Kennesaw State University

DigitalCommons@Kennesaw State University

Faculty Publications

2009

Animated Courseware Support for Teaching Database Design

Meg C. Murray
Kennesaw State University, mcmurray@kennesaw.edu

Mario Guimaraes
Kennesaw State University, mguimara@kennesaw.edu

Follow this and additional works at: http://digitalcommons.kennesaw.edu/facpubs

Part of the Databases and Information Systems Commons, and the Instructional Media Design Commons

Recommended Citation

Animated Courseware Support for Teaching Database Design

Meg Murray and Mario Guimaraes
Kennesaw State University, Kennesaw, GA, USA

mcmurray@kennesaw.edu; mguimara@kennesaw.edu

Abstract

Database technologies are a core component of computing technology curricula. Their importance will only continue to expand as more and more data are retained in database systems. An effective database system is predicated on a good data model. Database design is the conceptualization of an underlying data model that progresses from identifying user requirements to graphically depicting those requirements and finally mapping the graphical models to a set of tables implemented in a physical database system. The abstract nature of much of database design makes it challenging to teach. This paper presents a set of software animations designed to support the teaching of database design concepts. Topic areas covered include mapping problem descriptions to Entity Relationship (ER) diagrams, mapping ER diagrams to tables, normalization and denormalization and comparison of various ER notation sets. The animations are fairly intuitive to use and are independent of any specific database text or product. They are intended to provide supplemental instructional support and also provide students with additional learning opportunities.

Keywords: database, database design, entity relationship diagrams, normalization, database software animations.

Introduction

Database technologies are foundational to effective information systems and an effective database system is predicated on a good data model. The goal of database design is to conceptualize an underlying data model and determine an appropriate structure to be used in the physical implementation of that model. Consequently, understanding database design is central to understanding a good data model and an effective database implementation. However, the abstract nature of database design can make it challenging to teach and challenging to learn (Chilton, McHaney, & Chae, 2006). One way to address this challenge is through the use of supplemental instructional materials. We are developing a set of software animations to support the teaching of database design as well as other database related concepts. The animations are part of a larger animated database courseware project funded through a National Science Foundation Course, Curriculum and Laboratory Improvement grant. The courseware is freely available and can be accessed at http://adbc.kennesaw.edu

Animations and visualizations have been touted as a means to provide additional instructional support to convey complex and dynamic topics especially when commonly used teaching method-

ologies such as lectures and textbooks do not appear to be sufficient. Animations have been used in many disciplines, including computer science, where animations exist to support the teaching of algorithms, machine execution, networking and programming. While the achievement of learning outcomes as a result of using animations has been mixed, there is evidence that animations do support the teaching and learning process. Animations have been found to increase student motivation and visualizations have been found to help students develop understanding of abstract concepts which are otherwise considered to be 'invisible' (Steinke, Huk, & Floto, 2003). In their study of ways to introduce the idea of pervasive computing to students, Woodcock, Burns, Mount, Newman, and Gaura (2005) cite that using words and pictures helped students construct verbal and pictorial mental models as well as to build connections between them and increased their ability to understand the precepts of the topic under study. Further, Mayer (2001) notes that the capacity of animations to portray information in a different way engages the use of an additional cognitive channel, thereby, reducing the learner's overall cognitive load. Whatever the case, providing multiple venues of learning support such as supplemental instructional materials does contribute to heightened levels of student learning.

A review of the literature revealed very few animations or visualized tutorials that support the teaching of database concepts. A majority of these programs focus specifically on some aspect of SQL or query execution. For instance, W3Schools (W3Schools, 2009) offers an interactive SQL tutorial and Allenstein, Yost, Wagner and Morrison (2008) provide a simulation system to enable students to visualize the steps required to execute a database query. The WINRDBI Educational Tool developed at Arizona State University (WINRDBI Educational Tool, n.d.) provides an interpreter that allows students to program in interactive SQL, relational algebra, domain calculus or tuple calculus. While several drawing tools are available that provide support for the mechanical creation of ER diagrams, no animated instructional support materials were found that that address the topic of database design.

The process of database design includes identifying user requirements, conceptualizing those requirements into a graphical data model, mapping that model into a physical database system and assessing the model to insure data anomalies are reduced, data integrity maintained and data redundancy eliminated, a process known as normalization. The animations included for database design in the Animated Database Courseware (ADbC) project follow this process for the development of a relational database system. Specifically they include sub-modules for mapping scenarios to data models using Entity-Relationship (ER) Diagrams, converting ER diagrams to tables and normalization and denormalization. They also include a sub-module demonstrating various ER notation sets. In this paper, each of these modules will be discussed including a short discussion of the design principle, the purpose of the animations and how they might be utilized in the classroom.

ER Notation Sets

An Entity-Relationship (ER) Diagram is a conceptual representation of data and their relationships. ER diagrams depict the organization of data into thematically characterized entities supported by correlated details referred to as attributes or fields and are comprised of four components, entities, relationships, attributes and cardinalities. (Although, for simplicity and to avoid clutter, attributes are sometimes not displayed.) Relationships between entities are identified as the cardinality or quantitative minimum or maximum number of instances of an entity that may participate in a relationship. For example, in a typical academic scenario, entities might be organized into students, faculty and courses. 'Faculty teach courses' and 'students take courses' are examples of relationships. Cardinality defines minimum and maximum participation such as 'a faculty member may teach from one to many courses.' Cardinality is closely aligned with predefined rules.

ER Diagrams were first articulated by Peter Chen in 1976 (Chen, 1976) but no standard notation set has been adopted for their creation. This can be confusing as different software designers, database texts and different software programs use different symbol sets. The purpose of the ER Notation Sets animation is to allow users to quickly associate one notation set with another. The animation consists of seven different relationships with six different notation sets. Relationships include 1-N Binary Non Identifying, 1-N Binary Identifying, N-N Binary, 1-1 Binary, 1-N Unary, Generalization (incomplete, overlap) and Generalization (complete, disjoint). Notation sets depicted include the Maximum, Minimum/Maximum, Martin (1989) notation (as used in Hoffer, Prescott & Topi, 2008), UML Class and the IDEF1X adopted by the US federal government. Also included is the notation used by Kroenke (2006) and Kroenke and Auer (2008) in their many database textbooks. Other notation sets exist but this collection, while including some of the more well-known notation sets, orients students to the non-standardization that exists in symbols employed in ER Diagram model development. Figure 1 depicts the ER Notations animation interface.
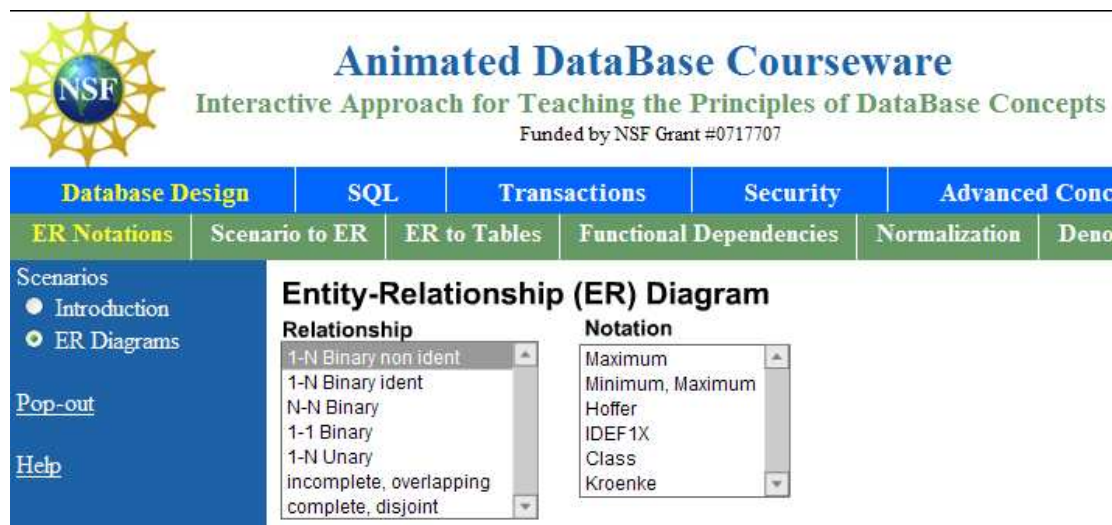

Figure 1: ER Notations Animation Interface

Figure 2 shows a 1-N Binary Non-Identifying Relationship in three different notations: Maximum, Minimum/Maximum, and Martin (from Hoffer, Prescott, & Topi, 2008). It is a binary relationship because there are two entities involved. It is non-identifying as neither entity identifies the other. The numbers on each side of the relationship indicate cardinality. Note that when there is only one number on each side, it always refers to maximum cardinality. According to the diagrams in Figure 2, an Employee may belong to a maximum of one Department and a Department may have a maximum of many Employees. The minimum and maximum notation is a more complete description of the relationship, but also more complex. It reads, an employee may not work in any Department (zero) and may work in at most one department. A department has at least one employee (1) and may have many employees (N). Students often benefit from reviewing the maximum notation first followed by the minimum/maximum groupings. Also included in this figure is the Martin notation which utilizes 'crow's feet' to depict cardinality. The oval corresponds to optional, the hashmark to one or mandatory and the crowfoot to many. It is easy to explain this notation, when it is compared side-by-side with the Minimum/Maximum notation.
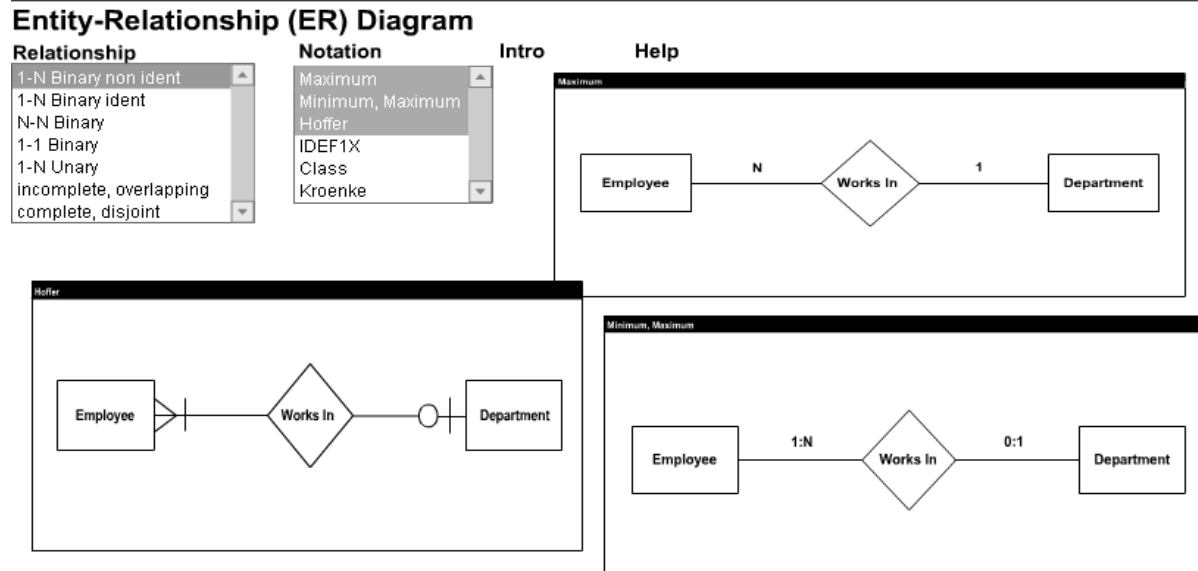
Figure 2: Example One-to-Many Binary Relationship using Three Different Notations

Figure 3 demonstrates the same relationship using three different notation sets. IDEF1X uses the dashed line to distinguish a one-to-many non-identifying relationship. The Kroenke notation shows the maximum cardinality inside the diamond and the minimum cardinality close to the respective entities. The class diagram notation uses an asterisk to show a one-to-many relationship.
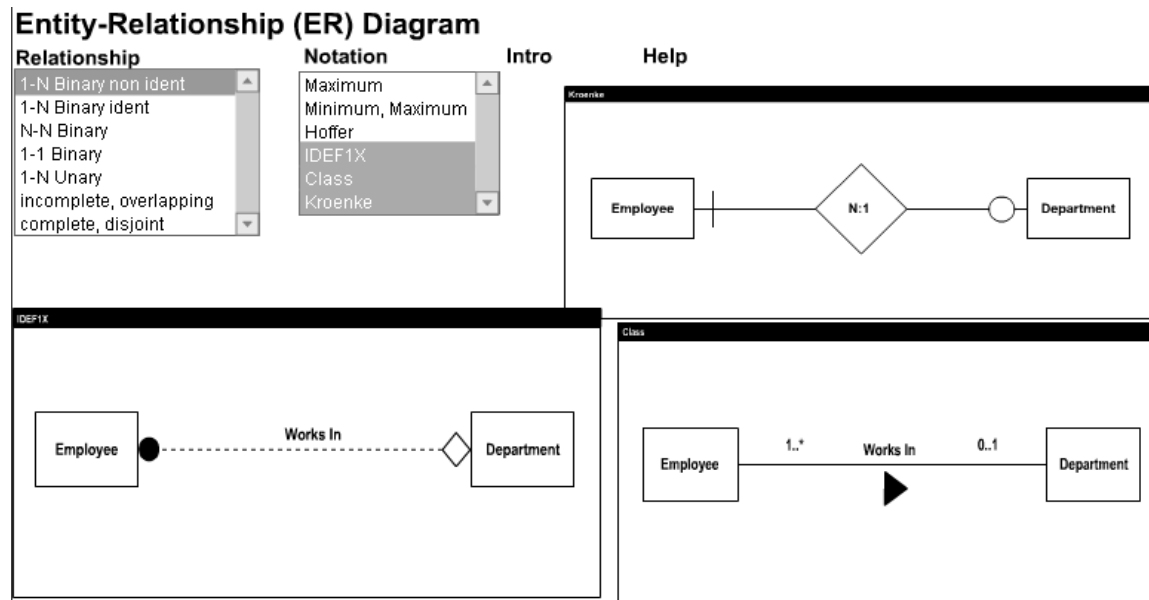

Figure 3: Example One-to-many Binary Relationship Using Three Different Notation Sets

Figure 4 demonstrates how various notations sets depict an identifying relationship. In an identifying relationship, a weak or dependent entity exists. A weak entity is one that cannot exist on its own. In this example, the weak entity is named 'Section.' A section of a course cannot exist with-

out a corresponding course. In an identifying relationship, the minimum cardinality of the parent entity is always one or mandatory. In these notation sets, the weak/dependent entity is either depicted as a round rectangle or a rectangle inside a rectangle.
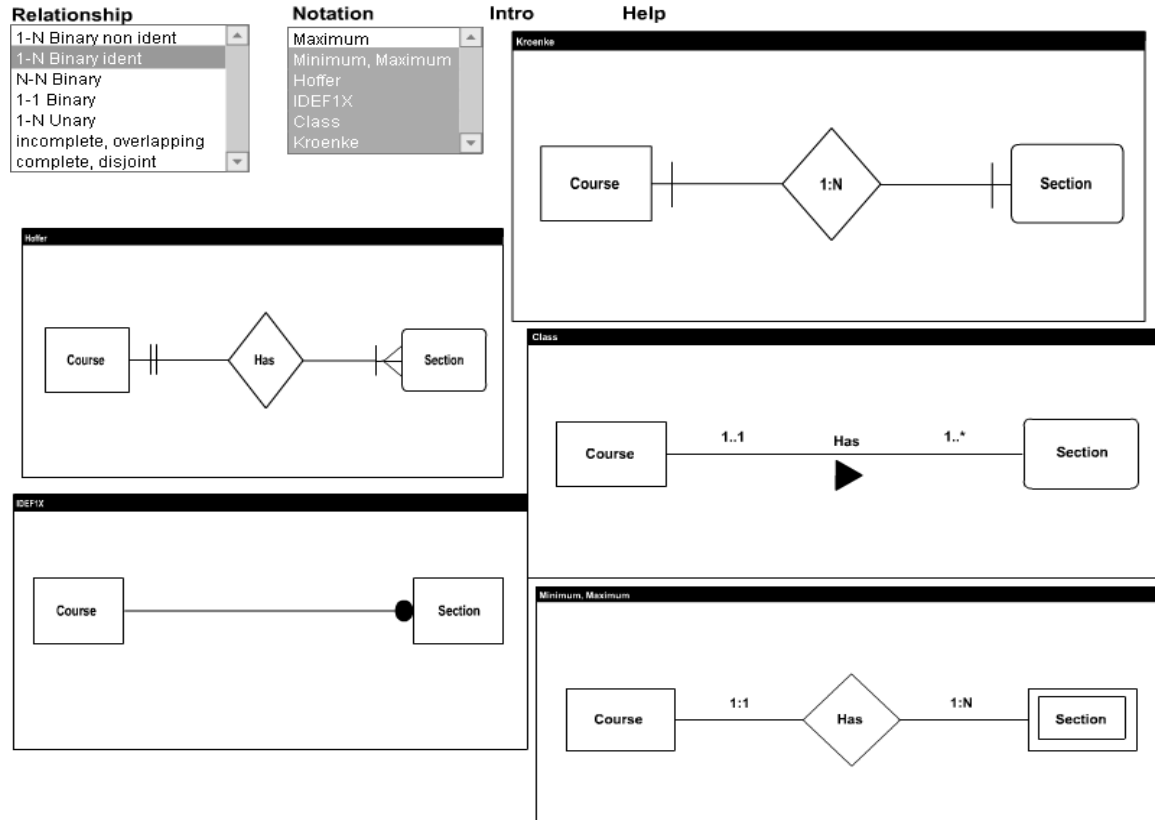

Figure 4: Binary One-to-Many Identifying Relationship

Other relationships included in this sub-module include unary relationships and generalization. In a unary relationship (also called recursive) there is only one entity involved and the relationships are between instances of the same entity. Generalization, referred to as an 'is-a relationship' represents supertype/subtypes. Figure 5 shows an example generalization in which the subtypes are overlapping (the supertype may also be represented in more than one subtype).

Entity-Relationship (ER) Diagram

Figure 5: Generalization Relationship Depicted in Different ER Notation Sets

The ER notation animations may be used in the classroom when the topic of ER diagrams is introduced. One approach is to ask students to work through each of the relationships using only one of the notation sets. Then students are asked to explore various combinations of the relationships using various notation sets. Students should be directed to use this animation for future reference when creating or interpreting ER diagrams.

Scenario to ER Diagrams

Data models are built from an analysis of a problem scenario and user requirements. These requirements are obtained by collecting data from users, input forms, reports and other related documentation. Conceptual design includes the visualization of the requirements in a graphical data model. As noted previously, the ER diagram is the most commonly used technique for modeling data for relational databases. The ER diagram visually displays entities and the relationships that exist between them.

This sub-module includes a set of eight different problem scenarios in which users are presented with four possible ER diagram solutions and asked to choose the best one. The scenario provided may be a problem specification, a form or a report. The first four scenarios are problem descriptions (Employees, Automobiles, Medical and Car Dealer), the next two are forms (Orders, Phone) and the last two are reports (Suppliers and University). Figure 6 shows possible ER diagram solutions for a scenario based on a problem description. Solutions may be viewed individually or all together. Users select the solution they believe to be correct and corresponding feedback is provided.
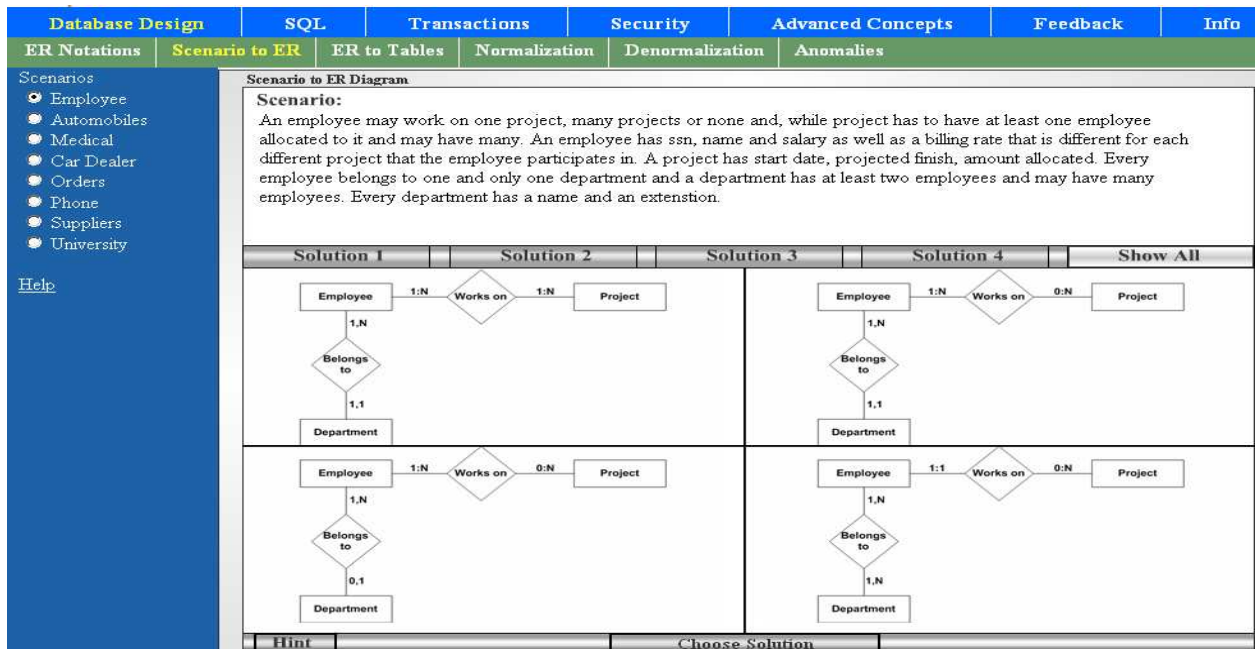
Figure 6: Example Scenario to ER Diagram Exercise

Animations in this sub-module may be introduced to students to reinforce classroom instruction related to constructing ER diagrams. Completing the exercises may be assigned as an in-class or out-of-class assignment. Students are asked to complete each of the listed exercises. For each scenario, students are instructed to read the scenario, view each of the possible ER diagram solutions and then choose the diagram they feel best meets the specifications outlined in the scenario. Students are encouraged to work through the exercises until they feel they are able to easily identify the ER diagram that most adequately meets the specifications provided.

ER to Table

The next step of the database design process are to map the data model into a physical database system. Entity Relationship diagrams form the basis for this physical implementation. Entities are mapped to tables and relationships are mapped to corresponding foreign keys. In essence, an attribute column is duplicated between two tables. A set of rules exist that that define the best methods for implementing foreign keys. While these rules may be memorized, memorization does not necessarily result in understanding why those rules apply. This sub-module includes a set of seven different animated exercises that demonstrate where foreign keys should be placed. Five different relationships are provided including one-to-many binary and unary, one-to-one binary and many-to-many binary and unary. Each exercise offers the user the option to view the construction of an associated E-R diagram. The ER Diagram is constructed and data is automatically loaded in the tables. However, the tables have no relationships implemented. The user is asked to determine, from a choice of four possible options, the best way to map that relationship to tables. After each option, an animation is shown that completes the tables by adding foreign keys or association tables. The pros and cons of each option are illustrated through animation. An example of the initial construction of the ER diagram and possible solutions is provided in Figure 7.
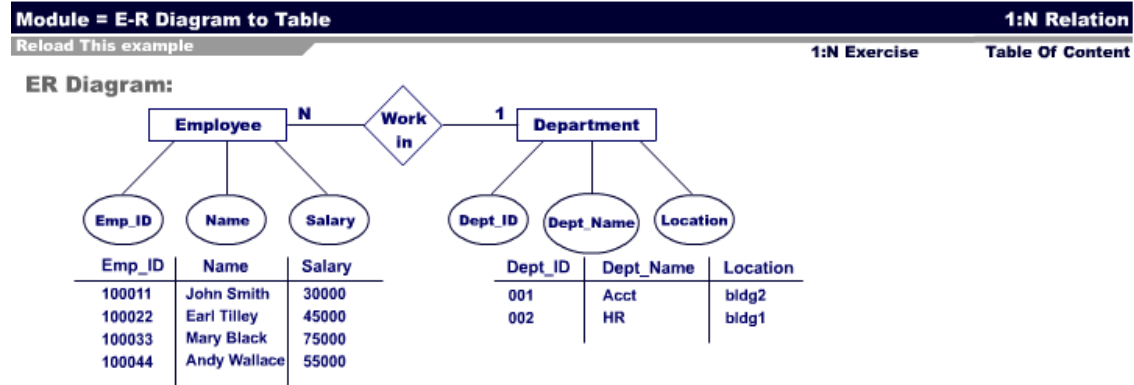
Figure 7: Example ER Diagram Mapping to Tables

As with the Scenario to ER Diagram animations, these animations are used to reinforce classroom instruction. They can either be used in-class or as out-of-class assignments. For each scenario, students are asked to read the scenario, view the ER diagrams and the associated tables and then proceed to the exercise. Students should complete the exercise by choosing one of the possible answers and then reviewing the animation of the result of the answer they chose. Students are encouraged to work through each exercise and each answer until they feel they are able to apply the standard rules for mapping ER diagrams to tables for each of the relationship types presented.

Normalization

The objective of a good database design is to reduce redundancy and normalization is the process of organizing data into tables with the goal of reducing those redundancies. Unnecessary redundancy often results in poor performance and has the potential to introduce data anomalies increasing the risk of data inconsistency. Designing normalized tables is a subject of great difficulty, yet, great importance. Normalization in categorized into forms with each higher-level form interjecting a refinement of the lower-form. In practice normalizing data to the third normal form is most common. This sub-module includes a tutorial that steps through the normalization of a large table into a set of tables complying to the rules of the third normal form. It also includes descriptions and an example of how first, second and third normal forms might be violated. Figure 8 shows the tutorial screen which begins by displaying an unnormalized table.

The table below represents unnormalized data. This can be indicated by (1) an intended primary key including null values; (2) multiple values in a column/row intersection; or (3) anomalies occuring during updates, or deletions. For example, if professor Harley changes his office, the room number would have to be changed in each row it occurs in.

UnNormalized

CourseID	CourseName	StudName	SSN	DOB	Prof	Bldg	Floor	RmNo	Grade
10058	CSIS 2520	Sam Hall	254638589	5/19/59	Harley	BB	4	405	C
		Ted Ringly	752135423	11/7/72	Harley	BB	4	405	B
25600	ART 1101	Jane King	232522654	1/5/81	Bartett	PI	2	201	A
38569	MATH 4454	John Smith	458963232	8/1/69	Harton	CL	1	106	D
10654	CSIS 3600	Jane King	232522654	1/5/81	Snipes	JM	2	217	A

Figure 8: Normalization Tutorial Introduction Screen Showing an Unnormalized Table

The tutorial is followed by a set of exercises in which users are presented with a scenario, a set of data and a set of functional dependencies, and asked to normalize the data into tables adhering to the rules of third normal form. Users choose attributes and create tables containing these attributes. Clicking the finish button, reports on whether the normalization process has been completed correctly or not. Table 9 shows an example normalization exercise.

Instructions

From the unnormalized table(s) below and the Functional Dependencies, construct new tables such that all tables are in 3NF.

1) Click on the fields to add to a new table,
2) Click "Create Table",
3) Click on the fields in the new table to set as primary key,
4) Repeat steps 1-3 until finished,
5) Click the "Finish" button

Controls

Create Table

Restart

Finish

Functional Dependencies

VID	→	Vtype
		OwnerID
OwnerID	→	OwnerName
SuspectID	→	SuspectName

VID	Vtype	OwnerID	OwnerName	SuspectID	SuspectName
207	Escort	17	Jack	10	John
205	Quest	22	Mary	10	John
336	Neon	17	Jack	14	Joe
208	Camry	15	Sue	18	James

Figure 9: Example Normalization Exercise

The tutorial can be used in class to demonstrate the normalization process. Then students may be assigned to complete each of the six exercises working through them until they feel confident they understand the steps to normalize database tables to third normal form. In addition, a sub-module entitled 'Anomalies' is also included. This sub-module demonstrates problems that can occur when un-normalized tables are used.

Denormalization

Although the usual approach in relational database design is to normalize tables in order to reduce redundancy and its associated problems, normalization may not always be the most desirable design strategy. Certain situations, including read only databases and the need for optimized query retrieval time, are more suitable to denormalization. The denomalization animation presents a scenario and asks users to choose the situation in which denormalization might be appropriate. The result of each option, whether the best solution or not, is demonstrated. Figure 10 presents an example of a denormalization exercise.

Figure 10: Example Denormalization Exercise

Conclusion

As more and more data is retained digitally in database systems, understanding what makes a good data model will grow in importance. Database technology is a core knowledge area in computing curriculum and database design is a core component of this curriculum. However, the abstract nature of data modeling makes it challenging for many students. Supplemental instructional materials provide students with additional learning opportunities. This paper provided an overview of a set of animated courseware developed to support the teaching of database concepts. Topic areas covered in these animations include mapping scenarios to ER diagrams, mapping ER diagrams to database tables and database normalization. The animations can be used in the classroom to provide additional means for demonstrating concepts. They can also be assigned as in-class or out-of-class assignments. Preliminary evaluations of the software have been positive. Student responses to surveys of their perceptions of the value of the animations and the results of a recently conducted usability study indicate that students perceive the animations to be valuable and contribute to their learning. In a pilot control group evaluation, students taught using the software outperformed those who did not on classroom examinations. Providing students with access to multiple learning channels and multiple modes of presentation, such as visualizations and animations, has been shown to increase student motivation, help students develop deeper understandings and achieve higher levels of learning.

References

Allenstein, B., Yost, A., Wagner, P., & Morrison, J. (2008). A query simulation system to illustrate database query execution. *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education, SIGCSE '08*. ACM, New York, NY, 493-497.

Chen, P. (1976). The entity-relationship model - Toward a unified view of data. *ACM Transactions on Database Systems, 1*(1), 9-36.

Chilton, M. A., McHaney, R., & Chae, B. (2006). Data modeling education: The changing technology. *Journal of Information Systems Education, 17*(1), 17-20.

Hoffer, J. A, Prescott, M., & Topi, H. (2008). *Modern database management* (9th ed.). Upper Saddle River, NJ: Prentice Hall.

Kroenke, D. (2006). *Database processing: Fundamentals, design, and implementation* (10th ed.). Upper Saddle River, NJ: Prentice Hall

Kroenke, D., & Auer, D. (2008). *Database concepts* (3rd ed.). Upper Saddle River, NJ: Prentice Hall

Martin, J. (1989). *Information engineering: Introduction*. Upper Saddle River, NJ: Prentice Hall.

Mayer, R. E. (2001). *Multimedia learning*. Cambridge: Cambridge University Press.

Steinke, M., Huk, T., & Floto, C. (2003). Helping teachers develop computer animations for improving learning in science education. C. Crawford et al. (Eds.), *Proceedings of the Society for Information Technology and Teacher Education International Conference 2003*. Chesapeake, VA: AACE, 3022-3025.

W3 Schools. (2009). *SQL Tutorial*. Retrieved March 8, 2009 from http://www.w3schools.com/

WINRDBI Educational Tool. (n.d.). Retrieved March 8, 2009 from http://www.eas.asu.edu/~winrdbi/

Woodcock, A., Burns, J., Mount, S., Newman, R., & Gaura, E. (2005). Animating pervasive computing. *Proceedings of the 23rd Annual International Conference on Design of Communication, SIGDOC '05*. ACM, New York, NY, 116-122.

Biography

Dr. Meg Murray is an Associate Professor of Information Systems at Kennesaw State University. She has extensive experience in both academe and industry in the areas of software design, development and architecture. Her current work has focused on web services and using XML as a medium for data exchange. She is also actively involved in defining what it means to be IT literate in today's world. She developed a model for a healthcare information systems portal to facilitate the exchange of healthcare data stored in a variety of backend database systems. Twice she has been a semifinalist for 'Women of the Year in Technology in Georgia.'

Dr. Guimaraes is an Associate Professor of Computer Science at Kennesaw State University where he teaches Database Management Systems. He has extensive experience in the development of educational software and in teaching database content. Dr. Guimaraes has many years of experience both developing and coordinating the development of educational software and his software, ASA, has been commercialized throughout Brazil. He was a database administrator for the Brazilian Navy. Most recently, he completed a post-doctoral fellowship in Information Assurance with an emphasis in Database Security.