Association for Information Systems
AIS Electronic Library (AISeL)

ICIS 1999 Proceedings

International Conference on Information Systems (ICIS)

December 1999

An Empirical Investigation of Entity-based and Object-oriented Data Modeling: A Development Life Cycle Approach

Atish Sinha
University of Dayton

Iris Vessey
Indiana University

AN EMPIRICAL INVESTIGATION OF ENTITY-BASED AND OBJECT-ORIENTED DATA MODELING: A DEVELOPMENT LIFE CYCLE APPROACH

Atish P. Sinha
School of Business Administration
University of Dayton
U.S.A.

Iris Vessey
School of Business
Indiana University
U.S.A.

Abstract

This paper examines end-user performance with conceptual and logical data models in the context of the database development life cycle. Both entity-based and object-oriented modeling methods were examined in a within-subjects study using 19 graduate students as subjects. The first method employed the extended entity-relationship (EER) model and relational data model (RDM), while the second method employed the object-oriented diagram (OOD) and object-oriented text (OOT) models. The models were assessed on the accuracy of modeling entities/classes and attributes, association relationships, and generalization relationships. Conceptual models (EER and OOD) were more effective than logical models (RDM and OOT) for representing all types of constructs. Further, the OOD model was superior to the EER model for representing entities/classes and attributes, while the OOT model was superior to the RDM for representing generalization relationships. Finally, mapping from conceptual to logical design proved to be more effective using the OOD-OOT method than the EER-RDM method.

Keywords: Data modeling, entity-relationship model, relational model, object-oriented DBMS, end-user computing, human factors, empirical research

1. INTRODUCTION

In the 1990s, the role of end-user computing became a well-established aspect of enterprise information systems. The gradual diffusion of all types of office automation tools—such as those for word processing, spreadsheets, presentation graphics, and databases—in the workplace has facilitated the decentralization of the IS function within organizations, opening up new possibilities for end-user systems development.

The majority of end users undergo training only in the use of software tools, however, resulting in their learning the syntax of the commands rather than the semantics associated with the concepts embedded in the tools (Hayen , Cook and Jecker 1990); i.e., little attention is paid to training end-users in how to use the software tools to address business problems. As Kettlehut (1991) states:

> Most professional MIS personnel would not build a system without some attention to formal analysis and design rules....End-users, on the other hand, may begin to develop spreadsheet or database applications without any formal analysis or design.

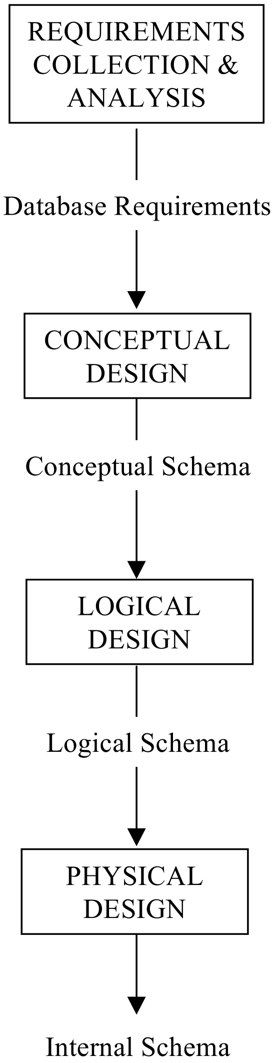

Figure 1. Database Development Life Cycle

With the proliferation of end-user constructed database applications, it is important that end-users pay attention to design principles, otherwise all types of problems will arise (Rob, Coronel and Adams 1991).

Just as the systems development life cycle starts with a set of functional requirements and goes through the analysis, design, and implementation phases, the database development life cycle starts with a set of data requirements and progressively evolves through the phases of conceptual design, logical design, physical design, and final implementation (see Figure 1, adapted from Navathe 1992).

In line with current development principles, therefore, this research examines end-user performance with conceptual and logical data models in the context of the database development life cycle. The paper examines both entity-based and object-oriented models. Although relational database management systems (DBMSs) dominate the market, object-oriented DBMSs are emerging as the most promising technology for the next generation of database systems. Object-oriented databases (OODBs) are becoming increasingly popular because of their support for representing complex data structures in applications such as CAD/CAM, multimedia, and the web (Watterson 1998), and it is likely that their use will spread to end users.

2. BACKGROUND

Prior research has examined the relative effectiveness of different data modeling formalisms for different types of database interactions (e.g., design, user validation, query writing). Most often, the relational data model (RDM) has been compared with a semantic data model such as the entity-relationship (ER) model. Most studies have found that users are more effective in all aspects of their interactions with databases when using the ER model compared with the RDM (Batra and Srinivasan 1992).

While a conceptual data model such as the ER model uses concepts that are close to the way users view data, a logical data model such as the RDM supports data descriptions that can be implemented directly on a computer system. Studies by Batra, Hoffer and Bostrom (1990), Jarvenpaa and Machesky (1989), Juhn and Naumann (1985), and Shoval and Even-Chaime (1987) specifically address data modeling, and all do so by comparing a conceptual data model to the RDM. The study by Shoval and Even-Chaime, which used the complex NIAM method (Nijssen's Information Analysis Method), is perhaps the only exception to studies that show the superiority of a conceptual model over the RDM.

Kim and March (1995) examined two conceptual data models, the *extended entity-relationship* (EER) model and the NIAM model. The researchers found that analysts using EER produced designs of higher semantic quality (overall, as well as on five different individual modeling constructs) than those using NIAM. The EER analysts also perceived their model to be less difficult to use and more valuable than did the NIAM analysts. There was no significant difference in syntactic performance between the two groups, however.

The majority of the researchers have compared the ER model with the RDM by treating them independently of one another; the study by Kim and March is an exception. In viewing user performance with the ER and RDM formalisms independently, rather than viewing them as successive techniques applicable to the first two phases of database design, these studies negate the well-established tradition of moving from analysis (requirements definition), through design, to implementation. We firmly believe that for users to develop effective databases, conceptual design should precede logical design, a point underscored by Navathe (1992):

> One of the shortcomings of the database design activity in organizations has been the lack of regard for the conceptual database design and a premature focus on some specific target DBMS. Designers are increasingly realizing the importance of the conceptual database design activity.

We examine end-user performance in developing conceptual schemas and, subsequently, the corresponding logical schemas.

3. REPRESENTATIONS

We use a university database case for the purpose of illustrating the modeling constructs under investigation: entities/classes and attributes, association relationships, and generalization relationships. The EER diagram for the university database is shown in Figure 2. The conceptual object-oriented diagram (OOD) schema is shown in Figure 4. The notation is adopted from the popular Coad and Yourdon notation, as presented in McFadden and Hoffer (1994).

Figure 3 presents the RDM schema. The RDM does not support generalization directly. To overcome that, one strategy is to create a relation for the superclass containing the attributes that are common to all the subclasses, and a separate relation for each subclass containing only the attributes that are unique to that subclass. Figure 5 presents the logical object-oriented text (OOT) schema for the university case. The notation is based on the object definition language (ODL), a standard prescribed by the Object Database Management Group (Cattell 1996). Association relationships are represented in both directions using the **relationship** and **inverse** keywords. Generalization relationships are captured directly in an OOT schema by specifying the superclass within the class definition.

4. THEORY AND HYPOTHESES

Data models vary in the extent to which the constructs they provide faithfully reflect the real world. Navathe (1992) suggests that a semantic model used for conceptual design should possess the properties of expressiveness, simplicity, minimality, and unique semantic interpretation. *Expressiveness* refers to the fact that the model should be expressive enough to distinguish between different types of data, relationships, and constraints. *Simplicity* implies that the model should be simple, so that the resulting schemas are easily understandable to both designers and users. *Minimality* means that every concept present in the model has a distinct meaning with respect to every other concept. And, for a given schema to have a *unique semantic interpretation,* each modeling construct must have complete and precise semantics.

4.1 Comparing Conceptual and Logical Data Models

Conceptual and logical models differ in their form of representation. While diagrammatic notations support conceptual models, textual notations support logical models. This observation suggests that the related literatures on pictorial and symbolic representation in cognitive psychology and on graphical and tabular representation in information systems are an appropriate basis for theoretical considerations in this area.

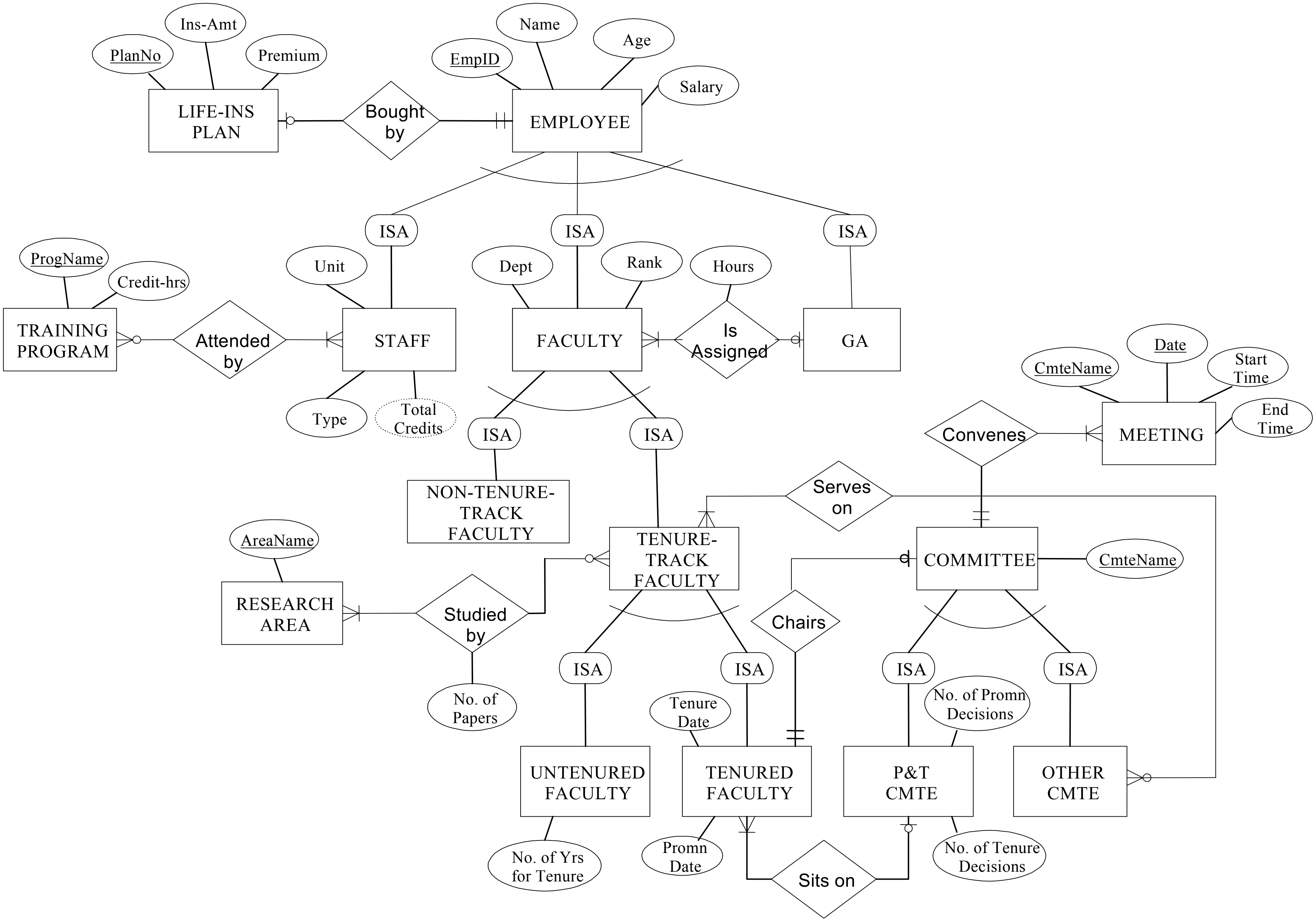

Figure 2. EER Schema for University Database

```
EMPLOYEE (EMP_ID, NAME, AGE, SALARY)

STAFF (EMP_ID, UNIT, TYPE, TOTAL_CREDITS)
  FK EMP_ID → EMPLOYEE

FACULTY (EMP_ID, DEPT, RANK, GRAD_ASST, GA_HRS)
  FK EMP_ID → EMPLOYEE
  FK GRAD_ASST → GA

GA (EMP_ID)
  FK EMP_ID → EMPLOYEE

NON_TEN_TR_FACULTY (EMP_ID)
  FK EMP_ID → FACULTY

TEN_TR_FACULTY (EMP_ID)
  FK EMP_ID → FACULTY

UNTENRD_FACULTY (EMP_ID, NO_YRS_FOR_TENRE)
  FK EMP_ID → TEN_TR_FACULTY

TENRD_FACULTY (EMP_ID, TENRE_DATE, PROMN_DATE, CMTE)
  FK EMP_ID → TEN_TR_FACULTY
  FK CMTE → P&T_CMTE

LIFE_INS_PLAN (PLAN_NO, INS_AMT, PREMIUM, SUBSCRIBER)
  FK SUBSCRIBER → EMPLOYEE

TRAINING_PROGRAM (PROG_NAME, CREDIT_HRS)

RESEARCH_AREA (AREA_NAME)

COMMITTEE (CMTE_NAME, CHAIR)
  FK CHAIR → TENRD_FACULTY

P&T_CMTE (CMTE_NAME, NO_TENRE_DECSNS, NO_PROMN_DECSNS)
  FK CMTE_NAME → COMMITTEE

OTHER_CMTE(CMTE_NAME)
  FK CMTE_NAME → COMMITTEE

MEETING (CMTE_NAME, DATE, START_TIME, END_TIME)
  FK CMTE_NAME → COMMITTEE

STAFFTRAIN (EMP_ID, PROG_NAME)
  FK EMP_ID → STAFF
  FK PROG_NAME → TRAINING_PROGRAM

FAC_RESEARCH (EMP_ID, AREA_NAME, NO_PAPERS)
  FK EMP_ID → TEN_TR_FACULTY
  FK AREA_NAME → RESEARCH_AREA

CMTESERVICE (EMP_ID, CMTE_NAME)
  FK EMP_ID → TEN_TR_FACULTY
  FK CMTE_NAME → OTHER_CMTE
```

Figure 3. RDM Schema for University Database

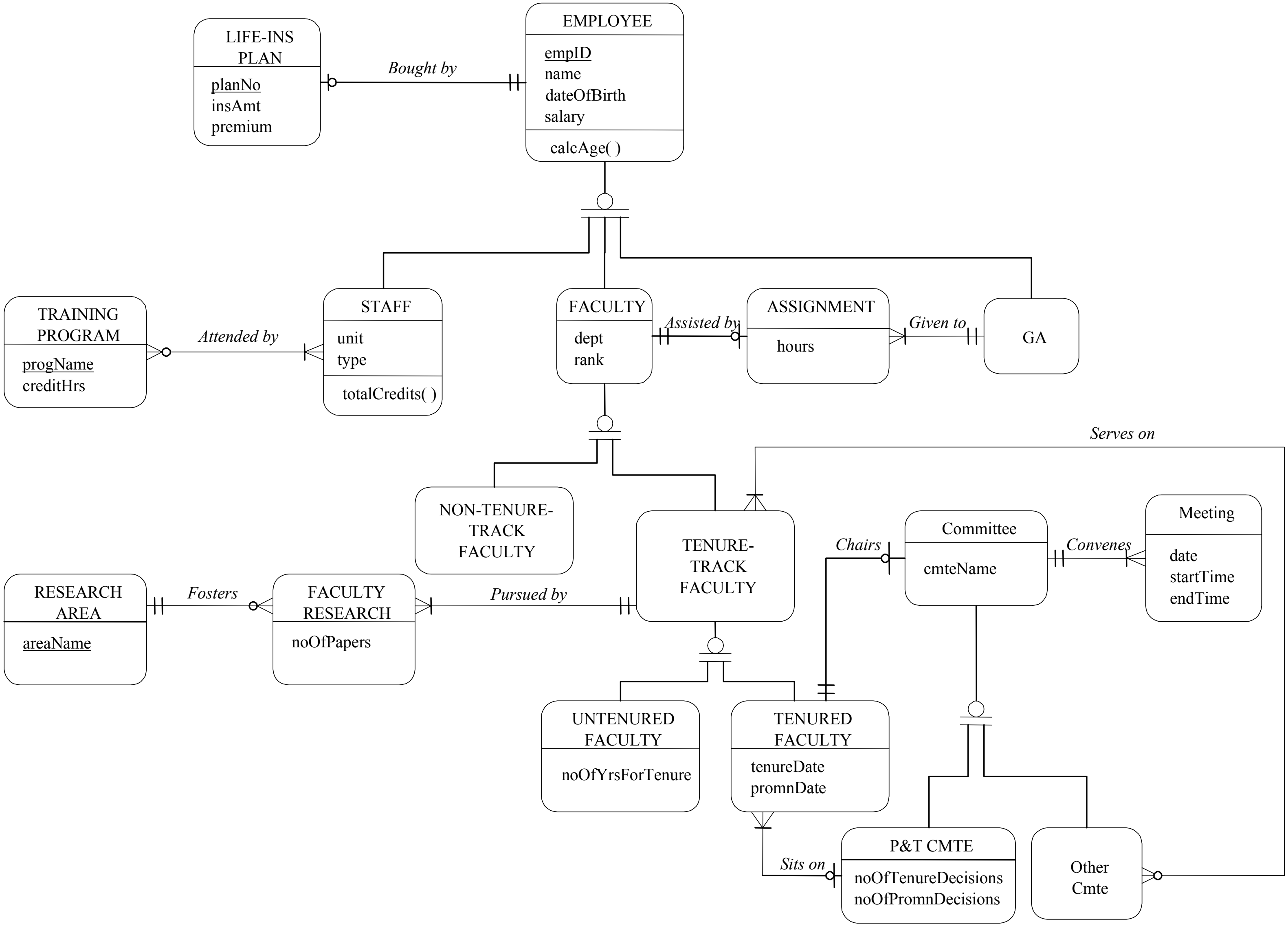

Figure 4. OOD Schema for University Database

```
interface Employee {
(   key empID)
   attribute string empID
   attribute string name
   attribute Date dateOfBirth
   attribute float salary
   relationship LifeInsPlan buys
     inverse LifeInsPlan::bought_by
   integer calcAge( ) }

interface LifeInsPlan {
(   key plan_no)
   attribute string plan_no
   attribute float ins_amt
   attribute float premium
   relationship Employee bought_by
     inverse Employee::buys }

interface Staff : Employee {
   attribute string unit
   attribute string type
   relationship set<TrainProgram> attends
     inverse TrainProgram::attended_by
   float totalCredits( ) }

interface TrainProgram {
(   key progName)
   attribute string progName
   attribute float creditHrs
   relationship set<Staff> attended_by
     inverse Staff::attends }

interface Faculty : Employee {
   attribute string dept
   attribute string rank
   relationship Assignment assisted_by
     inverse Assignment::for }

interface GA : Employee {
   relationship set<Assignment> works_in
     inverse Assignment::given_to }

interface Assignment {
   attribute integer hours
   relationship Faculty for
     inverse Faculty::assisted_by
   relationship GA given_to
     inverse GA::works_in }

interface TenTrFaculty : Faculty {
   relationship set<FacResearch> involved_in
     inverse FacResearch::pursued_by
   relationship set<OtherCmte> serves_on
     inverse OtherCmte::served_by }

interface NonTenTrFaculty : Faculty {  }

interface ResearchArea {
(   key areaName)
   attribute string areaName
   relationship set<FacResearch> fosters
     inverse FacResearch::conducted_in }

interface FacResearch {
   attribute integer noOfPapers
   relationship TenTrFaculty pursued_by
     inverse TenTrFaculty::involved_in
   relationship ResearchArea conducted_in
     inverse ResearchArea::fosters }

interface Committee {
   attribute string cmteName
   relationship TenrdFaculty chaired_by
     inverse TenrdFaculty::Chairs
   relationship set<Meeting> convenes
     inverse Meeting::convened_by }

interface Meeting {
   attribute Date date
   attribute Time startTtime
   attribute Time endTime
   relationship Committee convened_by
     inverse Committee::convenes }

interface UntenrdFaculty : TenTrFaculty {
   attribute integer noOfYrsForTenre }

interface TenrdFaculty : TenTrFaculty {
   attribute Date tenreDate
   attribute Date promnDate
   relationship Committee chairs
     inverse Committe::chaired_by
   relationship P&TCmte sits_on
     inverse P&TCmte::consists_of }

interface P&TCmte : Committee {
   attribute integer noOfTenreDecsns
   attribute integer noOfPromnDecsns
   relationship set<TenrdFaculty> consists_of
     inverse TenrdFaculty::sits_on }

interface OtherCmte : Committee {
   relationship set< TenTrFaculty> served_by
     inverse TenTrFaculty::serves_on }
```

Figure 5. OOT Schema for University Database

We base our analysis on the theory of cognitive fit (Vessey 1991), which states that most effective and efficient problem solving occurs when the process needed to complete the task is the same as (matches) that needed to interact with the problem representation and any methods, tools, or techniques used. Establishing cognitive fit requires analyzing both the task, to determine the processes needed to solve the problem, and the problem representation (in this case, the diagrammatic and textual schemas), to determine the process database designers use to access the information in the representation. Clearly, cognitive fit results when these two processes are similar, i.e., focus on the same type of information.

A diagrammatic schema is inherently pictorial in nature and therefore emphasizes spatial relationships in the data; perceptual processes, which show at a glance important relationships among data points, are used to access the data in a picture or graph. The diagrammatic schema, however, represents the details (attributes) in textual format, which is symbolic in nature. Analytical processes, which address individual data points, are used to access data in a textual representation. The textual schema emphasizes symbolic information alone, which, again, is accessed via analytical processes.

From the viewpoint of the tasks involved in system development, Vessey and Weber (1986) differentiate between design and coding. They argue that the design process is based on *taxonomizing,* and is, therefore, two-dimensional in nature. They further argue that the coding task is based on sequencing and is, therefore, one-dimensional in nature. The task of design is, therefore, best supported by a diagrammatic representation, which itself is two-dimensional, while the coding task, in which the programmer converts the application logic into code, is best supported by a textual representation, which is one-dimensional.

Both conceptual and logical database schemas address database design. A conceptual schema is represented by a diagram, which is two-dimensional in nature and which, therefore, supports the database design process, i.e., a fit exists between the cognitive process emphasized in the task and that emphasized in the representation. On the other hand, a logical schema is represented as text, which is unidimensional in nature. A fit does not exist, therefore, between the cognitive process emphasized in the task and that emphasized in the representation. Hence, a conceptual model better supports the design process than a logical model. We state the following hypotheses relating to user performance in modeling three constructs: entities/classes and their attributes; association relationships; and generalization relationships.

H1a: Using a conceptual data model will result in a more accurate representation of entities/classes and attributes in a database schema than a logical data model.

H1b: Using a conceptual data model will result in a more accurate representation of association relationships in a database schema than a logical data model.

H1c: Using a conceptual data model will result in a more accurate representation of generalization relationships in a database schema than a logical data model.

Next, we consider the EER and relational models. As we have seen, the diagrammatic, two-dimensional representation constructs that facilitate design, and that are supported by the EER model, are not available in the RDM. According to Navathe (1992), the ER model "is fairly simple to use, has only three basic constructs which are fairly, but not completely, orthogonal, has been formalized, and has a reasonably unique interpretation." The RDM, however, "clearly lacks the features for expressiveness and semantic richness for which the semantic models are preferred." Note that the EER model supports the concepts of classes and subclasses, and of inheritance hierarchies based on generalization by providing a special construct (an ISA link), while the RDM formalism does not. Also, the association relationship construct in the EER model does not have a direct RDM counterpart; associations are represented indirectly through foreign keys. Hence we state the following hypotheses:

H2a: Using the EER model will result in a more accurate representation of entities/classes and attributes in a database schema than the RDM.

H2b: Using the EER model will result in a more accurate representation of association relationships in a database schema than the RDM.

H2c: Using the EER model will result in a more accurate representation of generalization relationships in a database schema than the RDM.

We now consider the conceptual OOD and logical OOT models. The object-oriented model is appropriate for both conceptual and logical design (Navathe 1992). However, the requirement that the relationship construct be specified in both classes participating in a binary relationship, along with inverse references, tends to make logical modeling more difficult than its conceptual counterpart. We state the following exploratory hypotheses:

H3a: Using the OOD model will result in a more accurate representation of entities/classes and attributes in a database schema than the OOT model.

H3b: Using the OOD model will result in a more accurate representation of association relationships in a database schema than the OOT model.

H3c: Using the OOD model will result in a more accurate representation of generalization relationships in a database schema than the OOT model.

4.2 Comparing Entity-Based Approaches with Object-Oriented Approaches

The conceptual EER and OOD models are equally expressive in representing entities/classes and their attributes, association relationships, and generalization relationships. The two models also satisfy the property of unique interpretation. Further, they appear to be equally simple to use and we, therefore, do not expect that one would be better than the other for representing the constructs. We state the following hypotheses:

H4a: There will be no difference in the accuracy of representation of entities/classes and attributes in database schemas produced using the EER model and the OOD model.

H4b: There will be no difference in the accuracy of representation of association relationships in database schemas produced using the EER model and the OOD model.

H4c: There will be no difference in the accuracy of representation of generalization relationships in database schemas produced using the EER model and the OOD model.

Although the RDM and OOT models are equally expressive in terms of representing entities and attributes, a relation (table) in a relational schema does not have a unique interpretation because it could represent an entity or a (M:N) relationship. However, we believe this would result in more problems in user comprehension than user modeling. We, therefore, do not expect any difference in performance between the two models in representing entities and attributes.

In a logical OOT schema, an association relationship is specified explicitly using the **relationship** construct in the participating object classes. In an RDM schema, on the other hand, association relationships are represented implicitly using foreign keys. Further, in representing an M:N relationship, a third relation has to be introduced to decompose the relationship into two 1:N relationships. The OOT model, therefore, appears to be more expressive than the RDM for representing association relationships.

The RDM formalism does not directly support generalization. On the other hand, the OOT model allows explicit representation of generalization relationships in the schema through the specification of superclasses in the class definition. Therefore, the OOT model is much more expressive than the RDM with respect to generalization. Generalization can be captured indirectly in a relational schema by creating separate relations for a given superclass and its subclasses in which case the primary key in each subclass relation becomes a foreign key referencing the superclass relation. Foreign keys are usually employed to represent association relationships. Using the foreign key construct to represent generalization might confuse end users because it does not have a unique interpretation. Because of the problems with expressiveness and interpretation, we expect OOT users to perform better than RDM users for representing generalization relationships.

H5a: There will be no difference in the accuracy of representation of entities/classes and attributes in database schemas produced using the RDM and OOT models.

H5b: Using the OOT model will result in a more accurate representation of association relationships in a database schema than the RDM.

H5c: Using the OOT model will result in a more accurate representation of generalization relationships in a database schema than the RDM.

Finally, we consider the issue of transforming a conceptual schema into a logical schema using the entity-based and object-oriented approaches. The conceptual OOD and logical OOT models represent a natural progression of representations to be used as part of an OODB design process. A one-to-one mapping exists between the modeling constructs available in the two phases. In contrast, the EER-RDM mapping is not so direct, especially in translating association and generalization relationships. We, therefore, believe that the EER-RDM transformation will suffer much greater loss in modeling accuracy than the OOD-OOT transformation for association and generalization relationships. However, we expect that both the mapping methods will be equally effective for modeling entities and attributes.

H6a: There will be no difference in the effectiveness of mapping entities/classes and attributes from a conceptual database schema to a logical schema using the EER-RDM and OOD-OOT transformation methods.

H6b: Mapping association relationships from a conceptual database schema to a logical schema will be more effective using the OOD-OOT transformation method than the EER-RDM transformation method.

H6c: Mapping generalization relationships from a conceptual database schema to a logical schema will be more effective using the OOD-OOT transformation method than the EER-RDM transformation method.

5. METHODOLOGY

To test the hypotheses, we conducted an experiment in which the participants developed conceptual and logical database schemas using the EER and RDM, as well as the OOD and OOT models. The participants in this study were 19 MBA students enrolled in a database management course. As an incentive to perform well, the participants were awarded extra credit based on their performance.

The participants received instruction in each of the four data models as part of their coursework. Knowledge of the data models at the time of the study was assessed via a questionnaire. On a scale of 1 (“not very skilled”) to 7 (“very skilled”), the participants reported their level of skill in using the techniques of EER-RDM and OOD-OOT modeling as 4.25 and 4.00, respectively; the difference was not statistically significant (p = .427). They also reported their level of confidence in using the EER-RDM and OOD-OOT modeling techniques as 4.54 and 4.18, respectively; again, the difference was not statistically significant (p = .285). Therefore, as desired, equivalent training levels were achieved for both methods.

The experimental tasks involved developing conceptual and logical database schemas for a university database system, which was described as a case in a printed text format.[1] The same case was used in section 3 for the purpose of illustrating the modeling constructs. The participants were allowed to consult their database textbook and notes during the experiment.

[1] The university case is available from the first author upon request.

We used a repeated-measures design in which the participants developed schemas using each of the four data models: EER, RDM, OOD, and OOT. There were, therefore, four experimental treatments, one for each data model. The participants acted as their own controls. As Stevens (1986) notes, such designs are "much more powerful than completely randomized designs, where different subjects are randomly assigned to the different treatments." In a completely randomized design, with 15 subjects per treatment, we would have required 60 subjects, whereas in a repeated-measures design, we would only need 15. We used a repeated-measures design with 19 subjects per treatment.

The experiment was conducted over two sessions. Each participant developed schemas for the university database using the EER-RDM and OOD-OOT methods. Presentation of the methods was counterbalanced to control for potential learning effects. Those participants who used the EER-RDM models in the first session used the OOD-OOT models in the second session, and vice versa. To minimize any carryover effects, the two sessions were separated by an interval of three weeks.

The participants received in-class training on data modeling using all the models. They were also trained to map EER diagrams into RDM schemas, and OOD diagrams into OOT schemas. The total time devoted to training for each of the EER-RDM and the OOD-OOT modeling methods was approximately six hours.

Prior to the experiment proper, a pilot test was conducted with six MBA students taking another section of the same course to identify problems with the experimental tasks, the allotted time, and any other issues relating to the conduct of the experiment. Based on the feedback from the pilot study, changes were made in the wording of the tasks and the allotted time for each experimental session was increased from one and one half hours to two hours.

5.1 Experimental Variables

The two independent variables examined in this study were (1) the type of data model used and (2) the type of modeling construct. The dependent variable was the accuracy of the schema produced using a given model for a specific construct.

The schemas developed by the participants were evaluated using procedures employed by Batra, Hoffer and Bostrom (1990) and Kim and March (1995). Each of the modeling constructs under investigation—entities and attributes, association relationships, and generalization relationships—was evaluated with respect to the solution of an expert (regarded as the "correct" solution). One of the researchers, who has several years of experience in database design, developed the solutions.

Both syntactic performance and semantic performance play a role in performance (Kim and March 1995). We identified the semantic and syntactic mistakes in the subjects' solutions (see Figure 6). We classified the errors into major (M1) or minor (M2), depending on its severity. A major error was assigned a 0.5 penalty, while a minor error was assigned a 0.3 penalty. A construct was considered to be present as long as there was a semantically equivalent construct in the subject's solution. If a construct was missing altogether, a score of 0 was assigned. Accuracy (performance) was computed as the percentage correct on a given construct in a subject's solution using the following formula:

$$\text{Accuracy (\%)} = \frac{\underline{N - 0.5 * M1 - 0.3 * M2}}{N} * 100$$

where N is the number of instances of the construct in the expert solution.

6. RESULTS

Table 1 presents the descriptive statistics for user modeling performance with the conceptual and logical models. As expected, performance using a conceptual model in general exceeded that using a logical model. Paired *t*-tests were conducted on the performance data; each pair consisted of two accuracy scores for the same participant: average score using the two conceptual models and average score using the two logical models.

Entities/Classes and Attributes

Major Errors:

1. Missing primary key (EER, RDM)
2. Wrong primary key
3. Attributes/methods present in subclasses, other entities/classes, and other relationships
4. Entity/class not named
5. Attribute represented as a relationship with a superclass

Minor Errors:

1. Attribute not properly named
2. Duplicate names
3. Multivalued attribute (RDM)
4. One entity/class mistaken for another
5. Two foreign key attributes with the same name (RDM)

Association Relationships

Major Errors:

1. wrong cardinality
2. missing cardinality (EER, OOD, OOT)
3. missing foreign key reference (RDM)
4. relationship with wrong entity/class
5. wrong degree (ternary)
6. relationship with wrong entity/class
7. inverse relationship not specified (OOT)
8. attribute belonging to the other entity/class in the relationship present

Minor Errors:

1. wrong name
2. duplicate names
3. unnamed (OOD)
4. redundant relationship
5. associative entity attribute placed in base entity/relation (EER, RDM)
6. primary key of a participating entity/class present in relationship (EER, OOD, OOT)
7. relationship stores attribute of another entity (EER, RDM)
8. relationship represented as an entity (EER)
9. foreign key attribute better placed in the other participating relation (RDM)
10. foreign keys do not tally (RDM)
11. relationship specified in only one of the two classes (OOT)
12. inconsistent naming of inverse relationship (OOT)

Generalization Relationships

Major Errors:

1. inheritance not recognized
2. represented as an association relationship
3. represented as an aggregation relationship (OOD, OOT)
4. foreign key is not the primary key (RDM)
5. missing foreign key/superclass reference (RDM, OOT)
6. wrong foreign key/superclass reference (RDM, OOT)

Minor Errors:

1. subclass not named
2. superclass and subclass have the same name
3. cardinality error
4. wrong symbol

Figure 6. Error Categories

Table 1. Means (SDs) for the Conceptual versus Logical Schemas

	Conceptual			Logical		
Construct	**EER**	**OOD**	**Overall**	**RDM**	**OOT**	**Overall**
Entities and attributes	82.72 (13.08)	87.62 (6.81)	85.17 (9.21)	77.91 (13.96)	83.50 (11.90)	80.70 (11.28)
Association Relationships	68.75 (20.68)	59.98 (15.26)	64.36 (15.28)	48.25 (19.10)	54.61 (20.50)	51.43 (15.70)
1:1	63.16 (26.83)	56.58 (20.14)	59.87 (19.58)	53.95 (27.97)	63.82 (26.32)	58.88 (18.67)
1:N	72.04 (24.95)	58.22 (21.05)	65.13 (20.76)	41.45 (20.54)	49.34 (25.42)	45.39 (19.80)
M:N	71.05 (25.62)	65.13 (22.66)	68.09 (18.18)	49.34 (24.20)	50.66 (25.16)	50.00 (21.07)
Generalization Relationships	82.24 (24.96)	90.13 (15.91)	86.18 (17.74)	45.39 (37.61)	75.33 (22.84)	60.36 (24.51)

Table 2. Conceptual versus Logical Models

Construct	**Conceptual vs. Logical**		**Hypotheses Supported**
	t	p-value	
Entities and attributes	3.849	.001	H1a: Conceptual > Logical
Association Relationships	4.730	.000	H1b: Conceptual > Logical
1:1	.210	.836	
1:N	6.633	.000	
M:N	4.620	.000	
Generalization Relationships	5.861	.000	H1c: Conceptual > Logical

Table 2 presents the results of the paired *t*-tests. All three hypotheses (H1a, H1b, and H1c) relating to the differences between conceptual and logical models were supported. The conceptual models were superior to the logical model for modeling entities and attributes (*p* = .001), association relationships (*p* = .000) and generalization relationships (*p* = .000). When the three types of association relationships were considered individually, we found that there was no significant difference in accuracy between the conceptual and logical models for 1:1 relationships (*p* = .836), although the differences were significant for both 1:N relationships (*p* = .000) and M:N relationships (*p* = .000).

A repeated-measures ANOVA procedure was applied to test the second, third, fourth, and fifth sets of hypotheses. Recall that we compared the performance of the same participants under four different treatments: EER, RDM, OOD, and OOT. The within-subjects factor was the data model; there was no between-subjects factor. For each construct, we selected the "repeated" contrast type in SPSS to transform the dependent variables (scores using the four data models) into three difference variables on the adjacent repeated measures. Next, we conducted a multivariate analysis on those difference variables to test the null hypothesis that performance using the four data models is the same for a given construct.

The null hypothesis was rejected for the entities and attributes construct (*p* = .007), association relationship construct (*p* = .003), and the generalization construct (*p* = .000). We then conducted tests of within-subjects contrasts to find where the differences lay. Table 3 presents the results. Hypotheses H2a, H2b, and H2c were all supported. EER was superior to RDM for modeling entities and attributes (*p* = .028), association (*p* = .001), and generalization (*p* = .000). Although hypothesis H2b was supported individually for 1:N relationships (*p* = .000) and M:N relationships (*p* = .003), it was not supported for 1:1 relationships (*p* = .247).

Table 3. Pairwise Conceptual-Logical Model Comparisons

Construct	EER vs. RDM		OOD vs. OOT		Hypotheses Supported
	F	p-value	F	p-value	
Entities and attributes	5.752	.028	5.774	.027	H2a: EER > RDM H3a: OOD > OOT
Association Relationships	16.235	.001	4.472	.049	H2b: EER > RDM H3b: OOD > OOT
1:1	1.432	.247	2.821	.110	
1:N	37.623	.000	6.243	.022	
M:N	11.699	.003	10.184	.005	
Generalization Relationships	24.640	.000	8.492	.009	H2c: EER > RDM H3c: OOD > OOT

Table 4. Comparison of Logical Models Across the Methods

Construct	EER vs. OOD		RDM vs. OOT		Hypotheses Supported
	F	p-value	F	p-value	
Entities and attributes	4.785	.042	3.626	.073	H4a not supported H5a: RDM = OOT
Association Relationships	3.777	.068	1.316	.266	H4b: EER = OOD H5b not supported
1:1	1.145	.299	1.190	.290	
1:N	8.929	.008	2.085	.166	
M:N	.655	.429	.050	.826	
Generalization Relationships	2.402	.139	11.594	.003	H4c: EER = OOD H5c: OOT > RDM

Similar results were obtained for the object-oriented models. Hypotheses H3a, H3b, and H3c were all supported. OOD proved to be better than OOT for modeling entities and attributes (p = .027), association (p = .049), and generalization (p = .009). However, hypothesis H3b was supported for 1:N relationships (p = .022) and M:N relationships (p = .005), but not for 1:1 relationships (p = .110).

Table 4 presents the results of comparison of conceptual models as well as logical models across both approaches, EER-RDM and OOD-OOT. Hypotheses H4a, H4b, and H4c predict that there will be no difference between EER and OOD in terms of representing the three types of constructs accurately. Hypotheses H4b and H4c were supported (the null hypotheses were not rejected at the .05 significance level). However, H4a was not supported: performance was better for representing entities and attributes using the OOD model than the EER model (p = .042). Problems associated with specifying primary keys in the EER diagram contributed to the difference in performance. An object, by definition, has its own identity and, therefore, in an OOD schema, primary keys are not necessary for enforcing uniqueness (see Figure 4).

Hypothesis H5a, which postulates that there will be no difference between RDM and OOT for representing entities and attributes, was supported (see Table 4). Hypotheses H5b and H5c predict the superiority of OOT over RDM for representing association and generalization relationships, respectively. However, while hypothesis H5c was supported (p = .003), hypothesis H5b was not; both models were equally effective in representing association relationships.

Finally, we compared the effectiveness of mapping a conceptual schema to a logical schema with the two transformation methods using paired *t*-tests (H6a, H6b, and H6c). All of those hypotheses were supported (see Table 5). As predicted by hypothesis H6a, there was no difference in mapping entities and attributes (p = .817). However, as posited by hypotheses H6b and H6c, mapping from OOD to OOT was easier than from EER to RDM for both association relationships (p = .019) and generalization relation-

Table 5. Comparison of Mapping Methods

Construct	**EER-RDM Mapping vs. OOD-OOT Mapping**		**Hypotheses Supported**
	t	p-value	
Entities and attributes	.235	.817	H6a: EER-RDM = OOD-OOT
Association Relationships	2.566	.019	H6b: OOD-OOT > EER-RDM
1:1	1.999	.061	
1:N	3.450	.003	
M:N	.931	.364	
Generalization Relationships	2.403	.027	H6c: OOD-OOT > EER-RDM

ships (p = .027). Notice that although H6b was supported for 1:N relationships, it was not supported for 1:1 and M:N relationships.[2]

7. DISCUSSION

In this study, we conducted an empirical investigation of end-user data modeling performance using the EER-RDM and OOD-OOT methods. In contrast to prior research, our research assessed the effectiveness of the data models by considering them in their natural sequence within the context of the database development life cycle.

Our analysis of the effectiveness of the data modeling formalisms was based on the theory of cognitive fit and construct adequacy. We hypothesized that, because of the two-dimensional nature of the design process, using conceptual, diagrammatic schemas would lead to more accurate data models than their logical, textual counterparts. The results indicate that conceptual models are indeed more effective than logical models for representing all types of constructs using both the entity-based and object-oriented approaches; the only exception was in modeling 1:1 relationships, which were equivalent in both models. Future research could investigate the factors that lead to a loss in accuracy during the conceptual-to-logical transformation, as well as seek to understand the types of problems users experience during the mapping process.

We then applied the notions of construct adequacy to compare the individual models. We found that, in general, when a data model does not satisfy one or more of those properties, performance with the model deteriorates. For instance, the lack of expressiveness and simplicity of the relational model compared to the EER model resulted in designs of inferior quality. We also found that the OOD model was better than the OOT model for all the three constructs.

Our finding that the EER model is superior to the RDM formalism is consistent with the findings of prior studies, with the important difference that we studied modeling effectiveness within the context of the database development life cycle, while others did not. The limitation of the study hinges upon the use of only one data modeling problem. Future studies could examine the effectiveness of the models for different types of problems.

[2] Because the dependent variable did not satisfy normality in some instances, we conducted the non-parametric Friedman and Wilcoxon tests, which do not make any assumptions about the shape of the underlying distributions. The results of these tests were the same as those obtained using the parametric tests. We, therefore, report only the results of the parametric tests.

8. REFERENCES

Batra, D.; Hoffer, J. A.; and Bostrom, R. P. "Comparing Representations with Relational and EER Models," *Communications of the ACM* (33:2), 1990, pp. 126-139.

Batra, D., and Srinivasan, A. "A Review and Analysis of the Usability of Data Management Environments," *International Journal of Man-Machine Studies* (36), 1992, pp. 395-417.

Cattell, R. G. G. *The Object Database Standard: ODMG – 93*, San Francisco: Morgan Kaufmann, 1996.

Hayen, R. L.; Cook, W. F.; and Jecker, G. H. "End User Training In Office Automation: Matching Expectations," *Journal of Systems Management*, March 1990, pp. 7-12.

Jarvenpaa, S. L., and Machesky, J. J. "Data Analysis and Learning: An Experimental Study of Data Modeling Tools," *International Journal of Man-Machine Studies* (31), 1989, pp. 367-391.

Juhn, S. H., and Naumann, J. D. "The Effectiveness of Data Representation Characteristics on User Validation," *Proceedings of the Sixth International Conference on Information Systems*, L. Gallegos, R. Welke and J. Wetherbe (eds.), Indianapolis, Indiana, 1985, pp. 212-226.

Kettlehut, M. C. "Don't Let Users Develop Applications Without Systems Analysis," *Journal of Systems Management*, July 1991, pp. 23-26.

Kim, Y., and March, S. "Comparing Data Modeling Formalisms," *Communications of the ACM* (38:6), 1995, pp. 103-115.

McFadden, F. R., and Hoffer, J. A. *Modern Database Management*, 4th ed., Redwood City, CA: Benjamin/Cummings, 1994.

Navathe, S. B. "Evolution of Data Modeling for Databases," *Communications of the ACM* (35:9), 1992, pp. 112-123.

Rob, P.; Coronel, C.; and Adams, C. N. "Relational Database Design at a Construction Company: A Problem or a Solution?" *Journal of Systems Management*, August 1991, pp. 23-27 and 36.

Shoval, P., and Even-Chaime, M. "Database Schema Design: An Experimental Comparison between Normalization and Information Analysis," *Database* (18:3), 1987, pp. 30-39.

Stevens, J. *Applied Multivariate Statistics for the Social Sciences*, Hillsdale, NJ: Lawrence Erlbaum, 1986.

Vessey, I. "Cognitive Fit: A Theory-Based Analysis of the Graphs Versus Tables Literature," *Decision Sciences* (22:2), 1991, pp. 219-240.

Vessey, I., and Weber, R. "Structured Tools and Conditional Logic: An Empirical Investigation," *Communications of the ACM* (29:1), 1986, pp. 48-57.

Watterson, K. "When it Comes to Choosing a Database, the Object is Value," *Datamation*, December/January 1998, pp. 100-107.