OmniaScience

Journal of Technology and Science Education
JOTSE, 2018 – 8(4): 238-253 – Online ISSN: 2013-6374 – Print ISSN: 2014-5349
https://doi.org/10.3926/jotse.365

APPLYING TSSL IN DATABASE SCHEMA MODELING: VISUALIZING THE SYNTAX OF GRADUAL TRANSITIONS

Adi Katz
Sami Shamoon College of Engineering (SCE) (Israel)
adis@sce.ac.il

Received December 2017
Accepted February 2018

Abstract

Since database conceptual modeling is a complex cognitive activity, finding an appropriate pedagogy to deliver the topic to novice database designers is a challenge for Information Systems (IS) educators. The four-level TSSL model that is known in the area of human-computer interactions (HCI) is used to explain and demonstrate how instructional design can minimize extraneous cognitive load in the conceptual modeling task of designing a database schema. The instructional design approach puts focus on the syntactic level of TSSL, to explain how visualizing gradual transitions between hierarchic levels of the schema is effective in database modeling. The current work demonstrates the approach, and at the next phase we plan to experimentally test the effectiveness of the approach by comparing performance and attitudes of students who are exposed to emphasizing the syntax of the gradual transitions in schema structure to those who are not exposed to it.

Keywords – Conceptual modeling, Database schema, TSSL, Pedagogy, Visualization.

----

1. Introduction

This is a pedagogic research that offers an approach for effectively delivering and instructing the activity of relational database schema modeling to novice database designers. Modeling users' requirements is a very important task in information system analysis and design, to ensure that the intended system would meet organizational goals (Dahan, Shoval & Sturm, 2014). The purpose of data modeling is to correctly capture users' reality. The activity of modeling involves the creation of relations, attributes, and indicating relationships among relations with a set of integrity constraints (Elmasri & Navathe, 2011). Modeling a database schema according to user or organization scenarios is an essential skill, but a complex cognitive process for novice database designers. Therefore it is error prone. Students in database courses demonstrate over and over again the difficulty to think like data modelers (Watson, 2006). While many database textbooks present various approaches, methods, and techniques for database design, an ongoing debate exists with regards to the effectiveness of certain approaches within the classroom and in practice (Fotache, 2006). It is important to find effective approaches so that IS students, in their professional occupations, will design systems that would correctly model the reality of organizations, and that would successfully support business activities. Previous pedagogic works in database modeling show attempts to deal with the challenge of effectively delivering the complex topic using different approaches. Interesting attempts include the integrated spiral approach (Watson, 2006), the cognitive apprenticeship based approach (Al-Dmour, 2010), and the learning from errors approach (Katz & Shmallo, 2016).

The intention in the current paper is to theoretically explain how an approach of emphasizing the syntax of a hierarchical structure of database schemas with two main gradual transitions between different levels is useful in the process of learning relational database schema modeling. TSSL, a four-level model (Foley, van Dam, Feiner, & Hughes, 1990) known in the area of human computer interactions (HCI) is applied as a theoretic framework to examine the interaction between novice database designers and organization scenarios. A textual description of an online flower shop scenario is used to demonstrate how this pedagogic approach simplifies the conceptual activity of database schema modeling.

1.1. Challenges in Teaching Database Schema Modeling

In database courses, students learn various activities that are related to defining, creating and manipulating databases. We focus on the conceptual level, in which a relational schema is defined to express the database requirements of a specific organization, institute or any other context for using an information system. In this stage, database designers need to accurately analyze the organization's needs and semantic constraints (business rules), to form a database schema that includes a list of relations (or entities), attributes, relationships, and interrelation referential integrity constraints which are used to maintain consistency of reference among records from different relations (Elmasri & Navathe, 2011).

Novice database designers struggle with the cognitive complexity needed for transforming an organizational description into a proper database schema of relations and relationships. Previous studies analyzed students' solutions to find typical errors in order to investigate their misconceptions about relations and relationships. These studies found that during conceptual database design, students face significant cognitive difficulties and mental challenges (Fessakis, Dimitracopoulou & Komis, 2005; Katz & Shmallo, 2015). A previous research examined students' errors in database modeling using a course exercise, in the form of a textual scenario. According to the scenario, the students were required to identify relations, attributes, keys (identifiers), and to draw foreign key-primary key (hereafter FK and PK, respectively) relationships between relations. Their solutions to the modeling exercise were analyzed to map modeling errors into distinct categories and sub-categories. The two main categories of errors revealed a dichotomous distinction between additions and omissions. Among other error types revealed, within the sub-categories for additions were the prominent errors of adding redundant relations, and redundant FK-PK relationships between relations. On the contrary, within the sub-categories for omissions were repetitive errors of missing relations, and a failure to define crucial FK-PK relationships between existing relations (Katz & Shmallo, 2015). The approach described in this paper focuses primarily on reducing these types of errors.

Relational database modeling is a central topic in database courses, and therefore it is extremely important that novice designers will be received the material effectively, to understand it correctly. The learners' working memory resources, particularly novices, can quickly become overwhelmed by task complexity and, as a result, their learning process will suffer (Anthony, 2008). Cognitive load theory provides a framework for designing instructional materials and is focused on identifying instructional designs that can effectively reduce unnecessary cognitive burden on the learner (van Merriënboer & Sweller, 2005). According to cognitive load theory, among three types of cognitive load is an "extraneous cognitive load", an ineffective cognitive load that results from instructional techniques that require learners to be engaged in working memory activities that are not related to schema construction or automation (Sweller, 1998). To avoid confusion, unlike the database- oriented use of "schema" throughout this paper, the notion of schema in cognitive theories refers to knowledge structures that are organized and stored in the human long-term memory. Since extraneous cognitive load is imposed by the ways information is presented, it is encouraging for educators to acknowledge that this type of cognitive load is under the control of the instructional design, and can be reduced by it. How complex or simple something is depends critically upon the way in which it is described. To achieve simplification, we must find the right representation. In line with cognitive load theory, it has been found, that visualization supports learning by decreasing cognitive load and enhancing comprehension as it seems to reduce extraneous cognitive processing (Schwamborn, Thillmann, Opfermann & Leutner, 2011). In addition, hierarchy is one of the most effective ways of organizing complexity for human comprehension as it allows representations at different levels of detail, with a manageable complexity at each level (Flood & Carson, 1993). The fact that many complex systems have a nearly decomposable, hierarchic structure is a major facilitating factor enabling human beings to understand and to describe such systems (Simon, 1962). In line with these findings, the approach presented in this paper puts emphasize on visualizing hierarchic levels in a database schema.

In the following section we apply the TSSL model that originally explains human computer interaction in the context of interface design to the context of instructional design in the area of relational database schema modeling. The TSSL approach is used as a conceptual framework to demonstrate how visualizing the syntax of a hierarchical schema structure characterized by gradual transitions, has a high potential of reducing the extraneous cognitive load in database modeling.

2. Approach

2.1. Applying TSSL in Database Schema Modeling

The TSSL model is a four level model to explain human interaction with a system (Foley et al., 1990). TSSL is an abbreviation for: task, semantics, syntax and lexicon. TSSL is a multilayer model for user activity, originally developed as a high-level theory, which can be used as a conceptual framework for designing user interfaces, as well as for conducting evaluations and inspecting possible problems in existing designs.

The idea of different levels of interaction can be used not only in HCI but in different contexts of interaction between humans and artifacts; therefore, TSSL is applied as a lens for understanding and analyzing the conceptual activity of database schema modeling. In HCI, TSSL is used to instruct interface designers when developing a system for users that need to accomplish certain tasks. In the current study, TSSL is used to show how instructional designers (educators) can help novice database-designers (students) accomplish the task of modeling organizational requirements in terms of a relational database schema.

In HCI, while the upper levels of task and semantics can be viewed independently of their physical implementation, the syntax and lexicon are tied to their implementation. A dichotomous distinction between semantic and syntactic interactions to explain programmers' behaviors, treat semantic knowledge as general and meaningful sets of information that are independent of the syntactic knowledge of particular programming languages or facilities (Shneiderman & Mayer, 1979). In a similar manner, in the area of relational database schema design, while the two upper levels of TSSL, task and semantics, are close to the organizations' or users' realities, the two lower levels of TSSL, syntax and lexicon, are affected by the pedagogic implementation of educational practitioners. Database educators must be able to consistently integrate across the distinct levels of interaction when delivering the topic of database modeling.

In problem solving, top-down implementation for a problem demands that the highest levels be set first, followed by more detailed analysis, a process that is referred to as "working backwards" or "reformulating the goal". A bottom-up implementation would permit low level elements to be generated first, in a process referred to as "working forward" or "reformulating the given" (Shneiderman & Mayer, 1979; Wickelgren, 1974). As in many other areas of problem solving, schema modeling is a process often solved by both techniques, top-down and bottom-up, interchangeably. Database educators should integrate both top-down and bottom-up approaches in database design (Kung, Kung & Gardiner, 2013).

In TSSL, since each level provides the context for the level below (Te'eni, Carey & Zhang, 2005), it is convenient to describe the model's four levels from top to bottom for demonstrating its applicability as a pedagogic framework for guiding database educators on how to deliver the topic of database schema modeling. However, alongside the top-down description, we pursue a bottom-up influence, in which certain visualization choices deliberately made by educators at the lower and physical levels (lexical and syntactic levels), will promote the higher levels of comprehending (semantic level) and fulfilling the goal of designing an accurate schema for organizational users (task level). Just as interface designers in HCI make lexical and syntactic choices when developing a system that is used to accomplish certain tasks, instructional designers should make lexical and syntactic choices in textbooks and in the classroom when teaching the complex topic of database schema modeling. The lowest lexicon level holds the building blocks for the next syntactic level. The syntactic level, with a proper instructional design using visualizations aids (such as relative locations, sizes, and colors) can hold information regarding the relative differences between the building blocks at different levels of the schema. In turn, these visualization aids would be utilized at the following semantic level to support the comprehension of a gradual shift in the levels of abstraction of entities in a database schema. Understanding the different levels of abstraction will lead to a higher performance in the task of schema modeling. Since TSSL is applied to the pedagogy of database modeling, from an instructional designer (educator) perspective, we will dwell on the lower levels and particularly on the syntactic level. An appropriate bottom-up influence can be achievedc when the syntactic level integrates the lexical level's building blocks, in a way that creates a deeper understanding of the entities (relations) and relationships between entities at the semantic level, to accomplish an appropriate expression of organizational database requirements at the highest task level.

As mentioned earlier, it is convenient to describe the TSSL model's levels from top to bottom. In the following section the TSSL is explained and demonstrated from the highest task level to the lowest lexicon level. Table 1 demonstrates each level of TSSL in three different contexts: human computer interactions, interpersonal communications, and database schema modeling.

TSSL levels	**HCI: job application website**	**Communication: spoken or written message transfer**	**Databases: schema modeling**
Task	applying to 6 different jobs by submitting a CV	A department manager sends an email to his employees in order to have them perform a certain task by the end of the day	Form a relational conceptual schema that properly describes organization requirements
Semantic	Users understand that there is no need to attach the CV file 6 times, only once. Users know that the CV file is stored in a folder that is placed in a hierarchical organization of folders in their computers.	The manager wants his employees to understand that the request is mandatory and urgent	Database designers understand the meaning of entities, the nature of the relationships among entities (in terms of cardinality ratios – one-to-one, one-to-many, many-to-many), and the entities' different levels of abstraction
Syntax	Sequence of user actions: first mark all the desired jobs, then attach the CV file, and finally submit	Sentences' structure, the sequence of the words (subject, verb, and object) in the sentences	The schema structure has a sequence of a gradual shift in levels of abstraction and a gradual shift in the expansion of primary keys from top to bottom
Lexicon	The interface separate elements: job labels, checkboxes, a button with an attachment clip icon, a submit button, etc.	The sentences' building blocks – each and every word chosen, and appearance (e.g. bold text, text size).	Each component in the schema's diagram - relation, key, field, and relationship. For example, displaying a relation as a table with columns and rows, specifying a PK attribute by underlining its text, etc.

Table 1. Applying TSSL to different contexts

2.1.1. Task

At the uppermost level, the user's task should be identified. In HCI, at this level, the user is interacting with a system in order to achieve certain goals, such as sending a birthday card, purchasing a product, or applying to a job. In database modeling, the task level is about organization requirements and constraints that have to be met in a relational conceptual schema. The goal of database designers is to define a database schema that properly and accurately expresses an organization's description of needs. Each schema includes relations with attributes (fields), keys (primary and foreign) of relations, and connections (relations) between relations.

2.1.2. Semantics

In HCI, the semantic level is related to the user's world of meaning and also to the computer's logical structure (Te'eni et al., 2005). This level supports the task, since at this level the user needs to understand certain aspects that are related to the task itself or to computerized systems. Using the example presented in Table 1, when a user is interacting with an online job application website in order to apply for jobs, the semantic level refers to objects and operations through which the computer becomes meaningful to the user. A computer-related semantic level aspect is to know that a CV is saved as a file object inside a certain folder (another object), and that the file can be duplicated and kept in different locations in the computer, updated, deleted, printed, and submitted multiple times to various destinations.

In database modeling, designers are required to understand the meaning of entities and relationships between entities. Different questions are raised at this level, such as: What attribute of an entity unambiguously identifies its records? What is the meaning of an identity? Also, given different attributes that are candidates for serving as the PK of a relation, which one is most preferable? Different consideration can be taken into account regarding each question. At the semantic level, database designers are required to think about the consequences of their choices. For demonstration, referring to deciding what PK is best among alternative candidates, a department can be unambiguously identified by a unique name, code, or the SSN (social security number) of its manager (in accordance with a constraint that a manager can manage only one department). Department names vary in their length, and are usually longer than numeric codes, and therefore the name solution is wasteful in terms of storage space. Choosing the manager's SSN to identify a department is conceptually odd since SSNs identify people, not departments, and in addition, managers can be replaced periodically.

A most crucial aspect in modeling a relational database at the semantic level is to identify the nature of the relationships among relations (entities) derived from the organizational scenario, in terms of cardinality ratios –one-to-one, one-to-many, many-to-many (1:1, 1: N, M: N respectively).

2.1.3. Syntax

In HCI, the syntactic level is about rules that combine objects and operations in a specific computerized system. At this level, the design of the specific system and the relationships between the various design elements affect the ability of users to execute their intentions for achieving their goals, or succeeding in their tasks. Following the example of users interacting with an online job application interface, at the syntactic level the users should know that in this specific website, they need to first select all the desired jobs from a list of jobs, then upload their CV file, and at the end they need to submit the request of proposing their candidacy for the job. A system that forces the users to upload their CV file before they can submit a job request has a syntactic constraint that prevents them from sending incomplete information.

As aforementioned, in the area of relational database schema design, while the upper levels of TSSL, task and semantics, are close to the organizations' realities, the lower levels of TSSL, syntax and lexicon, are affected by the pedagogic implementation of educational practitioners. When teaching database modeling, educational practitioners can use different syntax to display database schemas. According to cognitive load theory, extraneous cognitive load is imposed by the way information is presented, and therefore is under the control of instructional (educational) designers. Hence, this level is extremely important for reducing and coping with the cognitive load that arises from an organizational description of requirements and constraints. Visualization supports learning by reducing extraneous cognitive processing (Schwamborn et al., 2011). A proper syntactic representation organizes the information and supports the understanding at the upper semantic level. A representation without a suitable syntax to guide the database designer might be overly complicated, may increase extraneous cognitive load, and will probably lead to difficulties in developing understanding of the entities and relationships of a given organizational context.

Research findings point to the fact that hierarchy is one of the most effective ways of organizing complexity for human comprehension. The nature of many complex systems, characterized by a hierarchic structure enables the comprehension of such systems (Simon, 1962; Flood & Carson, 1993). Hierarchical structuring has been a key tool for abstraction, as it removes the complexity of large schemes generated by organizational modeling (Gandhi, Robertson & Van Gucht, 1994). Research on memory indicates that hierarchical organization of materials serves as a retrieval cue for rec all, with a general–specific structure, that helps locate particular items (Najarian, 1981). In line with the findings regarding the advantages of proper visualization and of hierarchal structures, our pedagogic approach puts emphasize on visualizing both the hierarchic levels in a database schema, and the gradual shifts between the hierarchical levels.

As aforementioned, a previous study in relational database modeling found that prominent errors made by database students are creating redundant relations, and adding redundant FK-PK relationships between relations, and on the contrary, omitting required relations, and a failure to define crucial FK-PK relationships between existing relations (Katz & Shmallo, 2015). These errors can be reduced by simplifying the relational schema to a hierarchic structure, and applying the syntactic level of the TSSL framework by emphasizing two parallel transitions that occur when moving up or down the hierarchy. Section 2.2 demonstrates in the context of an online flower shop scenario, how all types of relationships can be simplified by treating them as hierarchical. According to the current pedagogic approach, visualizing the syntax of gradual transitions across levels of a schema's hierarchy, will serve as a guide for novice database designers in the process of modeling various and entirely diverse organizational descriptions.

In order to promote the semantic understanding of relationships between relations of any organization, our approach focuses on a structured diagram with a syntax that visually highlights the hierarchical nature of a database schema. The hierarchical structure is characterized by two main gradual transitions from parent to child levels of the hierarchy (or vice versa when going bottom-up). One gradual transition is a semantic transition in terms of changes in the degree of abstraction. Going top-down, the first level starts with entities (things) in the real-world that are clear, usually tangible and straightforward and gradually going down the hierarchic levels, relations represent entities that are relatively either more abstract (e.g. events), more specific, more detailed, and usually more complex to comprehend. The other transition that can stand out easily, and therefore simple to emphasize visually is that going from top to bottom, the relations' PK gradually include more fields, in other words, PKs gradually expand. They include the PK of their parent and an additional field (or fields), or a combination of the PKs of mutual parents.

All types of relationship cardinality ratios, one to one (1:1), one to many (1: N) and many to many (M: N) can be represented by the syntax of an hierarchical structure. A FK of a child relation is a PK of a parent relation, used to reference the parent. There are three patterns of FK-PK relationships:

1. The FK is a regular field in the child relation: This pattern usually expresses a 1: N, parent-child relationship. Sometimes this pattern presents a 1: 1 relationship that is more difficult to present hierarchically, because it is not easy to determine who the parent is and who the child is. For example, in most organizations, a department has only one employee who manages it, and a manager-employee can manage only one department. Since not every employee manages a department, but every department has a manager, the referential integrity constraint would be that a department's manager has to exist in the employees' relation. Therefore, the departments' relation will be defined as the referencing relation, hence, the child.
2. The FK is part of the PK in the child relation: A pattern that expresses an M: N relationship. Symmetrically, each entity can be treated as both the parent and the child of the other. In

relational databases, M: N is implemented by means of a cross-reference (also called junction) relation, in a way that forms a pair of 1: N relationships. Treating an M: N relationship as two hierarchic relationships by considering two parents of a joint child (the cross-reference relation) simplifies a relatively complex relationship.

3. The FK is also the PK in the child relation: expresses a 1:1 relationship. This pattern usually expresses an "IS A" relationship, known as a generalization-specialization pattern. The child relation is a specific sub-type of its parent (paralleling the inheritance concept in the object-oriented approach). This is the only case in which the PK of a child does not expand the PK of its parent.

2.1.4. Lexicon

In the area of HCI, the lowermost lexicon level relates to the specific elements such as computer devices and specific operations. In the example of a user interacting with an online job application website, users choose several desired jobs by clicking with the mouse device inside squares (checkboxes), attach a CV file by clicking on an attachment clip icon, and search for their CV file in a "window" that displays the hierarchical organization of content on their computer, using icons of yellow folders.

In database modeling, lexicon is about the visual appearance of each component in the schema's diagram - relation, key, field, and relationship. For example, it is common to display a relation as a table with columns and rows, to specify that an attribute is a PK by underlining its text, and to draw lines that connect relations to express relationships between them.

2.1.5. Semantic and Syntactic Parallel Transitions in Database Modeling

The gradual semantic transition in relations' abstraction levels and the gradual syntactic transition in PKs' expansion occur in parallel. As previously mentioned, schema modeling is a process in which top-down and bottom-up are interchangeably used techniques. The fact that the semantic and syntactic transitions are parallel supports and enables the integration of both approaches in the database schema modeling. On the one hand, the semantic level serves as conceptual guidance for physically positioning relations in the syntactic level and for properly defining the PKs. On the other hand, the syntactic level of relative relation locations and relative PK sizes (key expansion) increases the comprehensibility of the hierarchic schema structure and the meaning of the entities. Painstaking attention to both transitions encourages database students to go both ways, top-down and bottom-up interchangeably.

In the following section, we demonstrate how to visually and conceptually emphasize the two gradual transitions in the hierarchical structure when modeling data, on an online flower shop scenario.

2.2. Demonstrating TSSL in Database Schema Modeling - The Online Flower Shop Scenario

We now demonstrate the emphasis on the syntax of hierarchical structure of a database schema with gradual transitions, on an online flower shop scenario.

An online flower shop holds data on bouquets, flowers, suppliers, customers and orders. Customers order bouquets and pay by credit card at the shop's web site. Customers can choose to buy prepared bouquets ("Catalog Bouquets") or to assemble bouquets by themselves from a variety of flowers ("Self Bouquets"). Every prepared bouquet has a catalog code, a name, a fixed catalog price, and different types of flowers are included in it, in a specific predetermined amount.

Each flower has a code, a name, and a description. The shop owner can order a type of flower from a number of different suppliers, and of course each supplier can provide multiple types of flowers. Flower prices may vary from one supplier to another. Each type of flower grows in a certain season or in several seasons. All suppliers are identified by a unique code, and have a name, an address, and a telephone number.

The store holds data on customers. Each customer has an ID, name, last name, an address and a telephone. The store keeps the birth date of VIP customers. Also, the store keeps data on customer orders. For simplicity, assume that a customer can order at a certain date only once, and that at the same date he can order both catalog bouquets (an unlimited amount of each) and self-assembly bouquets (an unlimited amount of each). There is a need to know how many bouquets of each type a customer ordered at each date.

Since a self-assembly bouquet has no catalog number, for each customer order, self-assembly bouquets receive a serial number. To optimize the customer's future orders, the shop owner enables the display of previous self-assembly bouquets for each customer, including the types and the amount of flowers that were included in them. The shop maintains the following payment information (assume one payment for each date): credit card company, credit card number, validity, CVV, the amount paid, and the number of payments. The shop also maintains information about credit card companies.

Figure 1 displays a database schema for the online flower shop scenario. For a practical reason of limited space and since we mainly focus on defining relations and FK-PK relationships, not all attributes that are mentioned in the above textual description appear as relation fields in the figure.

2.2.1. The Online Flower Shop Schema

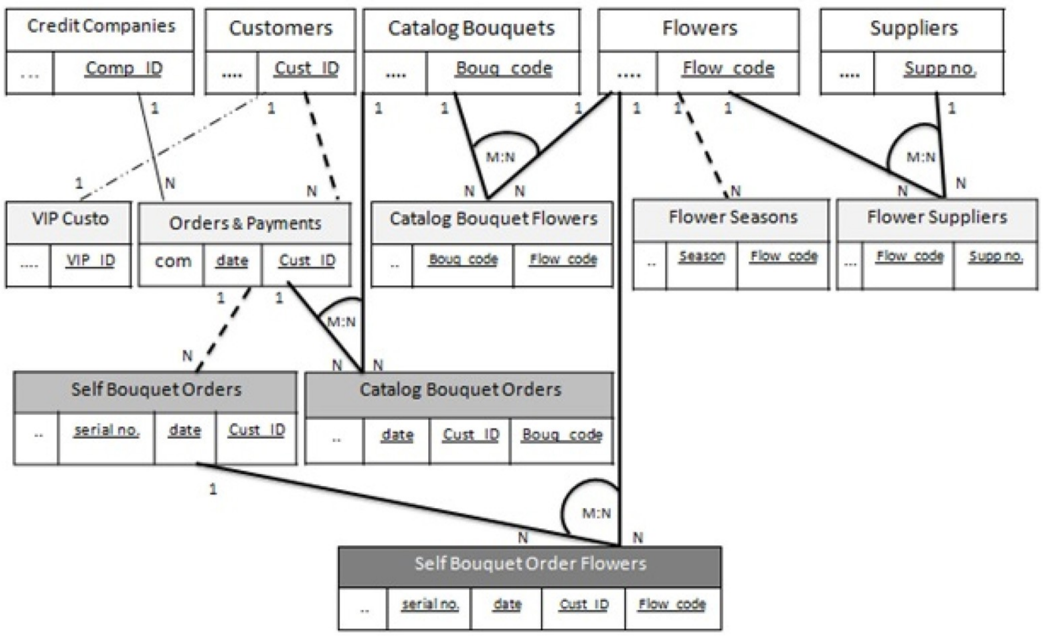


Figure 1. A database schema for an online flower shop scenario

In Figure 1, we see that parents are placed above their children. The relative location of relations is a choice made by educators at the syntax level: placing the parents above and closest to their direct children helps visualize the hierarchy and prevents the creation of incorrect FK-PK relationships. For clarity, direct parents and children should be placed at a minimum distance (one hierarchy level) from one another, but this is not always possible. Sometimes the parents of a joint child are not at the same hierarchical level, and then the distance between the "higher" parent and the child is the distance from the "higher" parent to the "lower" parent plus one more level.

Looking at the online flower shop schema from top to bottom, notice that the entities are gradually transitioning from being simple and tangible to being either more abstract, or more detailed and specific (concrete): "Flowers", "Suppliers" and "Customers" are simple and easy to comprehend. At the next hierarchical level we find relatively abstract entities that represent events, (such as "Orders and Payments", an event driven by a customer), or more specific types of entities (such as "VIP customer" that IS A customer type with specific attributes, such as a birth date), or more detailed entities that represent combinations of higher level entities (such as "Catalog Bouquet Flowers" and "Flower Seasons"). The recommendation is to start with tracking down simple entities that are easily identified in a scenario, and only then handle the more abstract entities that are related to the simple ones, to be represented (as relations) below them. In other words, it is easiest to create the hierarchy from top to bottom.

In the example, there are four M: N relationships marked by continuous thick lines, three 1: N relationships marked by dashed lines that actually are a part of a M: N relationship (one of the parents is not presented as a relation in the schema as we will demonstrate), one 1: N relationship marked by a continuous thin line, and one 1: 1 relationship marked by a dashed-dotted line.

M: N relationships: "Flowers" and "Suppliers" have an M: N relationship, but cannot be directly connected in a relational database, therefore they are joint parents of "Flower Suppliers" (cross-reference relation). There are 3 additional M: N relationships in the schema ("Catalog Bouquet Flowers", "Catalog Bouquet Orders", and "Self Bouquet Order Flowers"). The joint child's PK is an integration of his parents' PKs, demonstrating the second FK-PK pattern shown previously in the Syntax Section.

"Catalog Bouquet Orders" and "Self Bouquet Orders" are both children of the same parent, "Orders and Payments". Semantically, they show the specific details of the customers' orders in terms of what bouquets were purchased in an order. The next level, "Self Bouquet Order Flowers" is even more specific compared to its parent ("Self Bouquet Orders"), because it holds details regarding the combination of flowers in each self assembled bouquet.

Partial representations of M: N relationships: While "Catalog Bouquet Orders" is a joint child of "Orders and Payments" and "Catalog Bouquets", "Self Bouquet Orders" has only one parent, since the other "parent", serial number, is not defined as a relation according to the scenario (having no additional attributes), but appears as an attribute which is needed for identifying a self bouquet order. One can mistakenly perceive this as a regular 1: N relationship (that stands by itself), but instructors should emphasize that it is a partial presentation of the second FK-PK pattern described in the Syntax Section, an M: N relationship between customer orders and serial numbers. The same can be said regarding "Orders and Payments", and regarding "Flower Seasons" (respectively, the absent parent relations, are dates and seasons).

1: N relationship: "Credit Companies" and "Orders and Payments" have a 1: N relationship, demonstrating the first and simple FK-PK (parent-child) pattern mentioned in the Syntax Section.

1: 1 relationship: "Customers" and "VIP Custo" (short for VIP Customers) have a 1: 1 relationship, demonstrating the third FK-PK pattern (see the Syntax Section). Having additional unique attributes, a VIP customer IS-A specific sub-type of the general customer type.

In the Syntax Section, we pointed to another gradual transition that can be seen when moving top-down the hierarchy: the relations' PK include more fields, in other words, PKs gradually expand. Figure 2 shows two alternative solutions for only a part of the online flower shop scenario. Figure 2A on the left side presents a correct solution, derived from a syntactic emphasize on the gradual expansion of PKs. Figure 2B on the right side presents an erroneous solution, in which two important FK-PK relationships are omitted, while two invalid FK-PK relationships are added. The erroneous solution is based on prevalent errors that students had made in a modeling exercise in a previous study (Katz & Shmallo, 2015). The erroneous solution omits two requires (direct) parent-child FK-PK relationships, 1) between "Orders and Payments" and "Self Bouquet Orders", and 2) between "Self Bouquet Orders" and "Self Bouquet Order Flowers". The two invalid "grandparent-grandson" (indirect parent-child) relations added are 1) between "Orders and Payments" and "Self Bouquet Order Flowers", and 2) between "Customers" and "Self Bouquet Orders".

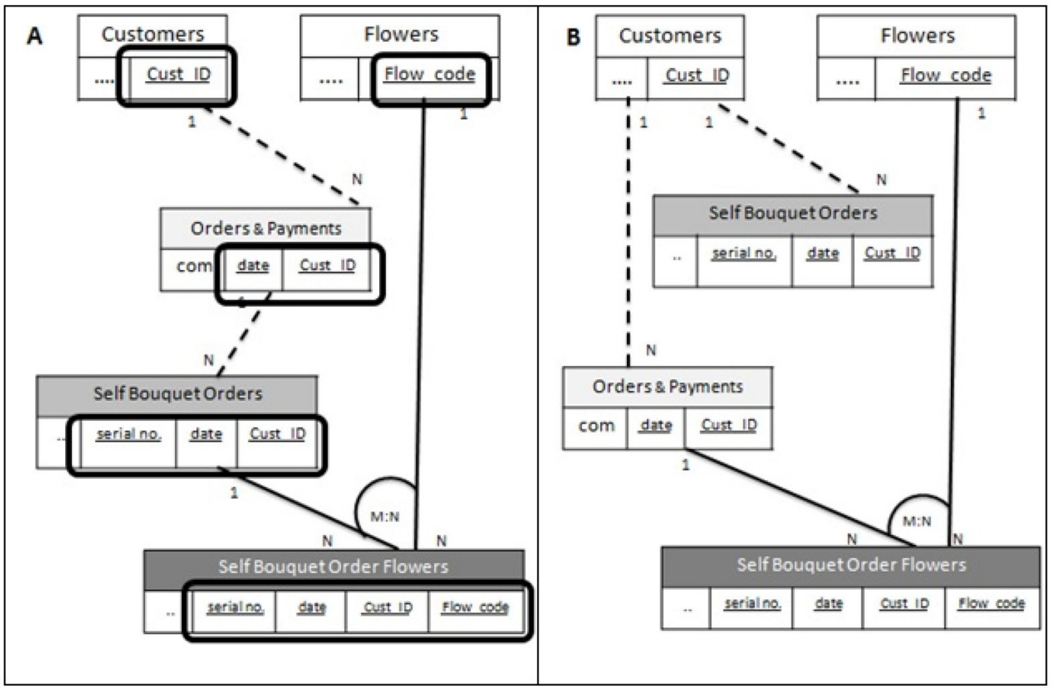
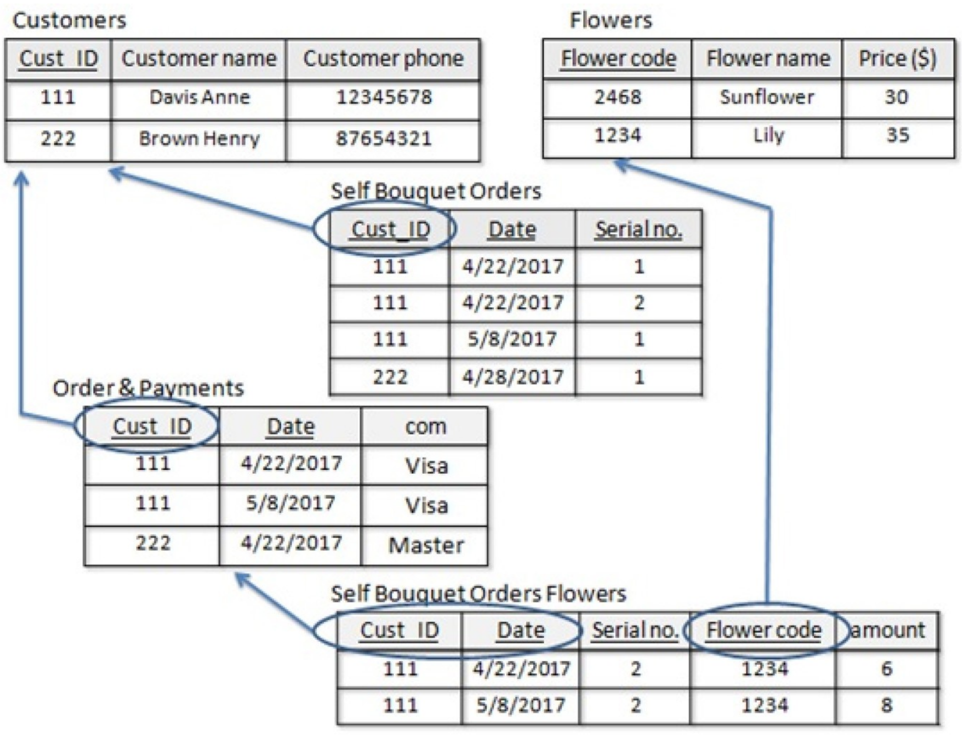


Figure 2. A partial view of the online flower shop schema: alternative solutions

As part of the approach of emphasizing the hierarchical syntax of gradual transitions in hierarchy levels, it is highly recommended to deliberately use erroneous examples in class to help students understand the importance of imposing the correct referential integrity constraints (Katz & Shmallo, 2016). Erroneous referential integrity constrains are liable to occur when the schema does not follow the hierarchical syntax of the gradual transitions.

An effective demonstration of the consequences of designing such a schema is to fill concrete records in the relations to show how the system will allow its users to insert records that violate the system's integrity. Figure 3 shows possible consequences of omitting direct parent-child, derived from the erroneous solution we had presented in Figure 2B, by demonstrating inconsistencies of reference among records, which violate data integrity. For convenience, the relations include only a partial view of attributes. The ellipses in the figure mark the defined FKs of each referencing relation. The erroneous omission of the FK-PK relationship between "Self Bouquet Orders" and "Orders and Payments" enables an abnormal data entry such as including a self bouquet order that could not occur at the inserted date (4/28/2017). The omission of the FK-PK relationship between "Self Bouquet Order Flowers" and "Self Bouquet Orders" enables an abnormal data entry such as a record appearing in "Self Bouquet Order Flowers", that shows a combination of an order with a non-existing serial number (customer 111 at 5/8/2017 did not order more than one self bouquet, so 2 does not make any sense). In other words, a record in "Self Bouquet Order Flowers" refers to a record that does not exist in "Self Bouquet Orders".



Customers

Cust_ID	Customer name	Customer phone
111	Davis Anne	12345678
222	Brown Henry	87654321

Flowers

Flower code	Flower name	Price ($)
2468	Sunflower	30
1234	Lily	35

Self Bouquet Orders

Cust_ID	Date	Serial no.
111	4/22/2017	1
111	4/22/2017	2
111	5/8/2017	1
222	4/28/2017	1

Order & Payments

Cust_ID	Date	com
111	4/22/2017	Visa
111	5/8/2017	Visa
222	4/22/2017	Master

Self Bouquet Orders Flowers

Cust_ID	Date	Serial no.	Flower code	amount
111	4/22/2017	2	1234	6
111	5/8/2017	2	1234	8

Figure 3. Inconsistent records that may result from an erroneous database schema (as in Figure.2B)

The consequences of adding invalid "grandfather-grandchild" relations along with proper "father-child" relations, is redundancy, since there is already an implicit FK relationship from a child to his grandfather through the father. At the semantic level, database designers understand that the consequence of redundancy is forcing unnecessarily checks of compliance with the defined referential integrity constraints on the system.

3. Approach Testing, Future Experimental Plan and Instructional Implications

This study adopts an "educational action research" methodology, in which the motivation for being involved in an educational action is the improvement of the teaching and learning quality. Educational action research aims at the development of an autonomous improvement ability for educators using systematic self observations, and testing pedagogic ideas using research procedure in class (Fessakis et al., 2005). Our pedagogic approach has already been implemented in the last two years (2016-2017) in an academic college of engineering in a "Databases" course, enrolled by Information Systems track 3rd year students learning in the department of Industrial Engineering and Management. The database curriculum includes a series of activities developed to learn and practice the topic of database modeling. The implementation of the schema modeling according to organizational textual scenarios lasts between two to three class meetings (of three academic hours each), and additional two to three practice meetings (of two academic hours each), followed by homework assignments. Most class and homework exercises are in the format of organizational textual descriptions like the online flower shop scenario described in Section 2.2.

Data collected from solutions for homework assignments and exam questions in the topic of schema modeling showed improvements in comparison to the solutions of the previous years. The students were more accurate in defining the PKs of relations, and made fewer errors in terms of needless additions (such as adding redundant relations, and redundant relationships between relations) and in terms of omitting required elements (such as missing relations, and failing to define crucial relationships between relations). In addition, the course's educators subjectively evaluated that the subject of database modeling is clearer when taught according to the new approach. They reported that the time devoted to the subject in class meetings was reduced, since students seemed to express less comprehension difficulties.

According to subjective evaluations, in the years before the approach was implemented in the course, students expressed more negative attitudes towards modeling database schemas. Therefore, it seems that the new approach may be efficient and useful for teaching database modeling, but a controlled experiment to investigate whether the approach improves database modeling significantly, was not yet conducted.

The following section describes an experimental plan to empirically test the approach and the next section refers to a computerized tool that is currently designed as an additional means to test the approach. If the experimental results indicate that the approach is useful, such a computerized supporting tool can be implemented along with the approach in database courses.

3.1. Experimental Plan

We intend to empirically test whether the approach is effective in educating modeling of relational databases. A controlled experiment will be conducted to compare a group of students who will learn schema modeling through a traditional database teaching approach (a control group) to a group of students who will be exposed to a learning process that emphasizes the syntax of gradual transactions between hierarchical levels of a schema (an experimental group). Students enrolled in a database course will be randomly divided into the two treatment groups. Each group will separately learn and practice identical textual scenarios that describe organization requirements, in class meetings with the same instructor. In a following class meeting, they will be given unseen textual scenarios (identical for both groups) and will be asked to create database schemas, i.e. identify relations, PKs, and draw FK-PK relationships between relations. Our experimental study is designed as a one factor with two treatments. This kind of assignment of participants to treatment groups is often used in experimental evaluation of modeling techniques (Dahan et al., 2014).

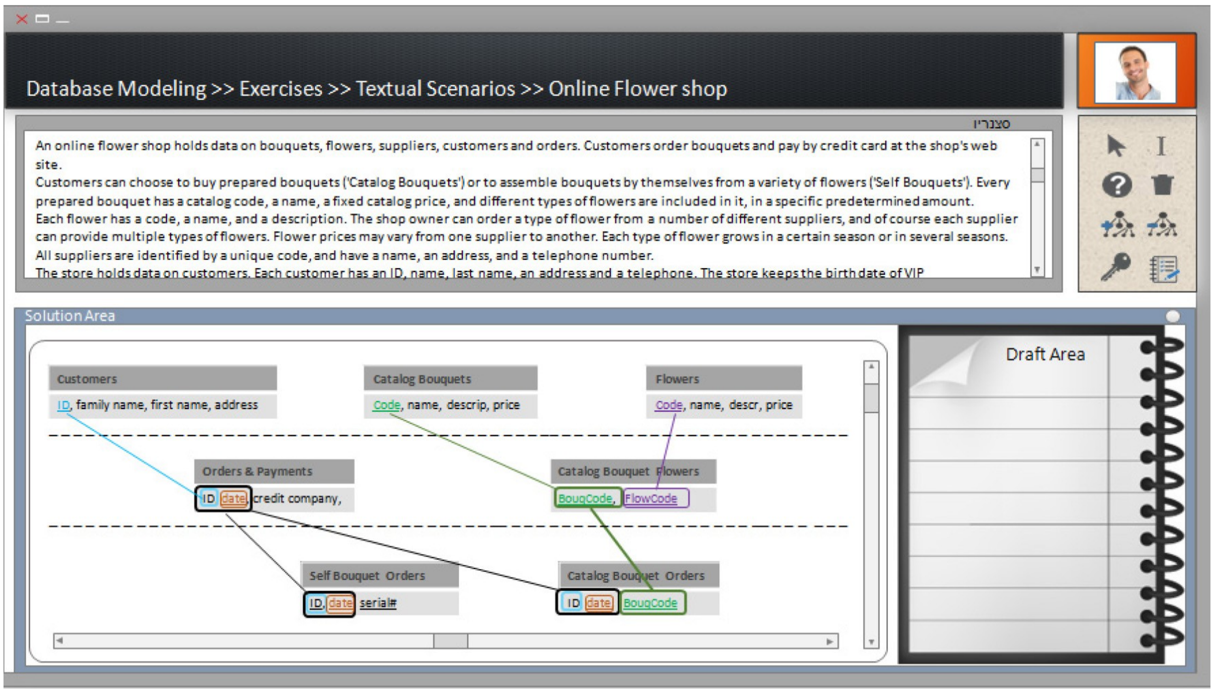
In order to compare between the traditional approach and the approach presented in this paper, several database designers and educators will analyze the quality of the students' created schemas. The quality of the solutions will be measured in terms of the number and types of errors found in the students' solutions. The error analysis will follow pre-defined categorization of errors based on the types found in a previous pedagogic study in the area of database modeling (Katz & Shmallo, 2015). Since evaluating the quality of the solutions can be subject to biases, we will use Cohen's Kappa test to ensure inter-rater reliability (agreement) among raters. We will also ask student participants in both groups to complete a questionnaire to examine their attitudes towards the pedagogic approaches, their satisfaction and evaluation of the level of their comprehension of database schema modeling. The results of the comparison will enable us to reach a conclusion regarding the effectiveness of the current approach in educating database schema modeling. The expectations regarding the planned experiment are to find higher-quality solutions, and more positive attitudes toward the learning of data models through the pedagogical approach of visually emphasizing the syntax of the hierarchical nature of schemas.

3.2. Computerized Supporting Tools for Database Modeling

In the process of evaluating the effectiveness of the offered approach, a computerized supporting tool is currently being designed to be implemented in database courses. The prototype serves to evaluate students' attitudes towards the pedagogic approach, using user testing techniques such as observations, think aloud, and interviews (Shneiderman, Plaisant, Cohen, Jacobs, Elmqvist & Diakopoulos, 2016). If the approach would be found useful, it could guide the development of computer-based learning environments for tutoring and exercising database schema modeling.

Previous works show interesting database modeling tutoring systems, many of them focus on normalization rules (Kung & Tung, 2006; Mitrovic, 2002; Taofiki & Tale, 2012), or on translating scenarios to ER (entity-relationship) diagrams (Murray & Guimaraes, 2008), but none refer to visually emphasizing the syntax of the hierarchical structure of a database scheme with gradual transitions between levels of the hierarchy as an effective way for guiding the database modeling process. Although the current paper's focus was not on discussing normalization, normal form (NF) rules are indeed a central topic in any database course. Database modeling is an activity characterized by a high level of element interactivity, and different topics should be delivered and understood with reference to other topics, and cannot be considered independently (Katz & Shmallo, 2016). Therefore, future designs of computerized supporting tools that would visually implement the emphasis on the hierarchical syntax should also integrate references to NF rules, to ensure normalized database schemas.

An initial mockup prototype for designing a database modeling tutoring system already exists, but a detailed description of its features is beyond the scope of this paper. The design process follows usability guidelines and strategies for effective user-computer interaction (Shneiderman et al., 2016). The general idea of a system that supports the emphasis of the hierarchical syntax contains an area on the screen that presents a textual scenario (uploaded from a pool of pre-written scenarios) and a working area that is divided horizontally into different levels (the user can at any time add or remove levels with dedicated hierarchical icons). The user will be able to select and then "drag and drop" words that represent entities from the textual scenario to the working area in order to create relations. When the user "drops" the dragged text in the area of a specific hierarchical level, a relation represented by a table will automatically appear there. Once a table is shown in the working area, the user is able to select and then "drag and drop" words that represent the table's attributes, from the textual scenario into that table. Throughout the solution process, visualization techniques such as colors and rectangles of different sizes will automatically mark the gradual transitions between hierarchical levels of the schema. For example, mutual PK fields of different relations will appear in identical colors to highlight the gradual expansion of PKs, and serve as a visual aid in the process of creating proper FK-PK connections. Figure 4 shows a print screen of the database modeling tutoring prototype at its current design stage.


Database Modeling >> Exercises >> Textual Scenarios >> Online Flower shop

סצנריו

An online flower shop holds data on bouquets, flowers, suppliers, customers and orders. Customers order bouquets and pay by credit card at the shop's web site.
Customers can choose to buy prepared bouquets ('Catalog Bouquets') or to assemble bouquets by themselves from a variety of flowers ('Self Bouquets'). Every prepared bouquet has a catalog code, a name, a fixed catalog price, and different types of flowers are included in it, in a specific predetermined amount.
Each flower has a code, a name, and a description. The shop owner can order a type of flower from a number of different suppliers, and of course each supplier can provide multiple types of flowers. Flower prices may vary from one supplier to another. Each type of flower grows in a certain season or in several seasons.
All suppliers are identified by a unique code, and have a name, an address, and a telephone number.
The store holds data on customers. Each customer has an ID, name, last name, an address and a telephone. The store keeps the birth date of VIP

Solution Area

Customers: ID, family name, first name, address
Catalog Bouquets: Code, name, descrip, price
Flowers: Code, name, descr, price
Orders & Payments: ID date, credit company,
Catalog Bouquet Flowers: BouqCode, FlowCode
Self Bouquet Orders: ID, date serial#
Catalog Bouquet Orders: ID date BouqCode

Figure 4. Database modeling tutoring prototype

The degree of system intervention and guidance will be controlled by the user. At one end of intervention range (maximum level), the user will define the system as a step-by-step guide at all stages of the task (a tutorial, leading the whole process). At the other end of intervention range (minimal level), the user will go through the whole modeling process alone, and only at the end the system will check whether the solution is correct, and mark errors if they were made. Between these two extremes, the user can request intervention throughout the process in the form of leading questions that are sensitive to the user's actions at any point. When a leading question appears, the user can either choose to proceed with modeling actions (since the question was helpful in directing to an action), or to display the answer for that question, and then proceed. Users will also be able to determine whether the system will divide the working area horizontally into different levels in advance, or that she or he will add and remove hierarchical levels, independently. The tutorials, the leading questions and the error corrections will all emphasize the gradual transitions and the hierarchic nature of the schema. A usability test with five students that were already enrolled in the database course last year was conducted. The student subjects were introduced with the initial mockup prototype in the context of solving the flower shop scenario demonstrated in Section 2.2. Preliminary results show satisfaction with such a computerized tool that supports the schema design by emphasizing the hierarchy. Especially, subjects were satisfied with the visualization aids, and noted that it was unfortunate that there was no such system in the course when they learned it.

The initial mockup prototype of a database modeling tutoring system will be developed iteratively to a satisfactory stage that implements the presented approach. When the prototype will reach a maturity state of an interactive prototype, the System Usability Scale (SUS) will be used as a tool to collect users' subjective ratings of the system's usability (Brooke, 1996).

4. Conclusions

The objective of this study was to propose a new pedagogic approach for educating database design to more effectively deliver and instruct the cognitively complex material of schema modeling. TSSL, a multilayer model originally developed in the area of HCI, as a conceptual framework for understanding user activity in the aim of guiding the design of user interfaces was applied in the area of relational database modeling. The four-level TSSL structure is used to divide the process of database modeling into four levels: task, semantic, syntax and lexicon. Examining database modeling through the TSSL lens can explain how a pedagogic approach of visually emphasizing the syntax of the hierarchical nature of schemas, may reduce the potential cognitive complexity of novice database designers. The theoretical framework was demonstrated on a particular example.

The pedagogic approach has already been implemented in the past two years in an academic college of engineering in a database course, enrolled by Information Systems students. Solutions for homework assignments and test questions in the topic of schema modeling showed improvements in comparison to the solutions of the previous years. With the new approach, students' solutions were more accurate and exhibited fewer errors. In addition, the course's educators noticed that the topic of database modeling seems clearer in the past two years, and in addition, delivering the topic in class required less time. Educators' perceptions derived from class observations and from personal interactions with the students are that students had fewer difficulties and fewer negative attitudes towards modeling database schemas in comparison to students who learned the topic before the pedagogic change was made. Since these are only subjective evaluations, it is not yet possible to draw conclusions regarding the effectiveness of the approach. Therefore, at the next phase, a controlled experiment to empirically investigate whether the approach is significantly efficient and useful will be conducted according to the proposed experimental plan.

According to the theoretical foundation of the educational action research methodology, specific solutions that are developed in the context of teaching, constitute suppositions for test by other educators (Fessakis et al., 2005). Following this line and the preliminary educators' evaluations derived from applying the approach in the last two years, a future research direction for educators who face the challenges and problems of relational database modeling, is to implement for test the pedagogic approach of visually emphasizing the syntax of the hierarchical nature of schemas.

An additional future research direction with a focus on visualization would be to compare several visual alternatives at the lexical and syntax levels, to find the most efficient and useful ways to emphasize the gradual changes between different hierarchy levels of a database schema.

Acknowledgments

We thank Sharon Malka and Hen Uzan for their contributions in designing the computerized supporting tool prototype.

Declaration of Conflicting Interests

The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author received no financial support for the research, authorship, and/or publication of this article.

References

Al-Dmour, A. (2010). A cognitive apprenticeship based approach to teaching relational database analysis and design. *Journal of Information & Computational Science*. 7, 2495-2502.

Anthony, Jr, R. (2008). Cognitive load theory and the role of learner experience: An abbreviated review for educational practitioners. *Aace Journal*, 16(4), 425-439.

Brooke, J. (1996). SUS- A quick and dirty usability scale. *Usability Evaluation in Industry*, 189(194), 4-7.

Dahan, M., Shoval, P., & Sturm, A. (2014). Comparing the impact of the OO-DFD and the Use Case methods for modeling functional requirements on comprehension and quality of models: a controlled experiment. *Requirements Engineering*, 19(1), 27-43. https://doi.org/10.1007/s00766-012-0155-2

Elmasri, R., & Navathe, S.B. (2011). *Fundamentals of database systems*. Pearson.

Fessakis, G., Dimitracopoulou, A., & Komis, V. (2005). Improving database design teaching in secondary education: action research implementation for documentation of didactic requirements and strategies. *Computers in Human Behavior*, 21(2), 159-194. https://doi.org/10.1016/j.chb.2004.06.006

Flood, R.L., & Carson, E.R. (1993?). *Dealing with complexity: an introduction to the theory and application of systems science*. Springer Science & Business Media.

Foley, J.D., Van Dam, A., Feiner, S.K., & Hughes, J.F. (1990). *Computer Graphics Principles and Practice*. Massachusetts: Addison-Wesley, Reading.

Fotache, M. (2006). *Why normalization failed to become the ultimate guide for database Designers?* Available at: http://ssrn.com/abstract=905060 or https://doi.org/10.2139/ssrn.905060

Gandhi, M., Robertson, E.L., & Van Gucht, D. (1994, December). Leveled entity relationship model. *In International Conference on Conceptual Modeling* (420-436). Berlin, Heidelberg: Springer. https://doi.org/10.1007/3-540-58786-1_94

Katz, A., & Shmallo, R. (2015). Improving relational data modeling through learning from errors. *The IADIS Multi Conference of Computer Science and Information Systems, Theory and Practice in Modern Computing track (IADIS TPMC 2015).* Las Palmas de Gran Canaria, Spain.

Katz, A., & Shmallo, R. (2016). Learning from errors as a pedagogic approach for reaching a higher conceptual level in database modeling, COGNISE workshop, as part of *CAiSE 2016, the 28th International Conference on Advanced Information Systems Engineering*. Ljubljana, Slovenia.

Kung, H.J., Kung, L., & Gardiner, A. (2013). Comparing top-down with bottom-up approaches: Teaching data modeling. *Information Systems Education Journal*, 11(1), 14.

Kung, H.J. & Tung, H.L. (2006, March). A web-based tool to enhance teaching/learning database normalization. *Proceedings of the 2006 Southern Association for Information Systems Conference* (251-258).

Mitrovic, A. (2002, December). NORMIT: A web-enabled tutor for database normalization. In Computers in Education, 2002. *Proceedings of the International Conference on* (1276-1280). IEEE.

Murray, M., & Guimaraes, M. (2008). Animated courseware support for teaching database design. *Journal of Computing Science*, 24(2), 144-150.

Najarian, S.E. (1981). Organizational factors in human memory: implications for library organization and access systems. *The Library Quarterly*, 51(3), 269-291. https://doi.org/10.1086/601111

Schwamborn, A., Thillmann, H., Opfermann, M., & Leutner, D. (2011). Cognitive load and instructionally supported learning with provided and learner-generated visualizations. *Computers in Human Behavior*, 27(1), 89-93. https://doi.org/10.1016/j.chb.2010.05.028

Shneiderman, B., & Mayer, R. (1979). Syntactic/semantic interactions in programmer behavior: A model and experimental results. *International Journal of Parallel Programming*, 8(3), 219-238. https://doi.org/10.1007/BF00977789

Shneiderman, B., Plaisant, C., Cohen, M.S., Jacobs, S., Elmqvist, N., & Diakopoulos, N. (2016). *Designing the user interface: strategies for effective human-computer interaction*. Pearson.

Simon, H.A. (1962). The architecture of complexity. *Proc. of the American Philosophical Society*, 106(6), 467-482.

Sweller, J. (1988). Cognitive load during problem solving: *Effects on learning*. *Cognitive Science*, 12(2), 257-285. https://doi.org/10.1207/s15516709cog1202_4

Taofiki, A.A., & Tale, A.O. (2012). A visualization tool for teaching and learning database decomposition system. *Journal of Information and Computing Science*, 7(1), 003-010.

Te'eni, D., Carey, J.M., & Zhang, P. (2005). *Human-computer interaction: Developing effective organizational information systems*. John Wiley & Sons.

Van Merrienboer, J.J., & Sweller, J. (2005). Cognitive load theory and complex learning: Recent developments and future directions. *Educational Psychology Review*, 17(2), 147-177. https://doi.org/10.1007/s10648-005-3951-0

Watson, R.T. (2006). The essential skills of data modeling. *Journal of Information Systems Education*, 17(1), 39-41.

Wickelgren, W.A. (1974). *How to solve problems: Elements of a theory of problems and problem solving*. San Francisco, Freemen.