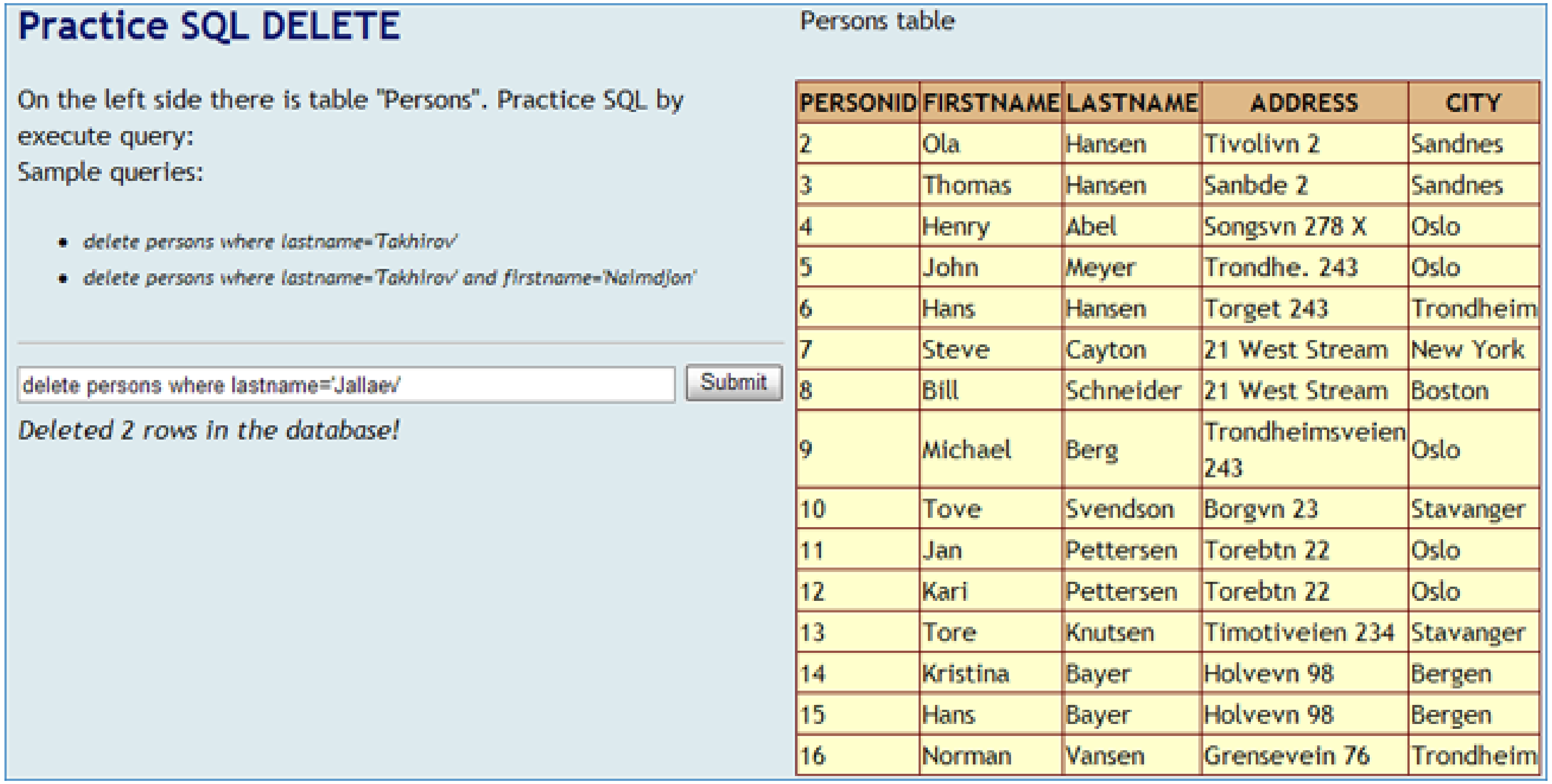


The DELETE Statement

The following query will delete persons with lastname 'Rasmussen' from person table.

```
DELETE FROM Person WHERE LastName = 'Rasmussen'
```

Figure 7.56: (1) “SQL Delete” module for a visual user.



Practice SQL DELETE

On the left side there is table "Persons". Practice SQL by execute query:
Sample queries:

- *delete persons where lastname='Takhirov'*
- *delete persons where lastname='Takhirov' and firstname='Nalmdjon'*

```
delete persons where lastname='Jallaev'
```

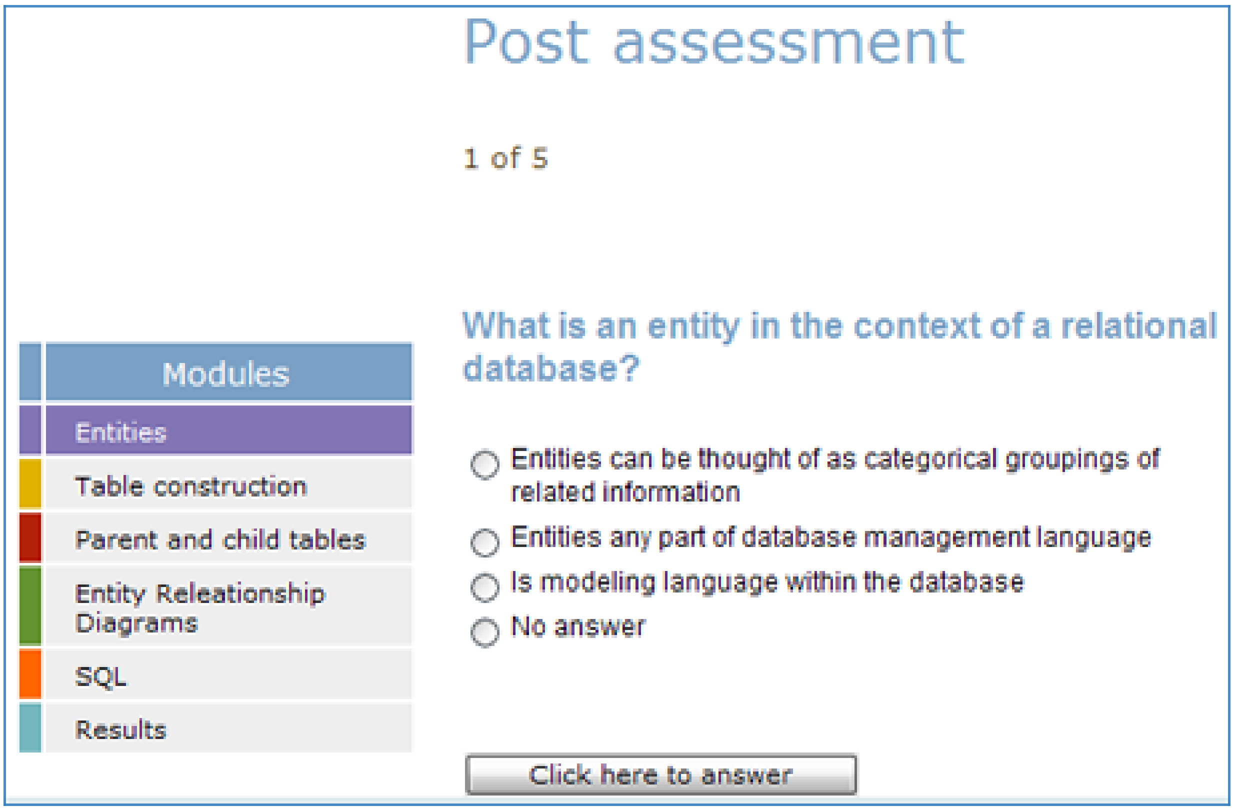
Deleted 2 rows in the database!

Persons table

PERSONID	FIRSTNAME	LASTNAME	ADDRESS	CITY
2	Ola	Hansen	Tivolivn 2	Sandnes
3	Thomas	Hansen	Sanbde 2	Sandnes
4	Henry	Abel	Songsvn 278 X	Oslo
5	John	Meyer	Trondhe. 243	Oslo
6	Hans	Hansen	Torget 243	Trondheim
7	Steve	Cayton	21 West Stream	New York
8	Bill	Schneider	21 West Stream	Boston
9	Michael	Berg	Trondheimsveien 243	Oslo
10	Tove	Svendson	Borgvn 23	Stavanger
11	Jan	Pettersen	Torebtn 22	Oslo
12	Kari	Pettersen	Torebtn 22	Oslo
13	Tore	Knutsen	Timotiveien 234	Stavanger
14	Kristina	Bayer	Holvevn 98	Bergen
15	Hans	Bayer	Holvevn 98	Bergen
16	Norman	Vansen	Grensevein 76	Trondheim

Figure 7.57: (2) “SQL Delete” practice item for a kinesthetic user featuring a live database.

Post assessment



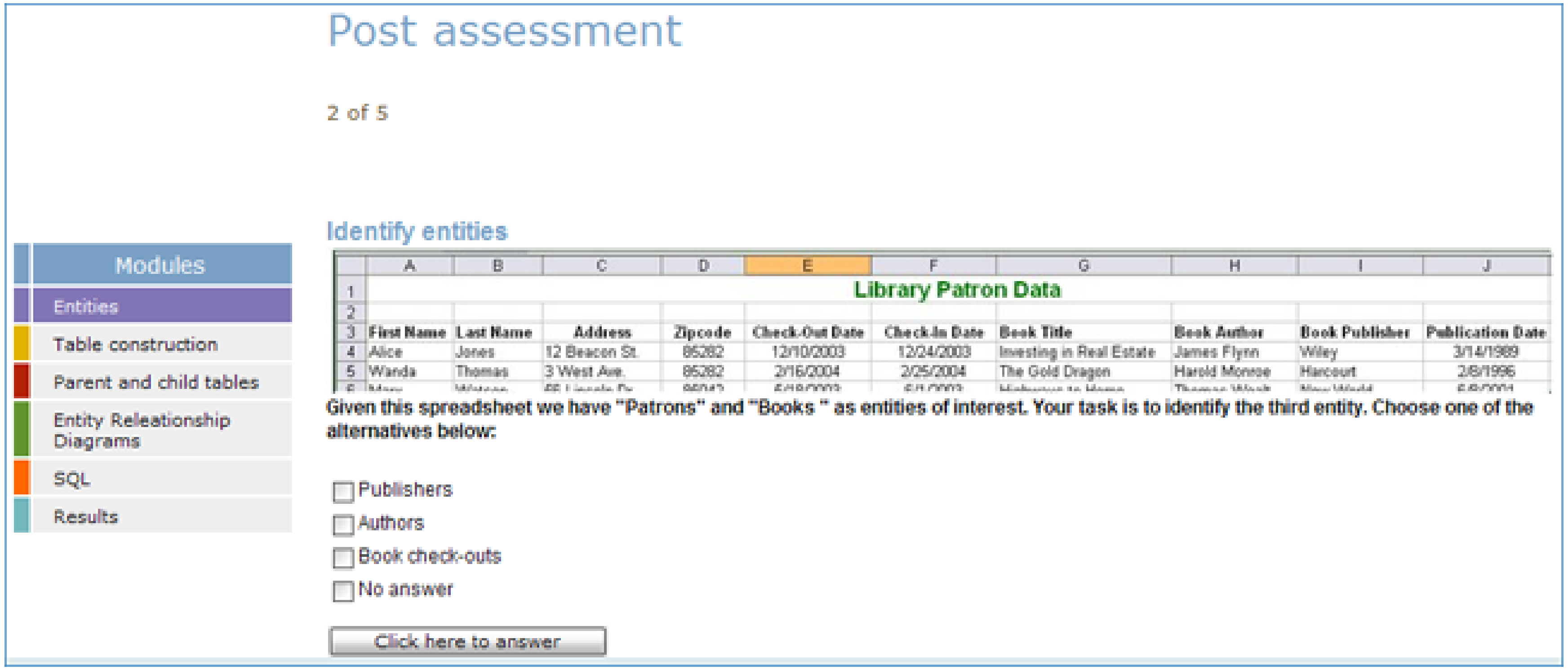
Post assessment

1 of 5

What is an entity in the context of a relational database?

- ◯ Entities can be thought of as categorical groupings of related information
- ◯ Entities any part of database management language
- ◯ Is modeling language within the database
- ◯ No answer

Figure 7.58: (1) Question on Entities.



Post assessment

2 of 5

Identify entities

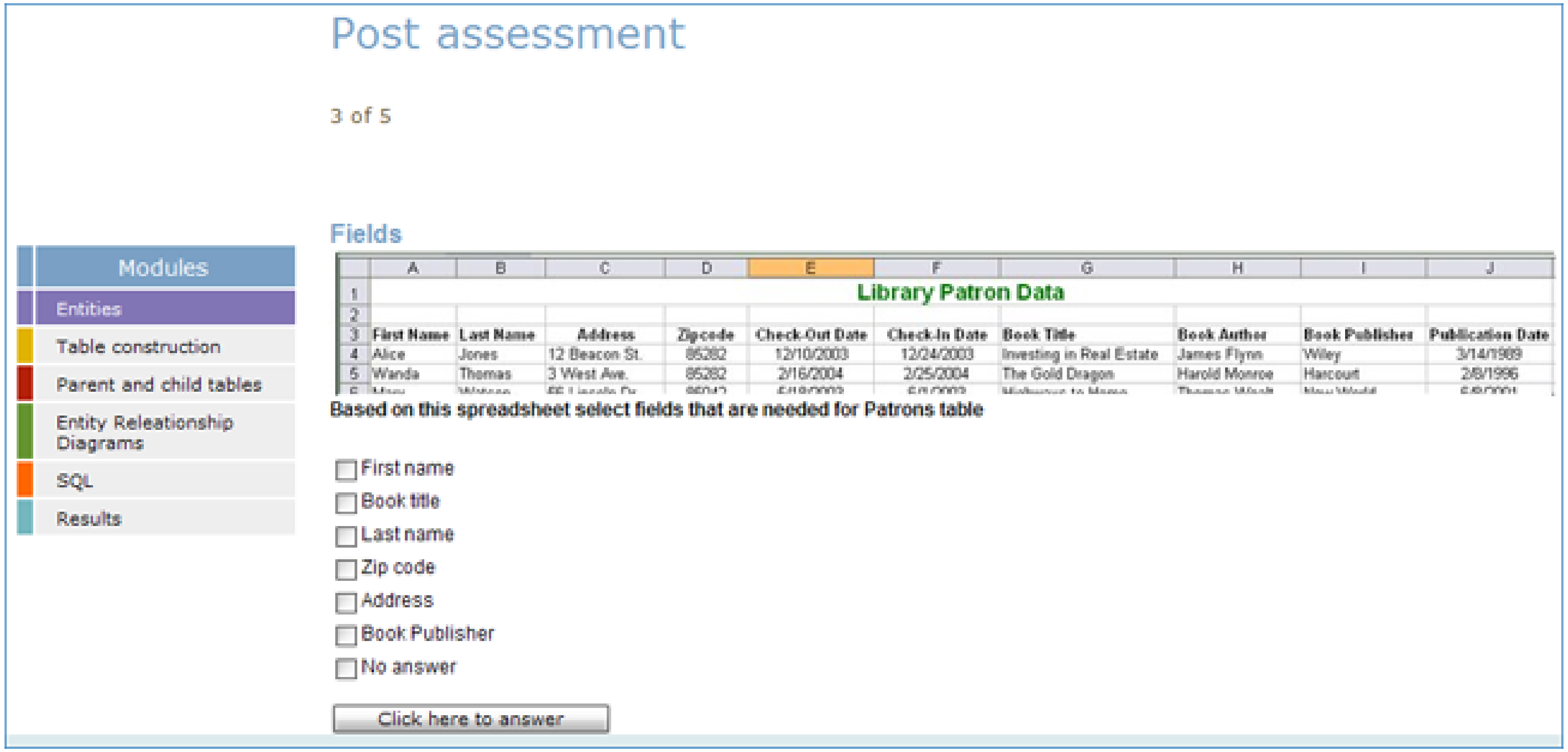
	A	B	C	D	E	F	G	H	I	J
1	Library Patron Data									
2										
3	First Name	Last Name	Address	Zipcode	Check-Out Date	Check-In Date	Book Title	Book Author	Book Publisher	Publication Date
4	Alice	Jones	12 Beacon St.	85282	12/10/2003	12/24/2003	Investing in Real Estate	James Flynn	Wiley	3/14/1989
5	Wanda	Thomas	3 West Ave.	85282	2/16/2004	2/25/2004	The Gold Dragon	Harold Monroe	Harcourt	2/8/1996
6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Given this spreadsheet we have "Patrons" and "Books " as entities of interest. Your task is to identify the third entity. Choose one of the alternatives below:

☐ Publishers
☐ Authors
☐ Book check-outs
☐ No answer

Click here to answer

Figure 7.59: (2) Question on Entities.



Post assessment

3 of 5

Fields

Library Patron Data

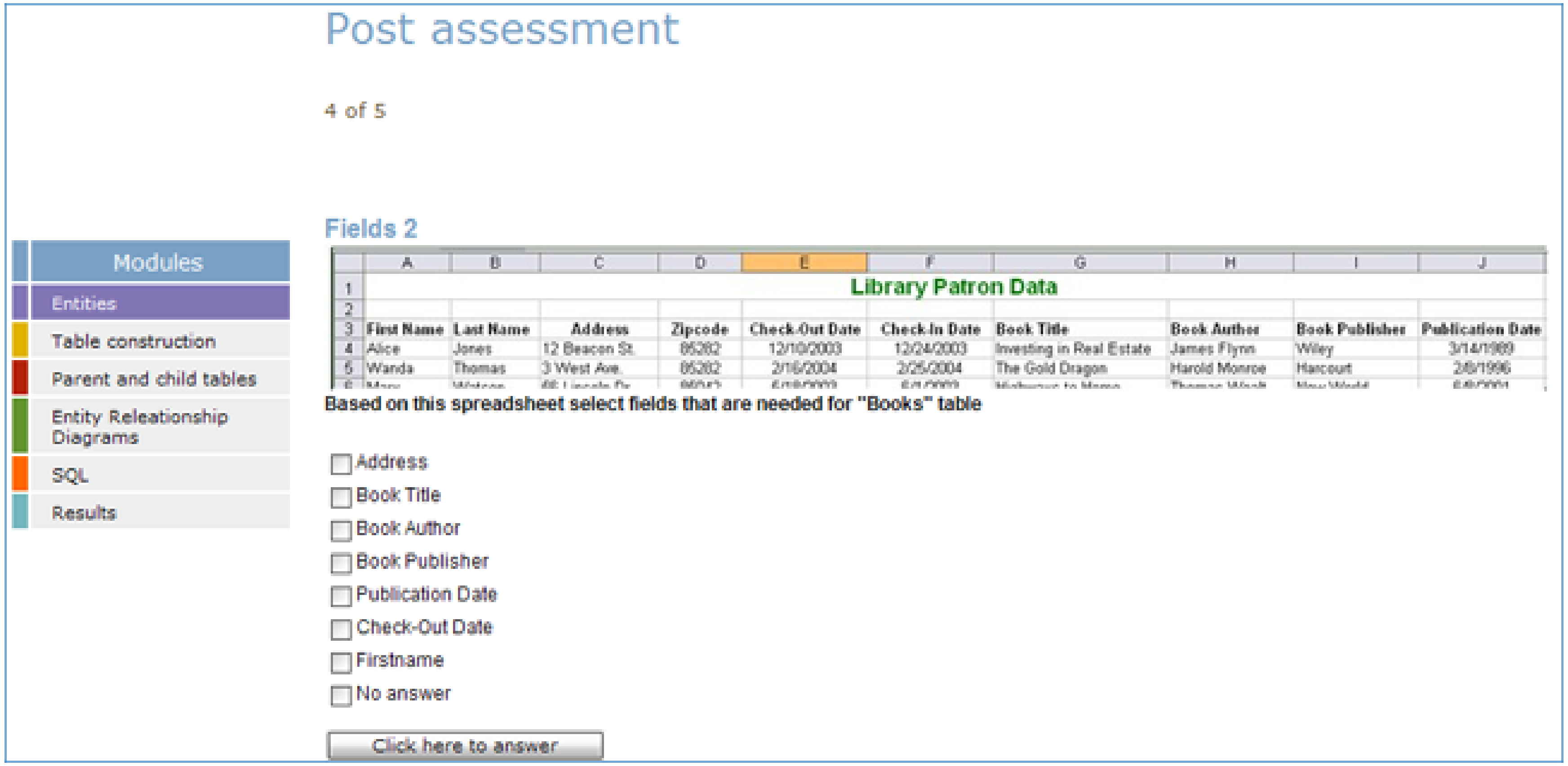
	A	B	C	D	E	F	G	H	I	J
1										
2										
3	First Name	Last Name	Address	Zipcode	Check-Out Date	Check-In Date	Book Title	Book Author	Book Publisher	Publication Date
4	Alice	Jones	12 Beacon St.	85282	12/10/2003	12/24/2003	Investing in Real Estate	James Flynn	Wiley	3/14/1989
5	Wanda	Thomas	3 West Ave.	85282	2/16/2004	2/25/2004	The Gold Dragon	Harold Monroe	Harcourt	2/8/1996
6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Based on this spreadsheet select fields that are needed for Patrons table

- ☐ First name
- ☐ Book title
- ☐ Last name
- ☐ Zip code
- ☐ Address
- ☐ Book Publisher
- ☐ No answer

Click here to answer

Figure 7.60: (3) Question on Entities.



Post assessment

4 of 5

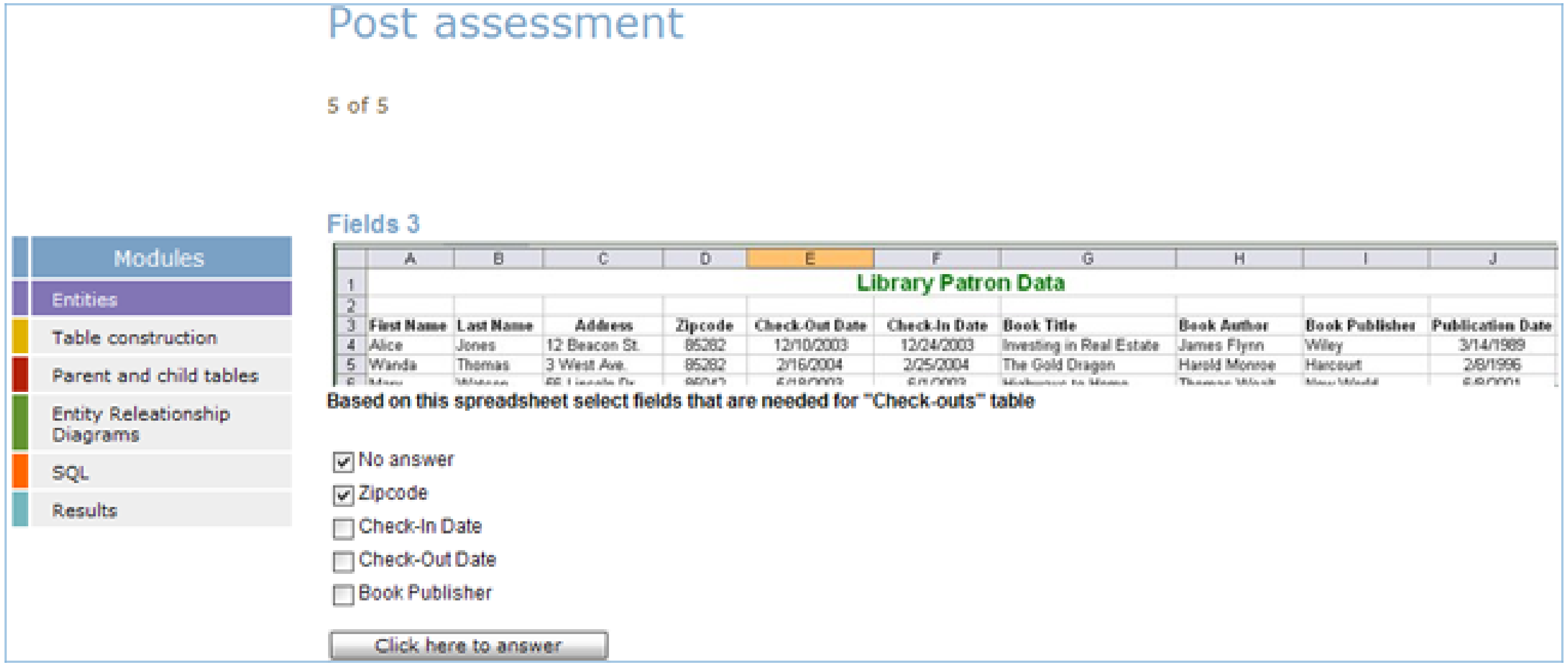
Fields 2

	A	B	C	D	E	F	G	H	I	J
1	Library Patron Data									
2										
3	First Name	Last Name	Address	Zipcode	Check-Out Date	Check-In Date	Book Title	Book Author	Book Publisher	Publication Date
4	Alice	Jones	12 Beacon St.	85282	12/10/2003	12/24/2003	Investing in Real Estate	James Flynn	Wiley	3/14/1989
5	Wanda	Thomas	3 West Ave.	85282	2/16/2004	2/25/2004	The Gold Dragon	Harold Monroe	Harcourt	2/8/1996
6	Mary	Watson	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	New World	[illegible]

Based on this spreadsheet select fields that are needed for "Books" table

- ☐ Address
- ☐ Book Title
- ☐ Book Author
- ☐ Book Publisher
- ☐ Publication Date
- ☐ Check-Out Date
- ☐ Firstname
- ☐ No answer

Figure 7.61: (4) Question on Entities.



Post assessment

5 of 5

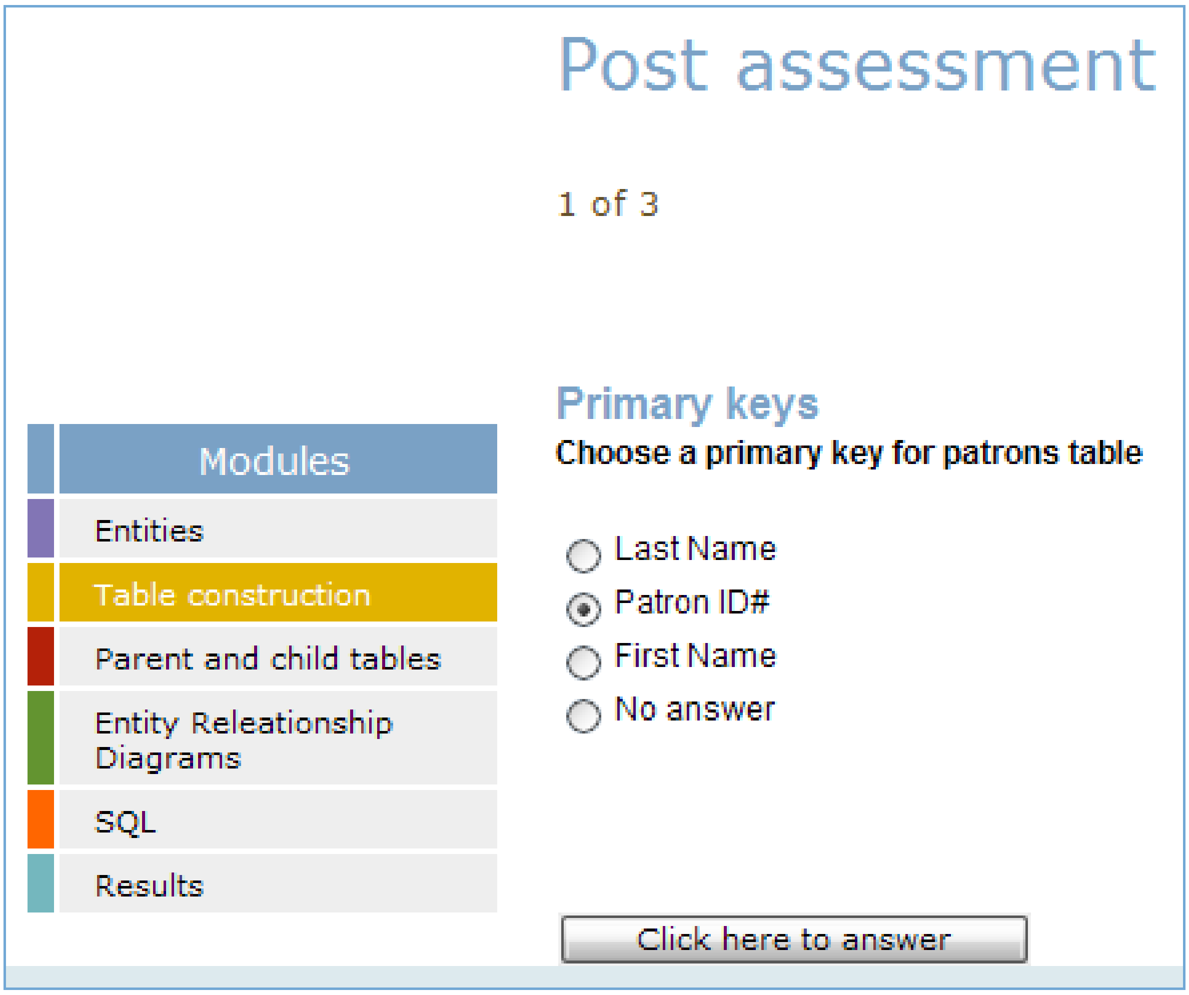
Fields 3

	A	B	C	D	E	F	G	H	I	J
1					Library Patron Data					
2										
3	First Name	Last Name	Address	Zipcode	Check-Out Date	Check-In Date	Book Title	Book Author	Book Publisher	Publication Date
4	Alice	Jones	12 Beacon St.	85282	12/10/2003	12/24/2003	Investing in Real Estate	James Flynn	Wiley	3/14/1989
5	Wanda	Thomas	3 West Ave.	85282	2/16/2004	2/25/2004	The Gold Dragon	Harold Monroe	Harcourt	2/8/1996
6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Based on this spreadsheet select fields that are needed for "Check-outs" table

- [✓] No answer
- [✓] Zipcode
- [ ] Check-In Date
- [ ] Check-Out Date
- [ ] Book Publisher

Figure 7.62: (5) Question on Entities.


Post assessment

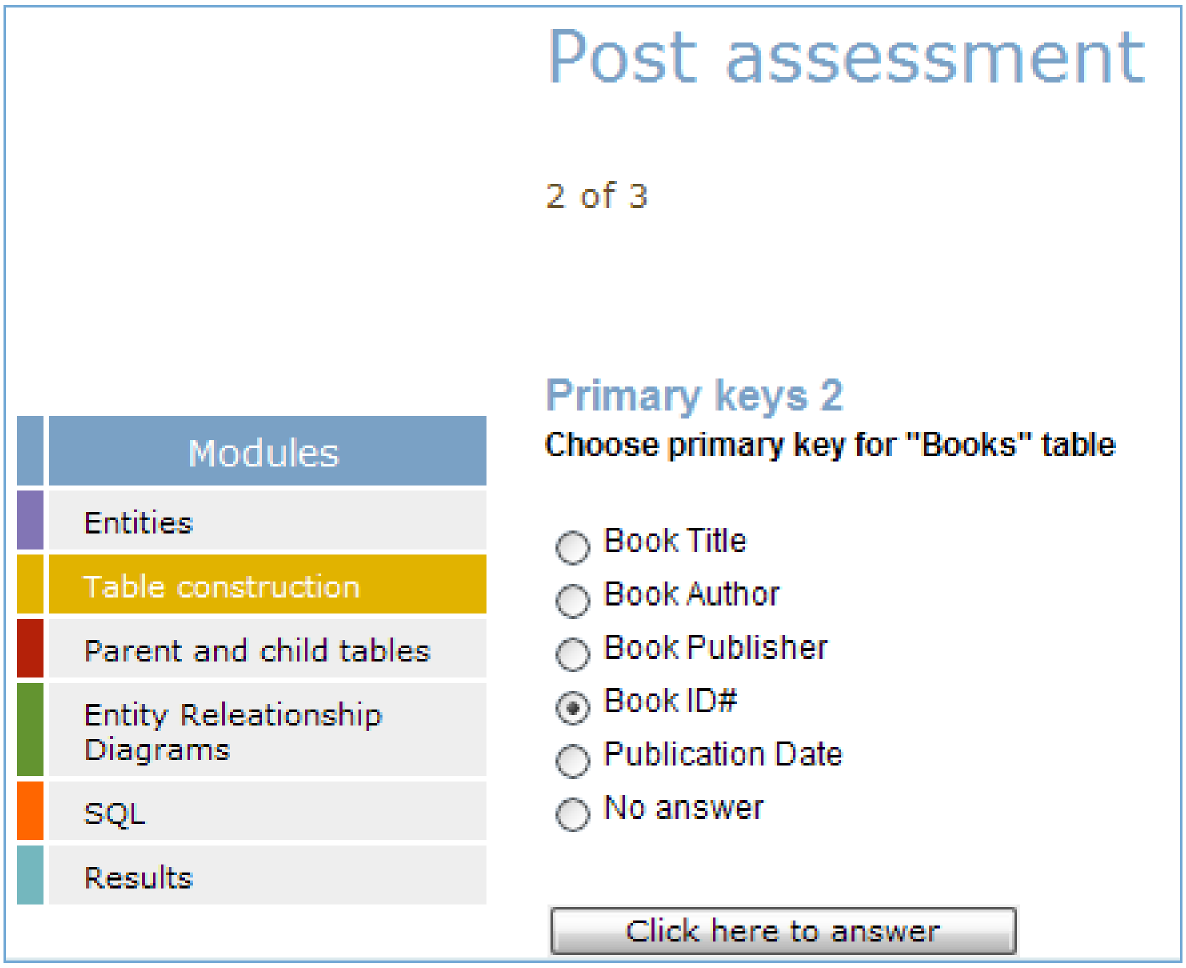
1 of 3

Primary keys

Choose a primary key for patrons table

- Last Name
- [✓ Patron ID#]
- First Name
- No answer

Figure 7.63: (1) Question on table construction.



Post assessment

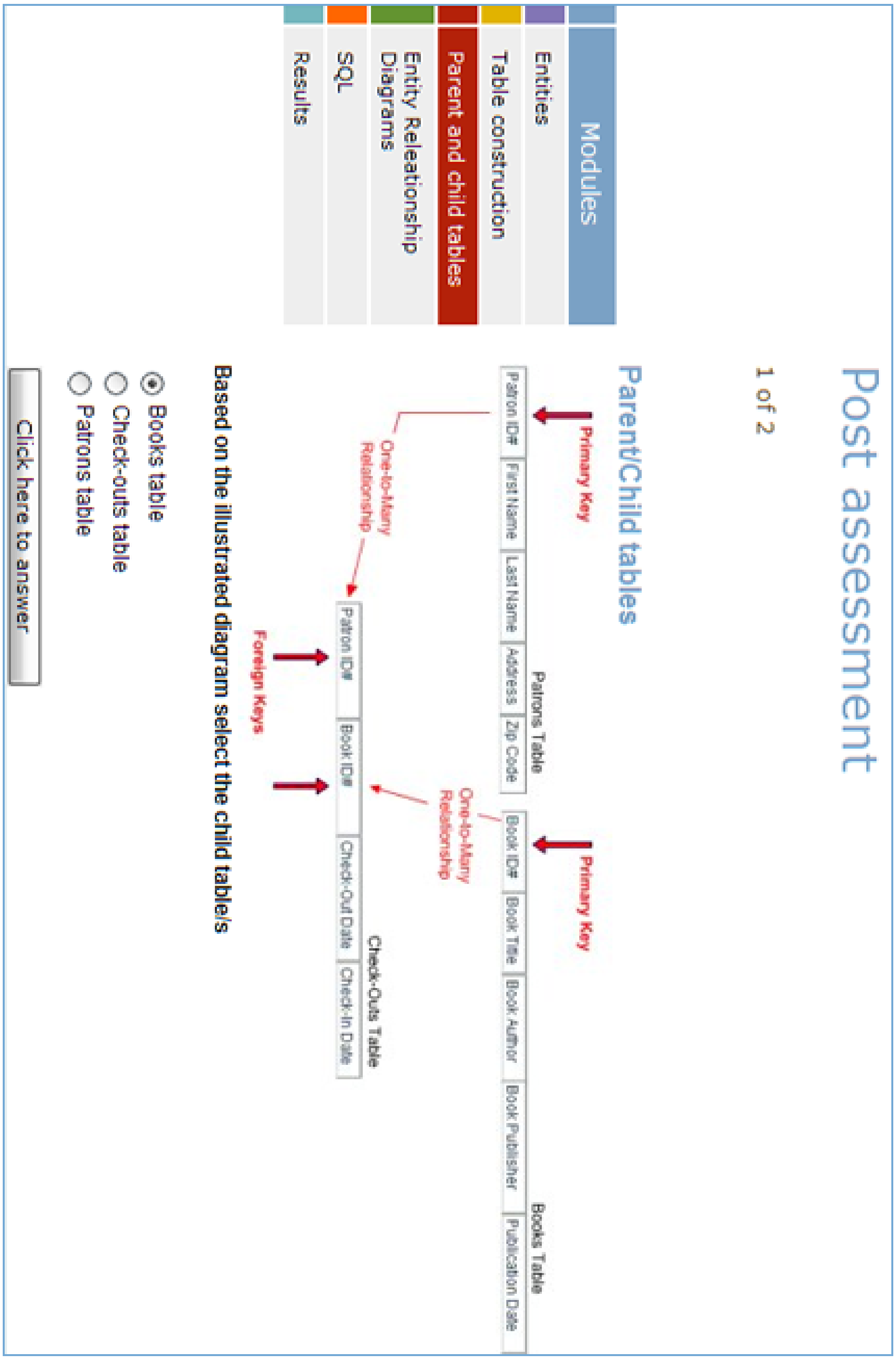
2 of 3

Primary keys 2

Choose primary key for "Books" table

- Book Title
- Book Author
- Book Publisher
- [✓ Book ID#]
- Publication Date
- No answer

Figure 7.64: (2) Question on table construction.



Post assessment

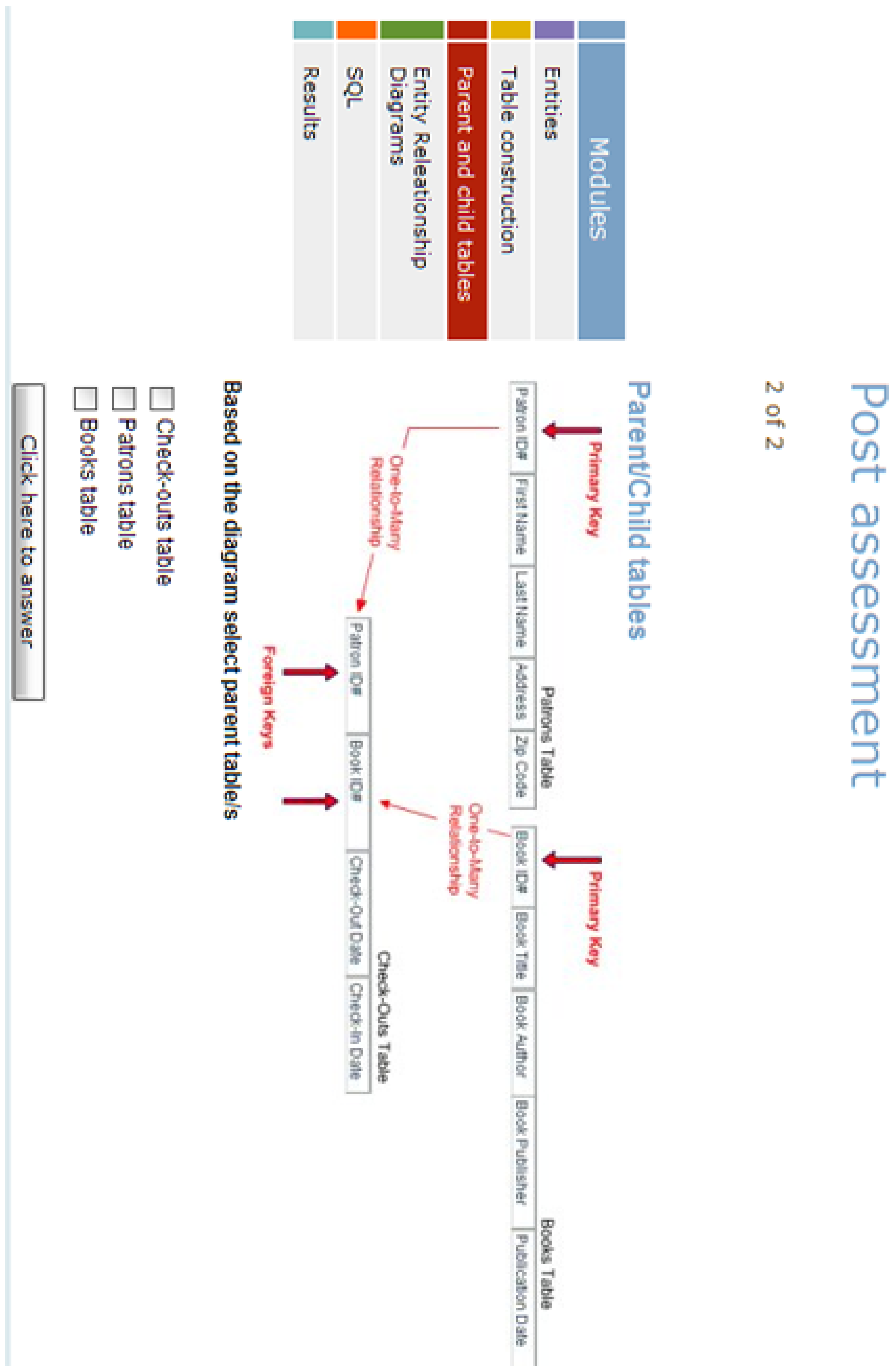
1 of 2

Parent/Child tables

Based on the illustrated diagram select the child table/s

- [✓ Books table]
- Check-outs table
- Patrons table

Figure 7.65: (1) Question on construction of parent and child tables.



Post assessment

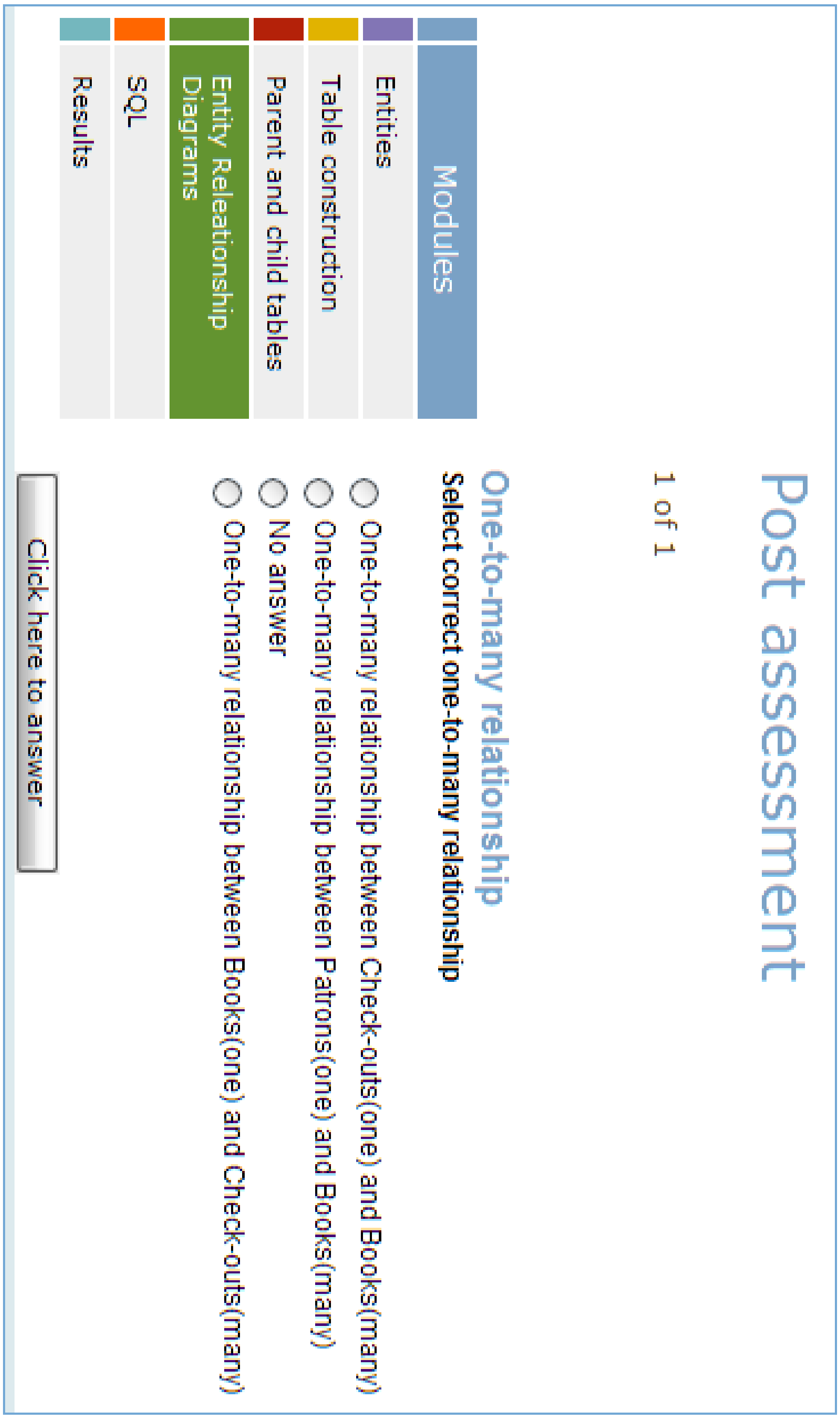
2 of 2

Parent/Child tables

Based on the diagram select parent table/s

- [ ] Check-outs table
- [ ] Patrons table
- [ ] Books table

Figure 7.66: (2) Question on construction of parent and child tables.



Post assessment

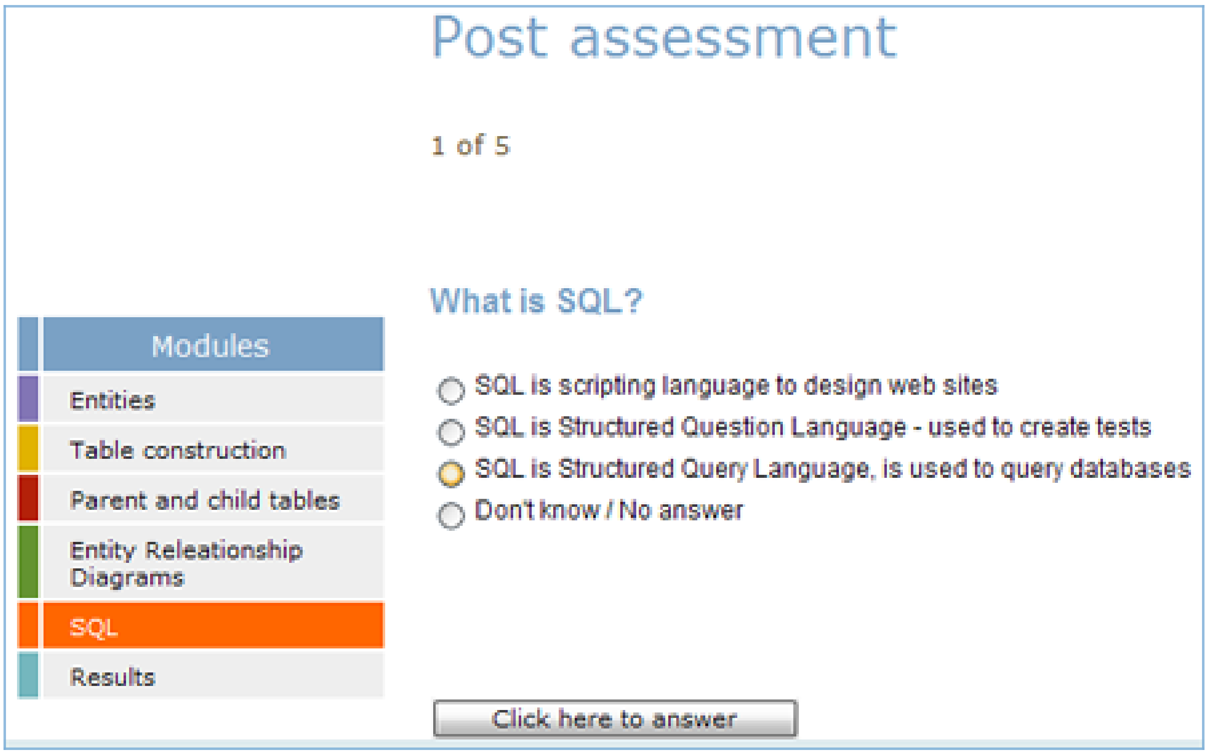
1 of 1

One-to-many relationship

Select correct one-to-many relationship

- One-to-many relationship between Check-outs(one) and Books(many)
- One-to-many relationship between Patrons(one) and Books(many)
- No answer
- One-to-many relationship between Books(one) and Check-outs(many)

Figure 7.67: (1) Question on construction of parent and child tables.



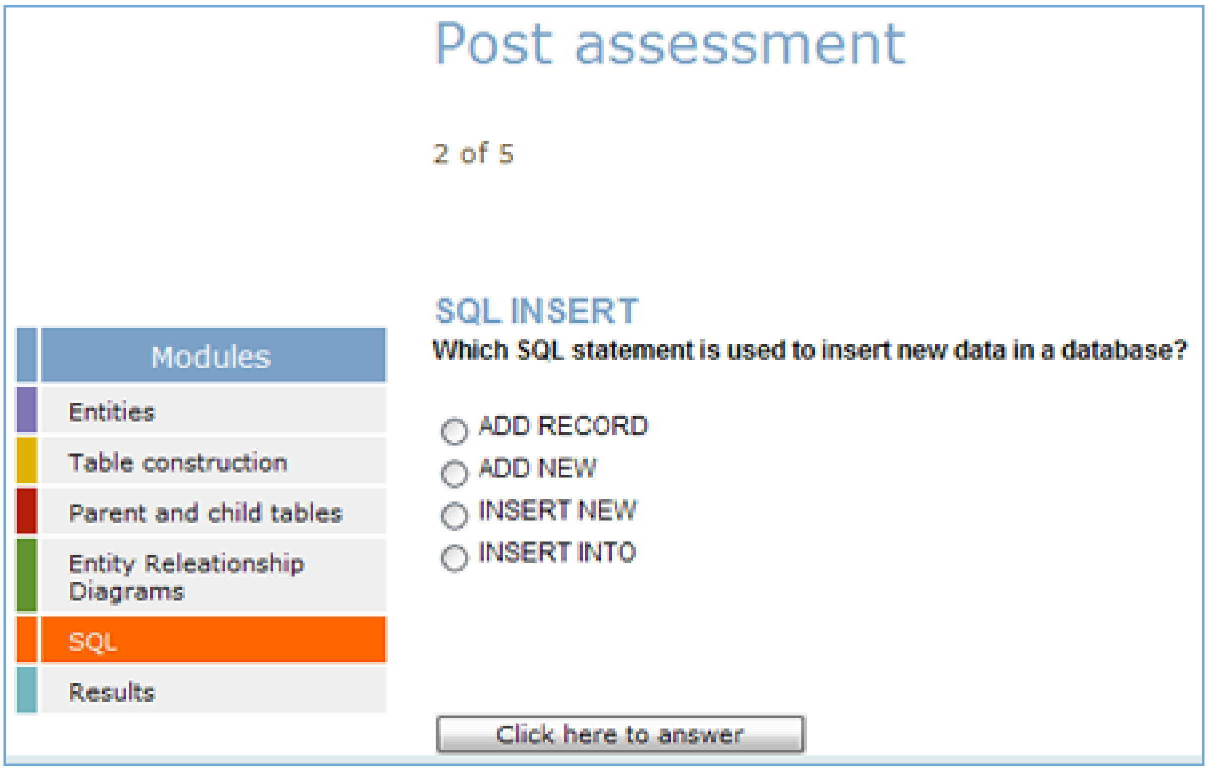
Post assessment

1 of 5

What is SQL?

- ○ SQL is scripting language to design web sites
- ○ SQL is Structured Question Language - used to create tests
- [✓ SQL is Structured Query Language, is used to query databases]
- ○ Don't know / No answer

Figure 7.68: (1) Question on SQL.



Post assessment

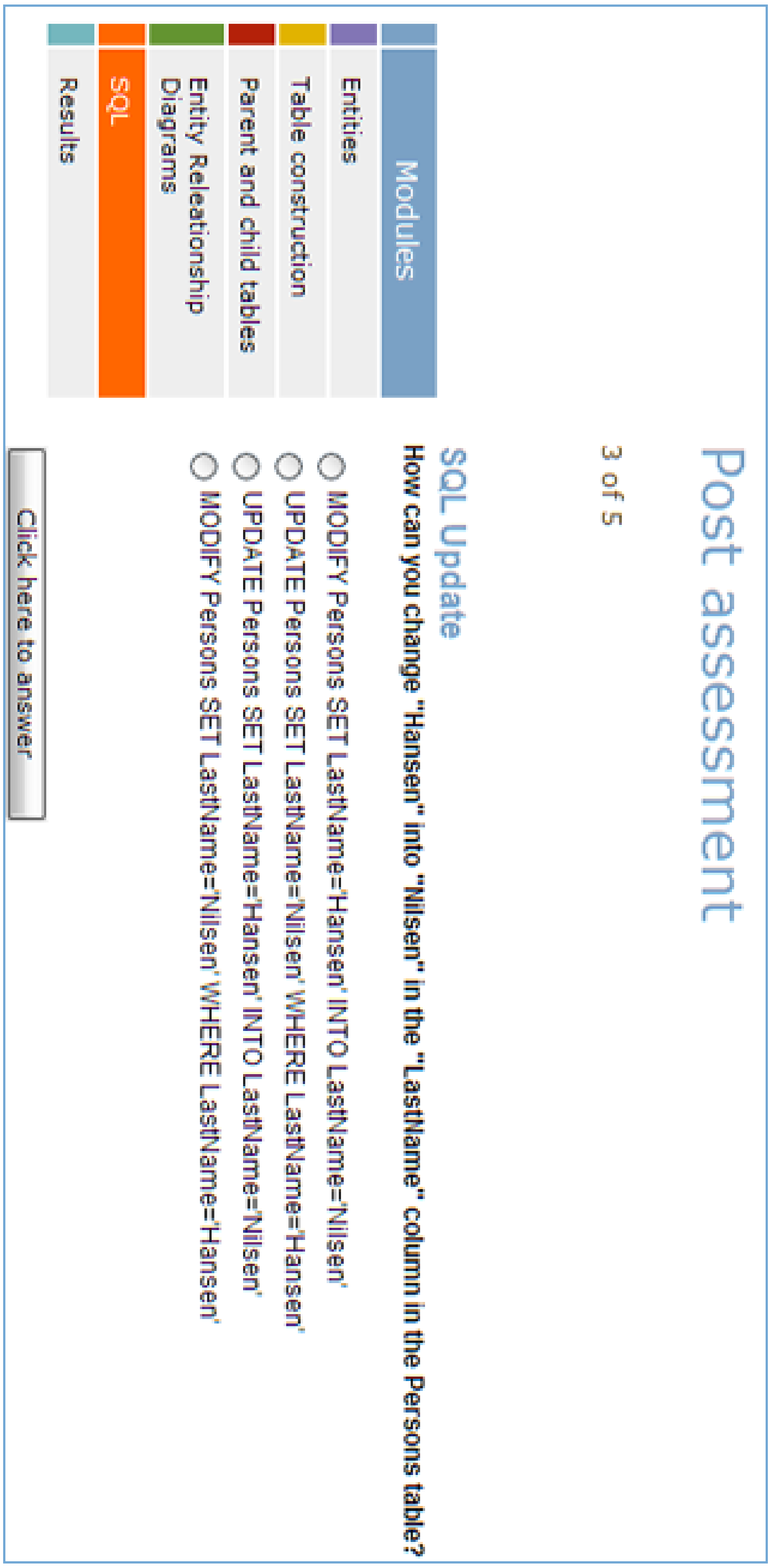
2 of 5

SQL INSERT

Which SQL statement is used to insert new data in a database?

- ADD RECORD
- ADD NEW
- INSERT NEW
- INSERT INTO

Figure 7.69: (2) Question on SQL.

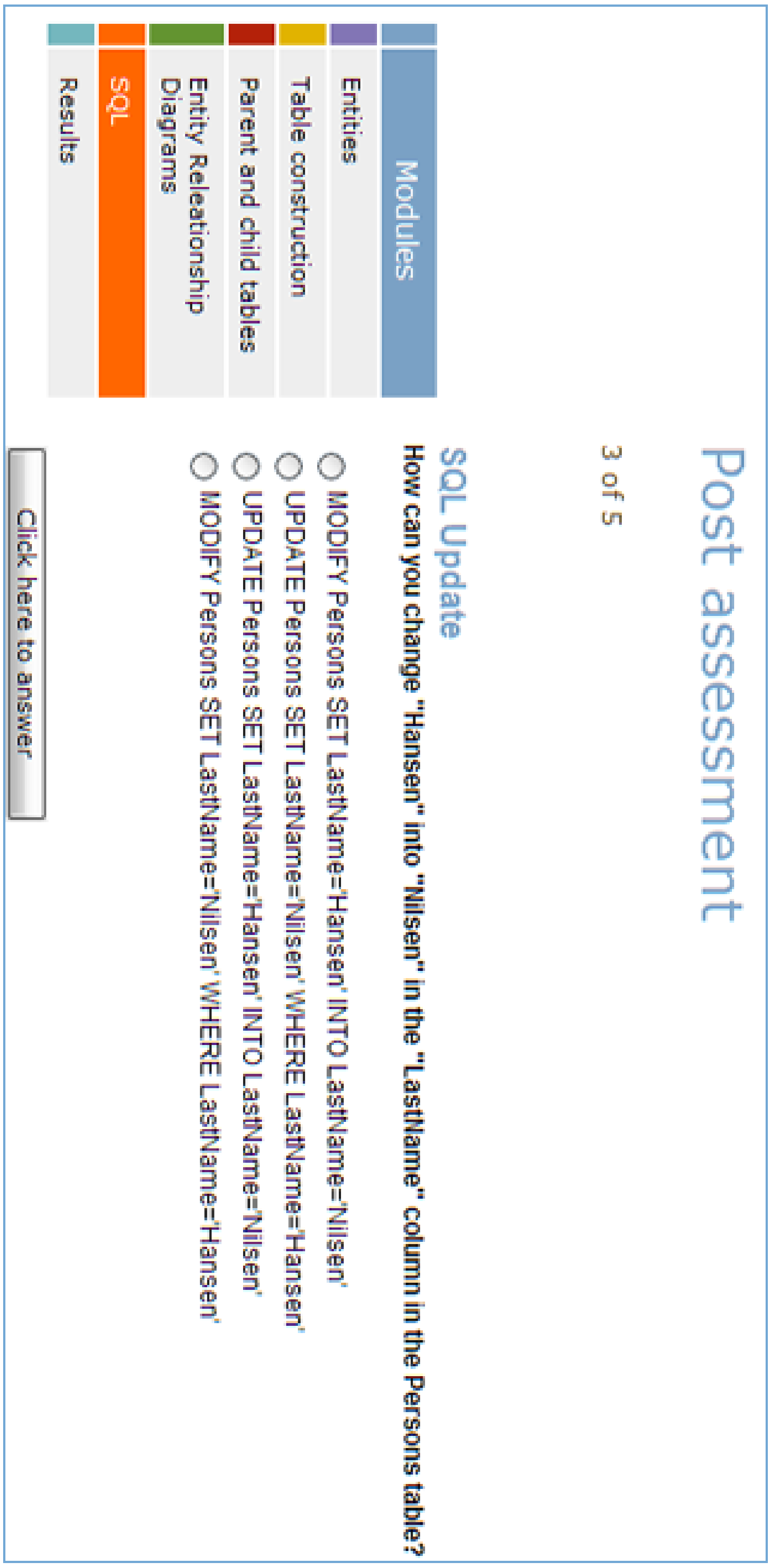


SQL Update

How can you change "Hansen" into "Nilsen" in the "LastName" column in the Persons table?

- MODIFY Persons SET LastName='Hansen' INTO LastName='Nilsen'
- UPDATE Persons SET LastName='Nilsen' WHERE LastName='Hansen'
- UPDATE Persons SET LastName='Hansen' INTO LastName='Nilsen'
- MODIFY Persons SET LastName='Nilsen' WHERE LastName='Hansen'

Figure 7.70: (3) Question on SQL.



Post assessment

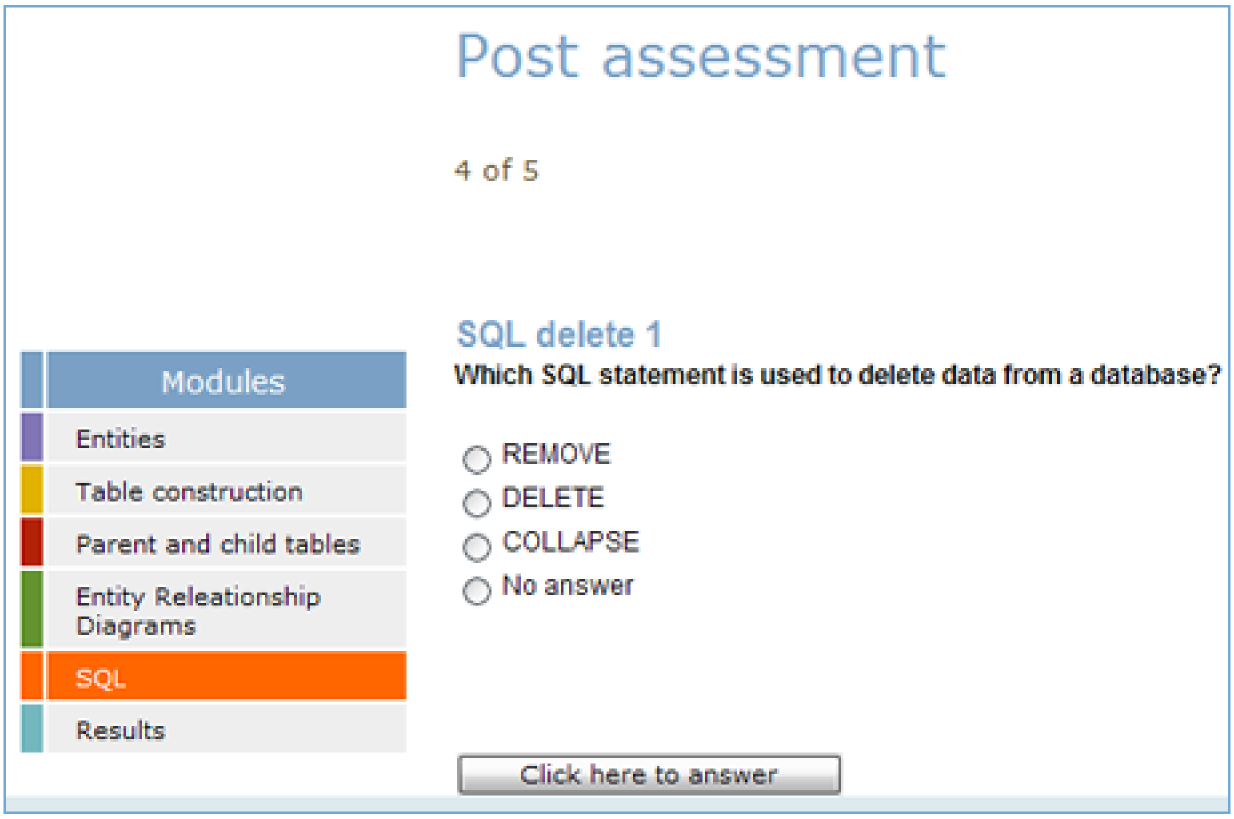
3 of 5

SQL Update

How can you change "Hansen" into "Nilsen" in the "LastName" column in the Persons table?

- ○ MODIFY Persons SET LastName='Hansen' INTO LastName='Nilsen'
- ○ UPDATE Persons SET LastName='Nilsen' WHERE LastName='Hansen'
- ○ UPDATE Persons SET LastName='Hansen' INTO LastName='Nilsen'
- ○ MODIFY Persons SET LastName='Nilsen' WHERE LastName='Hansen'

Figure 7.71: (3) Question on SQL.



Post assessment

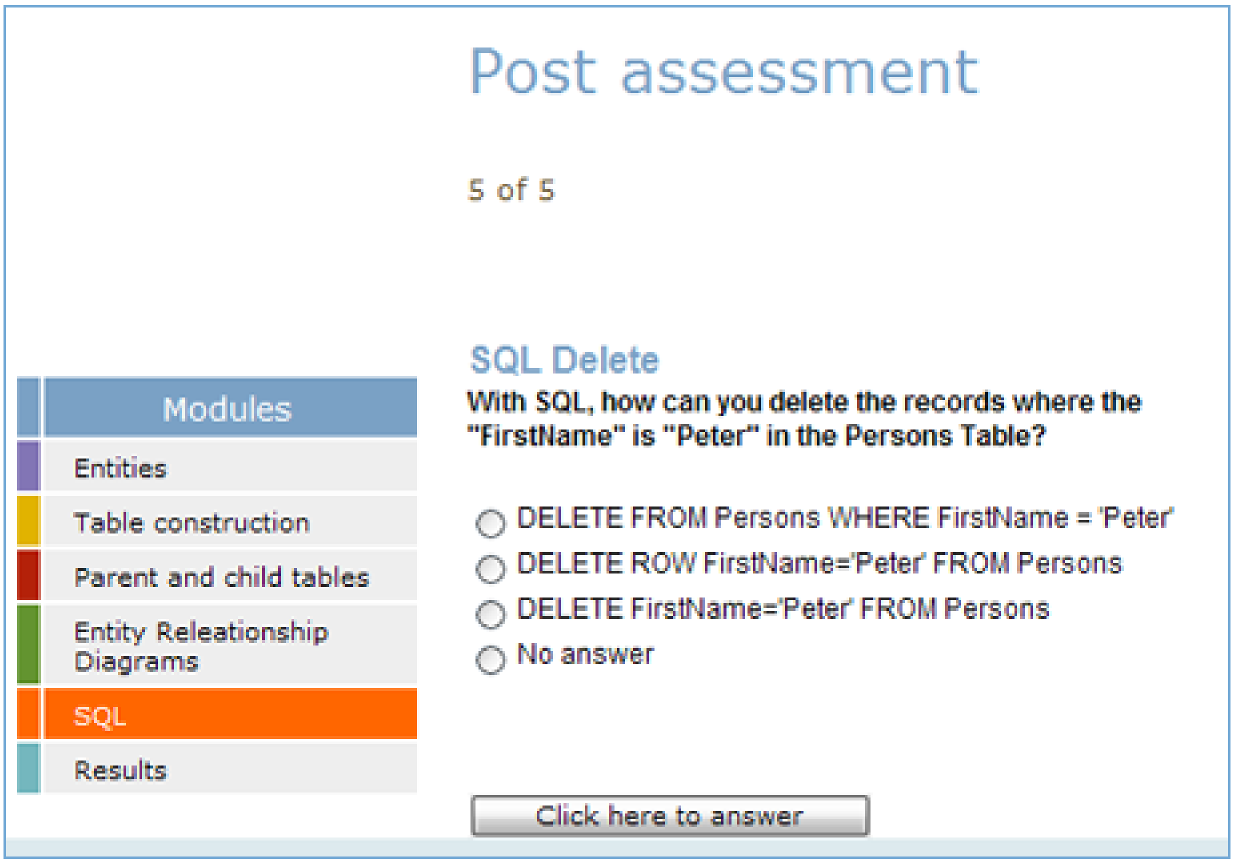
4 of 5

SQL delete 1

Which SQL statement is used to delete data from a database?

- REMOVE
- DELETE
- COLLAPSE
- No answer

Figure 7.72: (4) Question on SQL.



Post assessment

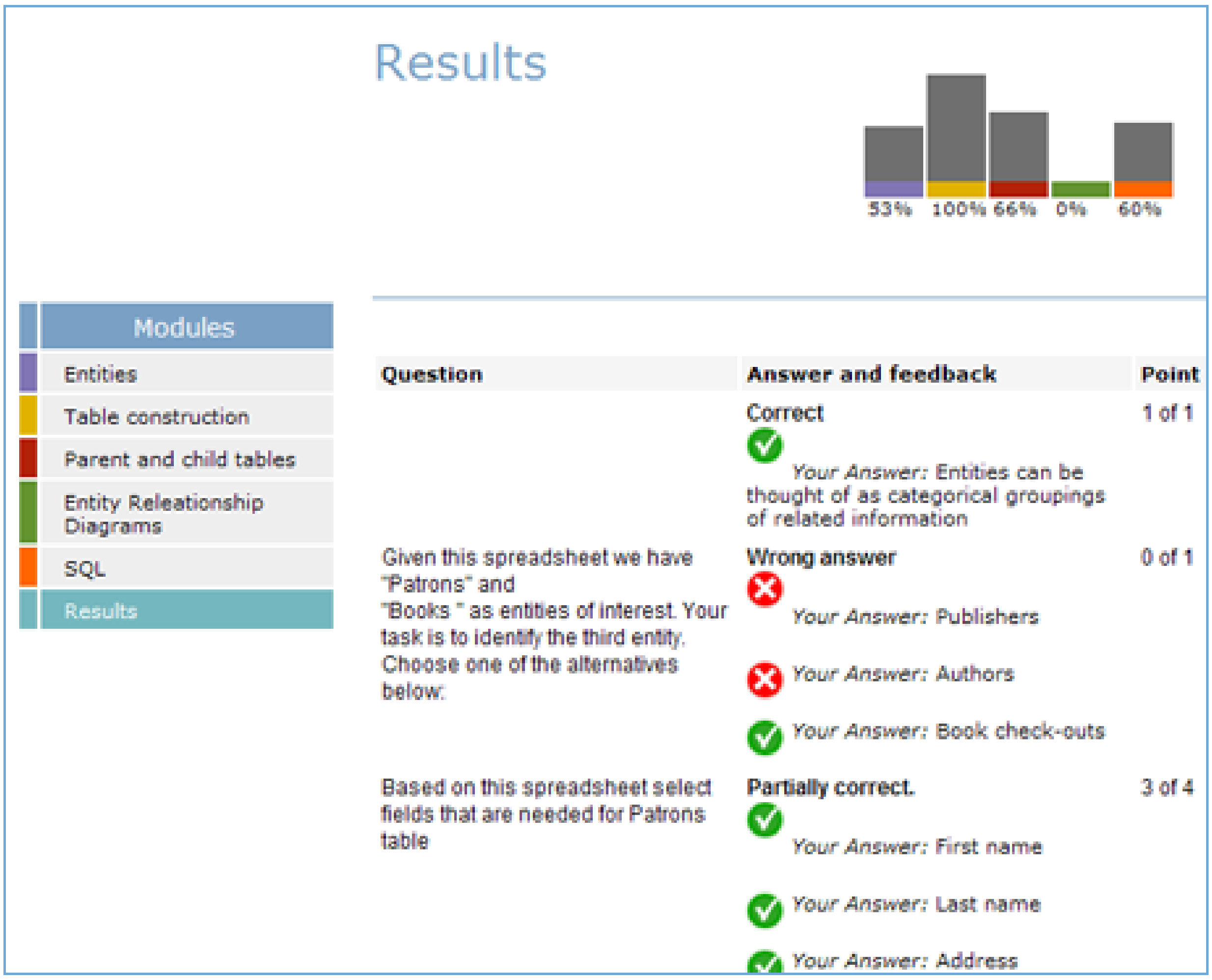
5 of 5

SQL Delete

With SQL, how can you delete the records where the "FirstName" is "Peter" in the Persons Table?

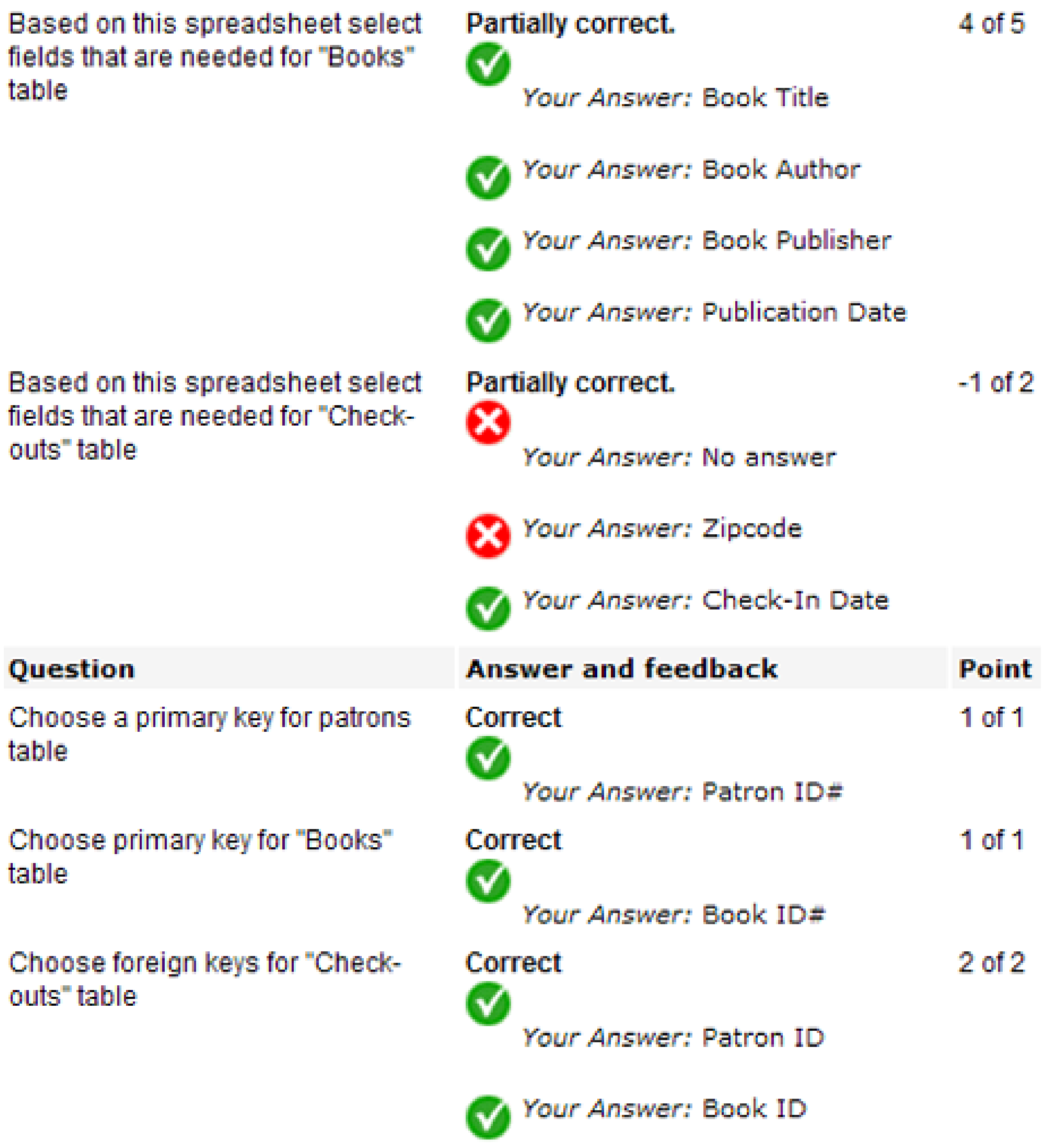
- ○ DELETE FROM Persons WHERE FirstName = 'Peter'
- ○ DELETE ROW FirstName='Peter' FROM Persons
- ○ DELETE FirstName='Peter' FROM Persons
- ○ No answer

Figure 7.73: (5) Question on SQL.



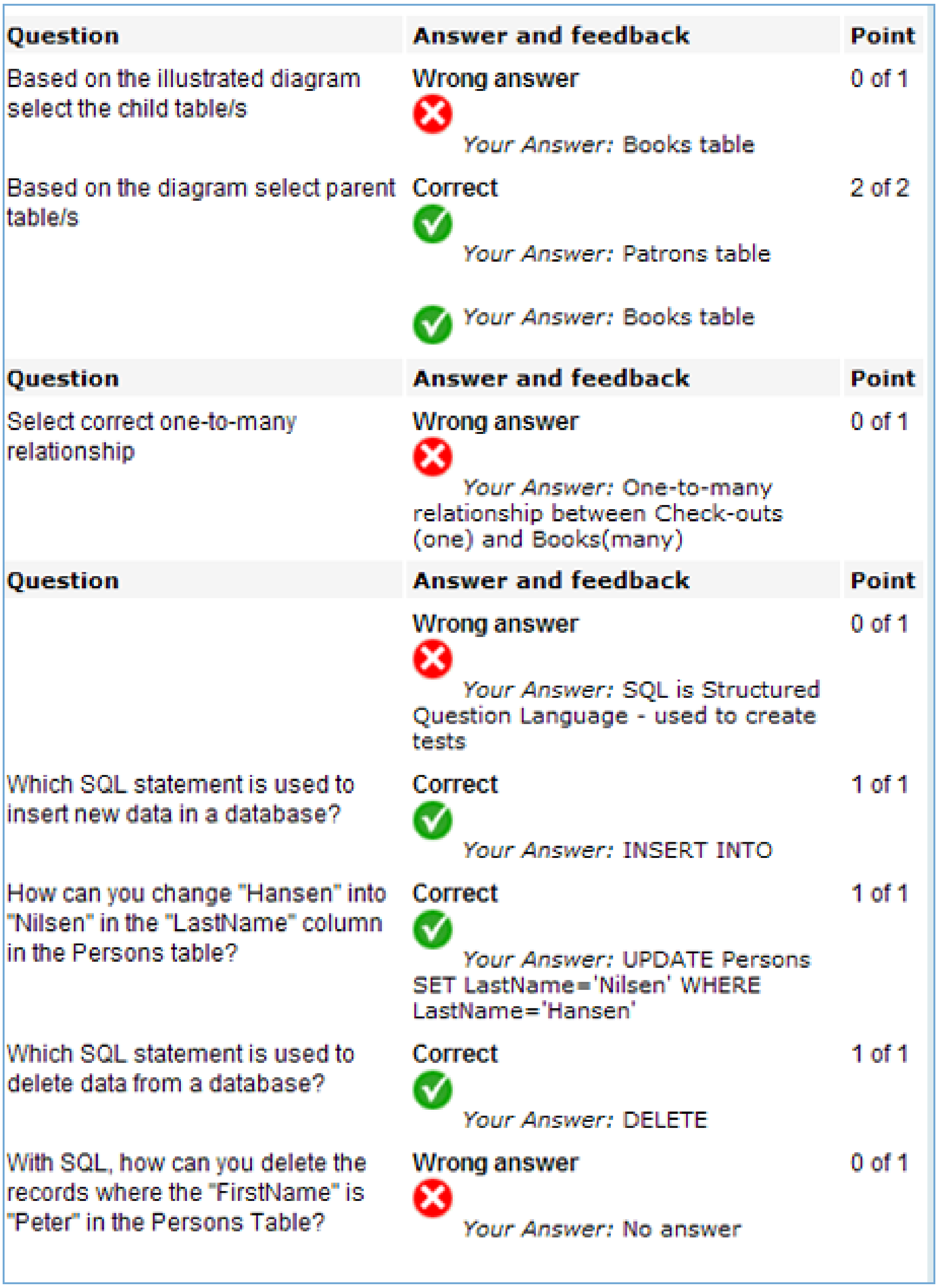
Question	Answer and feedback	Point
	Correct ✓ *Your Answer:* Entities can be thought of as categorical groupings of related information	1 of 1
Given this spreadsheet we have "Patrons" and "Books " as entities of interest. Your task is to identify the third entity. Choose one of the alternatives below.	Wrong answer ✗ *Your Answer:* Publishers ✗ *Your Answer:* Authors ✓ *Your Answer:* Book check-outs	0 of 1
Based on this spreadsheet select fields that are needed for Patrons table	Partially correct. ✓ *Your Answer:* First name ✓ *Your Answer:* Last name ✓ *Your Answer:* Address	3 of 4

Figure 7.74: (1) The results page showing scores obtained by the user.



Based on this spreadsheet select fields that are needed for "Books" table	Partially correct. ✓ *Your Answer:* Book Title ✓ *Your Answer:* Book Author ✓ *Your Answer:* Book Publisher ✓ *Your Answer:* Publication Date	4 of 5
Based on this spreadsheet select fields that are needed for "Check-outs" table	Partially correct. ✗ *Your Answer:* No answer ✗ *Your Answer:* Zipcode ✓ *Your Answer:* Check-In Date	-1 of 2
Question	**Answer and feedback**	**Point**
Choose a primary key for patrons table	Correct ✓ *Your Answer:* Patron ID#	1 of 1
Choose primary key for "Books" table	Correct ✓ *Your Answer:* Book ID#	1 of 1
Choose foreign keys for "Check-outs" table	Correct ✓ *Your Answer:* Patron ID ✓ *Your Answer:* Book ID	2 of 2

Figure 7.75: (2) The results page showing scores obtained by the user.



Question	Answer and feedback	Point
Based on the illustrated diagram select the child table/s	Wrong answer ✗ *Your Answer:* Books table	0 of 1
Based on the diagram select parent table/s	Correct ✓ *Your Answer:* Patrons table ✓ *Your Answer:* Books table	2 of 2

Question	Answer and feedback	Point
Select correct one-to-many relationship	Wrong answer ✗ *Your Answer:* One-to-many relationship between Check-outs (one) and Books(many)	0 of 1

Question	Answer and feedback	Point
	Wrong answer ✗ *Your Answer:* SQL is Structured Question Language - used to create tests	0 of 1
Which SQL statement is used to insert new data in a database?	Correct ✓ *Your Answer:* INSERT INTO	1 of 1
How can you change "Hansen" into "Nilsen" in the "LastName" column in the Persons table?	Correct ✓ *Your Answer:* UPDATE Persons SET LastName='Nilsen' WHERE LastName='Hansen'	1 of 1
Which SQL statement is used to delete data from a database?	Correct ✓ *Your Answer:* DELETE	1 of 1
With SQL, how can you delete the records where the "FirstName" is "Peter" in the Persons Table?	Wrong answer ✗ *Your Answer:* No answer	0 of 1

Figure 7.76: (3) The results page showing scores obtained by the user.

H Source code

no.inspera.services.pedal.PEDAL

```
package no.inspera.services.pedal;

/**
 * <tt>PEDAL</tt> is interface which holds constants.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: PEDAL.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 20.12.2007
 */
public interface PEDAL {

    /**
     * Index enum of fields that are used for in-memory indexing
        .
     */
    public static enum Index{
        type,
        title,
        id,
        parentId
    }

    /**
     * Learning style enum. Definition of learning styles
        supported by the system.
     */
    public static enum LearningStyle {
        visual,
        auditory,
        kinesthetic;
        public String s(){return this.toString();}
    }

    /**
     * Level enum. Definition of levels supported by the system.
     */
    public static enum Level {
        low(0,55),
        mid(56,75),
        advanced(76,100);
        private final int lowThreshold;
        private final int highThreshold;

        /**
         * Constructs a new <tt>Level</tt> enum with the
            specified low/high thresholds.
         *
         * @param lowThreshold the lower threshold.
         * @param highThreshold the high threshold.
         */
        Level(int lowThreshold,int highThreshold) {
            this.lowThreshold = lowThreshold;
            this.highThreshold = highThreshold;
        }
        public int lowThreshold(){return this.lowThreshold;}
        public int highThreshold(){return this.highThreshold;}

    }

    /**
     * The start page of the course.
     */
    public static final String startPage="/jsp/pedal-ng";

    /**
     * The in-memory index name of the course.
     */
    public static final String indexName="pedal-ng.index";

    /**
     * The session attribute name used to store course
        information in user's session.
     */
    public static final String course="pedal-ng.course";

    /**
     * The logical id of learning object used as prior knowledge
         assessment.
     */
    public static final String prior_knowledge_assessment="
        pre_assessment";

    /**
     * Identifier marker for currentModuleId.
     */
    public static final String currentModuleId="pedal-ng.
        currentModuleId";

    /**
     * Session attribute name for sequencing engine.
     */
    public static final String sequencingEngine="pedal-ng.
       sequencingEngine";

    /**
     * The learning style metadata element name
     */
    public static final String learningStyle="learning_style";

    /**
     * The level metadata element name
     */
    public static final String level="level";

    /**
     * Session attribute name to store  user's level for
        different modules.
     */
    public static final String moduleLevels ="moduleLevels";

    /**
     * Default logical id of the course.
     */
    public static  final String defaultLogicalTitle="pedal_ng";

    /**
     * Default marketplace id. This is required by Inspera
        Content Server.
     */
    public static  final long defaultMarketplaceId=629804;

    /**
     * The file path of the XSL that displays course. The path
        is absolute to the folder that contains the actual xsl
        file.
     */
    public final String xslPath=System.getProperty("webapp.path"
       ).concat(java.io.File.separator).concat("xsl").concat(
       java.io.File.separator);
}
```

no.inspera.services.pedal.PedalServlet

```
package no.inspera.services.pedal;

import no.inspera.applications.publish.datamodel.ContentRevision
   ;
import no.inspera.db.BaseObjectDataAccess;
import no.inspera.db.ContentPresentationDataAccess;
import no.inspera.services.languagenegotiator.LanguageNegotiator
    ;
import no.inspera.services.log.LogServices;
import no.inspera.services.pedal.actions.*;
import no.inspera.utils.Parameters;
import no.inspera.services.pedal.utils.ICollections;
import org.apache.log4j.Logger;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.util.Map;

/**
 * Servlet for personalising courses. This servlet acts as a
   dispatcher to all pages in the
 * personalised course.
 * <p>
 * It holds a list of possible actions that can be executed.
 * The actions implement {@link no.inspera.services.pedal.
   actions.Action} interface.
 * If this servlet is unable to identify the requested action
   the default action is login page.
 *  An example of url for the servlet:
 * <pre>
 *    http://&lt;hostname&gt;:&lt;port&gt;/pedalng?ac=view
 * </pre>
 *
 *  Which executes the view action(see {@link no.inspera.
   services.pedal.actions.View} for
 * more information).
 *
 * The View action displays the requested module.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
   </a>
 * @version $Id: PedalServlet.java 45069 2008-05-29 08:25:25Z
   naimdjon $
 * @since 20.12.2007
 */
public class PedalServlet extends HttpServlet {
    /**
     * Logging utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
```

```
     * Action map. Holds action id (string) and actual
        implementation.
     */
    private static Map<String, Action> pedalActions =
        ICollections.newMap();

    /**
     * Data access class for using content related methods.
     */
    private final ContentPresentationDataAccess cpda =
        ContentPresentationDataAccess.getInstance();

    static {
        pedalActions.put("idx", new Index());
        pedalActions.put("start", new Start());
        pedalActions.put("empty", new Empty());
        pedalActions.put("view", new View());
        pedalActions.put("login", new Login());
        pedalActions.put("showindex", new ShowIndex());
        pedalActions.put("tag", new TagObject());
        pedalActions.put("rate", new Rate());
    }

    /**
     * The entry gate to serve all requests.
     *
     * @param request the current request
     * @param response reponse object where the results will be
        written.
     * @throws ServletException if any error occurs.
     */
    public void service(HttpServletRequest request,
        HttpServletResponse response) throws ServletException {
        Parameters params = Parameters.create(request);
        final long mark = params.getMarketplaceId();
        if (mark <= 0) {
            params.setMarketplaceId(PEDAL.defaultMarketplaceId);
            params.setLanguageId(LanguageNegotiator.
                getHighestPriorityLanguageId(mark));
        }
        final String logicalTitle = params.getParameter("
            pedalCourseLogicalTitle",PEDAL.defaultLogicalTitle,
            true);
        log.debug("logicalname:" + logicalTitle);
        final BaseObjectDataAccess boda = BaseObjectDataAccess.
            getInstance();
        long contentItemId =boda.getObjectIdFromLogicalName(
            logicalTitle, mark);
        final long lang = params.getLanguageId();
        ContentRevision revision = cpda.
            getLiveContentRevisionWithEffectiveLanguage(
            contentItemId, lang, mark);
        log.debug("revisionrevision:"+revision);
        params.setRequestAttribute(PEDAL.course, revision);
        params.logParams();
        Action action = pedalActions.get(params.getParameter("ac
            "));
        if (action == null || params.getUserId()<=0) action =
            new Login();
        Result res = action.execute(params);
        res.processResult(request, response);
    }

}
```

no.inspera.services.pedal.Module

```
package no.inspera.services.pedal;

import no.inspera.services.log.LogServices;
import no.inspera.services.pedal.engine.SequencingState;
import no.inspera.services.pedal.utils.ICollections;
import org.apache.log4j.Logger;

import java.util.Collection;
import java.util.Map;

/**
 * Module represents a course module. Any given course consists
    of several logical modules.
 * <p>
 * Modules are instantiated when the user logs on to the system.
     The state of the module is not persisted,
 * which means every time the user logs on the modules are
    reinitialized. Therefore it is safe, to store the sequencing
     state which is specific to a user.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Module.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 20.12.2007
 */
public class Module {
    /**
     * Logger utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
     * the Id of the module. The modules are unique system-wide.
     */
    private long moduleId;

    /**
     * The example learning object of this module.
     */
    private LearningObject example;

    /**
     * The theory learning object of this module.
     */
    private LearningObject theory;

    /**
     * The exercise learning object of this module.
     */
    private LearningObject exercise;

    /**
     * A collection of unknown items.
     */
    private Collection<LearningObject> unknown= ICollections.
        newSet();

    /**
     * All items that are in this modules.
     */
    private Map<Long, LearningObject> allItems= ICollections.
        newMap();

    /**
     * The current state of the module.
     */
    private SequencingState currentState;

    /**
     * Constructs a new <tt>Module</tt> object with the
         specified moduleId
     * @param moduleId the id of module.
     */
    public Module(long moduleId) {
        this.moduleId = moduleId;
        this.currentState = new SequencingState(this);
    }

    /**
     * Adds this item to the list of available items within this
         module.
     *
     * @param learningObject item to be added
     */
    public void addItem(LearningObject learningObject){
        if(learningObject.isTheory())
            theory= learningObject;
        else if(learningObject.isExercise())
            exercise= learningObject;
        else if(learningObject.isExample())
            example= learningObject;
        else {
            log.debug("unknown:"+ learningObject);
            unknown.add(learningObject);
        }
        allItems.put(learningObject.getId(), learningObject);
    }

    /**
     * Looks up an item with specified id.
     *
     * @param id the id of an item (either theory, exercise or
        example)
     * @return an item matching specified id or null if there is
         no item with that id
     */
    public LearningObject getItem(long id ) {
        return allItems.get(id);
    }

    /**
     * Returns all items within this module.
     *
     * @return all items in this module.
     */
    public Collection<LearningObject> getAllItems() {
        return this.allItems.values();
    }

    /**
     * Getter for example.
     * @return example
     */
    public LearningObject getExample() {
        return example;
    }

    /**
     * Getter for theory
     * @return theory
     */
    public LearningObject getTheory() {
        return theory;
    }

    /**
     * Getter for exercise
     * @return exercise or null if exercise is not available
     */
    public LearningObject getExercise() {
        return exercise;
    }

    /**
     * Returns string representation of this object.
     * @return string representation of this object
     */
    public String toString() {
        return "ex:"+example +"\n exer:"+exercise+"\n theory:"+
            theory;
    }

    /**
     * Returns the current state of sequenced items.
     *
     * @return current state of sequenced items.
     */
    public SequencingState getCurrentState() {
        return currentState;
    }

    /**
     * Getter for moduleId
     *
     * @return moduleId
     */
    public long getModuleId() {
        return moduleId;
    }
}
```

no.inspera.services.pedal.LearningObject

```
package no.inspera.services.pedal;

```

```
import no.inspera.services.pedal.utils.StringUtils;

/**
 *
 * A <tt>LearningObject</tt> is an entity used for learning,
     education or training.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: LearningObject.java 45074 2008-05-29 10:04:54Z
     naimdjon $
 * @since 20.12.2007
 */
public class LearningObject {
    /**
     * Identifier of this LearningObject
     */
    private long id;

    /**
     * The level metadata element. LearningObject is tagged with
         different level (e.g. easy, intermediate, advanced.)
     */
    private String level;

    /**
     * The learningStyle  metadata element. LearningObject is
        tagged with any one of VAK entry (visual, auditory,
        kinesthetic.)
     */
    private String learningStyle;

    /**
     * The string defining the type of this LearningObject. The
        type can be example, image, video, sound or assessment.
     */
    private String type;

    /**
     * Boolean indicator whether or not this LearningObject is
        exercise.
     */
    private boolean isExercise;

    /**
     * Constructs a new LearningObject with specified id and
        type.
     *
     * @param id the identifier of this LearningObject.
     * @param type the content type of this LearningObject
     */
    public LearningObject(long id, String type) {
        this.id = id;
        this.type = type;
    }

    /**
     * Setter for id property
     * @param id new id property.
     */
    public void setId(long id) {
        this.id = id;
    }

    /**
     * Setter for level property
     * @param level new level property
     */
    public void setLevel(String level) {
        this.level = level;
    }

    /**
     * Setter for learningStyle property
     * @param learningStyle new learningStyle property
     */
    public void setLearningStyle(String learningStyle) {
        this.learningStyle = learningStyle;
    }

    /**
     * Setter for type property
     * @param type new type property
     */
    public void setType(String type) {
        this.type = type;
    }

    /**
     * Getter for Id
     *
     * @return Id of this LearningObject
     */
    public long getId() {
        return id;
    }

    /**
```

```
     * Getter for level property
     * @return level of this LearningObject
     */
    public String getLevel() {
        return level;
    }

    /**
     * Getter for learningStyle property
     * @return learningStyle of this LearningObject
     */
    public String getLearningStyle() {
        return learningStyle;
    }

    /**
     * Checks whether this LearningObject is example.
     * @return  true if this LearningObject has examlpe as type,
          false otherwise
     */
    public boolean isExample() {
        return StringUtils.equals(this.type, "
            content_document_example");
    }

    /**
     * Checks whether this LearningObject is theory.
     * @return  true if this LearningObject has theory as type,
         false otherwise
     */
    public boolean isTheory() {
        return StringUtils.equals(this.type, "
            content_document_theory");
    }

    /**
     * Checks whether this LearningObject is exercise.
     * @return  true if this LearningObject has exercise as type
         , false otherwise
     */
    public boolean isExercise() {
        return StringUtils.anyEquals(this.type, "
            content_activity_exercise", "content_exercise") ||
            isExercise;
    }

    /**
     * Setter for isExercise property
     *
```

```java
     * @param isExercise new value for isExercise property
     */
    public void setIsExercise(Boolean isExercise) {
        this.isExercise = isExercise;
    }

    /**
     * Returns string representation of this LearningObject
     * @return string representation of this LearningObject
     */
    public String toString() {
        return id + "-" + type;
    }

    /**
     * Indicates whether some other LearningObject is "equal to"
          this one.
     * <p>
     * More specifically, if two LearningObjects have the same <
          tt>Id</tt> identifier they are considered equal.
     * @param obj the other LearningObject
     * @return true if <tt>obj</tt> is equal to this object.
     */
    @Override
    public boolean equals(Object obj) {
        return obj != null && this.id == ((LearningObject) obj).
            id;
    }

    /**
     * Returns a hash code value for the LearningObject. This is
          among other things used to check the current state of
          the module.
     *
     * @return
     * @see no.inspera.services.pedal.engine.SequencingState
     */
    public int hashCode() {
        return Long.valueOf(id).hashCode();
    }


}
```

no.inspera.services.pedal.Tag

```java
package no.inspera.services.pedal;

import java.util.Calendar;
```

```java
/**
 *
 * A class representing a tag.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Tag.java 44417 2008-04-21 18:53:36Z naimdjon $
 * @since 23.02.2008
 */
public class Tag {

    /**
     * Descriptive label for the tag.
     */
    private String label;

    /**
     * The id of the object to be tagged.
     */
    private long objectId;

    /**
     * The Id of the user who tags
     */
    private long taggerId;

    /**
     * The time when tagging is performed.
     */
    private Calendar taggingTime;

    /**
     * Getter for label
     * @return label
     */
    public String getLabel() {
        return label;
    }

    /**
     * Setter for label
     * @param label new label
     */
    public void setLabel(String label) {
        this.label = label;
    }

    /**
```

```
     * Getter for objectId.
     *
     * @return objectId
     */
    public long getObjectId() {
        return objectId;
    }

    /**
     * Setter for objectId
     * @param objectId objectId
     */
    public void setObjectId(long objectId) {
        this.objectId = objectId;
    }

    /**
     * Getter for tagging time.
     * @return taggingTime
     */
    public Calendar getTaggingTime() {
        return taggingTime;
    }

    /**
     * Setter for taggingTime.
     * @param taggingTime new taggingTime
     */
    public void setTaggingTime(Calendar taggingTime) {
        this.taggingTime = taggingTime;
    }

    /**
     * Getter for taggerId
     * @return taggerId
     */
    public long getTaggerId() {
        return taggerId;
    }

    /**
     * Setter for taggerId
     * @param taggerId new taggerId
     */
    public void setTaggerId(long taggerId) {
        this.taggerId = taggerId;
    }
}
```

no.inspera.services.pedal.actions.Action

```java
package no.inspera.services.pedal.actions;

import no.inspera.utils.Parameters;

/**
 * An action is generic action to be performed in personalised
     course. Actions are invoked from {@link no.inspera.services.
     pedal.PedalServlet} servlet.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Action.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 14.12.2007
 */
public interface Action {

    /**
     * Executes the action based on parameters that are sent.
     *
     * @param params request parameters.
     * @return the result of the action.
     */
    public Result execute(Parameters params);

}
```

no.inspera.services.pedal.actions.Start

```java
package no.inspera.services.pedal.actions;

import no.inspera.services.pedal.PEDAL;
import no.inspera.utils.Parameters;

/**
 * The start is invoked once the user has been authenticate.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Start.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 29.12.2007
 */
public class Start implements Action {

    /**
```

```
     * Executes the action and forwards the user to the start
         page of the personalised course.
     * @param params the current request parameters.
     * @return to the start page of the course.
     */
    public Result execute(Parameters params) {
        params.removeSessionAttribute("moduleId");
        params.removeSessionAttribute(PEDAL.sequencingEngine);
        new Index().execute(params);
        try {
            new Configure(params.getMarketplaceId(), params.
                getLanguageId())
                     .execute(params);
        } catch (Exception e) {
            e.printStackTrace();
            params.setUserId(-1);
            params.setRequestAttribute("error","Could not
                configure personalised course");
            return new Forward("/jsp/pedal-ng/");
        }
        return new View().execute(params);
    }

}
```

no.inspera.services.pedal.actions.Empty

```
package no.inspera.services.pedal.actions;

import no.inspera.services.pedal.PEDAL;
import no.inspera.utils.Parameters;

/**
 * An empty action.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Empty.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 23.12.2007
 */
public class Empty implements Action {

    /**
     * An empty action.
     *
     * @param params current request parameters.
     * @return an empty {@link HTML} result
     */
    public Result execute(Parameters params) {
        return new HTML("<html><head><link rel=\"stylesheet\"
            type=\"text/css\" href=\""+ PEDAL.startPage+"/course.
            css\"/></head><body></body></html>");
    }
}
```

no.inspera.services.pedal.actions.Configure

```
package no.inspera.services.pedal.actions;

import no.inspera.applications.publish.datamodel.
    ContentActivityNode;
import no.inspera.applications.publish.datamodel.
    ContentActivityRoot;
import no.inspera.applications.publish.datamodel.ContentItem;
import no.inspera.applications.publish.datamodel.ContentRevision
    ;
import no.inspera.applications.publish.datamodel.assessment.*;
import no.inspera.applications.publish.util.PublishConstants;
import no.inspera.db.ContentPresentationDataAccess;
import no.inspera.db.MetadataDataAccess;
import no.inspera.services.log.LogServices;
import no.inspera.services.metadataservice.*;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.engine.*;
import no.inspera.utils.Parameters;
import no.inspera.services.pedal.utils.ICollections;
import org.apache.log4j.Logger;

import java.util.Collection;
import java.util.Map;

/**
 * A configuration action for the personalised course.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Configure.java 45069 2008-05-29 08:25:25Z
    naimdjon $
 * @since 23.12.2007
 */
public class Configure implements Action {
    /**
     * Logging utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());
    /**
```

```
     * Data access that provides database access functionality.
     */
    private final ContentPresentationDataAccess cpda =
       ContentPresentationDataAccess.getInstance();
    /**
     * The marketplace of the user.
     */
    private long userMarketplaceId;
    /**
     * Preferred language of the user
     */
    private long userLanguageId;

    /**
     * Constructs a new Configure object.
     * @param marketplaceId the marketplaceid of the user.
     * @param languageId the language of the user.
     */
    public Configure(long marketplaceId, long languageId){
        this.userMarketplaceId=marketplaceId;
        this.userLanguageId=languageId;
    }

    /**
     * Executes the Configure object which causes the engine to
        be configured. Usually this is done when the user has
        taken prior knowledge assessment and answered learning
        style questionnaire.
     *
     * @param params current request parameters.
     * @return the Result of the action.
     */
    public Result execute(Parameters params) {
        final Metadata md = MetadataService.getMetadataForObject
           (params.getUserId());
        final SequencingMetadataItem learningStyle =
           getMetadataItem(md, PEDAL.learningStyle);
        final SequencingMetadataItem level=getMetadataItem(md,
           PEDAL.level);
        if (learningStyle != null  /*&& level!=null*/) {
            if(setupModuleLevels(params)){
                Map<Long, PEDAL.Level>levels=params.
                    getSessionAttribute(PEDAL.moduleLevels);
            AdaptiveEngine se = EngineBuilder.create()
                    .setLearningStyle(learningStyle)
                            .setLevelMap(levels)
                            .addSequencingChain(new
                               LevelSequencing(level))
                    .addSequencingChain(new
                       LearningStyleSequencing(learningStyle))
                    .addEmpty()
                    .build();
                params.setSessionAttribute(PEDAL.
                   sequencingEngine, se);
            }
            //else log.warn("setup module levels returned false
               !!");
        }
        //else log.warn("did not find learning style and level
           for the user. Not configuring..."+(learningStyle)+",
           le:"+level);
        return new Empty().execute(params);
    }

    /**
     * Sets up module levels for the current user.
     *
     * @param params current request parameters.
     * @return true if the setup was finished successfully,
        false otherwise
     */
    private boolean setupModuleLevels(Parameters params) {
        ContentRevision revision = params.getRequestAttribute(
           PEDAL.course);
        Map<Long, PEDAL.Level> moduleLevels = ICollections.
           newMap();
        final ItemValue iv = MetadataService.
           getMetadataForObject(revision.getObjectId()).
           getItemValue(PEDAL.prior_knowledge_assessment);
        final ContentRevision assessment = cpda.
           getLiveContentRevisionWithEffectiveLanguage(Long.
           valueOf(iv.getValueText()), userLanguageId,
           userMarketplaceId);
        final QTIUserAssessment ua = AssessmentDataAccess.
           getInstance().getLastUserAssessment(assessment.
           getContentRevisionId(), params.getUserId());
        if(ua==null) {
             log.warn("user assessment not found!");
            return false;
        }
        final QTIUserAnswer[] useranswers = ua.getUserAnswers();
        ContentActivityRoot root = (ContentActivityRoot)
           revision.getContentItem();
        collectPreAssessmentScores(moduleLevels, useranswers,
           root);
        params.setSessionAttribute(PEDAL.moduleLevels,
           moduleLevels);
        return true;
    }

    /**
     * Collects prior knowledge assessment scores for specific
        modules.
     *
     * @param moduleLevels module to level mappings.
     * @param useranswers user's answers to questions in modules
        .
     * @param activityNode current module activity node.
     */
    private void collectPreAssessmentScores(Map<Long, PEDAL.
       Level> moduleLevels, QTIUserAnswer[] useranswers,
       ContentActivityNode activityNode) {
        for (Long childId : activityNode.getChildren()) {
            ContentItem ci = cpda.getContentItem(childId);
            ContentActivityNode can = (ContentActivityNode) ci;
            if (!can.isMandatory() && can.getTargetId() > 0) {
                ContentItem target = cpda.getContentItem(can.
                   getTargetId()); //folder...
                Iterable<QTIQuestion> questions = getQuestions(
                   target, userLanguageId, userMarketplaceId);
                int maxScore = 0, userScore = 0;
                for (QTIQuestion q : questions) {
                    maxScore = q.getMaxScore().intValue() +
                       maxScore;
                    userScore = getUserScore(q, useranswers) +
                       userScore;
                }
                if (maxScore > 0 && userScore > 0) {
                    final long normalizedScore = (long) ((Double
                       .valueOf(userScore) / Double.valueOf(
                       maxScore)) * 100);
                    log.debug("maxScore:" + maxScore + ",
                       userScore: " + userScore + ", for: " +
                       target.getName() + ", normalizedScore:" +
                        normalizedScore);
                    //if (normalizedScore > PEDAL.skipThreshold)
                    moduleLevels.put(can.getObjectId(),
                       calculateLevel(normalizedScore));
                }
            }
            collectPreAssessmentScores(moduleLevels,useranswers,
                can);
        }
    }

    /**
```

```
     * Calculates level based on the normalized score.
     *
     * @param normalizedScore the score that is normalized.
     * @return the level based on calculated score.
     * @see {@link no.inspera.services.pedal.PEDAL.Level}
     */
    private PEDAL.Level calculateLevel(long normalizedScore) {
        PEDAL.Level level=null;
        if (normalizedScore >= PEDAL.Level.low.lowThreshold() &&
             normalizedScore <= PEDAL.Level.low.highThreshold())
            level = PEDAL.Level.low;
        else if (normalizedScore >= PEDAL.Level.mid.lowThreshold
            () && normalizedScore <= PEDAL.Level.mid.
            highThreshold())
            level = PEDAL.Level.mid;
        else if (normalizedScore >= PEDAL.Level.advanced.
            lowThreshold() && normalizedScore <= PEDAL.Level.
            advanced.highThreshold())
            level = PEDAL.Level.advanced;
        return level;
    }

    /**
     * Gets the score of the user for the specified question
     * @param question the question.
     * @param useranswers the answers of the user.
     * @return the raw score for the question.
     */
    private int getUserScore(QTIQuestion question, QTIUserAnswer
        [] useranswers) {
        int userscore = 0;
        for (QTIUserAnswer ua : useranswers) {
            if (ua.getQuestionId() == question.getQuestionId())
                {
                final QTIAnswerOption ao = question.
                    getAnswerOption(ua.getAnswerOptionId());
                if (ao.getIsCorrect())
                    userscore = userscore + 1;
            }
        }
        return userscore;
    }

    /**
     * Returns an iterable object of questions for the content
        item.
     * @param contentItem the folder that contains all learning
        objects.
     * @param userLanguageId the language of the user.
     * @param marketplaceId the marketplaceid of the user.
     * @return an iterable of questions.
     */
    private Iterable<QTIQuestion> getQuestions(ContentItem
        contentItem, long userLanguageId, long marketplaceId) {
        Collection<QTIQuestion> questions = ICollections.newSet
            ();
        for (Long itemId : contentItem.getChildren()) {
            ContentItem item = cpda.getContentItem(itemId);
            if (item.getContentType().startsWith(
                PublishConstants.OBJECTTYPE_CONTENTQUESTION)) {
                final ContentRevision rev = cpda.
                    getLiveContentRevisionWithEffectiveLanguage(
                    item.getObjectId(), userLanguageId,
                    marketplaceId);
                if (rev instanceof QTIQuestion) {
                    final QTIQuestion q = (QTIQuestion) rev;
                    questions.add(q);
                }
            }
        }
        return questions;
    }

    /**
     * Constructs a new SequencingMetadataItem for the metadata
         element specified by <tt>itemproperty</tt>
     * @param md the metadata object
     * @param itemproperty requested itemproperty
     * @return new SequencingMetadataItem
     */
    private SequencingMetadataItem getMetadataItem(Metadata md,
        String itemproperty) {
        ItemProperty ip = MetadataService.getItemProperty(
            itemproperty);
        final ItemValue iv = md.getItemValue(ip.
            getItemPropertyId());
        if (iv == null) {
            return null;
        }

        final ItemPropertyValue ipv = ip.getItemPropertyValue(iv
            .getValueNumber());
        return new SequencingMetadataItem(ip, ipv);
    }


     public static String updateCompetenceLevel(long userId,
         String  totalScore, String maxPossibleScore) {
```

```
        try {
            if (userId <= 0) return "";
            String level = "";
            final long normalizedScore = (long) ((Double.valueOf(
                totalScore) / Double.valueOf(maxPossibleScore)) *
                100);
            if (normalizedScore <= PEDAL.Level.low.highThreshold
                ())
                level = "low";
            else if (normalizedScore >= PEDAL.Level.mid.
                lowThreshold() && normalizedScore <= PEDAL.Level.
                mid.highThreshold())
                level = "mid";
            else if (normalizedScore >= PEDAL.Level.advanced.
                highThreshold() && normalizedScore <= PEDAL.Level.
                advanced.highThreshold())
                level = "advanced";
            //log.info("updateCompetenceLevel totalScore:" +
                totalScore + ", userId:" + userId + ",
                maxPossibleScore:" + maxPossibleScore+",
                computedlevel:"+level+", normalizedscore:"+
                normalizedScore);
            final ItemProperty itemproperty = MetadataService.
                getItemProperty("level");
            final Metadata md = MetadataService.
                getMetadataForObject(userId);
            ItemValue iv = md.getItemValue(itemproperty.
                getItemPropertyId());
            for (ItemPropertyValue ipv : itemproperty.
                getValuesColl()) {
                if (ipv.getLogicalName().equals(level)) {
                    if (iv == null) {
                        iv = new ItemValue();
                        iv.setItemProperty(itemproperty.
                            getItemPropertyId());
                        iv.setObjectId(userId);
                        iv.setValueNumber(ipv.
                            getItemPropertyValueId());
                        MetadataDataAccess.getInstance().
                            addItemValueValueNumber(iv);
                    } else {
                        iv.setValueNumber(ipv.
                            getItemPropertyValueId());
                        MetadataDataAccess.getInstance().
                            updateItemValue(iv);
                    }
                    //log.debug("ipv.getLogicalName():"+ipv.
                        getLogicalName());
                    MetadataService.
                        removeMetadataForObjectFromCache(userId);
                }
            }
        } catch (Throwable e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

no.inspera.services.pedal.actions.Index

```
package no.inspera.services.pedal.actions;

import no.inspera.applications.publish.datamodel.*;
import no.inspera.applications.publish.util.PublishConstants;
import no.inspera.db.ContentPresentationDataAccess;
import no.inspera.services.log.LogServices;
import no.inspera.services.metadataservice.ItemPropertyValue;
import no.inspera.services.metadataservice.ItemValue;
import no.inspera.services.metadataservice.Metadata;
import no.inspera.services.metadataservice.MetadataService;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.utils.StringUtils;
import no.inspera.utils.Parameters;
import no.inspera.services.pedal.utils.ICollections;
import org.apache.log4j.Logger;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.document.Field;
import org.apache.lucene.index.IndexWriter;
import org.apache.lucene.store.RAMDirectory;

import java.io.IOException;
import java.util.Collection;

/**
 * An Index action causes in-memory indexing the whole course
    structure.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Index.java 45074 2008-05-29 10:04:54Z naimdjon
    $
 * @since 18.12.2007
 */
public class Index implements Action{
    /**
```

```
     * Logging utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
     * The class that provides data access functionality.
     */
    private final ContentPresentationDataAccess cpda =
        ContentPresentationDataAccess.getInstance();

    /**
     * Executes the action which results in indexing the course
         structure.
     * @param params current request parameters.
     * @return Result of the index. An empty XML string to be
         transformed.
     */
    public Result execute(Parameters params) {
        RAMDirectory idx = new RAMDirectory();
        try {
            IndexWriter writer =new IndexWriter(idx, new
                StandardAnalyzer(), true);
            try {
                ContentRevision revision=(ContentRevision)
                    params.getRequestAttribute(PEDAL.course);
                ContentActivityRoot root=(ContentActivityRoot)
                    revision.getContentItem();
                Collection<Indexable> col= ICollections.newSet()
                    ;
                for (Long child : root.getChildren())
                    addIndexables(child,params.getLanguageId(),
                        params.getMarketplaceId(),col);
                for (Indexable ind : col) {
                    Document doc= new Document();
                    doc.add(new Field(PEDAL.Index.parentId.
                        toString(),""+ ind.parentId,Field.Store.
                        YES,Field.Index.UN_TOKENIZED));
                    doc.add(new Field(PEDAL.Index.id.toString(),
                        ""+ ind.id,Field.Store.YES,Field.Index.
                        UN_TOKENIZED));
                    doc.add(new Field(PEDAL.Index.title.toString
                        (), ind.title,Field.Store.YES,Field.Index
                        .TOKENIZED));
                    doc.add(new Field(PEDAL.Index.type.toString
                        (), ind.type,Field.Store.YES,Field.Index.
                        TOKENIZED));
                    for (ItemValue iv : ind.md.getItemValues())
                        {
```

```
                        final ItemPropertyValue[] values = iv.
                            getValues();
                        if(values!=null && values.length>0) {
                            for(ItemPropertyValue val: values){
                                if(val.isSet())
                                    doc.add(new Field(iv.
                                        getItemProperty(),val.
                                        getLogicalName()+"",Field
                                        .Store.YES,Field.Index.
                                        UN_TOKENIZED));
                            }
                        }else if (StringUtils.hasLength(iv.
                            getValueText())) {
                            doc.add(new Field(iv.getItemProperty
                                (),iv.getValueText(),Field.Store.
                                YES,Field.Index.UN_TOKENIZED));
                        }

                        //final ItemPropertyValue value =
                            MetadataService.getItemProperty(iv.
                            getItemProperty()).
                            getItemPropertyValue(iv.
                            getValueNumber());
                        //doc.add(new Field(iv.getItemProperty()
                            ,value.getLogicalName()+"",Field.
                            Store.YES,Field.Index.UN_TOKENIZED));
                    }
                    writer.addDocument(doc);
                }
                log.debug("gathered indexables. Added ["+col.
                    size()+"] documents to the index");
            } finally {
                writer.optimize();
                writer.close();
            }
            params.setSessionAttribute(PEDAL.indexName,idx);
        } catch (IOException e) {
            e.printStackTrace();
            params.setRequestAttribute("error",e.getMessage());
        }

        return new XMLTransform("<content/>");
    }

    /**
     * Adds indexable objects to the ilst of indexables.
     *
     * @param child the id of item that is mapped to the module.
     * @param languageid the language of the user.
     * @param mpId the marketplace id of the user.
     * @param indexables collection of indexables.
     */
    private void addIndexables(Long child,long languageid,long
        mpId,final Collection<Indexable> indexables) {
        ContentRevision revision = cpda.
            getLiveContentRevisionWithEffectiveLanguage(child,
            languageid, mpId);
        final ContentItem ci = revision.getContentItem();
        if (ci instanceof ContentActivityNode) {
            ContentActivityNode can=(ContentActivityNode) ci;
            final long contentItemId = can.getTargetId();
            if (contentItemId > 0) {
                final ContentItem folder = cpda.getContentItem(
                    contentItemId);
                if (folder instanceof ContentFolder) {
                    ContentFolder f=(ContentFolder) folder;
                    for (Long item : f.getChildren()) {
                        ContentRevision rev=cpda.
                            getLiveContentRevisionWithEffectiveLanguage
                            (item, languageid, mpId);
                        final ContentItem revItem = rev.
                            getContentItem();
                        if (!revItem.getContentType().startsWith
                            (PublishConstants.
                            OBJECTTYPE_CONTENTIMAGE)) {
                            final Metadata md = MetadataService.
                                getMetadataForObject(rev.
                                getObjectId());
                            indexables.add(new Indexable(rev.
                                getTitle(),rev.getObjectType(),
                                revItem.getObjectId(),
                                contentItemId,md));
                        }
                    }
                }
            }
            for (Long ch : can.getChildren())
                addIndexables(ch,languageid,mpId,indexables);
        }else
            log.warn("revision.getContentItem():"+ ci.getClass()
                .getName());
    }


    /**
     * Class that holds indexing information.
     */
    private class Indexable{
        /**
         * The title of LearningObject
         */
        private String title;
        /**
         * The type of learning object.
         */
        private String type;
        /**
         * The id of learning object.
         */
        private long id;
        /**
         * The folder containing this learning object.
         */
        private long parentId;
        /**
         * Metadata associated with this learning object.
         */
        private Metadata md;

        /**
         * Constructs a new Indexable object.
         * @param title the title of the learning object
         * @param type the type of the learning object
         * @param id the identifier of the learning object
         * @param parentId the folder containing  learning
            object
         * @param md the metadata associated with the learning
            object
         */
        private Indexable(String title, String type, long id,
            long parentId, Metadata md) {
            this.title = title;
            this.type = type;
            this.id = id;
            this.parentId = parentId;
            this.md = md;
        }
    }
}
```

no.inspera.services.pedal.actions.Login

```
package no.inspera.services.pedal.actions;

import no.inspera.services.pedal.PEDAL;
import no.inspera.utils.Parameters;
```

```
/**
 * The entry point to the personalised course. This class also
    responsible for ensuring security.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Login.java 45069 2008-05-29 08:25:25Z naimdjon
    $
 * @since 23.12.2007
 */
public class Login implements Action {
    /**
     * Executes login action which checks if the user is logged
        in and cofigures the personalised course.
     * @param params the current request parameters.
     * @return the result of the action.
     */
    public Result execute(Parameters params) {
        if(params.getUserId()<=0 || params.getMarketplaceId()
            <=0) {
            return new Forward(PEDAL.startPage);
        }else if(params.getSessionAttribute(PEDAL.indexName)==
            null){
            return new Start().execute(params);
        }else if (params.getSessionAttribute(PEDAL.
            sequencingEngine) == null) {
            new Configure(params.getMarketplaceId(),params.
                getLanguageId()).execute(params);
        }
        //params.setRequestAttribute("user",UserDataAccess.
            getInstance().getUser(params.getUserId()));
        return new View().execute(params);
    }
}
```

no.inspera.services.pedal.actions.Rate

```
package no.inspera.services.pedal.actions;

import no.inspera.db.BaseObjectDataAccess;
import no.inspera.db.MetadataDataAccess;
import no.inspera.services.metadataservice.MetadataTemplate;
import no.inspera.services.pedal.services.RatingService;
import no.inspera.services.rating.Rating;
import no.inspera.utils.Parameters;

/**
 * This action is invoked when the user wants to rate the
    content.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
   </a>
 * @version $Id: Rate.java 45069 2008-05-29 08:25:25Z naimdjon $
 * @since 03.01.2008
 */
public class Rate implements Action {

    /**
     * Rates the content. The actual rating is done on metadata
       associated with Learning Object.
     * The result of this action is an empty html if the rating
       was successful or an error code is sent back to client
       in case an error occurs.
     *
     * @param params the current request parameters.
     * @return to the start page of the course.
     */
    public Result execute(Parameters params) {
        try {
            final long objectId = params.getLong("objectId");
            final String logicalName = "metadata_template_for_"
               + objectId;
            long metadataTemplateId = BaseObjectDataAccess.
               getInstance().getObjectIdFromLogicalName(
               logicalName, params.getMarketplaceId());
            if (metadataTemplateId <=1) {
                MetadataTemplate md= new MetadataTemplate();
                md.setMarketplaceId(params.getMarketplaceId());
                md.setLogicalName(logicalName);
                metadataTemplateId=MetadataDataAccess.
                   getInstance().newMetadataTemplate(md,objectId
                   );
            }
            Rating r= new Rating();
            r.setObjectId(metadataTemplateId);
            r.setUserId(params.getUserId());
            r.setScore(params.getInt("score"));
            r.setRatedDate(System.currentTimeMillis());
            RatingService.rate(r);
        } catch (Exception e) {
            e.printStackTrace();
            params.setUserId(-1);
            params.setRequestAttribute("error","Could not
               configure personalised course");
        }
        return new HTML("<html/>");
    }

}
```

no.inspera.services.pedal.actions.ShowIndex

```
package no.inspera.services.pedal.actions;

import no.inspera.services.pedal.PEDAL;
import no.inspera.utils.Parameters;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.document.Field;
import org.apache.lucene.queryParser.QueryParser;
import org.apache.lucene.search.Hits;
import org.apache.lucene.search.IndexSearcher;
import org.apache.lucene.search.Searcher;
import org.apache.lucene.store.RAMDirectory;

import java.util.List;

/**
 *
 * This action shows what is in-memory indexed.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: ShowIndex.java 45069 2008-05-29 08:25:25Z
    naimdjon $
 * @since 23.12.2007
 */
public class ShowIndex implements Action {

    /**
     * Executes the show index action and prints out all objects
         in the index.
     *
     * @param params the current request parameters.
     * @return the html result showing index.
     */
    public Result execute(Parameters params) {
        StringBuffer html= new StringBuffer();
        RAMDirectory index=(RAMDirectory) params.
            getSessionAttribute(PEDAL.indexName);
        try {
            Searcher searcher = new IndexSearcher(index);
            final Hits hits =searcher.search(new QueryParser("id
                ", new StandardAnalyzer()).parse("id:[2000 TO
                99999999]"));
            for (int i = 0; i < hits.length(); i++) {
                final Document doc = hits.doc(i);
                for (Field f : (List<Field>) doc.getFields()) {
                    html.append("<div>");
                    html.append(f.name()).append(":").append(f.
                       stringValue());
                    html.append("</div>");
                }
                html.append("<hr/>");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        return new HTML(html.toString());
    }
}
```

no.inspera.services.pedal.actions.View

```
package no.inspera.services.pedal.actions;

import no.inspera.applications.publish.datamodel.
   ContentActivityNode;
import no.inspera.applications.publish.datamodel.
   ContentActivityRoot;
import no.inspera.applications.publish.datamodel.ContentRevision
   ;
import no.inspera.applications.publish.structure.
   StructureElementConverter;
import no.inspera.db.ContentPresentationDataAccess;
import no.inspera.services.log.LogServices;
import no.inspera.services.pedal.LearningObject;
import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.engine.AdaptiveEngine;
import no.inspera.services.pedal.utils.StringUtils;
import no.inspera.utils.Parameters;
import org.apache.log4j.Logger;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.queryParser.ParseException;
import org.apache.lucene.queryParser.QueryParser;
import org.apache.lucene.search.Hits;
import org.apache.lucene.search.IndexSearcher;
import org.apache.lucene.search.Searcher;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.RAMDirectory;

import java.io.IOException;
import java.util.Map;
```

```java
/**
 * The View action displays the module. The sequencing engine is
     used to ensure the correct sequencing of learning objects.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: View.java 45074 2008-05-29 10:04:54Z naimdjon $
 * @since 23.12.2007
 */
public class View implements Action {

    /**
     * Logging utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
     * The class that provides data access functionality.
     */
    private final ContentPresentationDataAccess db =
        ContentPresentationDataAccess.getInstance();

    /**
     * Executes action and displays the module page with the
         next LearningObject.
     * @param params the current request parameters.
     * @return result of the action. The result is xml
         transformation that displays a learning object in the
         module.
     */
    public Result execute(Parameters params) {
        ContentRevision revision = params.getRequestAttribute(
            PEDAL.course);
        ContentActivityRoot root = (ContentActivityRoot)
            revision.getContentItem();
        final AdaptiveEngine engine = params.getSessionAttribute
            (PEDAL.sequencingEngine);
        StringBuffer xml = new StringBuffer("<?xml version
            =\"1.0\" encoding=\"ISO-8859-1\" ?>");
        xml.append("<content>\n");
        try {
            long moduleId = params.getLong("moduleId");
            if (moduleId > 0) {
                if (engine != null && params.getRequest().
                    getRequestURI().contains("/clear")) {
                    log.debug("refreshing module:"+moduleId);
                    engine.getModule(moduleId).getCurrentState().refresh();
                }
                ContentActivityNode ci = (ContentActivityNode) db.getContentItem(moduleId);
                log.debug("ci.isMandatory():"+ci.isMandatory());
                log.debug("ci.getTargetId() :"+ci.getTargetId());
                if (ci.isMandatory() && ci.getTargetId() > 0) // pre/post assessments and learning style questionnaire.
                {
                    xml.append("<forwardContent mandatory=\"true\">").append(ci.getTargetId()).append("</forwardContent>");
                    if (engine != null)
                        engine.getModule(moduleId).getCurrentState().markCurrent(new LearningObject(ci.getTargetId(), ci.getContentType()));
                } else if ((ci.getTargetId() > 0 || ci.hasChildren()) && engine!=null) {
                    viewModule(params, engine, xml, moduleId, ci);
                }
            }
            if (engine == null) xml.append("<noAssessments/>");
            if (engine != null) {
                xml.append("<learningstyle>").append(engine.getLearningStyleValue()).append("</learningstyle>");
                xml.append("<level>").append(engine.getLevelValue(moduleId)).append("</level>");
            }
            StructureElementConverter.appendAsActivityPlanXML(root, params.getUserId(), xml);
            //showIndex(params);
        } catch (Exception e) {
            log.error("Exception", e);
        }
        appendModuleLevels(xml,params);
        params.appendPageInfoXML(xml);
        xml.append("</content>\n");
        return new XMLTransform(xml.toString());
    }

    /**
     * Gets the next entry in the chain for the current module.
     *
     * @param params current parameters
     * @param engine the engine
     * @param xml xml buffer
     * @param moduleId current moduleid
     * @param ci activity node
     * @throws IOException if any IO exception occurs.
     * @throws ParseException if any exception occurs when
        parsing query
     */
    private void viewModule(Parameters params, AdaptiveEngine
       engine, StringBuffer xml, long moduleId,
       ContentActivityNode ci) throws IOException,
       ParseException {
        StringBuffer queryBuf = new StringBuffer("+(");
        queryBuf.append("parentId:").append(ci.getTargetId());
        queryBuf.append(")");
        final StringBuffer query = queryBuf.append(" +").append(
           engine.getLearningStyleProp()).append(":").append(
           engine.getLearningStyleValue());
        final String levelValue = engine.getLevelValue(moduleId)
           ;
        if(StringUtils.hasLength(levelValue))
            query.append(" +").append(PEDAL.level).append(":").
               append(levelValue);
        final Module module = engine.getModule(moduleId);
        final Hits hits = search(query.toString(), (RAMDirectory
           ) params.getSessionAttribute(PEDAL.indexName));
        log.debug("hits.length():" + hits.length());
        if (hits.length() > 1) {
            addHits(engine, module, hits);
            LearningObject currentItem = engine.sequence(module)
               ;
            if (module.getCurrentState().isAllProcessed()  ||
               currentItem ==null) {
                xml.append("<allProcessed/>");
                if(currentItem!=null)
                    xml.append("<forwardContent>").append(
                       currentItem.getId()).append("</
                       forwardContent>");
            }else {
                log.info("same module:"+currentItem.getId());
                xml.append("<sameModule/>");
                xml.append("<forwardContent>").append(
                   currentItem.getId()).append("</forwardContent
                   >");
            }
        } else if (hits.length() == 1) {
            processSingleHit(engine, xml, ci, module, hits);
        }
    }

    /**
     * Processes search result with single hit.
     *
     * @param engine the engine
     * @param xml xml buffer
     * @param ci activity node
     * @param module current module.
     * @param hits search hits
     * @throws IOException if any IO error occurs.
     */
    private void processSingleHit(AdaptiveEngine engine,
       StringBuffer xml, ContentActivityNode ci, Module module,
       Hits hits) throws IOException {
        engine.sequence(module);
        if (module.getCurrentState().isAllProcessed()) xml.
           append("<allProcessed/>");
        final Document firstHit = hits.doc(0);
        final String id = firstHit.getField("id").stringValue();
        log.debug("id:"+id);
        xml.append("<forwardContent>").append(id).append("</
           forwardContent>");
        module.getCurrentState().markCurrent(new LearningObject(
           Long.valueOf(id), ci.getContentType()));
    }


    /**
     * Appends user's level for specific module.
     *
     * @param xml the xml buffer
     * @param params the current request parameters.
     */
    private void appendModuleLevels(StringBuffer xml, Parameters
        params) {
        Map<Long, PEDAL.Level>  moduleLevels= params.
           getSessionAttribute( PEDAL.moduleLevels);
        if(moduleLevels==null || moduleLevels.size()<1)return;

        xml.append("<moduleLevels>");
        for (Map.Entry<Long, PEDAL.Level> en : moduleLevels.
           entrySet())
            xml.append("<module id=\"").append(en.getKey()).
               append("\" level=\"").append(en.getValue()).
               append("\"/>");
        xml.append("</moduleLevels>");
    }

    /**
     * Adds hits to the module.
     *
     * @param engine the sequencing engine
     * @param module the current module.
     * @param hits search hits for the module.
     * @throws IOException if any error occurs when accessing
       documents in  hits.
     */
    private void addHits(AdaptiveEngine engine, Module module,
        Hits hits) throws IOException {
        for (int i = 0; i < hits.length(); i++) {
            final Document doc = hits.doc(i);
            final LearningObject contentItem = new
                LearningObject(Long.valueOf(doc.get(PEDAL.Index.
                id.toString())), doc.get(PEDAL.Index.type.
                toString()));
            contentItem.setIsExercise(Boolean.valueOf(doc.get("
                is_exercise")));
            contentItem.setLearningStyle((doc.get(engine.
                getLearningStyleProp())));
            module.addItem(contentItem);
        }
    }

    /**
     * Performs in-memory search for the learning objects
       matching query specified by <tt>query</tt>.
     *
     * @param query the query string.
     * @param dir the directory containing lucene index.
     * @return search hits
     * @throws IOException if any error occurs accessing the
       index
     * @throws ParseException if any error occurs when parsing
       the search query.
     */
    private Hits search(String query, Directory dir) throws
        IOException, ParseException {
        log.debug("query:" + query);
        Searcher searcher = new IndexSearcher(dir);
        return searcher.search(new QueryParser("id", new
            StandardAnalyzer()).parse(query));
    }
}
```

no.inspera.services.pedal.actions.View

```
package no.inspera.services.pedal.actions;
```

```java

import no.inspera.applications.publish.datamodel.
    ContentActivityNode;
import no.inspera.applications.publish.datamodel.
    ContentActivityRoot;
import no.inspera.applications.publish.datamodel.ContentRevision
    ;
import no.inspera.applications.publish.structure.
    StructureElementConverter;
import no.inspera.db.ContentPresentationDataAccess;
import no.inspera.services.log.LogServices;
import no.inspera.services.pedal.LearningObject;
import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.engine.AdaptiveEngine;
import no.inspera.services.pedal.utils.StringUtils;
import no.inspera.utils.Parameters;
import org.apache.log4j.Logger;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.queryParser.ParseException;
import org.apache.lucene.queryParser.QueryParser;
import org.apache.lucene.search.Hits;
import org.apache.lucene.search.IndexSearcher;
import org.apache.lucene.search.Searcher;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.RAMDirectory;

import java.io.IOException;
import java.util.Map;

/**
 * The View action displays the module. The sequencing engine is
     used to ensure the correct sequencing of learning objects.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: View.java 45074 2008-05-29 10:04:54Z naimdjon $
 * @since 23.12.2007
 */
public class View implements Action {

    /**
     * Logging utility.
     */
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
```

```
     * The class that provides data access functionality.
     */
    private final ContentPresentationDataAccess db =
        ContentPresentationDataAccess.getInstance();

    /**
     * Executes action and displays the module page with the
        next LearningObject.
     * @param params the current request parameters.
     * @return result of the action. The result is xml
        transformation that displays a learning object in the
        module.
     */
    public Result execute(Parameters params) {
        ContentRevision revision = params.getRequestAttribute(
            PEDAL.course);
        ContentActivityRoot root = (ContentActivityRoot)
            revision.getContentItem();
        final AdaptiveEngine engine = params.getSessionAttribute
            (PEDAL.sequencingEngine);
        StringBuffer xml = new StringBuffer("<?xml version
            =\"1.0\" encoding=\"ISO-8859-1\" ?>");
        xml.append("<content>\n");
        try {
            long moduleId = params.getLong("moduleId");
            if (moduleId > 0) {
                if (engine != null && params.getRequest().
                    getRequestURI().contains("/clear")) {
                    log.debug("refreshing module:"+moduleId);
                    engine.getModule(moduleId).getCurrentState()
                        .refresh();
                }
                ContentActivityNode ci = (ContentActivityNode)
                    db.getContentItem(moduleId);
                log.debug("ci.isMandatory():"+ci.isMandatory());
                log.debug("ci.getTargetId() :"+ci.getTargetId()
                    );
                if (ci.isMandatory() && ci.getTargetId() > 0) //
                     pre/post assessments and learning style
                    questionnaire.
                {
                    xml.append("<forwardContent mandatory=\"true
                        \">").append(ci.getTargetId()).append("</
                        forwardContent>");
                    if (engine != null)
                        engine.getModule(moduleId).
                            getCurrentState().markCurrent(new
                            LearningObject(ci.getTargetId(), ci.
                            getContentType()));
                } else if ((ci.getTargetId() > 0 || ci.
                   hasChildren()) && engine!=null) {
                    viewModule(params, engine, xml, moduleId, ci
                       );
                }
            }
            if (engine == null) xml.append("<noAssessments/>");
            if (engine != null) {
                xml.append("<learningstyle>").append(engine.
                   getLearningStyleValue()).append("</
                   learningstyle>");
                xml.append("<level>").append(engine.
                   getLevelValue(moduleId)).append("</level>");
            }
            StructureElementConverter.appendAsActivityPlanXML(
               root, params.getUserId(), xml);
            //showIndex(params);
        } catch (Exception e) {
            log.error("Exception", e);
        }
        appendModuleLevels(xml,params);
        params.appendPageInfoXML(xml);
        xml.append("</content>\n");
        return new XMLTransform(xml.toString());
    }

    /**
     * Gets the next entry in the chain for the current module.
     *
     * @param params current parameters
     * @param engine the engine
     * @param xml xml buffer
     * @param moduleId current moduleid
     * @param ci activity node
     * @throws IOException if any IO exception occurs.
     * @throws ParseException if any exception occurs when
        parsing query
     */
    private void viewModule(Parameters params, AdaptiveEngine
       engine, StringBuffer xml, long moduleId,
       ContentActivityNode ci) throws IOException,
       ParseException {
        StringBuffer queryBuf = new StringBuffer("+(");
        queryBuf.append("parentId:").append(ci.getTargetId());
        queryBuf.append(")");
        final StringBuffer query = queryBuf.append(" +").append(
           engine.getLearningStyleProp()).append(":").append(
           engine.getLearningStyleValue());
        final String levelValue = engine.getLevelValue(moduleId)
            ;
        if(StringUtils.hasLength(levelValue))
            query.append(" +").append(PEDAL.level).append(":").
                append(levelValue);
        final Module module = engine.getModule(moduleId);
        final Hits hits = search(query.toString(), (RAMDirectory
            ) params.getSessionAttribute(PEDAL.indexName));
        log.debug("hits.length():" + hits.length());
        if (hits.length() > 1) {
            addHits(engine, module, hits);
            LearningObject currentItem = engine.sequence(module)
                ;
            if (module.getCurrentState().isAllProcessed()  ||
                currentItem ==null) {
                xml.append("<allProcessed/>");
                if(currentItem!=null)
                    xml.append("<forwardContent>").append(
                        currentItem.getId()).append("</
                        forwardContent>");
            }else {
                log.info("same module:"+currentItem.getId());
                xml.append("<sameModule/>");
                xml.append("<forwardContent>").append(
                    currentItem.getId()).append("</forwardContent
                    >");
            }
        } else if (hits.length() == 1) {
            processSingleHit(engine, xml, ci, module, hits);
        }
    }

    /**
     * Processes search result with single hit.
     *
     * @param engine the engine
     * @param xml xml buffer
     * @param ci activity node
     * @param module current module.
     * @param hits search hits
     * @throws IOException if any IO error occurs.
     */
    private void processSingleHit(AdaptiveEngine engine,
        StringBuffer xml, ContentActivityNode ci, Module module,
        Hits hits) throws IOException {
        engine.sequence(module);
        if (module.getCurrentState().isAllProcessed()) xml.
            append("<allProcessed/>");
        final Document firstHit = hits.doc(0);
```

```java
        final String id = firstHit.getField("id").stringValue();
        log.debug("id:"+id);
        xml.append("<forwardContent>").append(id).append("</forwardContent>");
        module.getCurrentState().markCurrent(new LearningObject(Long.valueOf(id), ci.getContentType()));
    }


    /**
     * Appends user's level for specific module.
     *
     * @param xml the xml buffer
     * @param params the current request parameters.
     */
    private void appendModuleLevels(StringBuffer xml, Parameters params) {
        Map<Long, PEDAL.Level>  moduleLevels= params.getSessionAttribute( PEDAL.moduleLevels);
        if(moduleLevels==null || moduleLevels.size()<1)return;

        xml.append("<moduleLevels>");
        for (Map.Entry<Long, PEDAL.Level> en : moduleLevels.entrySet())
            xml.append("<module id=\"").append(en.getKey()).append("\" level=\"").append(en.getValue()).append("\"/>");
        xml.append("</moduleLevels>");
    }

    /**
     * Adds hits to the module.
     *
     * @param engine the sequencing engine
     * @param module the current module.
     * @param hits search hits for the module.
     * @throws IOException if any error occurs when accessing documents in  hits.
     */
    private void addHits(AdaptiveEngine engine, Module module, Hits hits) throws IOException {
        for (int i = 0; i < hits.length(); i++) {
            final Document doc = hits.doc(i);
            final LearningObject contentItem = new LearningObject(Long.valueOf(doc.get(PEDAL.Index.id.toString())), doc.get(PEDAL.Index.type.toString()));
            contentItem.setIsExercise(Boolean.valueOf(doc.get("is_exercise")));
```

```
                contentItem.setLearningStyle((doc.get(engine.
                    getLearningStyleProp())));
                module.addItem(contentItem);
            }
    }

    /**
     * Performs in-memory search for the learning objects
         matching query specified by <tt>query</tt>.
     *
     * @param query the query string.
     * @param dir the directory containing lucene index.
     * @return search hits
     * @throws IOException if any error occurs accessing the
         index
     * @throws ParseException if any error occurs when parsing
         the search query.
     */
    private Hits search(String query, Directory dir) throws
        IOException, ParseException {
        log.debug("query:" + query);
        Searcher searcher = new IndexSearcher(dir);
        return searcher.search(new QueryParser("id", new
            StandardAnalyzer()).parse(query));
    }
}
```

no.inspera.services.pedal.actions.TagObject

```
package no.inspera.services.pedal.actions;

import no.inspera.utils.Parameters;
import no.inspera.services.pedal.Tag;
import no.inspera.services.pedal.services.TaggingService;

import java.util.Calendar;

/**
 *
 * Tags object with specific tags.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Tag.java 44417 2008-04-21 18:53:36Z naimdjon $
 * @since 23.02.2008
 */
public class TagObject implements Action {

    public Result execute(Parameters params) {
        try {
            Tag tag= new Tag();
            tag.setLabel(params.getParameter("tag"));
            tag.setObjectId(params.getLong("objectId"));
            tag.setTaggerId(params.getUserId());
            tag.setTaggingTime(Calendar.getInstance());
            TaggingService.tagObject(tag);
        } catch (Exception e) {
            e.printStackTrace();
            params.setRequestAttribute("error","Could not tag
                object");
        }
        //redirect user to the same page
        return new View().execute(params);
    }
}
```

no.inspera.services.pedal.actions.Result

```
package no.inspera.services.pedal.actions;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * This object represents a result of the action in the
     personalised course page.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Result.java 45069 2008-05-29 08:25:25Z naimdjon
     $
 * @since 15.12.2007
 */
public interface Result {

    /**
     * Processes result.
     *
     * @param request current request.
     * @param response current reponse.
     * @throws ServletException if any exception occurs when
         forwarding.
     */
     public void processResult(HttpServletRequest request,
         HttpServletResponse response)throws ServletException;

}
```

no.inspera.services.pedal.actions.Forward

```java
package no.inspera.services.pedal.actions;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 *
 * Forward result forwards user to some URL.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Forward.java 45069 2008-05-29 08:25:25Z
    naimdjon $
 * @since 15.12.2007
 */
public class Forward implements Result{

    /**
     * Constructs a new Forward object with the url
     * @param url the url to which user is to be forwarded.
     */
    protected Forward(String url){
        this.forwardURL=url;
    }

    /**
     * the forwarding url
     */
    private String forwardURL;

    /**
     * Processes result which causes forwarding user to the url.
     *
     * @param request current request.
     * @param response current reponse.
     * @throws ServletException if any exception occurs when
        forwarding.
     */
    public void processResult(HttpServletRequest request,
       HttpServletResponse response)throws ServletException {
        try {
            response.setHeader("Cache-Control","no-cache"); //
               HTTP 1.1
            response.setHeader("Pragma","no-cache"); //HTTP 1.0
            response.setDateHeader ("Expires", 0); //prevents
               caching at the proxy server
            if(!response.isCommitted())
            request.getRequestDispatcher(this.forwardURL).
               forward(request, response);
        } catch (IOException e) {
            e.printStackTrace();
            throw new ServletException(e);
        }
    }

    /**
     * Returns string representation of this object.
     * @return string representation of this object.
     */
    public String toString() {
        return "Forward:"+this.forwardURL;
    }

}
```

no.inspera.services.pedal.actions.HTML

```
package no.inspera.services.pedal.actions;

import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * HTML action simply prints out html to the response.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: HTML.java 45069 2008-05-29 08:25:25Z naimdjon $
 * @since 23.12.2007
 */
public class HTML implements Result{

    /**
     * HTML result content
     */
    private String html;


    /**
```

```
     * Constructs a new HTML object with the specified html
        content
     * @param html the html content
     */
    protected HTML(String html){
        this.html =html;
    }

    /**
     * Processes result which causes printing an HTML content to
         the response.
     *
     * @param req current request.
     * @param res current reponse.
     * @throws ServletException if any exception occurs when
        forwarding.
     */
    public void processResult(HttpServletRequest req,
       HttpServletResponse res)throws ServletException {
        try {
            res.setHeader("Cache-Control","no-cache"); //HTTP
                1.1
            res.setHeader("Pragma","no-cache"); //HTTP 1.0
            res.setDateHeader ("Expires", 0); //prevents caching
                 at the proxy server
            final ServletOutputStream out = res.getOutputStream
                ();
            try {
                out.println(html);
            } finally {
                out.flush();
                out.close();
            }

        } catch (java.io.IOException e) {
            e.printStackTrace();
            throw new ServletException(e);
        }
    }

    /**
     * Returns string representation of  this HTML object.
     * @return string representation of the html
     */
    public String toString() {
        return "HTML:"+this.html;
    }

}
```

no.inspera.services.pedal.actions.XMLTransform

```java
package no.inspera.services.pedal.actions;

import no.inspera.services.pedal.PEDAL;

import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stream.StreamSource;
import java.io.FileInputStream;
import java.io.StringReader;

/**
 * The XMLTransform transforms xmls into HTML pages using XSLT.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: XMLTransform.java 45069 2008-05-29 08:25:25Z
    naimdjon $
 * @since 16.12.2007
 */
public class XMLTransform implements Result{

    /**
     * The xml string to be transformed.
     */
    private String xml;

    /**
     * Constructs a new XMLTransform with the specified xml to
        be transformed.
     * @param xml xml  string to be transformed.
     */
    protected XMLTransform(String xml){
        this.xml=xml;
    }

    /**
     * Processes result which results in actual xml
        transformation.
     *
     * @param request current request.
     * @param response current reponse.
     * @throws ServletException if any exception occurs when
        forwarding.
     */
    public void processResult(HttpServletRequest request,
       HttpServletResponse response)throws ServletException {
        response.setHeader("Cache-Control","no-cache"); //HTTP
            1.1
        response.setHeader("Pragma","no-cache"); //HTTP 1.0
        response.setDateHeader ("Expires", 0); //prevents
            caching at the proxy server
        try {

            final ServletOutputStream out = response.
                getOutputStream();

            if(request.getRequestURI().indexOf("xml")>-1){
                response.setContentType("text/xml");
                out.write(xml.getBytes());
                out.flush();
                out.close();
                return;
            }

            final String xslFile = PEDAL.xslPath + "
                Common_Adaptive_CourseDisplay.xsl";
            final StreamSource xslSource = new StreamSource(new
                FileInputStream(xslFile));
            xslSource.setSystemId(xslFile);
            Transformer tr = TransformerFactory.newInstance().
                newTemplates(xslSource).newTransformer();
            tr.transform(new StreamSource(new StringReader(xml))
                , new javax.xml.transform.stream.StreamResult(out
                ));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
     * Returns string representation of this object.
     * @return string representation of this object.
     */
    public String toString() {
        return "XML:"+this.xml;
    }

}
```

no.inspera.services.pedal.engine.Chain

```
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.Module;

/**
 * A SequencingChain is an object provided by the sequencing
    engine giving a view into the invocation chain of a sequence
     request for a module.
 *
 * Sequences use the SequencingChain to invoke the next sequece
    object  in the chain.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: Chain.java 45069 2008-05-29 08:25:25Z naimdjon
    $
 * @since 27.12.2007
 */
public class Chain {

    /**
     * The sequence object
     */
    protected Sequence seq;

    /**
     * The next sequence in the chain
     */
    protected Chain next;


    /**
     * Causes the next sequence in the chain to be invoked.
     *
     * @param module the module to pass along the chain
     */
    public void doSequence(Module module){
        if(seq!=null)
            seq.doSequence(next, module);
    }

    /**
     * Returns the string representation of this object. Prints
        out objects in the sequence chain.
     *
     * @return the string representation of this object.
     */
    public String toString() {
        StringBuffer s=new StringBuffer();
        s.append("seq:").append(seq)
                .append("next:").append(next);
        return s.toString();
    }
}
```

no.inspera.services.pedal.engine.AdaptiveEngine

```
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.LearningObject;
import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.utils.StringUtils;
import no.inspera.services.pedal.utils.ICollections;

import java.util.Map;

/**
 * AdaptiveEngine is the engine that adapts LearningObject in
     the sequencing chain.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: AdaptiveEngine.java 45074 2008-05-29 10:04:54Z
     naimdjon $
 * @since 23.12.2007
 */
public class AdaptiveEngine {
    /**
     * A map of modules in the course identified by moduleId key
         and a Module object.
     */
    private final Map<Long,Module> modules= ICollections.
        newHashtable();

    /**
     * The sequencing chain.
     */
    protected Chain chain;

    /**
     * The user's learning style.
     */
    protected SequencingMetadataItem learningStyle;
    /**
     * A map of user level for each module. Modules are
        represented by moduleId key.
     */
    protected Map<Long, PEDAL.Level> levels;

    /**
     * Returns a logical identifier for user's learning style.
     * @return user's learning style.
     */
    public String getLearningStyleValue(){
        return learningStyle.getValue().getLogicalName();
    }

    /**
     * Returns user's level for the specified module.
     *
     * @param moduleId module identifier
     * @return user's level for the specified module.
     */
    public String getLevelValue(long  moduleId){
        final PEDAL.Level level = levels.get(moduleId);
        return StringUtils.checkNull(level);
    }

    /**
     * Returns the learning style metadata element.
     * @return the learning style metadata element.
     */
    public String getLearningStyleProp(){
        return learningStyle.getProperty().getItemPropertyId();
    }

    /**
     * Returns the module identified by specified moduleId.
     * @param moduleId the identifier of the module.
     * @return  Module
     */
    public Module getModule(long moduleId){
        Module module = modules.get(moduleId);
        if (module == null) {
            module= new Module(moduleId);
            modules.put(moduleId,module);
        }
        return module;
    }

    /**
     * Returns string representation of this object
     * @return string representation of this object
     */
    public String toString() {
        return chain.toString();
    }

    /**
     * Causes the sequencing of LearningObjects in the chain.
     *
     * @param module LearningObjects in the module to be
           sequenced
     * @return next LearningObject
     */
    public LearningObject sequence(Module module) {
        chain.doSequence(module);
        module.getCurrentState().checkAllProcessed(module.
            getAllItems());
        return module.getCurrentState().getCurrentItem();
    }
}
```

no.inspera.services.pedal.engine.EngineBuilder

```
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.PEDAL;

import java.util.Map;

/**
 *
 * SequencingBuilder uses builder pattern to build
     SequencingEngine object.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: EngineBuilder.java 45069 2008-05-29 08:25:25Z
     naimdjon $
 * @since 27.12.2007
 */
public class EngineBuilder {

    /**
     * the sequencing engine to be built.
     */
    private AdaptiveEngine se;

    /**
```

```
     * The first entry in the SequencingChain.
     */
    private Chain start;

    /**
     * The last configured entry in SequencingChain.
     */
    private Chain lastConfiguredChain;

    /**
     * Creates  a new SequencingBuilder.
     * @return new SequencingBuilder.
     */
    public static EngineBuilder create(){
        final EngineBuilder builder = new EngineBuilder();
        builder.se= new AdaptiveEngine();
        return builder;
    }

    /**
     * Sets the learning style of a user to the sequencing
        engine.
     * @param learningStyle the user learning style.
     * @return SequencingBuilder
     */
    public EngineBuilder setLearningStyle(SequencingMetadataItem
        learningStyle) {
        this.se.learningStyle=learningStyle;
        return this;
    }

    /**
     * Sets the mapping of module to learner level.
     *
     * @param levelMap the level mapping.
     * @return SequencingBuilder
     */
    public EngineBuilder setLevelMap(Map<Long,PEDAL.Level>
       levelMap) {
        this.se.levels =levelMap;
        return this;
    }

    /**
     * Adds an empty sequence to the chain.
     * @return SequencingBuilder
     */
    public EngineBuilder addEmpty() {
        Sequence empty= new Sequence() {
                public void doSequence(Chain chain, Module module) {
                    //System.out.println("empty");
                }

                public String toString(){
                    return "empty anonymous inner ";
                }

            };
            return addSequencingChain(empty);
        }

        /**
         * Adds the specified implementation of <tt>Sequence</tt> to
             the sequencing chain.
         * @param seq the sequence to be added to the chain.
         * @return SequencingBuilder
         */
        public EngineBuilder addSequencingChain(Sequence seq) {
            Chain chain=new Chain();
            chain.seq=seq;
            if(this.lastConfiguredChain !=null) {
               this.lastConfiguredChain.next = chain;
            } else{
                this.start=chain;
                chain.next= new Chain();
            }
            this.lastConfiguredChain=chain;
            return this;
        }

        /**
         * This method is usually called when we are done with
            building a sequencing engine.
         * @return SequencingEngine that has been built.
         */
        public AdaptiveEngine build(){
            this.se.chain=start;
            //System.out.println("this.se.chain:"+this.se.chain);
            return se;
        }


}
```

no.inspera.services.pedal.engine.LearningStyleSequencing

```
package no.inspera.services.pedal.engine;
```

```
import no.inspera.services.log.LogServices;
import no.inspera.services.pedal.LearningObject;
import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.PEDAL;
import no.inspera.services.pedal.utils.ICollections;
import org.apache.log4j.Logger;

import java.util.Map;
import java.util.Queue;

/**
 *
 * This class is an implementation of <tt>Sequence</tt>
     interface. It is reponsible for sequencing LearningObjects
     based on learning style.
 * <p>
 * At the time of implementation the VAK learning style model is
      supported.
 * <ul>
 *          <li>visual: the sequencing is done as follows: 1)
     theory, 2) example, 3) exercise.</li>
 *          <li>auditory: the sequencing is done as follows: 1)
     theory, 2) example, 3) exercise.</li>
 *          <li>kinesthetic: the sequencing is done as follows:
     1) exercise, 2) example , 3) theory .</li>
 * </ul>
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: LearningStyleSequencing.java 45069 2008-05-29
     08:25:25Z naimdjon $
 * @since 27.12.2007
 */
public class LearningStyleSequencing implements Sequence {
    private final transient Logger log = LogServices.getLogger(
        getClass());

    /**
     * A map of learning styles. The key is string identifier
         for learning style and the LearningObject collector is
         stored as value
     */
    private Map<String, LearningStyleContentCollector>
        contentCollectorMap = ICollections.newMap();

    /**
     * The learning style of the user.
     */
    private SequencingMetadataItem learningStyle;
    /**
     * Constructs a new LearningStyleSequencing object for the
       specified user learningStyle
     * @param learningStyle user's learning style.
     */
    public LearningStyleSequencing(SequencingMetadataItem
        learningStyle) {
        this.learningStyle = learningStyle;
        contentCollectorMap.put(PEDAL.LearningStyle.auditory.s()
            , new AuditoryContentCollector());
        contentCollectorMap.put(PEDAL.LearningStyle.visual.s(),
            new VisualContentCollector());
        contentCollectorMap.put(PEDAL.LearningStyle.kinesthetic.
            s(), new KinestheticContentCollector());
    }

    /**
     * The implementation of doSequence method.
     * @param next the next Sequence object in the chain-
     * @param module the current module.
     */
    public void doSequence(Chain next, Module module) {
        Queue<LearningObject> q = new java.util.LinkedList<
            LearningObject>();
        SequencingState currentState = module.getCurrentState();
        final LearningStyleContentCollector collector =
            contentCollectorMap.get(learningStyle.getValue().
            getLogicalName());
        collector.addContentItems(q, module);
        Queue<LearningObject> newQueue = new java.util.
            LinkedList<LearningObject>();
        log.debug("new queue:"+q);
        for (LearningObject learningObject : q) {
            log.debug(learningObject.getId()+" processed:"+
                currentState.isProcessed(learningObject)+"....");
            if (!currentState.isProcessed(learningObject)) {
                newQueue.offer(learningObject);
            }
        }
        log.debug("newQueue:"+newQueue);
        final LearningObject peek = newQueue.peek();
        log.debug("peek:"+peek);
        if(peek==null)
            currentState.refresh();
         else
            currentState.markCurrent(peek);
        next.doSequence(module);
    }

    /**
     * Returns string representation of this object.
     * @return string representation of this object.
     */
    public String toString() {
        return "learning style\n";
    }

    /**
     * An interface used to abstract the collection process of
        items.
     */
    private interface LearningStyleContentCollector {
        /**
         * Add contentitems that are in the module to the queue,
             that is create a sequence of items based on
            learning style.
         * @param queue the queue of items.
         * @param module the module.
         */
        void addContentItems(Queue<LearningObject> queue, Module
            module);
    }

    /**
     * LearningStyleCollector for users with visual learning
        style preference.
     */
    private class VisualContentCollector implements
        LearningStyleContentCollector {

        /**
         * Add contentitems that are in the module to the queue,
             that is create a sequence of items based on
            learning style.
         * <p>
         * The sequencing is done as follows:
         * <ol>
         * <li>theory</li>
         * <li>example.</li>
         * <li> exercise.</li>
         * </ol>
         * @param queue the queue of items.
         * @param module the module.
         */
        public void addContentItems(Queue<LearningObject> queue,
            Module module) {
            if (module.getTheory() != null) {
```

```java
                queue.add(module.getTheory());
            }
            if (module.getExample() != null) {
                queue.add(module.getExample());
            }
            if (module.getExercise() != null)
                queue.add(module.getExercise());
        }
    }

    /**
     * LearningStyleCollector for users with auditory learning style preference.
     */
    private class AuditoryContentCollector implements LearningStyleContentCollector {
        /**
         * Add contentitems that are in the module to the queue, that is create a sequence of items based on learning style.
         * <p>
         * The sequencing is done as follows:
         * <ol>
         * <li>theory</li>
         * <li>example.</li>
         * <li> exercise.</li>
         * </ol>
         * @param queue the queue of items.
         * @param module the module.
         */
        public void addContentItems(Queue<LearningObject> queue, Module module) {

            if (module.getTheory() != null) {
                queue.add(module.getTheory());
            }
            log.debug("module.getExample():"+module.getExample());
            if (module.getExample() != null) {
                queue.add(module.getExample());
            }
            log.debug("module.getExercise()"+module.getExercise());
            if (module.getExercise() != null)
                queue.add(module.getExercise());
        }
    }

    /**
```

```
     * LearningStyleCollector for users with kinesthetic
        learning style preference.
     */
    private class KinestheticContentCollector implements
        LearningStyleContentCollector {
        /**
         * Add contentitems that are in the module to the queue,
             that is create a sequence of items based on
            learning style.
         * <p>
         * The sequencing is done as follows:
         * <ol>
         * <li> exercise.</li>
         * <li>example.</li>
         * <li>theory</li>
         * </ol>
         * @param queue the queue of items.
         * @param module the module.
         */
        public void addContentItems(Queue<LearningObject> queue,
            Module module) {
            if (module.getExercise() != null)
                queue.add(module.getExercise());

            if (module.getExample() != null) {
                queue.add(module.getExample());
            }
            if (module.getTheory() != null) {
                queue.add(module.getTheory());
            }
        }
    }
}
```

no.inspera.services.pedal.engine.LevelSequencing

```
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.Module;

/**
 * Sequnces Learning Objects based on level metadata.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: LevelSequencing.java 45069 2008-05-29 08:25:25Z
      naimdjon $
 * @since 27.12.2007
 */
```

```java
public class LevelSequencing implements Sequence {
    private SequencingMetadataItem level;

    /**
     * Constructs a new LevelSequencing based on level metadata.
     * @param level level metadata
     */
    public LevelSequencing(SequencingMetadataItem level) {
        this.level = level;
    }

    /**
     * Filters out only items that are in accordance with the
          specified level.
     *
     * @param next the next Sequence in the chain
     * @param module current module
     */
    public void doSequence(Chain next, Module module) {
        next.doSequence(module);
    }

    /**
     * Returns string representation of this object
     * @return string representation of this object
     */
    public String toString() {
        return "level\n";
    }

}
```

no.inspera.services.pedal.engine.Sequence

```java
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.Module;

/**
 * The <tt>Sequence</tt> interface does sequencing of
     LearningObject that are in a module.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: Sequence.java 45069 2008-05-29 08:25:25Z
     naimdjon $
 * @since 27.12.2007
 */
public interface Sequence {

    /**
     *  Sequences LearningObjects that are in the module.
     *
     * @param chain the sequencing chain
     * @param module the current module
     */
    public void doSequence(Chain chain, Module module);

}
```

no.inspera.services.pedal.engine.SequencingMetadataItem

```java
package no.inspera.services.pedal.engine;

import no.inspera.services.metadataservice.ItemProperty;
import no.inspera.services.metadataservice.ItemPropertyValue;

import java.io.Serializable;

/**
 * SequencingMetadataItem represents metadata item that can be
     used for sequencing.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: SequencingMetadataItem.java 45069 2008-05-29
     08:25:25Z naimdjon $
 * @since 23.12.2007
 */
public class SequencingMetadataItem implements Serializable{
    /**
     * The metadata element
     */
    private ItemProperty property;

    /**
     * The property value for the metadata element.
     */
    private ItemPropertyValue value;

    /**
     *  Constructs a new SequencingMetadataItem with specified
         parameters.
     *
     * @param property metadata element
     * @param value the property value
     */
    public SequencingMetadataItem(ItemProperty property,
       ItemPropertyValue value) {
        this.property = property;
        this.value = value;
    }

    /**
     * Getter for property metadata element
     * @return property
     */
    public ItemProperty getProperty() {
        return property;
    }

    /**
     * Setter for value
     * @return value.
     */
    public ItemPropertyValue getValue() {
        return value;
    }

}
```

no.inspera.services.pedal.engine.SequencingState

```
package no.inspera.services.pedal.engine;

import no.inspera.services.pedal.LearningObject;
import no.inspera.services.pedal.Module;
import no.inspera.services.pedal.utils.ICollections;

import java.util.Collection;
import java.util.concurrent.locks.ReadWriteLock;
import java.util.concurrent.locks.ReentrantReadWriteLock;

/**
 * SequencingState that holds the sequencing information for a
    module.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
    </a>
 * @version $Id: SequencingState.java 45069 2008-05-29 08:25:25Z
     naimdjon $
 * @since 27.12.2007
 */
public class SequencingState {
    //private final transient org.apache.log4j.Logger log = no.
       inspera.services.log.LogServices.getLogger(getClass());
    /**
     * The currentItem in the queue
     */
    private LearningObject currentItem;

    /**
     * The module this state belongs to
     */
    private Module module;

    /**
     * the boolean flag indicates whether all items are
         processed
     */
    private boolean allProcessed=false;

    /**
     * Collection of processed items in the module.
     */
    private java.util.Collection<LearningObject> processedItems=
         ICollections.newSet();

    /**
     * Constructs a new <tt>SequencingState</tt> for the
         specified module
     * @param module the module of this <tt>SequencingState</tt>
     */
    public SequencingState(Module module) {
        this.module = module;
    }

    /**
     * Returns the current item in the queue under processing.
     * @return the current item in the queue under processing.
     */
    public synchronized LearningObject getCurrentItem() {
          return currentItem;
    }

    /**
     * Marks the specifed item as current in the queue of
         processing items for the module
     *
     * @param learningObject the item to be marked as current
     */
    public synchronized void markCurrent(LearningObject
        learningObject) {
        if(learningObject ==null) {
            System.out.printf("null item attempted to be marked
                for module: %d", module.getModuleId() );
            return;
        }
        ReadWriteLock rwl = new ReentrantReadWriteLock();
        try {
            rwl.writeLock().lock();
            this.currentItem = learningObject;
            this.processedItems.add(currentItem);
        } finally {
            rwl.writeLock().unlock();
        }
        this.checkAllProcessed(this.module.getAllItems());
    }

    /**
     * Checks whether all items in the module are processed.
     *
     * @return true if all items are processed, false otherwise.
     */
    public synchronized boolean isAllProcessed() {
        return this.allProcessed;
    }

    /**
     * Refreshes the queue of processing items by removing
         current item. Calling this method will result in fresh
         sequencing state..
     */
    public synchronized void refresh() {
        ReadWriteLock rwl = new ReentrantReadWriteLock();
        try {
            rwl.readLock().lock();
            this.currentItem=null;
            this.processedItems.clear();
            this.allProcessed=false;
        } finally {
            rwl.readLock().unlock();
        }
    }

    /**
     * Checks whether the specified item is processed.
     *
     * @param learningObject item to be checked
     * @return true if the item is processed, false otherwise.
     */
    public synchronized boolean isProcessed(LearningObject
        learningObject) {
```

```
        ReadWriteLock rwl = new ReentrantReadWriteLock();
        try {
            rwl.readLock().lock();
            return allProcessed
                            || this.processedItems.contains(
                                learningObject);
        } finally {
            rwl.readLock().unlock();
        }
    }

    /**
     * Checks whether all items in the specified collection are
        processed.
     *
     * @param learningObjects collection of items to be checked.
     */
    public synchronized void checkAllProcessed(Collection<
       LearningObject> learningObjects) {
        ReadWriteLock rwl = new ReentrantReadWriteLock();
        try {
            rwl.readLock().lock();
            this.allProcessed= this.processedItems.containsAll(
                learningObjects);
        } finally {
            rwl.readLock().unlock();
        }
    }
}
```

no.inspera.services.pedal.utils.ICollections

```
package no.inspera.services.pedal.utils;

import java.util.*;

/**
 * Utility class to create generic collections. This is usefull
    and makes the code somewhat cleaner when the generic types
    in collection
 * include long class names and/or include nested collections.
    Example:<br/><br/>
 * <code>
 *  Map&lt;Long,Map&lt;String,Map&lt;Long,Long>>> myMap= new
    HashMap&lt;Long,Map&lt;String,Map&lt;Long,Long>>>();
 * </code>
 *      <br/><br/>
 * can be written: <br/><br/>
 * <code>
 * Map&lt;Long,Map&lt;String,Map&lt;Long,Long>>> myMap=
     ICollections.newMap();
 * </code>
 * <br/><br/>
 * which would be essentially the same, only with cleaner
     declaration.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: ICollections.java 43983 2008-03-25 00:06:35Z
     naimdjon $
 * @since 28.08.2007
 */
public class ICollections {

    /**
     * Creates a new instance of HashMap
     *
     * @return new map.
     */
     public static <K, V> Map<K, V> newMap() {
        return new HashMap<K, V>();
    }

    /**
     * Constructs an empty <tt>HashMap</tt> with the specified
         initial
     * capacity and the default load factor (0.75).
     *
     * @return hashmap.
     * @param initialcapacity the initial capacity
     */
     public static <K, V> HashMap<K, V> newHashMap(int
         initialcapacity) {
        return new HashMap<K, V>(initialcapacity);
    }

    /**
     * Creates a new instance of hashtable.
     * @return hashtable
     */
     public static <K, V> Hashtable<K, V> newHashtable() {
        return new Hashtable<K, V>();
    }

    /**
     * Constructs a new set containing the elements in the
         specified collection.
     *
     * @param c the collection whose elements are to be placed
        into this set.
     * @return new set
     */
    public static <E> Set<E> newHashSet(Collection<? extends E>
         c) {
        return new HashSet<E>(c);
    }

    /**
     * Constructs a new set.
     *
     * @return new set
     */
    public static <E> Set<E> newSet() {
        return new HashSet<E>();
    }

    /**
     * Constructs a new list.
     *
     * @return new list
     */
    public static <E> List<E> newList() {
        return new ArrayList<E>();
    }

    /**
     * Constructs a new list.
     *
     * @return new list
     */
    public static <E> Queue<E> linkedList() {
        return new LinkedList<E>();
    }

    /**
     * Constructs and returns a new  vector.
     *
     * @return new vector
     */
    public static <E> Vector<E> newVect() {
        return new Vector<E>();
    }

    /**
     * Returns a synchronized (thread-safe) map backed by the
        specified
     * map.  In order to guarantee serial access, it is critical
        that
     * <strong>all</strong> access to the backing map is
       accomplished
     * through the returned map.<p>
     *
     * </pre>
     * Failure to follow this advice may result in non-
       deterministic behavior.
     *
     * <p>The returned map will be serializable if the specified
        map is
     * serializable.
     *
     * @return a synchronized view of HashMap.
     */
    public static <K, V> Map<K, V> synchronizedMap() {
        return Collections.synchronizedMap(new HashMap<K, V>());
    }

    public static <K, V> SortedMap<K, V> sortedMap() {
        return new TreeMap<K,V>();
    }

}
```

no.inspera.services.pedal.utils.StringUtils

```
package no.inspera.services.pedal.utils;

import java.io.PrintWriter;
import java.io.StringWriter;
import java.util.regex.Pattern;
/**
 * Utility class for working with {@link String} class.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
     </a>
 * @version $Id: StringUtils.java 44171 2008-04-02 21:39:14Z
     naimdjon $
 * @since 20.12.2007
 */
public class StringUtils {
    public static final String ENCODING = System.getProperty("
        file.encoding");
    private static final transient Pattern numberPattern =
        Pattern.compile("-?[0-9]+(\\.[0-9]+)?([Ee][\\+-]?[0-9]+)?
        $");
```

```java
    /**
     * Compares this string to the specified object.
     * The result is <code>true</code> if and only if both arguments are
     * <code>null</code> or <code>s1</code> is a <code>String</code> object that represents
     * the same sequence of characters as <code>s2</code> object.
     *
     * @param s1 first String
     * @param s2 second String
     * @return <code>true</code> if both arguments are null or {@link String#equals(Object)} returns true
     */
    public static boolean equals(String s1, String s2) {
        return s1 == null ? s2 == null : s1.equals(s2);
    }

    /**
     * Compares <tt>firstString</tt> to the other specified strings. The method permits null <tt>firstString</tt> argument and any of <tt>strings</tt> can be null as well.
     *
     * @param firstString the first string to be compared
     * @param strings an array of strings to be compared with <tt>firstString</tt>
     * @return true if any of <tt>strings</tt> equals to <tt>firstString</tt>.
     */
    public static boolean anyEquals(String firstString, String ... strings) {
        if(strings==null || strings.length<=0)
            return false;

        for (String string : strings)
            if (equals(string, firstString))
                return true;
        return false;
    }

    public static boolean anyEqualsIgnoreCase(String firstString, String ... strings) {
        if(strings==null || strings.length<=0)
            return false;

        for (String string : strings)
            if (equals(string.toLowerCase(), firstString.toLowerCase()))
```

```
                return true;
        return false;
    }

    public static boolean anyStartsWithIgnoreCase(String
       firstString, String ... strings) {
        if(strings==null || strings.length<=0) {
            return false;
        }

        for (String string : strings) if (startsWith(firstString
           .toLowerCase(), string.toLowerCase())) {
            return true;
        }
        return false;
    }

    /**
     * Prints and returns a string that contains  the stack
        trace of this exception
     *
     * @param t throwable that needs to be printed
     * @return a string containing stack trace.
     */
    public static String stackTraceToString(Throwable t) {
        StringWriter str = new StringWriter();
        PrintWriter wr = new PrintWriter(str);
        t.printStackTrace(wr);
        return str.toString();
    }


    /**
     * Checks if the string is null. Returns an empty string if
        this string is null or the string itself otherwise.
     * <p> This method is usefull when we would like to avoid
        null strings.
     *
     * @param s string to be checked.
     * @return an empty string if this string is null or the
        string itself otherwise.
     */
    public static String checkNull(String s) {
        return s == null ? "" : s;
    }

    public static String checkNull(Object s) {
        return s == null ? "" : s.toString();
    }
```

```java


    /**
     * Appends the following value as tag specified by <tt>
        tagName</tt> if value is not null
     *
     * @param tagName the tag name of xml element
     * @param value   the value to be appended.
     * @param s       stringbuffer to be appended to.
     */
    public static void appendIfNotNull(String tagName, Object
        value, StringBuffer s) {
        if (value == null) {
            return;
        }
        s.append("<").append(tagName).append(">").append(value).
            append("</").append(tagName).append(">");
    }

    /**
     * Encodes the given string with the default system encoding
        .
     *
     * @param str string to be encoded.
     * @return encoded string.
     */
    public static String encode(String str) {
        if (str == null)
            return "";

        try {
            return java.net.URLEncoder.encode(str, ENCODING);
        } catch (java.io.UnsupportedEncodingException e) {
            e.printStackTrace(System.err);
            return "";
        }
    }

    /**
     * Decodes the string with default system encoding.
     *
     * @param str string to be decoded
     * @return decoded string.
     */
    public static String decode(String str) {
        if (str == null)
            return "";

        try {
            return java.net.URLDecoder.decode(str, ENCODING);
        } catch (java.io.UnsupportedEncodingException e) {
            e.printStackTrace(System.err);
            return "";
        }
    }

    /**
     * Checks if the specified string represents a number.
         Usefull for parsing strings that are numbers.
     *
     * @param str string representing number
     * @return true if the <tt>str</tt> is a number, false
         otherwise
     */
    public static boolean isNumber(String str) {
        return str != null && str.length() > 0 && numberPattern.
            matcher(str.trim()).matches();
    }

    /**
     * Checks if the type of the Object is string and checks if
         it can be parsed as a Number.
     *
     * @param obj Object to be checked.
     * @return true if it can be parsed as a Number.
     */
    public static boolean isParseableNumber(Object obj) {
        return obj != null && (obj instanceof String) &&
            isNumber((String) obj);
    }

    /**
     * Checks if the specified string is true. The check is not
         case-cesitive. The values to be checked true|false are
         (1|0, true|false, on|off).
     *
     * @param str string to be checked
     * @return true if the str is true, false otherwise.
     */
    public static boolean isTrue(String str) {
        return hasLength(str)
                && (str.equalsIgnoreCase("1")
                            || str.equalsIgnoreCase("on")
                            || str.equalsIgnoreCase("true"));
    }

    /**
```

```
     * Checks if the specified string is null or not. NOTE: This
        method trims the string before checking whether or not
       it is empty, thus
     * sending an empty string will result in false.
     *
     * @param s string to be checked
     * @return true if the string is empty and false otherwise.
     */
    public static boolean empty(Object s) {
        return s == null || s.toString() == null || s.toString()
            .trim().length() <= 0;
    }

    public static boolean startsWith(String s, String prefix) {
        return s != null && s.startsWith(prefix);
    }


    /**
     * Checks if the specified string is null or not. NOTE: This
        method trims the string before checking whether or not
       it is empty, thus
     * sending an empty string will result in false.
     *
     * @param s string to be checked
     * @return true if the string is not empty and false
       otherwise.
     */
    public static boolean hasLength(String s) {
        return s != null && s.trim().length() > 0;
    }

    /**
     * Tests if this string ends with the specified suffix.
     * @param original
     * @param suffixes the suffixes.
     * @return  <code>true</code> if the character sequence
       represented by the
     *          <tt>suffixes</tt> argument is a suffix of the
       character sequence represented by
     *          this object; <code>false</code> otherwise. Note
       that the
     *          result will be <code>true</code> if the argument
        is the
     *          empty string or is equal to this <code>String</
       code> object
     *          as determined by the {@link #equals(Object)}
       method.
     */
    public static boolean endsWith(String original, String ...
        suffixes) {
        if(!hasLength(original) || (suffixes==null || suffixes.
            length<=0)) {
            return false;
        }

        for (String s : suffixes) {
            if(original.endsWith(s)) {
                            return true;
                        }
        }

        return false;
    }

    public static String capitalize(String s) {
        if(!hasLength(s)) {
            return s;
        }

        return s.substring(0, 1).toUpperCase().concat( s.
            substring(1).toLowerCase());
    }
}
```

no.inspera.services.pedal.services.RatingService

```
package no.inspera.services.pedal.services;

import no.inspera.db.BaseObjectDataAccess;
import no.inspera.db.exception.DataAccessException;
import no.inspera.services.rating.Rating;

import java.sql.Connection;
import java.sql.PreparedStatement;

/**
 *
 * Service class for rating. This class contains methods to rate
    an object and retrieve a set of ratings related to a
   specific object or metadata.
 *
 * @author <a href="mailto:naimdjon@gmail.com">Naimdjon Takhirov
   </a>
 * @version $Id: RatingService.java 44417 2008-04-21 18:53:36Z
   naimdjon $
 * @since 23.02.2008
 */
```

```java
public class RatingService {

    /**
     * Stores rating to the db.
     *
     * @param rating the rating object that is to be stored in
        the database.
     * @throws no.inspera.db.exception.DataAccessException if
        any data access error occurs.
     */
    public static void rate(Rating rating) throws
       DataAccessException{
        BaseObjectDataAccess bo=BaseObjectDataAccess.getInstance
            ();
        final Connection con = bo.getConnection();
        PreparedStatement ps= null;
        try {
            int counter=1;
            ps=con.prepareStatement("insert into rating(objectid
                ,userid,rateddate,score) values(?,?,?,?)");
            ps.setLong(counter++,rating.getObjectId());
            ps.setLong(counter++,rating.getUserId());
            ps.setLong(counter++,rating.getRatedDate());
            ps.setInt(counter,rating.getScore());
            final int i = ps.executeUpdate();
            if(i!=1) {
                throw new DataAccessException("Could not insert
                    rating to the database");
            }
        } catch (Exception e){
            e.printStackTrace();
        }finally{
            bo.close(ps);
            bo.close(con);
        }

    }
}
```

no.inspera.services.pedal.services.TaggingService

```java
package no.inspera.services.pedal.score.services;

import no.inspera.services.pedal.Tag;
import no.inspera.services.pedal.utils.ICollections;
import no.inspera.db.BaseObjectDataAccess;
import no.inspera.db.exception.DataAccessException;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Collection;


/**
 * Service class for tagging learning objects.  This class
    contains methods to tag an object and retrieve a set of tags
     related to a specific object or metadata.
 *
 * @author <a href=”mailto:naimdjon@gmail.com”>Naimdjon Takhirov
    </a>
 * @version $Id: TaggingService.java 44417 2008-04-21 18:53:36Z
    naimdjon $
 * @since 23.02.2008
 */
public class TaggingService {

    /**
     * Tags object with the specified label.
     *
     * @param tag the tag object
     */
    public static void tagObject(Tag tag) {
        BaseObjectDataAccess bo = BaseObjectDataAccess.
            getInstance();
        final Connection con = bo.getConnection();
        PreparedStatement ps = null;
        try {
            int counter = 1;
            ps = con.prepareStatement(”insert into tag(objectid,
                label,taggingtime,score) values(?,?,?,?)”);
            ps.setLong(counter++, tag.getObjectId());
            ps.setString(counter++, tag.getLabel());
            ps.setLong(counter, tag.getTaggingTime().
                getTimeInMillis());
            final int i = ps.executeUpdate();
            if (i != 1) {
                throw new DataAccessException(”Could not insert
                    a new tag to the database”);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            bo.close(ps);
            bo.close(con);
        }
    }
    /**
     * Retrievs tags made by the specified user.
     *
     * @param taggerId the id of tagger.
     * @return an iterable of Tag objects.
     */
    public static Iterable<Tag> getTagsByTaggerId(long taggerId)
         {
        Collection<Tag> tags = ICollections.newSet();
        BaseObjectDataAccess bo = BaseObjectDataAccess.
            getInstance();
        final Connection con = bo.getConnection();
        PreparedStatement ps = null;
        try {
            ps = con.prepareStatement("select * from tag where
                taggerid=?");
            ps.setLong(1, taggerId);
            fillResult(tags, ps);
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            bo.close(ps);
            bo.close(con);
        }
        return tags;
    }

    /**
     * Retrievs objects specified the label for the user.
     * @param label the label of tags
     * @param taggerId the id of tagger
     * @return
     */
    public static Iterable<Long> getObjects(String label, long
        taggerId) {
        Collection<Long> objectIds = ICollections.newSet();
        BaseObjectDataAccess bo = BaseObjectDataAccess.
            getInstance();
        final Connection con = bo.getConnection();
        PreparedStatement ps = null;
        try {
            ps = con.prepareStatement("select objectid from tag
                where label=? and taggerId=?");
            ps.setString(1, label);
            ps.setLong(2, taggerId);
            final ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                objectIds.add(rs.getLong("objectId"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            bo.close(ps);
            bo.close(con);
        }
        return objectIds;
    }


    /**
     * Fills the object with the rows from the result set.
     *
     * @param tags a collection where the  retrieved tag is
         added.
     * @param ps prepared statement
     * @throws SQLException if any data access error occurs.
     */
    private static void fillResult(Collection<Tag> tags,
       PreparedStatement ps) throws SQLException {
        final ResultSet rs = ps.executeQuery();
        while (rs.next()) {
            Tag tag = new Tag();
            tag.setLabel(rs.getString("label"));
            tag.setObjectId(rs.getLong("objectId"));
            tag.setTaggerId(rs.getLong("taggerid"));
            tags.add(tag);
        }
    }

}
```

/jsp/pedal-ng/index.jsp

```
<jsp:forward page="login/index.jsp"/>
```

/jsp/pedal-ng/login/index.jsp

```

<%@ page import="no.inspera.Marketplace" %>
<%@ page import="no.inspera.db.MarketplaceDataAccess" %>
<%@ page import="no.inspera.services.metadataservice.
    ItemProperty" %>
<%@ page import="no.inspera.services.metadataservice.
    ItemPropertyValue" %>
<%@ page import="no.inspera.services.metadataservice.
    MetadataService" %>
```

```
<%@ page import="no.inspera.services.pedal.PEDAL" %>
<%@ page import="no.inspera.utils.Parameters" %>
<%@ page import="no.inspera.utils.StringUtils" %>
<%
    Parameters params= Parameters.create(request);
    params.setSessionAttribute("pedalCourseLogicalTitle",PEDAL.
       defaultLogicalTitle);
    //System.out.println("params.getUserId():"+params.getUserId
       ());
    if(params.getUserId()>0){
        %><jsp:forward page="/pedalng"/><%
    }else {
        if (params.getMarketplaceId() <= 0) {
            final Marketplace mp = MarketplaceDataAccess.
               getInstance().getMarketplaceFromName("pedal");
            params.setMarketplaceId(mp.getMarketplaceId());
        }
%>
<html>
<head>
    <title>PEDAL-NG: Welcome</title>
    <link rel="stylesheet" href="/jsp/pedal-ng/pedal-ng.css"/>
    <script type="text/javascript" src="/jsp/pedal-ng/pedal-ng.
       js">
        //void
    </script>
    <script language="javascript" src="/scripts/jquery/jquery.js
       ">
        //void
    </script>
    <jsp:include page="/jsp/pedal-ng/login.js"/>
</head>
<body>
<div align="center">
<div id="welcomeContent">
    <%
        String err = params.getRequestAttribute("error");
        if (StringUtils.hasLength(err)) {
    %><%=err%><%
    }
%>
</div>
<div id="loginDiv">
    <div class="headerDiv">&#160;Login to database course</div>
    <div>
        <!--<form action="/login" method="post" id="loginForm"
           >-->
        <table>
            <tr>
```

```
                <td align="center" colspan="2" id="loginMessage"
                    >Username and password are required!</td>
            </tr>
            <tr>
                <td>Username/Email:</td>
                <td><input type="text" name="username" id="
                    loginUsername" class="textField"/></td>
            </tr>
            <tr>
                <td>Password:</td>
                <td><input type="password" name="password" id="
                    loginPassword" class="textField"/></td>
            </tr>
            <tr>
                <td colspan="2" align="right">
                    <button name="submitBtn" id="loginBtn">Login
                        </button>
                    <a id="forgotPass" href="#">Forgot password
                        ?</a> &#160;<a id="newUser" href="#">New
                        user</a>
                </td>
            </tr>
        </table>
        <!--</form>-->
    </div>
</div>
<div id="registerDiv">
    <div class="headerDiv">&#160;Register for the course</div>
    <div>
        <table>
            <tr>
                <td align="center" colspan="2" id="
                    registerMessage">All fields marked with (*)
                    must be filled out!</td>
            </tr>
            <tr>
                <td>First name(*):</td>
                <td><input type="text" name="firstname" id="
                    firstname" class="textField"/></td>
            </tr>
            <tr>
                <td>Last name(*):</td>
                <td><input type="text" name="lastname" id="
                    lastname" class="textField"/></td>
            </tr>
            <tr>
                <td>Mobile:</td>
                <td><input type="text" name="mobile" id="mobile"
                     class="textField"/></td>
        </tr>
        <tr>
            <td>Email(*):</td>
            <td><input type="text" name="email" id="email"
                class="textField"/></td>
        </tr>
        <tr>
            <td>Password(*):</td>
            <td><input type="password" name="password" id="
                password" class="textField"/></td>
        </tr>
        <tr>
            <td colspan="2">
                <hr/>
            </td>
        </tr>
        <tr>
            <td>Interests:</td>
            <td>
                <%
                    try {
                        final ItemProperty ip =
                           MetadataService.getItemProperty("
                           interests");
                        final ItemPropertyValue[] values =
                           ip.getValues();
                        for (ItemPropertyValue ipv : values)
                            {

                %>
                <input type="checkbox" name="interests"
                    value="<%=ipv.getItemPropertyValueId()%>"
                        id="<%=ipv.getValue()%>"/>
                <label for="<%=ipv.getValue()%>"><%=ipv.
                    getText(2, params.getMarketplaceId())%>
                </label>
                <%
                %><br/><%
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            %>
            </td>
        </tr>
        <tr>
            <td colspan="2" align="right">
                <a id="toLoginPage2" href="#">&lt;&lt;To
                    login page</a>&#160;
```

```
                        <button name="submitBtn" id="registerBtn">
                            Register</button>
                    </td>
                </tr>
            </table>
        </div>
    </div>
    <div id="forgotPasswordDiv">
        <div class="headerDiv">&#160;Forgot password</div>
        <div>
            <table>
                <tr>
                    <td align="center" colspan="2" class="message"><
                        img src="/jsp/pedal-ng/loader.gif" alt="
                        Loading..."/>
                    </td>
                </tr>
                <tr>
                    <td>Email:</td>
                    <td><input type="text" name="email" id="
                        forgotPasswordEmail" class="textField"/></td>
                </tr>
                <tr>
                    <td colspan="2" align="right">
                        <a id="toLoginPage" href="#">&lt;&lt;To
                            login page</a>&#160;
                        <button name="submitBtn" id="
                            sendPasswordButton">Send password</button
                            >
                    </td>
                </tr>
            </table>
        </div>
    </div>
</div>
</body>
</html>


<%
    }
%>
```

/jsp/pedal-ng/login/auth.jsp

```
<%@ page import="no.inspera.db.UserDataAccess" %>
<%@ page import="no.inspera.kernel.User" %>
<%@ page import="no.inspera.services.resourcestring.
    ResourceStringService" %>
```

```
<%@ page import="no.inspera.utils.Parameters" %>

<%
    Parameters params= Parameters.create(request);
    final String username = params.getParameter("username");
    final String password = params.getParameter("password");
    //System.out.println("username:"+username+", password:"+
        password);
    response.setContentType("text/xml");
    final ServletOutputStream outputStream = response.
        getOutputStream();

    try {
        final Long userId = UserDataAccess.getInstance().
            checkPassword(username, password, params.
            getMarketplaceId());
        final User user = UserDataAccess.getInstance().getUser(
            userId);
        response.setHeader("Cache-Control","no-cache"); //HTTP
            1.1
        response.setHeader("Pragma","no-cache"); //HTTP 1.0
        response.setDateHeader ("Expires", 0); //prevents
            caching at the proxy server
        params.setSessionAttribute("returnLogOutURL",request.
            getRequestURL().toString().replace("login/auth.jsp","
            "));
        params.setUserId(userId);
        params.setMarketplaceId(user.getMarketplaceId());
        params.setLanguageId(user.getLanguageId());
        outputStream.print("<msg>Welcome "+user.getFirstName()+"
              "+user.getLastName()+"</msg>");
    } catch (Exception e) {
        outputStream.print("<msg>Error:"+ ResourceStringService.
            getResourceString(e.getMessage(),params.getLanguageId
            (),params.getMarketplaceId(),false) +"</msg>");
    }

%>
```

/jsp/pedal-ng/login/register.jsp

```
<%@ page import="no.inspera.db.MetadataDataAccess" %>
<%@ page import="no.inspera.db.RelationDataAccess" %>
<%@ page import="no.inspera.db.UserDataAccess" %>
<%@ page import="no.inspera.kernel.User" %>
<%@ page import="no.inspera.services.metadataservice.*" %>
<%@ page import="no.inspera.services.relation.MembershipRelation
    " %>
```

```
<%@ page import=”no.inspera.services.relation.
    RelationTypeConstants” %>
<%@ page import=”no.inspera.utils.CommonTools” %>
<%@ page import=”no.inspera.utils.Parameters” %>
<%@ page import=”no.inspera.utils.StringUtils” %>

<%
    final long orgID=629805;
    Parameters params= Parameters.create(request);
    final String email = request.getParameter(”email”);
    final String firstname = request.getParameter(”firstname”);
    final String lastname = request.getParameter(”lastname”);
    final String mobile = request.getParameter(”mobile”);
    final String password = request.getParameter(”password”);
    final String[] interests = request.getParameterValues(”
       interests”);
    final User user = UserDataAccess.getInstance().
       getUserByEmail(email, params.getMarketplaceId());
    response.setContentType(”text/xml”);
    final ServletOutputStream outputStream = response.
       getOutputStream();
    if (!CommonTools.isValidEmail(email)) {
        outputStream.print(”<msg>Invalid email address!</msg>”);
    }else if(user!=null){
        outputStream.print(”<msg>Email is already registered!</
           msg>”);
    }else if(!StringUtils.hasLength(firstname)){
        outputStream.print(”<msg>First name missing!</msg>”);
    }else if(!StringUtils.hasLength(lastname)){
        outputStream.print(”<msg>Last name missing!</msg>”);
    }else if(!StringUtils.hasLength(password)){
        outputStream.print(”<msg>Password missing!</msg>”);
    }else {
        User u= new User();
        try {
            u.setFirstName(firstname);
            u.setLastName(lastname);
            u.setEmail(email);
            u.setPassword(password);
            u.setUserName(email);
            u.setLanguageId(2);
            u.setURL(mobile);
            u.setMarketplaceId(params.getMarketplaceId());
            final Long userId = UserDataAccess.getInstance().
               createNewUser(u);
            MembershipRelation rel = new MembershipRelation(
               orgID, userId, RelationTypeConstants.
               EMPLOYEE_STUDENT, params.getMarketplaceId());
```

```
                RelationDataAccess.getInstance().
                    newMembershipRelation(rel);
                if (interests != null && interests.length > 0) {
                    Metadata md= new Metadata();
                    md.setObjectId(userId);
                    ItemProperty ip= MetadataService.getItemProperty
                        ("interests");
                    ItemValue iv= new ItemValue();
                    iv.setItemProperty(ip.getItemPropertyId());
                    iv.setObjectId(userId);
                    iv.setLanguageId(0L);
                    for (String s : interests) {
                        ItemPropertyValue ipv= new ItemPropertyValue
                            ();
                        ipv.setValue(Long.valueOf(s));
                        iv.addValue(ipv);
                    }
                    md.addMetadataItem(iv);
                    MetadataDataAccess.getInstance().storeMetaData(
                        md.getItemValues(),userId);
                }
            } catch (Exception e) {
                e.printStackTrace();
                outputStream.print("<msg>Error creating a user!</msg
                    >");
                return;
            }
            outputStream.print("<msg>User is created!</msg>");
        }
%>
```

/jsp/pedal-ng/login/sendPassword.jsp

```
<%@ page import="no.inspera.utils.CommonTools" %>
<%@ page import="no.inspera.db.UserDataAccess" %>
<%@ page import="no.inspera.utils.Parameters" %>
<%@ page import="no.inspera.kernel.User" %>

<%
    Parameters params= Parameters.create(request);
    final String email = request.getParameter("email");
    response.setContentType("text/xml");
    final ServletOutputStream outputStream = response.
        getOutputStream();
    if (!CommonTools.isValidEmail(email)) {
        outputStream.print("<msg>Invalid email address!</msg>");
    }else {
        final User user = UserDataAccess.getInstance().
            getUserByEmail(email, params.getMarketplaceId());
        if(user==null){
            outputStream.print("<msg>User with this email is not
                 registered!</msg>");
        }else{
            try {
                User u=UserDataAccess.getInstance().
                    getUserToSendPassword(user.getEmail(),params.
                    getMarketplaceId());
                CommonTools.sendUserRequestedPassword(params,u,
                    params.getMarketplaceObj());
                outputStream.print("<msg>Password has been sent.
                     Please check also your junk-mail folder if
                    you don't find the email in your inbox.</msg>
                    ");
            } catch (Exception e) {
                e.printStackTrace();
                outputStream.print("<msg>Error:"+e.getMessage()+
                    "</msg>");
            }
        }
    }
%>
```

/jsp/pedal-ng/lsa/index.jsp

```
<%@ page import="no.inspera.utils.xpath.XPath" %>
<%@ page import="no.inspera.utils.XMLTools" %>
<%@ page import="org.w3c.dom.Document" %>
<%@ page import="no.inspera.utils.FileUtils" %>
<%@ page import="org.w3c.dom.Node" %>
<%@ page import="org.xml.sax.SAXException" %>
<%@ page import="org.w3c.dom.traversal.NodeIterator" %>
<%@ page import="no.inspera.utils.collections.ICollections" %>
<%@ page import="java.util.Map" %>
<%@ page import="java.util.ArrayList" %>

<%
    response.setHeader("Cache-Control","no-cache"); //HTTP 1.1
    response.setHeader("Pragma","no-cache"); //HTTP 1.0
    response.setDateHeader ("Expires", 0); //prevents caching at
         the proxy server
%>

<div>
    <em>Welcome to learning style questionnaire. The system will
         try to identify your learning style based on answers you
         give below.</em>
</div>
<div style="font-size:10px;padding-top:2px;">
```

```
    <hr/>
        &#169; Copyright Version 7.0 (2006) held by Neil D.
           Fleming, Christchurch, New Zealand and Charles C.
           Bonwell, Green Mountain Falls, Colorado
        80819 U.S.A.  <br/><a href="http://www.vark-learn.com"
           style="color:#ccccdd;" target="_blank">VARK web-site
           </a>
</div>
<%
    try {
        Document doc= XMLTools.getDocument(FileUtils.readFile(
           config.getServletContext().getRealPath("/jsp/pedal-ng
           /lsa/questions.xml")));
        final NodeIterator it = XPath.selectNodeIterator(doc, "/
           vark-questions/question");
        for (Node questionNode = it.nextNode(); questionNode !=
           null; questionNode = it.nextNode()) {
            String q= XPath.eval(questionNode,"q");
            String id= XPath.eval(questionNode,"@id");
            %>
            <div class="VARKQuestion">
                <hr/>
                    <div class="VARKQuestionText"><%=id+"/16.    "
                        +  q%></div>
            <%
                Map<String,String> map= ICollections.newMap();
                final NodeIterator responseIterator = XPath.
                   selectNodeIterator(questionNode, "response");
            for (Node responseNode = responseIterator.nextNode()
               ; responseNode != null; responseNode =
               responseIterator.nextNode()) {
                final String type = XPath.eval(responseNode, "
                   @type");
                final String text = XPath.eval(responseNode, "."
                   );
                map.put(type,text);

            }
                final ArrayList<String> list = new ArrayList<
                   String>(map.keySet());
                java.util.Collections.shuffle(list);
                for (String type : list) {
                    %>
                        <div class="VARKResponse">
                            <label>
                            <input type="checkbox" value="<%=
                               type%>" name="<%=type%>"/>
                            <%=map.get(type)%>
                        </label>
```

```
                              </div>
                          <%
                      }
                  %>


                  </div><%
              }
              %></div><%
      } catch (SAXException e) {
          e.printStackTrace();
          %>Error:<%=e.getMessage()%><%
      }
%>
```

/jsp/pedal-ng/lsa/submit_lsa_form.jsp

```
<%@ page import="no.inspera.applications.publish.datamodel.
    ContentRevision" %>
<%@ page import="no.inspera.db.BaseObjectDataAccess" %>
<%@ page import="no.inspera.db.ContentPresentationDataAccess" %>
<%@ page import="no.inspera.db.MetadataDataAccess" %>
<%@ page import="no.inspera.services.log.LogServices" %>
<%@ page import="no.inspera.services.metadataservice.*" %>
<%@ page import="no.inspera.services.pedal.PEDAL" %>
<%@ page import="no.inspera.services.pedal.actions.Configure" %>
<%@ page import="no.inspera.services.pedal.actions.Index" %>
<%@ page import="no.inspera.utils.Parameters" %>
<%@ page import="no.inspera.utils.collections.ICollections" %>
<%@ page import="org.apache.log4j.Logger" %>
<%!
    private final transient Logger log = LogServices.getLogger(
        getClass());
    static java.util.Map<String,String>mappings= ICollections.
        newMap();
    static {
        mappings.put("V", PEDAL.LearningStyle.visual.s());
        mappings.put("A", PEDAL.LearningStyle.auditory.s());
        mappings.put("K", PEDAL.LearningStyle.kinesthetic.s());
    }

    class VARK implements Comparable<VARK>{
        int score;
        String type;

        VARK(int score, String type) {
            this.score = score;
            this.type = type;
        }

        public String toString() {
            return "score:"+ score +", t:"+type;
        }

        public int compareTo(VARK o) {
            return (this.score<o.score ? -1 : (this.score==o.score ? 0 : 1));
        }
    }

    VARK count(String s,Parameters params) {
        final String[] values = params.getParameterValues(s);
        if (values != null && values.length > 0)
            return new VARK(values.length,s);
        return new VARK(0,s);
    }
%>
<%
    Parameters params=Parameters.create(request);
    params.logParams();
    VARK[] reponses = new VARK[]{
            count("V",params),
            count("A",params),
//            count("R",params),
            count("K",params)
    };
    java.util.Arrays.sort(reponses);
    String type = reponses[2].type;

    try {
        final Metadata md = MetadataService.getMetadataForObject(params.getUserId());
        final ItemProperty itemproperty = MetadataService.getItemProperty(PEDAL.learningStyle);
        ItemValue iv = md.getItemValue(itemproperty.getItemPropertyId());
        final ItemPropertyValue[] values = itemproperty.getValues();
        ItemPropertyValue ipv=null;
        for (ItemPropertyValue value : values) {
            if (value.getLogicalName().equals(mappings.get(type)))
                ipv=value;
        }
        if(ipv!=null){
            System.out.println("ipv.getItemPropertyValueId():"+ipv.getItemPropertyValueId());
            ipv.setIsSet(true);
            }
            if (iv == null) {
                iv= new ItemValue();
                iv.setItemProperty(itemproperty.getItemPropertyId())
                    ;
                iv.setObjectId(params.getUserId());
                iv.setValueNumber(ipv.getItemPropertyValueId());
                MetadataDataAccess.getInstance().
                    addItemValueValueNumber(iv);
            }else {
                iv.setValueNumber(ipv.getItemPropertyValueId());
                MetadataDataAccess.getInstance().updateItemValue(iv)
                    ;
            }
            long contentItemId = BaseObjectDataAccess.getInstance().
                getObjectIdFromLogicalName((String) params.
                getSessionAttribute("pedalCourseLogicalTitle"),
                params.getMarketplaceId());
            MetadataService.removeMetadataForObjectFromCache(params.
                getUserId());
            ContentRevision revision = ContentPresentationDataAccess
                .getInstance().
                getLiveContentRevisionWithEffectiveLanguage(
                contentItemId, params.getLanguageId(), params.
                getMarketplaceId());
             params.setRequestAttribute(PEDAL.course, revision);
            new Configure(params.getMarketplaceId(), params.
                getLanguageId()).execute(params);
            new Index().execute(params);
            %>
              <body onload="javascript:parent.scroll(0,0);"/>
            <p>Great. Now you can click "Next" to continue the
                course. </p>
      <%
      } catch (Exception e) {
            log.error("Exception",e);
            %>An error occurred. <a href="/jsp/pedal-ng/lsa">Please
                try again</a>.<%
      }
%>
```

/jsp/pedal-ng/pka/index.jsp

```
<%@ page import="no.inspera.applications.publish.datamodel.
    ContentRevision" %>
<%@ page import="no.inspera.applications.publish.datamodel.
    assessment.AssessmentDataAccess" %>
<%@ page import="no.inspera.applications.publish.datamodel.
    assessment.QTIUserAssessment" %>
```

```
<%@ page import="no.inspera.db.BaseObjectDataAccess" %>
<%@ page import="no.inspera.db.ContentPresentationDataAccess" %>
<%@ page import="no.inspera.services.metadataservice.*" %>
<%@ page import="no.inspera.services.pedal.PEDAL" %>
<%@ page import="no.inspera.utils.Parameters" %>

<%
    Parameters params= Parameters.create(request);
    //String req=request.getRequestURL().toString().replace("
       index.jsp","");
    String url="/content?marketplaceId="+params.getMarketplaceId
       ()+"&contentItemId=629886";
    if (params.getUserId() <= 0) {
        %>You have to login to take prior knowledge assessment<%
        return;
    }else {
            final Metadata md = MetadataService.
               getMetadataForObject(params.getUserId());
            if (md != null) {
                ItemValue iv=md.getItemValue(PEDAL.level);
                if (iv != null && iv.getItemValueId() != null &&
                    iv.getItemValueId() > 0) {
                    final ItemProperty itemproperty =
                       MetadataService.getItemProperty(iv.
                       getItemProperty());
                    final ItemPropertyValue[] values =
                       itemproperty.getValues();
                    ItemPropertyValue value=null;
                    for (ItemPropertyValue val : values) {
                        if (iv.getValueNumber().equals(val.
                           getItemPropertyValueId()))
                            value=val;
                    }
                    final String logicalTitle = params.
                       getParameter("pedalCourseLogicalTitle",
                       PEDAL.defaultLogicalTitle, true);
                    long contentItemId = BaseObjectDataAccess.
                       getInstance().getObjectIdFromLogicalName(
                       logicalTitle, params.getMarketplaceId());
                    ContentRevision revision =
                       ContentPresentationDataAccess.getInstance
                       ().
                       getLiveContentRevisionWithEffectiveLanguage
                       (contentItemId, params.getLanguageId(),
                       params.getMarketplaceId());
                    final ItemValue ivAss = MetadataService.
                       getMetadataForObject(revision.getObjectId
                       ()).getItemValue(PEDAL.
                       prior_knowledge_assessment);
                    final ContentRevision assessment =
                        ContentPresentationDataAccess.getInstance
                        ().
                        getLiveContentRevisionWithEffectiveLanguage
                        (Long.valueOf(ivAss.getValueText()),
                        params.getLanguageId(), params.
                        getMarketplaceId());
                    final QTIUserAssessment ua =
                        AssessmentDataAccess.getInstance().
                        getLastUserAssessment(assessment.
                        getContentRevisionId(), params.getUserId
                        ());
                    if(ua!=null && value!=null){
                        %>You've taken this assessment.   <a
                            href="<%=url%>">Take again</a>.<%
                    }else
                        %><jsp:forward page="<%=url%>"/><%
                }else {
                    //response.sendRedirect(url);
                    %><jsp:forward page="<%=url%>"/><%
                }
            }
        }
%>
```

/jsp/pedal-ng/pka/sql.jsp

```
<%@ page import="no.inspera.db.BaseObjectDataAccess" %>
<%@ page import="no.inspera.db.DataAccess" %>
<%@ page import="no.inspera.db.TableColumns" %>
<%@ page import="no.inspera.utils.Parameters" %>
<%@ page import="no.inspera.utils.StringUtils" %>
<%@ page import="no.inspera.services.pedal.utils.ICollections"
    %>
<%@ page import="java.sql.*" %>
<%@ page import="java.util.Collection" %>
<%@ page contentType="text/html;charset=UTF-8" language="java"
    %>
<%
    java.util.Map<String,Iterable<String>> sampleSQLs=
        ICollections.newMap();
    Collection<String> col= ICollections.linkedList();
    col.add("select * from persons");
    col.add("select * from persons where firstname like '
        Naimdjon'");
    col.add("select firstname from persons where city = 'Oslo'")
        ;
    sampleSQLs.put("select",col);
    col= ICollections.linkedList();
    col.add("update persons set firstname='Naimjon' where
        lastname='Takhirov'");
    col.add("update persons set city='Trondheim', address='
        Strindheim 23'  where lastname='Takhirov'");
    sampleSQLs.put("update",col);
    col= ICollections.linkedList();
    col.add("delete persons where lastname='Takhirov'");
    col.add("delete persons where lastname='Takhirov' and
        firstname='Naimdjon'");
    sampleSQLs.put("delete",col);
    col= ICollections.linkedList();
    col.add("insert into persons values(20,'Your firstname','
        Your lastname','Your address','Your city')");
    col.add("insert into persons (personid,firstname,lastname,
        address,city) values(21,'Your firstname','Your lastname
        ','Your address','Your city')");
    sampleSQLs.put("insert",col);
    final Parameters params=Parameters.create(request);
    Connection con =null;
    Statement st=null;
    try{
        con = DriverManager.getConnection("jdbc:oracle:thin:
            @utv_db:1521:utvdb4","booking_dontdelete","
            booking_dontdelete");
        st = con.createStatement();
        final String query = request.getParameter("query");%>

<html>
  <head><title>Practice SQL</title>
      <link rel="stylesheet" type="text/css" href="/jsp/pedal-ng
          /course.css"/>
  </head>
  <body>
      <table width="100%">
          <tr>
              <td valign="top">
                  <table>
                      <tr>
                          <td>
                             <%
                                 String q = params.getParameter
                                     ("query");
                                 String statementSample;
                                 if(q==null){
                                     statementSample=params.
                                         getParameter("sample",
                                         "select");
                                 }else if (!StringUtils.
                                     anyStartsWithIgnoreCase(q,
                                "select", "update", "insert
                                ", "delete")) {
                                %><div class="error">An
                                    SQL statement should
                                    start with "select", "
                                    update", "insert" or "
                                    delete"</div> <%
                                statementSample=params.
                                    getParameter("sample",
                                    "select");
                            }else
                                statementSample =q.
                                    substring(0,q.indexOf("
                                    ")).toLowerCase();
                        %>
                        <h2>Practice SQL <%=
                            statementSample.toUpperCase()
                            %></h2>
                        On the left side there is table "
                            Persons". Practice SQL by
                            execute query:
                        <br/>
                            <%
                                System.out.println("
                                    statementSample:"+
                                    statementSample);
                                Iterable<String> sample=
                                    sampleSQLs.get(
                                    statementSample);
                          %>
                        <label for="sample"> Sample
                            queries:</label>
                        <ul id="sample">
                        <%
                                for (String s : sample) {
                                    %><li class="sql"><i
                                        ><%=s%></i></li><%
                                }
                            %>
                        </ul>
                    </td>
                </tr>
                <tr>
                    <td><hr/></td>
                </tr>
                <tr>
                    <td><input id="query" style="width:400
                        px;" type="text" value="<%=
                        StringUtils.checkNull(query)%>"
```

```
                                    />&#160;<input value="Submit" type=
                                    "button" onclick="document.location
                                    .href='<%=request.getRequestURL().
                                    toString()+"?sample="+
                                    statementSample+"&query="%>'+
                                    document.getElementById('query').
                                    value"/> </td>
                              </tr>
                              <tr>
                                  <td>
                                      <%
                                          if(StringUtils.hasLength(query
                                              )){
                                              try {
                                                  if(query.toLowerCase()
                                                      .startsWith("select
                                                      ")){
                                                       System.out.println
                                                           ("select...");
                                                  final ResultSet rs =
                                                      st.executeQuery(
                                                      query);
                                                  final
                                                      ResultSetMetaData
                                                      md = rs.getMetaData
                                                      ();
                                                  final int cc = md.
                                                      getColumnCount();
                                                  %>
                                                    <table class="
                                                        borderTable">
                                                        <thead>
                                                    <%
                                                  for (int i = 1; i <=cc
                                                      ; i++) {
                                                       final String
                                                           columnName = md
                                                           .getColumnName(
                                                           i);
                                                       %><th><%=
                                                           columnName%></
                                                           th><%
                                                  }
                                                         %></thead><%
                                                  while (rs.next()) {
                                                       %><tr><%
                                                       for (int i = 1; i
                                                           <=cc; i++) {
                                                              %><td><%=rs.
                                                                  getObject(i
                                                                  )%></td><%
                                                        }
                                                     %></tr><%
                                                  }
                                                  %></table><%
                                            }else if(StringUtils.
                                                anyStartsWithIgnoreCase
                                                (query,"delete","update
                                                ","insert")){
                                                System.out.println("up
                                                    ...");
                                                final int i = st.
                                                    executeUpdate(query
                                                    );
                                                %><div><i><em><%=
                                                    StringUtils.
                                                    capitalize(
                                                    statementSample)
                                                    %><%=
                                                    statementSample.
                                                    endsWith("e")?"d":"
                                                    ed"%>  <%=i%> row
                                                    <%=i>1?"s":""%> in
                                                    the database!</em
                                                    ></i></div><%
                                            }
                                            } catch (Exception e) {
                                                e.printStackTrace();
                                                %><div  class="error">
                                                    Error:<%=e.
                                                    getMessage().
                                                    startsWith("ORA")?e
                                                    .getMessage().
                                                    substring(e.
                                                    getMessage().
                                                    indexOf(":")+1):e.
                                                    getMessage()%></div
                                                    ><%
                                            }
                                      }
                                      %>
                                  </td>
                              </tr>
                          </table>
                      </td>
                      <td valign="top">
                          <table>
```

```
                                    <tr>
                                        <td>Persons table</td>
                                    </tr>
                                    <tr>
                                        <table class="borderTable">
                                            <%
                                                DataAccess db=
                                                    BaseObjectDataAccess.
                                                    getInstance();
                                                final TableColumns
                                                    tableColumns = db.
                                                    getTableColumns(con, "
                                                    PERSONS");
                                                for (String s : tableColumns
                                                    .getColumnNames()) {
                                                    %><th><%=s%></th><%
                                                }
                                                final ResultSet rs = st.
                                                    executeQuery("select *
                                                    from persons");
                                                while (rs.next()) {
                                                    %>
                                                        <tr>
                                                            <%
                                                                for (String
                                                                    s :
                                                                    tableColumns
                                                                    .
                                                                    getColumnNames
                                                                    ()) {
                                                                    %><td
                                                                        ><%=
                                                                        rs.
                                                                        getObject
                                                                        (s)
                                                                        %></
                                                                        td><%
                                                                }
                                                            %>
                                                        </tr>
                                                    <%
                                                }
                                            %>
                                        </table>
                                    </tr>
                                </table>
                            </td>
                        </tr>
                    </table>
```

```
  </body>
</html>
<%
    }catch(Exception e){
        e.printStackTrace();
    }finally{
        st.close();
        con.close();
    }
%>
```

/jsp/pedal-ng/course.js

```
var nextModuleId=0;
var previousModuleId=0;
$(document).ready(function() {
    $(".courseNodeLinkNoContent").click(function(){
        alert("No content!");
    });
//    $(".btnNext").click(function(){
//        nextContent();
//    });
//    $(".btnPrevious").click(function(){
//        previousContent();
//    });
//    $("#contIframe").hide().end();
//    $("#contentMsg").hide().end();
//    $("#contentMsg").html("<h2>Welcome to the database course
    ").slideDown(2000);
//    $(".courseNodeLink").click(function(){
//        loadModule(this.id);
//    });
});

//function  loadModule(id){
//    nextModuleId=id;
//    $("#contIframe").hide().end();
//        $("#contentMsg").html("<div style=\"padding:20px;font-
    weight:bold;\"><img src=\"/jsp/pedal-ng/loader.gif\" alt=\"
    Loading...\"/> Loading ...</div>")
//        .show("fast",function(){
//            $("#contIframe").attr({src:"/pedal?content=1&
    moduleId="+id+"&"+Math.random()});
//            $("#contentMsg").hide().end();
//            $("#"+id).css("background:blue;")
//            $("#contIframe").show().end();
//            nextModuleId=$("#"+id).parent().next().find("a").
    attr("id");
//              previousModuleId=$("#"+id).parent().prev().find("a
    ").attr("id");
//              if(previousModuleId){
//                  $(".btnPrevious").removeClass("
    courseBtnDisabled");
//                  $(".btnPrevious").attr({disabled:false});
//              }else {
//                  $(".btnPrevious").addClass("courseBtnDisabled
    ");
//                  $(".btnPrevious").attr({disabled:true});
//              }
//          });
//}

//function  nextContent(){
//     if(nextModuleId == 0 )nextModuleId=$(".btnNext").attr("
    name");
//     loadModule(nextModuleId);
//}
//
//function  previousContent(){
//     loadModule(previousModuleId);
//}
```

/jsp/pedal-ng/login.js

```

            <script type="text/javascript">
                    $(document).ready(function() {
                        $("#forgotPasswordDiv").hide().end();
                        $(".message").hide().end();
                        $("#loginMessage").hide().end();
                        $("#registerDiv").hide();
                        $("#loginUsername").focus();

                        $('#loginPassword').keyup(function(e) {
                            if(e.keyCode == 13) {
                                login();
                            }
                        });

                        $("#toLoginPage,#toLoginPage2").click(
                            function(){
                            $("#forgotPasswordDiv").hide();
                            $("#registerDiv").hide();
                            $("#loginDiv").fadeIn();
                        });

                        $("#registerBtn").click(function(){
```

```
                register();
            });

            $("#loginBtn").click(function(){
                login();
            });

            $("#forgotPass").click(function(){
                $(".message").hide().end();
                $("#loginDiv").fadeOut(function(){
                    $("#forgotPasswordDiv").fadeIn()
                        ;
                });
            return false;
        });

            $("#newUser").click(function(){
                $("#loginDiv").fadeOut(function(){
                    $("#registerDiv").fadeIn();
                });
            return false;
        });

            $("#sendPasswordButton").click(function
                (){
                var email= $("#forgotPasswordEmail")
                    .val();
                 if(email == null || email.length
                     <=0){
                     alert('Please, write your email
                          and we will send you your
                        password!');
                     $("#forgotPasswordEmail").focus
                        ();
                 }else {
                     $(".message").show();
                     $("#forgotPasswordEmail").attr(
                        "disabled","disabled");
                     $("#sendPasswordButton").attr("
                        disabled","disabled");
                     $.get("/jsp/pedal-ng/login/
                        sendPassword.jsp?marketplace
                        =pedal&email="+email,
                        function(xml){
                       var text = $(xml).text();
                         $(".message").html(text);
                          $("#forgotPasswordEmail").
                              removeAttr("disabled");
                         $("#sendPasswordButton").removeAttr("disabled");
                         $("#forgotPasswordEmail").val("");
                    });
                 }
         return false;
     });

});

function register(){
    var firstname= $("#firstname").val();
    var lastname= $("#lastname").val();
    var mobile= $("#mobile").val();
    var email= $("#email").val();
    var password= $("#password").val();
    var error=false;
    if(password == null || password.length<=0){
        $("#password").addClass("redBorder");
        $("#password").focus();
        error=true;
    }else $("#password").removeClass("redBorder");

    if(email == null || email.length<=0){
         $("#email").addClass("redBorder");
        $("#email").focus();
        error=true;
    }else $("#email").removeClass("redBorder");

    if(lastname == null || lastname.length<=0){
         $("#lastname").addClass("redBorder");
        $("#lastname").focus();
        error=true;
    }else $("#lastname").removeClass("redBorder");

    if(firstname == null || firstname.length<=0){
         $("#firstname").addClass("redBorder");
        $("#firstname").focus();
        error=true;
    }else $("#firstname").removeClass("redBorder");

    if(error){
        $("#registerMessage").show().end();
    }else {
        $("#registerMessage").html("<img src=\"/jsp/pedal-ng/loader.gif\" alt=\"Loading...\"/>").show().end();
        var interests="&";
        $("input[@name=interests]").each(
           function(){
            if(this.checked)
                interests+="interests="+this.value+"&";
         });
        $.get("/jsp/pedal-ng/login/register.jsp?marketplace=pedal&email="+email+"&firstname="+firstname+"&lastname="+lastname+"&mobile="+mobile+"&email="+email+"&password="+password+interests,function(xml){
            var text = $(xml).text();
            $("#registerMessage").html(text);
            if(text.indexOf("User is created")>-1){
                $("#registerDiv").animate({opacity: 0.0}, 1200, function(){
                    $("#registerDiv").hide().end();
                    $("#loginDiv").fadeIn();
                });
            }
        });
    }
}

function login(){
    var loginUsername= $("#loginUsername").val();
    var loginPassword= $("#loginPassword").val();
    var error=false;
    if(loginPassword == null || loginPassword.length<=0){
        error=true;
        $("#loginPassword").addClass("redBorder");
        $("#loginPassword").focus();
    }else
        $("#loginPassword").removeClass("redBorder");
        if(loginUsername == null || loginUsername.length<=0){
            error=true;
            $("#loginUsername").addClass("redBorder");
            $("#loginUsername").focus();
        }else
            $("#loginUsername").removeClass("redBorder");

        if(error){
            $("#loginMessage").html("Username and password are required!");
            $("#loginMessage").show().end();
        }else {
            $("#loginMessage").html("<img src=\"/jsp/pedal-ng/loader.gif\" alt=\"Loading...\"/>").end();
            $("#loginMessage").show().end();
            $.get("/jsp/pedal-ng/login/auth.jsp?marketplace=pedal&username="+loginUsername+"&password="+loginPassword,function(xml){
                var text = $(xml).text();
                $("#loginMessage").html(text);
                if(text.length>7 && (text.indexOf("Welcome")==0 || (text.indexOf("Welcome")>=0 && text.indexOf("Welcome")<10))){
                    $("#loginDiv").animate({opacity:0.0}, 1200, function(){
                        $("#welcomeContent").html("<div style=\"padding:170px;font-weight:bold;\"><img src=\"/jsp/pedal-ng/loader.gif\" alt=\"Loading...\"/> Loading...</div>").end();
                        $("#welcomeContent").show().end();
                        document.location.href='/pedalng?ac=start';
                    });
                }
            });
        }
    }
```

```
            </script>
```

/xsl/Common_Adaptive_CourseDisplay.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
<!-- $Id: Common_Adaptive_CourseDisplay.xsl 43914 2008-03-19 11
   :00:01Z naimdjon $ -->
<xsl:include href="Common_Includes.xsl"/>

<xsl:variable name="PEDALHOME"><xsl:value-of select="$COMMON_JSP
   "/>/pedal-ng/</xsl:variable>

<xsl:variable name="moduleId">
 <xsl:choose>
            <xsl:when test="$request/moduleId"><xsl:value-of
                select="$request/moduleId"/></xsl:when>
            <xsl:when test="/*/activityNode"><xsl:value-of
                select="/*/activityNode/activityNode/
                contentActivityNodeId"/></xsl:when>
            <xsl:otherwise><xsl:value-of select="/*/planItem/
                descendant::planSubject[not(objectType = '
                content_activity_node')]/ContentActivitySymlink/
                @id"/></xsl:otherwise>
 </xsl:choose>
</xsl:variable>

<xsl:variable name="cssOverride"/>
<xsl:param name="leftNavigation"/>
<xsl:param name="topNavigation"/>

<xsl:template match="/">
<html>
 <head>
  <title>
   <xsl:choose>
    <xsl:when test="/*/activityNode"><xsl:value-of select="/*/
        activityNode/name"/></xsl:when>
    <xsl:otherwise>
     <xsl:choose>
      <xsl:when test="/*/planItem/contentRevision"><xsl:value-of
           select="/*/planItem/contentRevision/title"/></xsl:when
          >
      <xsl:otherwise><xsl:value-of select="name"/></
          xsl:otherwise>
     </xsl:choose>
    </xsl:otherwise>
    </xsl:choose>
   </title>
   <link rel="stylesheet" type="text/css" href="{$COMMON_JSP}
       pedal-ng/course.css"/>
 </head>

     <script language="javascript" src="/scripts/jquery/jquery.js
         ">
          //void
      </script>
     <script language="javascript" src="{$PEDALHOME}course.js">
          //void
      </script>

<xsl:call-template name="pageView"/>

</html>
</xsl:template>

<xsl:template name="pageView">
<div id="top">
    <div id="logoDiv"/>
    <div id="topMenu">
        <xsl:choose>
            <xsl:when test="$request/moduleId">
                <xsl:apply-templates select="/content/
                    activityNode/descendant::activityNode[
                    contentActivityNodeId=$request/moduleId]"
                    mode="courseNavigation"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:apply-templates select="/content/
                    activityNode" mode="courseNavigation"/>
            </xsl:otherwise>
        </xsl:choose>
    </div>
</div>

<div id="leftNavigation">
    <div id="courseTitleDiv" class="courseTitle"><span><
        xsl:value-of select="/content/activityNode/name"/></span>
        </div>
    <div id="activityNodes">
        <ul id="courseChildrenNavigation">
            <xsl:apply-templates select="/content/activityNode/
                activityNode" mode="viewCourseLeftMenuChildren"/>
        </ul>
    </div>
</div>
```

```

<div id="contentDiv">
    <xsl:choose>
        <xsl:when test="/content/forwardContent">
            <iframe scrolling="yes" src="/content?contentItemId
                ={/content/forwardContent}&amp;marketplaceId={$
                marketplaceId}&amp;skipTracking=1" style="
                border:1px solid #2175bc;width:90%;height:90%;"
                frameborder="0" vspace="0" hspace="0" name="
                contIframe" id="contIframe"/>
        </xsl:when>
        <xsl:when test="/content/noAssessments and $request/
            moduleId">
                The system did not find your assessment results.
                    Please, go back and answer questions.
        </xsl:when>
        <xsl:when test="$request/moduleId">
                No content / Completed module.
        </xsl:when>
        <xsl:otherwise>
            <h3>Welcome to the course on Designing Relational
                Database</h3>
            <p>
            The first module in this course will help identify
                your current knowledge of databases. In this
                module you will be asked to answer some questions
                . If you don't know the answer to the question
                simply choose "No answer" option. The second
                module is a questionnaire that helps identify
                your preferred way of learning.
            </p>
             <p>
                <em>It is important that you complete these two
                    (pre) assessments accurately. The system will
                     try to give you the most appropriate
                    learning experience. All answers you provide
                    will be treated confidentially and no
                    personal information will be made public.</em
                    >
            </p>
            <hr/>
             <p>
            The objectives of the course are:
            <ul>
                <li>Distinguish between records and fields in
                    database table</li>
                <li>Distinguish between parent and child tables
                    in relational database</li>
                <li>Use primary and foreign keys to define
                    relationships  between tables</li>
                <li>Design  a relational database from an
                    existing flat-file system(eg: Excel)</li>
            </ul>
             </p>
            <p>
                Note that you might need headphones or speakers
                    depending on your learning style. Make sure
                    to connect speakers/headphones to your
                    computer.
            </p>
            <p>
                Click "Next" on the top of the page to start the
                     prior knowledge assessment.
            </p>
        </xsl:otherwise>
    </xsl:choose>
</div>

</xsl:template>

<xsl:template match="activityNode" mode="courseNavigation">
   <xsl:variable name="precedingModuleId">
    <xsl:choose>
     <xsl:when test="preceding::activityNode"><xsl:value-of
         select="preceding::activityNode[1]/contentActivityNodeId
         "/></xsl:when>
     <!--<xsl:otherwise><xsl:value-of select="contentItemId"/></
         xsl:otherwise>-->
    </xsl:choose>
   </xsl:variable>
   <xsl:choose>
    <xsl:when test="$precedingModuleId">
     &#160;<input alt="Go to {preceding::activityNode/name}"
         title="Go to {preceding::activityNode/name}" value="{
         java:no.inspera.utils.XSLTools.getCaption('
         PREVIOUS_CONTENT_ACTIVITY_NODE_CLASSIFIED_OBJECT',$
         languageId,$marketplaceId)}" onclick="parent.location.
         href='/pedalng?moduleId={$precedingModuleId}'" type="
         button" class="courseBtn btnPrevious"/>
    </xsl:when>
    <xsl:otherwise>
     &#160;<input disabled="true" value="{java:no.inspera.utils.
         XSLTools.getCaption('
         PREVIOUS_CONTENT_ACTIVITY_NODE_CLASSIFIED_OBJECT',$
         languageId,$marketplaceId)}" type="button" class="
         courseBtnDisabled btnPrevious"/>
    </xsl:otherwise>
    </xsl:choose>

    <xsl:variable name="nextNode">
     <xsl:choose>
      <xsl:when test="count(activityNode)>1 and not(nodeType='
          content_activity_root_adaptive_course') and not(/content
          /allProcessed)"><xsl:copy-of select="./node()"/></
          xsl:when>
                      <xsl:when test="/content/sameModule"><
                          xsl:copy-of select="/content/activityNode
                          /descendant::activityNode[
                          contentActivityNodeId=$request/moduleId]/
                          node()"/></xsl:when>
                      <xsl:when test="following::activityNode and
                          following::activityNode[1]/activityNode
                          "><xsl:copy-of select="
                          following::activityNode/activityNode[1]/
                          node()"/></xsl:when>
                      <xsl:when test="following::activityNode"><
                          xsl:copy-of select="
                          following::activityNode/node()"/></
                          xsl:when>
                      <xsl:when test="not($request/moduleId)"><
                          xsl:copy-of select="activityNode[1]/node
                          ()"/></xsl:when>
     </xsl:choose>
          </xsl:variable>

             <xsl:choose>
     <xsl:when test="$nextNode">
      <input  onclick="parent.location.href='/pedalng?moduleId={$
          nextNode/contentActivityNodeId}'" title="Go to {$
          nextNode/name}" alt="Go to {$nextNode/name}" value="{
          java:no.inspera.utils.XSLTools.getCaption('
          NEXT_CONTENT_ACTIVITY_NODE_CLASSIFIED_OBJECT',$
          languageId,$marketplaceId)}" type="button" class="
          courseBtn btnNext"  id="nextButton"/>
     </xsl:when>
     <xsl:otherwise>
      <input title="Finish" value="{java:no.inspera.utils.
          XSLTools.getCaption('ASSESSMENT_FINISH',$languageId,$
          marketplaceId)}" onclick="parent.window.close()" type="
          button" class="courseBtn btnFinish"/>
     </xsl:otherwise>
    </xsl:choose>
             &#160;&#160;&#160;<em><xsl:value-of select="/*/
                 activityUserPlan/creationUser/creatorFirstName
                 "/>&#160;<xsl:value-of select="/*/
                 activityUserPlan/creationUser/creatorLastName
```

```
                    "/></em>&#160;&#160;<a href="{$Servlet_Login}&amp
                    ;action=logout" onclick="parent.location.href='{$
                    Servlet_Login}&amp;action=logout&amp;'"><
                    xsl:value-of select="java:no.inspera.utils.
                    XSLTools.getCaption('LOG_OUT',$languageId,$
                    marketplaceId)"/></a>
</xsl:template>

<xsl:template match="activityNode" mode="
    viewCourseLeftMenuChildren">
    <li>
            <xsl:choose>
                <xsl:when test="/content/noAssessments">
                    <xsl:value-of select="name"/>
                </xsl:when>
                <xsl:when test="activityNode|activityResource">
                    <a class="courseNodeLink" href="/pedalng/
                        clear?moduleId={contentActivityNodeId}"
                        id="{contentActivityNodeId}">
                        <xsl:if test="number(
                            contentActivityNodeId) = number($
                            moduleId)"><xsl:attribute name="style
                            ">background-color: #2586d7;</
                            xsl:attribute></xsl:if>
                        <xsl:value-of select="name"/>
                    </a>
                </xsl:when>
                <xsl:otherwise><a href="#" class="
                    courseNodeLinkNoContent"><xsl:value-of select
                    ="name"/></a></xsl:otherwise>
            </xsl:choose>
        <xsl:if test="activityNode and not(/content/
            noAssessments)">
            <xsl:for-each select="activityNode">
             <a class="courseNodeLink" href="/pedalng/clear?
                moduleId={contentActivityNodeId}" id="{
                contentActivityNodeId}">
                <xsl:attribute name="style">
                        <xsl:if test="number(
                            contentActivityNodeId) = number($
                            moduleId)">background-color: #2586d7
                            ;</xsl:if>
                </xsl:attribute>
                <div><xsl:value-of select="name"/></div>
             </a>
            </xsl:for-each>
        </xsl:if>
    </li>
</xsl:template>
```

```
</xsl:stylesheet>
```

I Attachment

An attachment is included to this thesis (as a single zip file). This attachment contains the following items:

- src/ - folder containing the original source code including Java, javascript, css, and XSL source codes;
- api/ - folder containing the javadoc API for PEDAL-NG;
- doc/ - folder containing the other files (JMeter test plan, class diagram in higher resolution, and PEDAL-NG screenshots).