TOWARDS IMPROVING CONCEPTUAL MODELING: AN EXAMINATION OF COMMON ERRORS AND THEIR UNDERLYING REASONS

by

Sabah Currim

A Dissertation Submitted to the Faculty of the

COMMITTEE ON BUSINESS ADMINISTRATION

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY
WITH A MAJOR IN MANAGEMENT

In the Graduate College

THE UNIVERSITY OF ARIZONA

2008

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation

prepared by Sabah Currim

entitled Towards Improving Conceptual Modeling: An Examination Of Common Errors And Their Underlying Reasons

and recommend that it be accepted as fulfilling the dissertation requirement for the

Degree of Doctor of Philosophy

_____ Date: 12/17/2007
Dr. Sudha Ram

_____ Date: 12/17/2007
Dr. Alexandra Durcikova

_____ Date: 12/17/2007
Dr. Sue Brown

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

_____ Date: 12/17/2007
Dissertation Director: Dr. Sudha Ram

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at the University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

SIGNED: Sabah Currim

ACKNOWLEDGEMENTS

I am grateful to my advisor and mentor, Dr. Sudha Ram, for her insight, support and encouragement throughout the dissertation process. Without her patient guidance this dissertation would not have been possible. I would also like to thank her for valuable advice on issues related to academia that go well beyond the scope of the dissertation.

Many thanks are also due to Dr. Alexandra Durcikova who helped me with behavioral methodology and provided valuable guidance through the dissertation process.

Many thanks are also due to Dr. Faiz Currim, for his encouragement, support and help throughout the dissertation process.

I would also like to thank the other faculty members who served on my committee and provided valuable assistance and advice: Thank you Dr. Mohan Tanniru, Dr. Sue Brown, Dr. Joel Levin, and Dr. Richard Snodgrass.

My thanks go to many other faculty members and colleagues who provided valuable assistance throughout the doctoral process: Dr. Kurt Fenstermacher, Dr. Vijay Khatri, Dr. David Pingry, Dr. Pamela Slaten, Dr. Sherry Thatcher, Dr. Ramesh Venkataraman, Dr. Suzie Weisband, Dr. Limin Zhang, Dr. Huimin Zhao, and Dr. Leon Zhao.

If I have missed anyone else, it is my mistake, not due to your lack of assistance. Thank you as well.

Last, but not the least, I would like to thank the One to whom all thanks is due.

TABLE OF CONTENTS

TABLE OF CONTENTS - Continued

TABLE OF CONTENTS - Continued

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF TABLES - Continued

LIST OF TABLES - Continued

LIST OF TABLES - Continued

ABSTRACT

Databases are a critical part of Information Technology. Following a rigorous methodology in the database lifecycle ensures the development of an effective and efficient database. Conceptual data modeling is a critical stage in the database lifecycle. However, modeling is hard and error prone. An error could be caused by multiple reasons. Finding the reasons behind errors helps explain why the error was made and thus facilitates corrective action to prevent recurrence of that type of error in the future. We examine what errors are made during conceptual data modeling and why. In particular, this research looks at expertise-related reasons behind errors. We use a theoretical approach, grounded in work from educational psychology, followed up by a survey study to validate the model. Our research approach includes the following steps: (1) measure expertise level, (2) classify kinds of errors made, (3) evaluate significance of errors, (4) predict types of errors that will be made based on expertise level, and (5) evaluate significance of each expertise level. Hypotheses testing revealed what aspects of expertise influence different types of errors. Once we better understand why expertise related errors are made, future research can design tailored training to eliminate the errors.

CHAPTER 1: INTRODUCTION

1.1. Overview

This dissertation examines what errors are made during conceptual data modeling and why. In particular, our[1] research looks at expertise-related reasons behind errors. An error could be caused by multiple reasons. Finding the reasons behind errors helps explain why the error was made and thus facilitates corrective action to prevent recurrence of that type of error in the future. We use a theoretical approach, grounded in work from educational psychology, followed up by a survey study to validate the model.

1.2. Background

Databases are a critical part of Information Technology—from desktop applications, to web-based applications, to huge Enterprise Resource Planning (ERP) systems (Date 2000, Elmasri and Navathe 2003, Hoffer, et al. 2006). Following a rigorous methodology in the database lifecycle ensures the development of an effective and efficient database. The database development lifecycle consists of planning, design, implementation, evaluation and maintenance. At the planning stage, the problem and scope of the system (to address the business challenges) are defined. At the design stage, the problem is detailed into user requirements, and then modeled using comprehensive design documents. At the implementation stage, the design is converted to a working database system and integrated within the current production environment. During evaluation, the

[1] The first person plural is used through the dissertation due to the awkwardness of repeatedly using the first person singular in reference to the research

quality assurance staff verify that the database system conforms to the users' functional requirements, and finally over the rest of the system's life, routine tasks are performed to ensure the maintenance and continued availability of the database system. With changing needs in a dynamic business environment, the process is usually iterative.

Good design is an integral pre-requisite for a successful database. Database design involves the following key processes: requirements analysis, conceptual data modeling[2], logical design, and evaluation. At the analysis stage, the designer identifies key stakeholders and gathers requirements for the system. During the conceptual data modeling stage, the designer converts the requirements into a semantic database schema. This schema captures the important data characteristics from the application domain, independent of the implementation platform. The primary decisions made in the semantic data modeling stage relate to what data elements need to be captured, how they are related, and what business rules govern them (Topi and Ramesh 2004). During logical design, the analyst selects the database model (examples include relational, object-oriented, hierarchical, and network) and converts the semantic database schema to a relevant storage structure. For example, for the relational database model, the storage structure would be tables. At the evaluation stage, the analyst verifies the design, and assesses the degree to which it matches user requirements. Additional steps that are usually performed include ensuring that the system satisfies business rules, optimizes storage space and maximizes efficiency.

[2] From here on, we favor the use of *semantic data modeling* instead of *conceptual data modeling* to minimize overloading the term "conceptual"; the term is also used in the underlying theory framework adopted by this dissertation.

1.3. Motivation

The objective of our research is to understand the *reasons* behind errors in semantic data modeling. Naturally, we feel it is an important undertaking. In this section, we present the motivation for our research.

1.3.1. Importance of Semantic Data Modeling

Semantic data modeling is a critical step in the database development cycle (Batini, et al. 1992, Elmasri and Navathe 2003, Ramesh and Browne 1999) since it captures the characteristics of the data needed to solve a problem. Each stage in the database lifecycle depends on the previous one and without sound conceptual design, there is a strong likelihood of poor logical design and implementation (not to mention a maintenance nightmare). With the increasing use of CASE tools and the movement towards Model Driven Architecture (MDA), the importance of a good conceptual design cannot be over-emphasized. Since it is essential to get the semantic database schema correct, we chose semantic data modeling as the domain of focus for understanding and correcting analyst errors.

1.3.2. Eliminating Errors in Semantic Data Modeling

It is estimated that 45 to 65 percent of all errors committed are made during the design of a system (Rush 1985). The cost of fixing an error in software development is directly proportional to the time the error remains in the process (Browne and Ramesh 2002, McConnell 1996). The estimated cost savings are approximately 10 to 200 times greater when an error is fixed near the creation point, rather than at the implementation stage (Boehm and Papaccio 1988, McConnell 1996, Penrose and Seiford 1988).

Therefore, we feel it is crucial to fix errors at the semantic data modeling stage rather than later on in the database development process.

Fixing an error is however only one piece of the puzzle. To avoid the recurrence of the same kinds of errors, factors that contribute to the error formation need to be identified (and corrected). There are three major factors influencing errors in constructing semantic database schemas; i.e., task, modeling technique, and modeler constraints. Examples of task constraints include complexity (of the task), and incomplete requirements. Examples of modeling technique constraints include inability of data models to capture process information and set-based business rules (Ram and Khatri 2005). Examples of modeler constraints include both modeling expertise (our focus) and domain expertise. Previous work has looked at some of these dimensions such as task and modeling language. However, there has been scant attention paid to the effect of expertise in modeling on errors. Since lack of sufficient modeling expertise is the cause of a large number of errors, and it is not always possible to have the most-experienced veterans develop a schema, we feel there is much to be gained from understanding the mistakes made by new analysts, so that the most costly and pernicious classes of errors they make can be identified and corrected. This is why our research examines the impact of modeling expertise on errors.

Knowing the cause of an error made helps us in determining the direction to go in fixing it. The course of action to successfully eliminate an error would logically depend on the source of error. For example, if the error is caused by limitations of the modeling technique, the technique needs to be expanded or a new technique needs to be selected.

However, if the cause of the error is modeling expertise, it is important to measure what aspect of expertise is lacking so that appropriate training can be designed. To summarize, it is imperative to identify why the error is being made *before* attempting to fix it.

1.4. Research Objective

Our research goal is to understand the *reasons* behind errors in semantic data modeling. Within *reasons*, we analyze the impact of expertise-related reasons. Thus, the objective of this research is to study *why novices make expertise-related errors in semantic data modeling*. The results of this research will influence the design of training for novices. We feel it is important to understand the reasons behind an error (symptom), to explain the cause of the error and thus facilitate action to prevent recurrence of that type of error in the future, and hence reduce the total number of errors. In other words, concentrating on fixing an error, without determining the cause, will not result in reducing the total number of errors (Murphy and Cleveland 1995).

To address our objective, we propose the following approach: (1) measure expertise level, (2) classify kinds of errors made, (3) evaluate significance of errors, (4) predict types of errors that will be made based on expertise level, and (5) evaluate significance of each expertise level. The first step, *measure expertise level*, i.e., estimates a novice's familiarity with the different aspects of semantic database design. The second step, *classify errors,* organizes the types of errors made by a novice in creating a conceptual database schema. The third step, *evaluate significance of errors,* estimates the impact of errors made in semantic design on the database development lifecycle. The fourth step,

predict errors, gauges the types of errors that will be made by a novice, if her expertise level is known. Finally, the fifth step, *evaluate significance of each expertise level*, estimates the cost of not achieving a particular expertise level, based on the errors eliminated by achieving the level. In summary, to achieve the research objective, we need to answer the following questions:

RQ1. How do we measure expertise?

RQ2. What types of errors do novices make?

RQ3. What is the significance of each type of error?

RQ4. How does knowing designer expertise help predict the types of errors they will make?

RQ5. How do we evaluate the significance of each expertise level?

1.4.1. Motivation for the Research Objective

1.4.1.1. Importance of Expertise

This research uses novices as the target sample. We define novices as graduates in Information Systems, who are not experts, i.e., they have limited significant real world experience designing databases and in particular semantic data modeling. Novices are often required to design databases on their own because of the scarcity of database design experts, higher upfront costs by using experts, and easier availability of technology (Nelson 1991). Another motivating factor is a desire to avoid dependence on specialists (Zwass 1992). However, errors in semantic data modeling will result in a bad design and this translates to high costs to fix and maintain the system later. Therefore, it is important that graduates know semantic data modeling well and hence make minimal errors.

Semantic data modeling is a complex topic, which is typically covered in the MIS curriculum. However, there is a limited time frame provided for a novice to master the concepts. The model curriculum for Information Systems, at the university level, recommends two courses, one on systems analysis and the other on design and databases (Gorgone, et al. 2002). The systems analysis class covers the information system and database lifecycle, and spends time on gathering requirements and provides an overview of the different types of data and process models. These days many curricula include training on UML and class diagrams (which can be used as a semantic database design tool). Semantic data modeling is covered in more depth in the design and database class. However, even within that class, the curriculum coverage for semantic data modeling typically results in about four lectures over two weeks. Sadly, this time frame is not sufficient, because many novices have difficulty in grasping various concepts in semantic data modeling, as evidenced by the wide variation, across novices, in the types of errors made (Batra and Antony 1994). Most database faculty know from classroom and anecdotal evidence that students new to database design find conceptual data modeling hard. While a handful of students are able to grasp the concepts with reasonable ease, many are left with recurrence of errors that takes much practical experience to remedy. Better understanding the wide variation in the types of errors is a motivating factor in our measuring expertise and its effects.

By knowing their current level of modeling expertise, tailored training can be designed to improve modeling skill. Facilitating novices' transition to a higher level of performance is likely to result in significant cost saving from the reduction in errors and

decreased supervisory effort and quality assurance staff time and energy spent in its correction. From a practical standpoint, our case for studying the errors novices make (rather than those made by experts) is partly stimulated by the fact that novices are better able to verbalize the rules they use in developing a design. Experts have internalized the rules and have difficulty articulating the rules they use (Dreyfus and Dreyfus 1986). Therefore, studying novices allows for analyzing the reasons why they construct the schema in a particular way.

To summarize, this research focuses on the impact of modeling expertise on errors for many reasons. Firstly, the independent variable *modeling expertise* has not has not sufficiently been examined in existing literature, unlike variables like modeling technique (Gemino and Wand 2004, Topi and Ramesh 2004). Secondly, modeling expertise is a factor that can be influenced; i.e., training can be provided to improve expertise. Thirdly, by verifying the relationship between modeling expertise and errors, future research can take steps to reduce errors made. Fourthly, it serves us that modeling expertise can be studied in a controlled environment (unlike task constraints which are hard to isolate).

1.5. Research Methodology

The Entity Relationship (ER) model and its variants, tools for semantic data modeling, have been around for decades, and are widely used both in academia and industry. It made sense therefore, for us to consider the ER model while studying the research question. We would like to emphasize though, that the general approach we take is not limited to the ER model, and can be used in conjunction with any semantic model.

However, to conduct tests that were consistent across subjects, a specific choice had to be made—and the ER grammar was picked. For the purposes of our research, we assume that the database requirements have previously been gathered and collated in a textual format. This work then examines the process of converting a textual description into an ER schema.

Before we go further, we would like to briefly describe the methodology for each research question. To address the first question, "*How do we measure expertise?",* we apply a knowledge framework to ER modeling constructs. To address the second question, "*What types of errors do novices make*", we classify the types of errors, based on consultation with experts. To address the third question, "*What is the significance of each error?*", we evaluate, for each error, the implications of the differences between the logical design artifacts corresponding to (a) a valid and complete ER schema and (b) an ER schema containing the error. To address the fourth question, "*How does knowing designer expertise help predict the types of errors they will make?*", we develop and test hypotheses that predict types of errors, using on expertise levels. To address the fifth question, "*How do we evaluate the significance of each expertise level?*", we evaluate the relative importance of the different expertise levels based on the cost of errors it significantly predicts.

1.6. Scope of This Research

The Planning Evaluation Lifecycle (Figure 1) shows the overall process of how the planning and evaluation phases fit together. Our research is part of the evaluation phase. The goal of evaluation stage is to provide feedback about the object of research to a

variety of audiences by using a systematic approach to acquiring and assessing information (Trochim 1999). In contrast, the planning stage builds a system to fix problems identified by the evaluation phase. The planning phase is beyond the scope of this research and is part of future work. The output of our research can be fed into the planning phase to design artifacts that improve expertise and reduce errors. Therefore, the feedback about expertise levels and corresponding errors (this research) would help researchers design tailored training for novices (planning phase). More specifically, our research develops a systematic approach to evaluation for expertise so that appropriate training can be designed training based on the concepts they are struggling with.

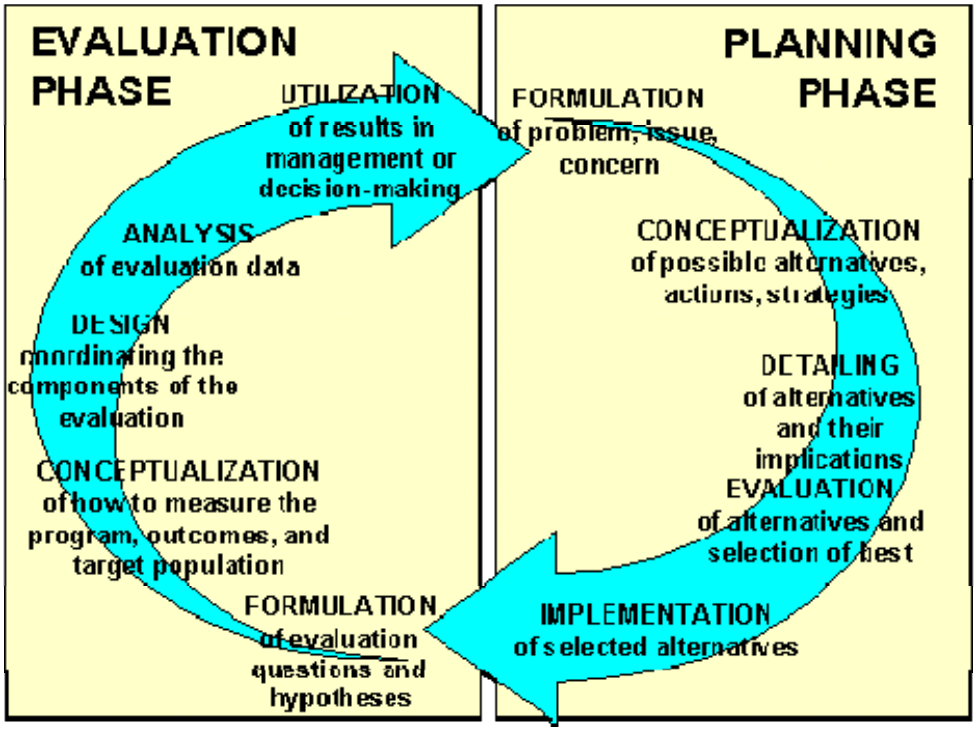

Figure 1: Planning evaluation lifecycle (Trochim 1999)

1.7. Dissertation Outline

The outline for the rest of the dissertation is as follows. Chapter 2, Literature review, describes what has been done in the past and why there is still a gap in the research to motivate this research. Chapter 3, Research model, develops the overall research model, the frameworks necessary to address Research Questions 1-5 and the associated hypotheses. Chapter 4, Research methodology, outlines methodology, data collection procedures, and sample used to test the hypotheses. Chapter 5 discusses the data analysis and results related to the hypotheses developed in the previous chapter. Finally, the conclusions and a discussion of future work are summarized in Chapter 6.

CHAPTER 2: LITERATURE REVIEW

The research objective is to understand why novices make expertise-related errors in semantic data modeling. The results of this research will influence the design of training for novices. This chapter looks at prior research in explaining errors, specifically in the area of database design. In addition, this chapter looks at research done in training novices in semantic data modeling, since the results of this research will influence the design of training for novices.

2.1. Understanding Errors Caused by People

An error could have many causes (ASEP 2002, Reason 1990). Therefore, it is important to understand the reason(s) behind the error before attempting to eliminate the error. To demonstrate the many possible reasons for an error and their implications for fixing the error, we illustrate with an error message, "This page cannot be displayed", in a browser window. Possible reasons for the error include: (a) the computer's internet connectivity is unavailable, (b) the user has typed the address incorrectly, (c) the server, on which the web page is located, is inaccessible, (d) the Internet Service Provider (ISP) has blocked the web page and (e) the web page does not exist. The strategy to eliminate the error depends on the reason. For (a), the internet connection was down, the ISP needs to repair it. For (b), the user typed the address incorrectly, the solution is re-typing the address. For (c), the page was temporarily unavailable, the fix is trying again. For (d), the ISP blocked the page, the resolution is requesting the ISP to unblock the site. For (e), the page no longer exists, the solution then is searching for archives of the page. This

example illustrates that there could be multiple reasons for an error, and the resolution depends on the reason.

Reasons for errors can be categorized based on three main factors: task, tool and person (Jenkins 1983) as illustrated in Figure 2. Our research seeks to understand the human factor and hence the rest of this section discusses the human grounds for errors. Human factors can be further classified into expertise and non-expertise (skill-based) errors. Expertise-based errors are caused by insufficient knowledge. Skill-based errors are caused by distraction. Therefore, experts make skill-based errors, even though they have understood all the concepts (Spohrer and Soloway 1986a). For example, experts make errors in simple tasks like reading or comparing digits (Bell, et al. 1997, Kantowitz and Sorkin 1983) and in logical activities, like writing programs (Panko 1997, Panko 1998). To summarize, it is not possible to eliminate skill-based (non-expertise) errors.

Our objective is to explain expertise-related errors, and thus we examine expertise errors in more depth. There are two main categories of expertise-related reasons for errors: rule-based and knowledge-based (Rasmussen 1986, Reason 1990). Rule-based errors occur due to a failure of expertise—when multiple rules appear to be applicable, and the person picks the wrong one without systematically eliminating all the options. Knowledge-based errors occur due to lack of expertise—when the person does not know any rules that will solve the task.

As discussed earlier, the course of action depends on the type of error. If the error is skill based, a checklist may suffice to minimize the possibility of the person overlooking something. However, if the error is rule-based or knowledge-based, it is important to

know what expertise is lacking so that appropriate training can be provided. Thus, it is important to be able to identify why the error is being made. We examine expertise-related errors because they are costly, recurring and can be fixed. However, not all causes for errors are expertise-related. Since we are interested in expertise-related errors, other reasons for errors need to be identified, measured, and if possible, controlled.

2.2. Understanding Database Design Errors

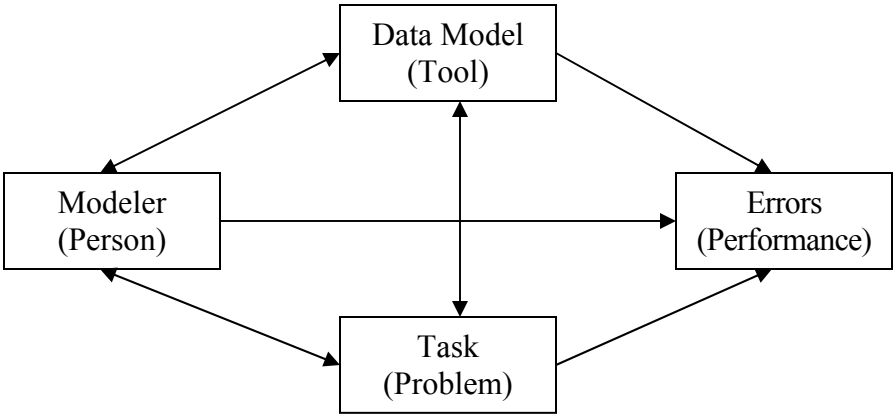

Figure 2: Factors Influencing Performance

To avoid the recurrence of errors, the factors that cause them need to be identified and corrected. Jenkins' Decision Support Systems framework (Jenkins 1983), as described in the previous section, was extended for the data modeling area (Batra, et al. 1990) to identify factors as shown in Figure 2. The dependent variable is Errors or Performance. The three factors affecting performance in designing a database are task, modeling technique, and modeler constraints. Examples of task constraints include complexity of the task, and incomplete requirements. Examples of modeling technique constraints include inability of semantic data models to capture process information and set-based

business rules (Ram and Khatri 2005). Examples of modeler constraints include modeling expertise (our focus) and domain familiarity.

The reasons for errors needs to be studied in further detail (Wand and Weber 2002). Previous work has looked the impact of some of the above-mentioned dimensions, modeling language and task, on errors or performance (Gemino and Wand 2004). However, there has been scant attention given to the influence of modeler, and in particular modeling expertise, on performance (Topi and Ramesh 2004). Since lack of sufficient modeling expertise is the cause of a large number of errors, and it is not always possible to have the most-experienced veterans develop a schema, we feel there is much to be gained from understanding the mistakes made by new analysts, so that the most costly and pernicious classes of errors they make can be identified and corrected. Therefore, our research fills this gap and studies the impact of the modeling expertise on errors.

Before we can describe how our research fills the existing gap in the literature, we need to describe, in further detail, what has been done in explaining errors in database design. As discussed earlier, there are three factors that influence errors: model, modeler and task. Since our research examines why novices make expertise-related errors, the modeler dimension is the most relevant. The model dimension is relevant because it developed classification scheme for errors. The task dimension is relevant because task and modeling expertise are related: higher the difficulty of the task, greater the level of expertise required. Therefore, the following sub-sections discuss the impact of the model, modeler and task.

2.2.1. Understanding Impact of Data Model on Errors

A number of studies compared errors within and across (a) ER model variants, (b) Relational model, (c) Object-Oriented design and (d) other miscellaneous modeling tools (Antony and Batra 2002, Batra, et al. 1990, Batra and Wishart 2004, Cambell 1992, Dey, et al. 1999, Fessakis, et al. 2005, Hardgrave and Dalal 1995, Kim, et al. 1995, Liao and Palvia 2000, Shoval and Even-Chaime 1987, Shoval and Shiran 1997, Sinha and Vessey 1999). All those studies examined the relationship between Data Model and Errors by highlighting pros and cons of different modeling grammars. To compare modeling grammars, these studies categorized errors, based on severity. For example, a sample scoring scheme awarded a score of 100% if there was no error, and subtracted a percentage based on the severity of the error, i.e., 25% was subtracted for minor errors, 50% for medium errors and 75% for a major errors (Batra, et al. 1990). Those scores were then averaged, across different instances and people, to develop a performance score for the construct. The drawback of this categorization was its subjectivity: it was based on instructor perception of severity. Our research addresses this limitation by using a more grounded and systematic evaluation: using pre-existing criteria to calculating the implications of the error.

Across multiple studies, results show that the average performance for (a) ternary relationships is below 50%, (b) binary relationships is greater than 80%, and (c) unary relationships is below 70% (Topi and Ramesh 2004). The range in average performance is large for each of the constructs. Possible reasons for the wide variance across studies include factors like expertise level, task type, application domain, type of cardinality captured, and grading scheme. Our research examines expertise level, a variable that has

not been studied as a variable before. Another reason is the aggregation of errors (done in the above studies) may mask the real distribution of errors types. Therefore, in this dissertation, we classified errors at a fine level of granularity and measured different factors.

Previous studies were designed to improve the modeling technique, not the modeler. Comparing grammars was important in early stages of developing data modeling to decide which techniques should be adopted and under what circumstances. However, now that the data modeling field has matured, the research in the field needs to move to areas like helping humans learn well-established techniques, like ER modeling, better. That is why our research considers expertise level. We select one modeling technique to examine errors at a finer granularity since the errors do not have to be generalized at a coarser level because of differences in modeling techniques. This would potentially highlight any patterns among errors that were hidden because of aggregated view of errors. This is where our research makes its contribution.

Drawbacks of current approaches	**How our research addresses limitations**
Subjective classification of errors	Systematic classification of errors
Unable to explain wide variance in types of errors made	Addresses this limitation by (a) classifying errors at a fine granularity, (b) measured different factors including modeling expertise, task type, and application domain
Focused on modeling technique, not modeling expertise	Examines modeling expertise

Table 1: Limitations of Research Examining Impact of Model on Errors

2.2.2. Understanding Impact of Modeler on Errors

Human error and its causes have been studied in a wide range of areas, including speech (Baars 1992), statistical modeling (Allwood 1984, Schutz, et al. 1998), creating spreadsheets (Panko 1998), programming (Katz and Anderson 1987-1988, Spohrer and Soloway 1986a, Spohrer and Soloway 1986b). Therefore, we felt it was important to study human (modeler) errors in database design too.

A few studies examined the impact of the modeler on errors, in database design. For example, in an extension of Reason's framework (Reason 1990) to database design, errors were classified, based on the distance between reality and its representation in the semantic data model, into syntactic and semantic errors (Batra 1993). Syntactic errors include errors in notation. Syntactic errors are easily corrected by using a template for creating a schema. Semantic errors are errors caused by the distance between user needs, represented by a requirement description, and the meaning of the corresponding expression in the data schema. Semantic errors are costly and difficult to correct automatically and thus interesting to study. Semantic errors are further classified into abstraction, simplification, overload, convergence and divergence errors. Abstraction errors are caused by inappropriate mapping from reality to representation. For example, an association between two roles maps to a unary relationship in the ER diagram (which might appear counter-intuitive, but is the correct way to model the scenario). Simplification errors occur when a complex application is erroneously separated into smaller pieces which results in distorted semantics. An example of the simplification error that leads to loss of information is when a complex ternary relationship is modeled as two (or more) easier-to-understand binary relationships. Overload errors are similar to

simplification errors except that the error is caused when the modeler missed modeling part of the situation because she did not see all parts. An example of the overload error is when a complex ternary relationship is modeled as one or more binary relationships. Convergence and divergence errors are caused by lack of consistency between the real world and the representation. Convergence errors occur when similar situations (in the real-world) are represented differently while divergence errors are caused when different situations are represented in similar ways (Burton-Jones and Weber 1999). Convergence and divergence errors do occur in ER diagrams when novices attempt to "plug-in" the constructs using pattern matching instead of trying to get a holistic view of the problem (Batra and Davis 1989, Batra and Davis 1992). Another study highlighted the influence of commonly used heuristics and biases like literal translation and anchoring heuristic in novices involved in creating an ER diagram (Batra and Antony 1994).

A major limitation of using the above classifications for errors is it is impossible to classify errors, by looking at the errors in isolation. For example, when a modeler represents a ternary relationship as one or more binary relationships, it can be simplification (he/she didn't know how to model ternary relationships) or it can be overload (he/she missed some information in the description) or an influence of the anchoring heuristic. Therefore, it is necessary to independently measure the underlying cause of the error. Another limitation is that the above classification is a high level classification designed to capture errors, irrespective of whether the cause is the data model, task expertise, skill, and other human factors. For example, abstraction errors could be viewed as a data model or an expertise error. Another example: overload errors

could be skill-based or expertise related. Both of these limitations motivate developing a framework for expertise-related reasons for predicting errors.

Drawbacks of current approaches	How our research addresses limitations
Does not independently measure modeling expertise (besides errors made)	Measures modeling expertise
High level of granularity for error classification	Classifies errors at a finer level of granularity

Table 2: Limitations of Research Examining Impact of Modeler on Errors

2.2.3. Understanding Impact of Task on Errors

Task is related to modeling expertise: greater difficulty in task requires higher levels of modeling expertise. Task can be measured along two major dimensions: type of task (like interpreting, validating and creating diagrams) and task complexity (measured by structure, difficulty, time). Most studies examined a single type of task (Agarwal, et al. 1996, Batra and Kirs 1993, Brosey and Shneiderman 1978, Jarvenpaa and Machesky 1989, Kim and March 1995, Siau, et al. 1995, Siau 1996). Some studies examined multiple levels of task types (Batra and Antony 2001, Brosey and Shneiderman 1978) and complexities (Hardgrave and Dalal 1995, Liao and Palvia 2000, Shoval and Even-Chaime 1987, Weber 1996). However, none of the studies, to the best of our knowledge, spanned multiple types and complexities.

Our research uses a knowledge framework as its underlying theory. By using a knowledge framework, the distinction between task type and complexity can be eliminated and the range of types of tasks and complexity can be viewed as a continuum of task characteristics. By doing this, the similarities and differences among the different

task types and complexities will be highlighted. This will allow research across different tasks can be aggregated. Our research demonstrates the approach of the process of converting a particular task, creating diagrams with one level of complexity (described in Chapter 3). We chose creating ER diagrams because it is error prone and the impact of errors has significant impacts on the whole database design.

Drawbacks of current approaches	How our research addresses limitations
Cannot compare results across different tasks	Our research provides an approach to compare results across different tasks

Table 3: Limitations of Research Examining Impact of Task on Errors

2.3. Training Systems for Database Design

Since the results of this research will influence the design of training for novices, research in developing training (Anderson, et al. 1985, Anderson, et al. 1995, Bostrom, et al. 1988) for semantic data modeling is also relevant. The motivation for the work done in the training area of research is to empower the user and reduce their dependence on database design experts. In recent years, there has been an increased interest in online tutoring systems (referred to as Intelligent Tutoring Systems) because they are easily accessible and cost effective. Since our objective is to determine the reasons for errors, the results can be applied to any tutoring system, including Intelligent Tutoring Systems (ITS).

Database design professionals obtain user requirements and design databases through labor intensive methods (Bouzeghoub, et al. 1985). The downside of using database design experts is that they are scarce. So, ITS were proposed to help end users obtain necessary skills to design simple databases (Ahrens and Sankar 1993, Bock and Yager

2001, Suraweera 2001, Suraweera and Mitrovic 2002). However, the tutors built to date in database design do not try to model novices' expertise accurately (Chi, et al. 2004)—they try to eliminate the errors (symptoms) without seeking out the cause of the error. Tutors designed solely based on errors (Antony and Batra 2002, Batra and Antony 2001, Suraweera 2001, Suraweera and Mitrovic 2002) fail to correct situations where incorrect cognitive process does not result in an error (Evans 1989). Also, as discussed previously (sections 2.1 and 2.2), it is not possible to know the cause of errors by looking at the error in isolation (Reason 1990). In fact, in the performance appraisal literature, focusing on the errors instead of the cause changed the pattern of errors and reduced overall accuracy (Murphy and Cleveland 1995). This motivates the study of the reasons behind errors, like done in this research.

Drawbacks of current approaches	How our research addresses limitations
Focused on modeling technique, not modeling expertise	Examines modeling expertise

Table 4: Limitations of Research in Training Systems for Database Design

2.4. Expert Systems

An alternative to training novices to construct semantic data schemas is designing an expert system to create the schemas. An expert system is a computer system that emulates experts' problem-solving abilities in a particular domain. If an expert system can be designed to create semantic data schemas for a problem automatically, then, there is no longer a need to train novice data designers. A number of systems have been developed in database design. The survey papers (Lloyd-Williams and Beynon-Davies 1992a, Lloyd-Williams and Beynon-Davies 1992b, Noah and Lloyd-Williams 1995,

Storey 1993, Storey and Goldstein 1993) found that, for the semantic data modeling phase, the users of the systems needed to know data modeling concepts to use the systems effectively. The main benefit from the systems were gained by expert semantic data modelers because the systems produced an initial approximation of the semantic data model that experts could then tailor for a particular situation. However, since the expert systems required expertise in semantic data modeling and just reduced the time involvement, they do not eliminate the need for experts.

Drawbacks of current approaches	How our research addresses limitations
Does not replace need for novice designers	Our research provides an approach to evaluate modeling expertise to help train novice designer

Table 5: Limitations of Research Examining Impact of Task on Errors

2.5. Summary

Current expert systems cannot replace an expert, so it is still important to design training to eliminate errors. Errors, a measure of Performance, have been studied extensively. However, the reasons, especially human-related and task related reasons, for errors in semantic data modeling still needs more research. This chapter motivated the need for examining reasons in addition to errors. In particular, it motivated examining expertise-related reasons. Therefore, our research makes a contribution by proposing to study, a new construct in semantic data modeling: expertise. Our research is also the first to develop, to the best of our knowledge, an in-depth classification of ER modeling errors. In addition, our research approach can be used as a framework to aggregate existing literature examining the impact of different types of tasks on performance. The

implications of this research include that training systems need to be designed to take into account why errors are made.

CHAPTER 3: RESEARCH MODEL

3.1. Error Research Model Overview

In this chapter, we introduce and describe the core research model. Our objective is to explain why novices make *expertise-related* errors in semantic data modeling. To address the research objective, it is first necessary to develop the *Modeling Expertise Framework* (MEF) and the *Performance Framework* (PF). Then, we build the *Error Research Model* (ERM) which will predict, based on the position of the designer in the MEF, the types of errors that will be made in the PF. The MEF uses a knowledge framework to measure levels of expertise for novices for different aspects of semantic data modeling. The PF classifies the types of errors made.

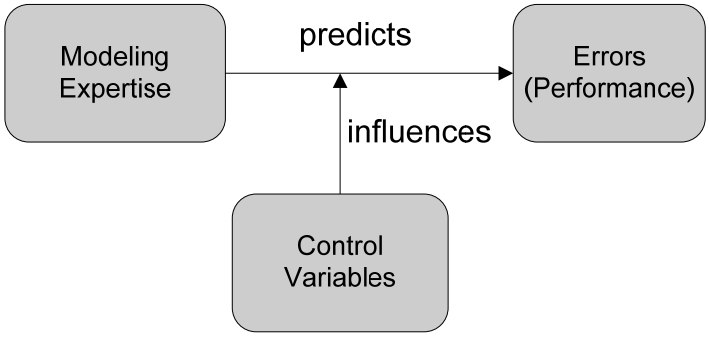

Figure 3: Error Research Model Overview

There are many factors, like data model, designer, and task that influence the errors made in semantic data schemas developed (Beach and Mitchell 1978). Designer-related factors include intelligence, knowledge, experience, learning style. A large number of studies show that intelligence and hence job-related knowledge is a good predictor of job performance (Schmidt and Hunter 2004). In the context of this research, job-related

knowledge would include designer's *modeling expertise* and job performance would include *errors* made. However, *modeling expertise* has not been studied sufficiently as a construct in semantic data modeling, and hence it should be isolated and eliminated before the impact of other factors, like changing requirements, are studied.

Now that we have provided an overview of the research and motivated the measurement of *modeling expertise*, the rest of our chapter develops ERM which is the core of our research.

3.2. Modeling Expertise Framework

3.2.1. Overview

The terms expertise may be construed as a synonym for analyst experience. However, a designer's *modeling expertise* is distinct from her experience because it is possible to have a lot of experience without truly becoming an expert. Experience can aid in improving expertise only if the designer receives feedback on her design, and doesn't attribute it to an incorrect or incomplete specifications or changing requirements. Users typically provide feedback about database design after implementation. If errors are discovered at this stage, it is difficult to identify their cause. The likelihood of tracing it back to the lack of expertise of a specific designer and providing her feedback on her work at this stage is quite small. Particularly since by then other factors interact with and often amplify an initial mistake (Penrose and Seiford 1988). Thus it would be hard to isolate, at the implementation stage, errors caused by the semantic modeler distinct from those caused by changing requirements, miscommunication, errors made at one of the later stages of database design: semantic modeling, logical modeling, and physical

modeling. It is however important to identify reasons for errors, so that the reasons can be addressed, and recurrence of the error can be prevented. Hence, our objective is to measure and isolate errors caused by inadequate semantic modeling expertise. As part of the theoretical foundation for our work, we introduce the concept of knowledge frameworks, and show how this helps us develop our own ideas of expertise-related errors in semantic modeling.

3.2.2. Background: Knowledge Frameworks

We believe that knowledge frameworks can be utilized to explain why novices make errors. A large number of knowledge frameworks have been developed in the instructional setting, to help with learning, and in other applied domains (Anderson, et al. 2001, Bloom, et al. 1956, Ebel and Frisbie 1986, Gagne, et al. 1988, Sein, et al. 1999). A key utility of knowledge frameworks is that they help us classify types of information and cognitive processes used in learning. For example, the initial version of Bloom's taxonomy (Bloom, et al. 1956) and Ebel's relevance guide (Ebel and Frisbie 1986) measured thinking processes while Gagné's five categories of learning outcomes (Gagne, et al. 1988) categorized types of information. However, the framework we found most useful and relevant for our work was the revised Bloom's taxonomy (Anderson, et al. 2001) which classifies both the cognitive processes and the types of knowledge.

Using knowledge frameworks provides us with a structure to measure model expertise. The revised version of Bloom's taxonomy (RBT) is the most comprehensive of the available knowledge frameworks. It has, in our view, the widest applicability. In testament to its usefulness we note that the original taxonomy has been translated into

more than twenty languages and has been used to design a variety of learning-related tests (Krathwohl 1994). Therefore, it is safe to conclude that it is a mature model. While many see its relevance in the humanities, it has been used in the sciences and areas that require logical thinking such as mathematics (Anderson, et al. 2001, Crawford and Brown 2002, Jolliffe and Ponsford 1989, Morris 2004, Vidakovic, et al. 2003) and physics (Anderson, et al. 2001, Dickie 2003). We feel it is appropriate for studying learning in a technology-related field such as database design as well. Thus, we were comfortable in using it as the foundation of MEF and ERM.

3.2.3. Applicability of Revised Bloom's Taxonomy

The previous section discussed why we selected Revised Bloom's Taxonomy (RBT). RBT aligns instructional objectives, activities and assessment tasks, so that the efficacy of instruction is optimized (Krathwohl 2002, Raths 2002). Therefore, RBT can help us define and measure expertise levels. We now describe RBT and outline how it can be used for semantic data modeling. The taxonomy has two dimensions, the *knowledge dimension* and the *cognitive processes dimension*. The knowledge dimension captures types of information while the cognitive processes dimension measures the level of expertise. The cognitive process dimension describes the hierarchical processes employed in learning. The two dimensions and their relevance to semantic data modeling are described in further detail in the next two subsections.

3.2.3.1. Knowledge Dimension

The knowledge levels are Factual, Conceptual, Procedural and Metacognitive. In this section, we describe what each level symbolizes and how the levels relate to semantic data modeling.

The objective of factual knowledge is to learn specific content or facts. The objective of conceptual knowledge is to learn the interrelationships between the different elements that allow them to function together. The objective of procedural knowledge is to learn a process, like using specific techniques and methods. The objective of metacognitive knowledge is to learn cognition and make a person more aware of her own cognition

The notation for the semantic data modeling fits under factual knowledge. All other aspects of semantic data modeling, as the name suggests, fits best with the *conceptual knowledge* dimension because their objective is to help organize data. Designers use semantic data modeling constructs, like entity classes and relationships, to structure all the data needed for a system. Thus, the learning objective goes beyond learning isolated facts like the definition of an entity class (factual knowledge). Instead, semantic data modeling is a tool to organize information (conceptual knowledge). ER modeling is an art and a science, and factors like application domain, information to be stored and queries to be generated, all impact what the final ER schema will look like. Therefore, there is no standard approach to create an ER schema, and hence procedural knowledge is not applicable. Since the objective is not to improve awareness of one's own cognition, metacognitive knowledge is not relevant.

3.2.3.2. Cognitive Process Dimension

The cognitive processes: *Remember, Understand, Apply, Analyze, Evaluate* and *Create* are hierarchically ordered, i.e., each builds on the previous one. *Remember* refers to the process of retrieving information from memory. For example, a learner who is at the *Remember* level for entity classes can recall the definition of entity classes from memory. *Understand* goes beyond just *remembering* the information and refers to deriving meaning from instruction. For example, a learner who is at the *Understand level* for entity classes can provide an example of an entity class in a specified environment. *Apply* builds on *understand* and uses a procedure on a new task. *Apply* is only relevant to aspects of data modeling that have a well-defined semantics, like cardinality. For example, a learner who is at the *Apply* level for cardinality knows that cardinality for a relationship shows how entity instances in that relationship are associated. *Analyze* level refers to a deeper comprehension of the material. At this stage, the learner grasps subtleties and knows how the different components fit together. For example, at the *Analyze* stage, the learner will recognize that the mini-world, application system, and processes are irrelevant pieces of information. At the *evaluate* level, the learner can judge the quality, effectiveness, efficiency and consistency of the product. For example, a learner at the *evaluate* level for relationships knows that a maximum of two binary relationships are needed to capture a single, connected semantic association among three entity classes. *Create* is the ability to generate an appropriate semantic data schema for a system. At the *Create* level, a learner can put different pieces together and invent a new

product. Novices are not expected to reach the *create* level. Therefore, we leave studying the impact of achieving the *create* level for future research.

3.2.3.3. Summary

As discussed above, the Semantic Data Modeling fits within Conceptual knowledge (along the knowledge dimension) and spans across *Remember* to *Evaluate* levels (along the cognitive process dimension).

3.2.4. Developing the Modeling Expertise Framework

The previous sub-section discussed which aspects of RBT are applicable to Semantic Data Modeling. Now that we have described RBT and defined the scope of its applicability, the Modeling Expertise Framework (MEF) can be described. An overview of MEF is shown in Figure 4. The first step in developing the Modeling Expertise Framework (MEF) is determining the main learning objectives for semantic data modeling. The objectives list all the components of learning that need to be attained to master the material. The second step is classifying the overall learning objectives using the knowledge framework, Revised Bloom's taxonomy (RBT). Placing the objectives in RBT clarifies which cognitive processes are applicable to semantic data modeling. The third step is translating the overall objectives to more specific objectives. Expertise can be examined at a finer granularity when specific objectives are used. The fourth step is classifying the specific objectives using the Revised Bloom's taxonomy. Classifying the specific objectives enables measurement of expertise and this completes the development of the MEF.

Once the MEF is developed, hypotheses are designed to examine the relationship among different degrees of expertise. The *Bloom's taxonomy verification hypotheses* examine the relationship between adjacent levels of expertise for a semantic data modeling construct. The *Relationship among Constructs hypotheses* examine the association across different semantic data modeling constructs within an expertise level.

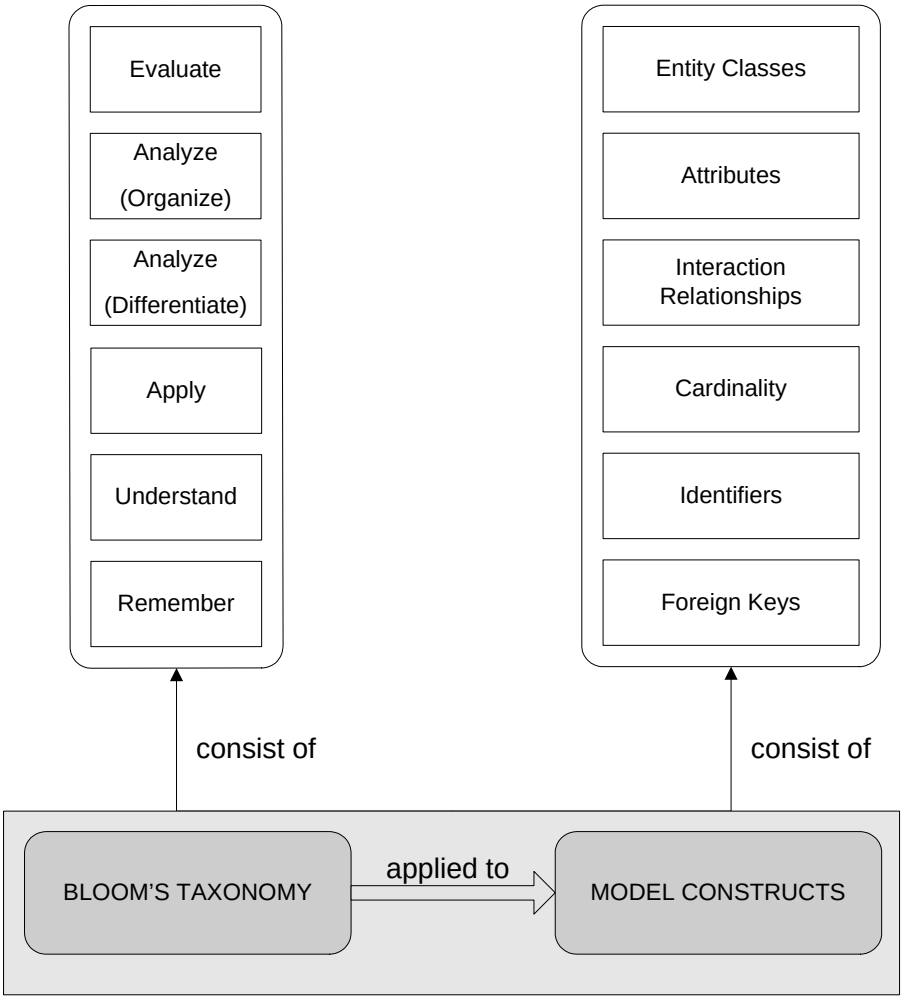

Figure 4: Overview of Modeling Expertise Framework

3.2.4.1. Overall Learning Objectives in Semantic data Modeling

Based on discussions with experts in the field, six overall learning objectives were chosen. The objectives are listed below:

I. Know the notation for every semantic data modeling construct
II. Understand the meaning of each semantic data modeling construct
III. Evaluate the comprehensiveness of the solution (compare the description with the solution to ensure no relevant information has been missed)
IV. Choose the most appropriate semantic data modeling constructs for a particular situation
V. Evaluate the semantic quality of the solution
VI. Produce a semantic data schema to accurately capture the data requirements in an organizational setting

The objectives are classified using the Revised Bloom's Taxonomy. As discussed in the previous section (3.2.3), the Revised Bloom's taxonomy has two dimensions: knowledge and cognitive processes. To summarize, the four types of knowledge are factual, conceptual, procedural and metacognitive. The six levels of cognitive processes are Remember, Understand, Apply, Analyze, Evaluate and Create.

We now explain how each objective is classified within the Revised Bloom's taxonomy (Table 6 contains a synopsis). *Know the notation for every semantic data modeling construct* (Objective I) fits within Factual knowledge (on the Knowledge dimension) and Remember (on the Cognitive Processes dimension) because it involves memorizing isolated pieces of information. Objectives II through VI span multiple

cognitive processes within Conceptual knowledge (on the Knowledge Dimension) because the objectives look at different aspects of organizing information, not just at isolated facts. Objective II, *Understand the meaning of every semantic data modeling construct*, involves grasping the meaning of each data modeling construct. Therefore, *Understand the meaning of every semantic data modeling construct* (Objective II) fits in best under Conceptual knowledge and Understand level. *Evaluate the comprehensiveness of the solution* (Objective III) involves two cognitive processes: (a) identifying overlap between the requirements and the corresponding schema (translated version of the requirements) (at the Understand level) and (b) differentiating relevant and irrelevant information (at the Analyze level). Since the more complex of the two cognitive processes is at the Analyze level, objective III fits in best with the Analyze level within Conceptual knowledge. *Choose the most appropriate semantic data modeling constructs for a particular situation* (Objective IV) goes beyond *understanding the meaning of each data modeling construct* (Objective I) and requires the novice to be able recognize information in the requirements description is relevant (at the Analyze level) and realize how it relates to other information captured (also at the Analyze level). Therefore, *Choose the most appropriate semantic data modeling constructs for a particular situation* (Objective IV) fits in with Conceptual knowledge and Analyze. *Evaluate the semantic quality of the solution* (Objective V) and *Produce a semantic data schema to accurately capture the data requirements in an organizational setting* (Objective VI) require advanced level of expertise, i.e., they require all the other objectives (I through IV) to be satisfied. Evaluating the semantic quality of the solution requires all the

preceding objectives to be satisfied, i.e., understand the meaning of every data modeling construct, know the notation for every data modeling construct, choose the most appropriate data modeling construct for a particular situation, evaluate the comprehensiveness of the solution. Before a designer undertakes *Producing a semantic data schema*, she should have mastered all the other learning objectives (I through V).

The scope of this research is objectives II (Understand the meaning of every semantic data modeling construct), III (Evaluate the comprehensiveness of solution) and IV (Choose the most appropriate semantic data modeling constructs for a particular situation) since those are the most relevant levels for novices.

	Remember	**Understand**	**Apply**	**Analyze**	**Evaluate**	**Create**
Factual	I					
Conceptual		II		III, IV	V	VI
Procedural	Not relevant					
Metacognitive	Not relevant					

Table 6: Overall Objectives in Revised Bloom's Taxonomy

Objectives I (Know the notation for every semantic data modeling construct), V (Evaluate the semantic quality of the solution) and VI (Produce a semantic data schema to accurately capture the data requirements in an organizational setting) were purposely scoped out from this research. Objective I (Know the notation for every semantic data modeling construct) is excluded because appropriate notation can always be refreshed from a database design book. Objectives V (Evaluate the semantic quality of the solution)

and VI (Produce a semantic data schema to accurately capture the data requirements in an organizational setting) are excluded because they require a level of expertise not attainable by the target audience—novices. Follow up studies can be designed to measure attainment of the last two objectives (V and VI) for designers who have mastered the other objectives (I through IV).

The next subsection converts the broad objectives to more measurable, specific objectives and classifies these specific objectives using RBT.

3.2.4.2. Specific Learning Objectives in Semantic Data Modeling

In this subsection, the overall objectives are translated to more specific objectives. We now proceed to explore Objectives II, III and IV in further depth.

We select five of the most commonly used data modeling constructs in semantic data modeling: *Entity classes, Attributes, Interaction Relationships, Cardinality* and *Identifiers*. In addition, *Foreign Keys*, a concept from the relational data model is selected. Although Foreign keys are not seen in the ER diagram, we are studying Foreign keys because to really learn ER modeling, it is necessary to appreciate the bigger picture of how Foreign Keys are connected to ER modeling. One reason for including Foreign Keys' learning objectives is because Foreign Keys is the only concept that is connected to semantic data modeling (because it is used to represent relationships in the corresponding logical schema) and knowledge of related concepts influences the rate or accuracy of learning (Murphy and Medin 1985, Pazzani 1991, Schank, et al. 1986, Wisniewski 1995). Another reason for including Foreign Keys is that Foreign Keys can be easily modeled using semantic data modeling constructs even though they should not be modeled. In the

next sub-section, we are going to elaborate more about Foreign keys and how they fit into the learning of ER modeling.

3.2.4.3. Learning Objectives for Foreign Keys

It is necessary to comprehend the bigger picture of how the semantic and logical design phases fit together, to grasp ER modeling. At the logical design phase, all the information contained in the semantic data schema is converted to tables. The semantic structure of the information organization gets lost in the logical schema. Relationships in the semantic data schema are converted into foreign keys in the logical schema. Foreign keys allow linkage between tables by repeating the identifier (called primary key in the logical model) of one table in another. There are two related contexts in which the term *foreign key* is used. The first usage for the term foreign keys is as a *constraint*: foreign keys constrain the value (instance level) an attribute can have to the associated primary key value or null. For example, if department number, from the departments table is a foreign key in the employees table: this implies that the value of department number is constrained to an existing department. The second use of the term foreign key is as an *attribute* that represents a relationship (schema level). For example, if department number, from the departments table is a foreign key in the employees table: this implies that there is a relationship (works in) between employee and department and an employee can work in at most one department. The second context indicates the connection to semantic data modeling and is the one we adopt. Novices should know that foreign keys should not be represented in the semantic data schema. In addition, novices should know

why, i.e., a relationship is a richer representation that clearly shows how different entity classes are associated.

Therefore, this objective translates to:

1. Know the definition of foreign keys (Remember level)
2. Understand that foreign keys should not be used in semantic data modeling (Understand)
3. Distinguish foreign keys as irrelevant information (Differentiate within Analyze)

Construct name	Remember	Understand	Analyze (Differentiate)
Foreign keys	✓	✓	✓

Table 7: Translating Learning Objectives for Foreign Keys

Now that the objectives for foreign keys have been identified and placed within RBT, hypotheses can be developed to examine the relationship among expertise levels identified. Based on the revised Bloom's taxonomy, the above sub-objectives are ordered. Since this learning objective spans three cognitive processes (Table 7), there will be two RBT hypotheses detailing the precedence among them. The first hypothesis details it is necessary to be at the Remember level before achieving the Understand level; i.e., the novice should 'Know what foreign keys are' (Remember) before she can 'Understand that foreign keys cannot be used in semantic data modeling' (Understand). The second hypothesis specifies that the Analyze level requires achievement of the Understand level; i.e., it is necessary that the novice should 'Understand that foreign keys cannot be used in

semantic data modeling' (Understand) before she can 'distinguish foreign keys as irrelevant information' (Differentiate within Analyze).

Bloom's taxonomy verification hypothesis 1: A novice should be at the Remember level for foreign keys before she is at the Understand level for foreign keys.

Bloom's taxonomy verification hypothesis 2: A novice should be at the Understand level for foreign keys before she is at the Differentiate sublevel for foreign keys.

3.2.4.4. Translating "Understand the meaning of every semantic data modeling construct" (Objective II)

Construct name	Understand the definition of construct
Entity classes	✓
Attributes	✓
Interaction relationships	✓
Cardinality	✓
Identifiers	✓

Table 8: Translating Objective II: Understand the meaning of every modeling construct

Objective II is the most basic level of the expertise for data modeling constructs. It requires the novice to understand the definition and hence to be able to provide examples of the modeling construct. This objective is applicable for all the five data modeling constructs as shown in Table 8.

There are two categories of hypotheses that can be tested based on MEF. The RBT hypotheses predict whether the hierarchy of the Revised Bloom's taxonomy is maintained for an ER modeling construct. There are no hypotheses to test based on RBT

since Objective II measures the lowest level of expertise, understand, that we are interested in measuring for the data modeling constructs, as evident from Table 8.

Relationship among data modeling constructs	Necessary to be at Understand level for	Before being at the Understand level for
Hypothesis 1	Interaction relationships	Cardinality
Hypothesis 4		Role of foreign keys
Hypothesis 2	Entity classes	Interaction Relationships
Hypothesis 3	Attributes	Identifiers

Table 9: Relationship among data modeling constructs associated with Objective II

The *relationship among data modeling constructs* hypotheses predict whether there is an ordering among learning ER modeling constructs for a particular expertise level. The hypotheses are explored to discover any precedence relationship among learning the different data modeling constructs at the same expertise level as shown in Table 9. For example, we believe it is necessary to be at the Understand level for interaction relationships before being at the Understand level for (a) cardinality because he method of calculating cardinality varies based on the degree of the relationship and (b) role of foreign keys because a novice needs to understand what relationships represent before they can see the connection between relationships and foreign keys. Understand level for entity classes is necessary to reach the Understand level for interaction relationships because relationships are described as connecting one or more entity classes. Understanding attributes precedes identifiers because identifiers are defined as the

attributes that uniquely identify instances of an entity class. If a positive relationship is found, subsequent studies can be designed examine the associations in further detail.

Relationship among data modeling constructs hypothesis 1: A novice should be at the Understand level for interaction relationships before she is at the Understand level for cardinality.

Relationship among data modeling constructs hypothesis 2: A novice should be at the Understand level for entity classes before she is at the Understand level for relationships.

Relationship among data modeling constructs hypothesis 3: A novice should be at the Understand level for attributes before she is at the Understand level for identifiers.

Relationship among data modeling constructs hypothesis 4: A novice should be at the Understand level for interaction relationships before she is at the Understand level for foreign keys.

3.2.4.5. Translating "Evaluate the comprehensiveness of the solution" (Objective III)

The first part of objective III requires the novice to be at the Understand level for data modeling constructs. This part of Objective III overlaps with Objective II and has been converted to more specific goals, as described in section 3.2.4.4. The second part of objective III requires the novice to be at the Differentiate sublevel of Analyze for the data modeling constructs. We convert this part of the objective into more specific goals. If the novice is at the Differentiate sublevel, she can detect (a) if any requirement is not represented in the corresponding schema and (b) if any information is in the schema, but not a requirement. In other words, when at the Differentiate sublevel, the novice can

identify relevant and irrelevant information for each modeling construct, as shown in Table 10. Identifying relevant information is applicable to all constructs. However, identifying irrelevant information is only applicable to (a) entity classes because novices should recognize mini-world and system information as extraneous and (b) relationships because novices should recognize irrelevant process information.

Therefore, this objective only needs to translate the Differentiate sublevel of the objective.

1. Identify relevant information for entity classes
2. Identify relevant information for attributes
3. Identify relevant information for interaction relationships
4. Identify relevant information for cardinality
5. Identify relevant information for identifiers
6. Identify irrelevant information for entity classes
7. Identify irrelevant information for interaction relationships

Construct name	**Understand**	**Analyze (Differentiate$_{Relevant}$)**	**Analyze (Differentiate$_{Irrelevant}$)**
Entity classes	✓	✓	✓
Attributes	✓	✓	
Interaction relationships	✓	✓	✓
Cardinality	✓	✓	
Identifiers	✓	✓	

Table 10: Translating Objective III: Evaluate Comprehensiveness of Solution

RBT requires an ordering between the Understand level and Differentiate sublevel of Analyze level for each data modeling construct as shown in Table 11. For example, the first row states that, based on Bloom's Taxonomy Verification Hypotheses 3, a novice should understand entity classes before she can identify relevant and irrelevant information for entity classes. Cardinality has a different arrangement because it is the only data modeling construct where the Apply level is relevant. Since the immediate predecessor to the Differentiate sublevel, for cardinality, is Apply, therefore, the corresponding Bloom's taxonomy verification hypothesis is that the novice needs to be at the Apply level before reaching the Differentiate sublevel.

Bloom's Taxonomy Verification Hypotheses	**Necessary to be at Understand level for**	**Necessary to be at Apply level for**	**Before being at the Differentiate sublevel for**
Hypothesis 3	Entity classes		Entity classes
Hypothesis 4	Attributes		Attributes
Hypothesis 5	Interaction Relationships		Interaction Relationships
Hypothesis 6		Cardinality	Cardinality
Hypothesis 7	Identifiers		Identifiers

Table 11: Revised Bloom's Taxonomy Verification Hypotheses associated with Objective III

The corresponding Bloom's taxonomy verification hypotheses are listed below.

Bloom's taxonomy verification hypothesis 3: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel for entity classes.

The ability to recognize relevant information is independent of the ability to recognize irrelevant information. For entity classes, recognizing irrelevant information involves recognizing objects like the mini-world or system are irrelevant information while relevant information is context dependent and could be objects like Books or persons like Employees. Therefore, this hypothesis can be separated into the following hypotheses:

Bloom's taxonomy verification hypothesis 3a: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel (relevant) for entity classes.

Bloom's taxonomy verification hypothesis 3b: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel (irrelevant) for entity classes.

Bloom's taxonomy verification hypothesis 4: A novice should be at the Understand level for attributes before she is at the Differentiate sublevel for attributes.

For attributes, we ignored the ability to recognize irrelevant information because some instructors recommend that novices add extra attributes as needed. Therefore, we only care about the ability to recognize relevant information for attributes.

Bloom's taxonomy verification hypothesis 5: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel for interaction relationships.

Interaction relationships are meaningful associations among entity classes. However, not every possible linkage among entity classes is meaningful. Therefore, novices should

recognize which interactions are important and which are not. Therefore, the hypothesis can be separated into the following hypotheses:

Bloom's taxonomy verification hypothesis 5a: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel (relevant) for interaction relationships.

Bloom's taxonomy verification hypothesis 5b: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel (irrelevant) for interaction relationships.

Cardinality is the only data modeling construct where the Apply level is relevant. Therefore, this Bloom's verification hypothesis is between the Apply level to the Differentiate sublevel of Analyze. At the Apply level, the novice can determine cardinality for binary relationships. However, at the Differentiate sublevel, the novice can determine ternary cardinality.

Bloom's taxonomy verification hypothesis 6: A novice should be at the Apply level for cardinality before she is at the Differentiate sublevel for cardinality.

At the Understand level for identifiers, the novice can determine one attribute identifiers. However, at the Differentiate sublevel, the novice can identify relevant information for identifiers.

Bloom's taxonomy verification hypothesis 7: A novice should be at the Understand level for identifiers before she is at the Differentiate sublevel for identifiers.

Based on the associations among data modeling constructs, it is anticipated that the novice would be able to distinguish relevant and irrelevant information for entity classes before doing so for interaction relationships as shown in Table 12.

Relationship among data modeling constructs hypothesis 5: A novice should be at the Differentiate sublevel for entity classes before she is at the Differentiate sublevel for interaction relationships

Relationship among data modeling constructs	Necessary to be at Differentiate sublevel for	Before being at the Differentiate sublevel for
Hypothesis 5	Entity classes	Interaction Relationships

Table 12: Relationship among data modeling constructs associated with Objective III

3.2.4.6. Translating "Choose the most appropriate semantic data modeling constructs for a particular situation" (Objective IV)

There are four main requirements to be able to successfully choose the most appropriate data modeling construct as summarized in Table 13. The first requirement is that the novice understands what each data modeling construct is. The second requirement is that the novice is able to apply the data modeling construct. The third requirement is that the novice can select the correct data modeling construct to be used. The fourth requirement is that the novice can select the correct variant of the data modeling construct and organize it within the overall schema.

The first requirement (understands what each modeling construct is) is achieved when Objective II is achieved.

The second requirement (ability to apply the data modeling construct) is only relevant to cardinality since cardinality is the only data modeling constructs with a standard procedure. At the Apply level for cardinality, the novice can successfully calculate cardinality. Therefore, this requirement translates to:

- Determine cardinality for binary relationships

Based on the revised Bloom's taxonomy, it is necessary to be at the Understand level, to be at the Apply level as shown in Table 14, i.e., a novice should understand cardinality before she can determine cardinality.

Construct name	**Understand**	**Apply**	**Analyze (Differentiate $_{\text{Relevant}}$)**	**Analyze (Organize$_{\text{Type}}$)**	**Analyze (Organize $_{\text{Grouping}}$)**
Entity classes	✓	✓	✓	✓	✓
Attributes	✓	✓	✓		✓
Interaction relationships	✓	✓	✓		✓
Cardinality	✓	✓	✓		✓

Table 13: Translating Objective IV: Choose the most appropriate construct for a situation

Bloom's taxonomy verification hypothesis 8: A novice should be at the Understand level for cardinality before she is at the Apply level for cardinality.

The third requirement indicates a novice's ability to select the correct data modeling construct. To select correctly, the novice should be able to recognize relevant information for the modeling construct. This requirement is attained when Objective III is achieved.

The fourth requirement involves choosing the correct variant of the modeling construct and organizing it within the overall schema (summarized by the last two columns of Table 13). Both aspects of the requirement require the novice to be at the Organize sublevel. The sub-requirement of organizing the data modeling construct within the schema is applicable entity classes, attributes, relationships and cardinality. Organize level for identifiers was excluded because some learners had been instructed to create an artificial "Id" identifier for all strong entity classes. Choosing the correct variant is only applicable when there are multiple variants; i.e., for entity classes, attributes and relationships. The different types of attributes (composite, derived, multi-valued) were excluded from the research because novices had received different instructions about their relevance. Also, we restricted the scope only one type of relationship: interaction relationships. Therefore, this sub-requirement of choosing the correct variant will only be examined for entity classes.

Bloom's Taxonomy Verification Hypotheses	**Necessary to be at Understand level for**	**Necessary to be at Apply level for**	**Before being at the Organize level for**
Hypothesis 8	Cardinality	Cardinality	
Hypothesis 9	Attributes		Attributes
Hypothesis 10	Entity classes		Entity classes
Hypothesis 11	Interaction Relationships		Interaction Relationships
Hypothesis 12		Cardinality	Cardinality

Table 14: Revised Bloom's Taxonomy Verification Hypotheses associated with Objective IV

To summarize, the fourth requirement includes:

1. Correctly associating attributes with appropriate entity classes
2. Correctly selecting between strong and weak entity classes
3. Correctly combining or separating entity classes
4. Correctly organizing shared information among entity classes and choosing the correct degree for the relationships
5. Correctly organizing cardinality for relationships

Based on the revised Bloom's taxonomy, it is necessary to be at the Understand level to be at the Organize sublevel (select the correct variant of the data modeling construct and organize it within the schema) as shown in Table 14. This translates to the following hypotheses:

Bloom's taxonomy verification hypothesis 9: A novice should be at the Understand level for attributes before she can correctly associate attributes with entity classes (Organize sublevel).

Bloom's taxonomy verification hypothesis 10: A novice should be at the Understand level for entity classes before she is at the Organize sublevel for entity classes.

This hypothesis can be separated into the following two hypotheses:

Bloom's taxonomy verification hypothesis 10a: A novice should be at the Understand level for entity classes before she can correctly choose between strong and weak entity classes (Organize sublevel).

Bloom's taxonomy verification hypothesis 10b: A novice should be at the Understand level for entity classes before she can correctly decide when to combine and when to separate entity classes (Organize sublevel).

Bloom's taxonomy verification hypothesis 11: A novice should be at the Understand level for relationships before she can correctly organize information shared by entity classes into appropriate relationships (Organize sublevel).

Bloom's taxonomy verification hypothesis 12: A novice should be at the Apply level for cardinality before she can correctly organize cardinality (Organize sublevel).

3.2.5. Summary

This section motivated the use of Knowledge frameworks and in particular Bloom's taxonomy for measuring expertise. Then, we described how we developed the Modeling Expertise Framework (MEF). We scoped our work by selecting the most commonly used data modeling constructs and the most important learning objectives for semantic data modeling as shown in Table 15.

	Remember	Understand	Apply	Analyze	Evaluate
Entity Classes		✓		✓	
Attributes		✓		✓	
Interaction Relationships		✓		✓	
Cardinality		✓	✓	✓	
Identifiers		✓		✓	
Foreign Keys	✓	✓		✓	

Table 15: Specific Objectives in Revised Bloom's Taxonomy

We also investigated the Bloom's taxonomy verification hypotheses which evaluate the hierarchical nature of cognitive processes of the revised Bloom's taxonomy, for each modeling construct. These hypotheses are summarized in Table 16. We explored associations among constructs within an expertise level (summarized in Table 17).

H#	Hypothesis description
BV1	Remember(Foreign keys)→ Understand(Foreign keys)
BV2	Understand(Foreign keys) → Differentiate(Foreign keys)
BV3	Understand(Entity classes) → Differentiate(Entity classes)
BV4	Understand(Attributes) → Differentiate(Attributes)
BV5	Understand(Relationships) → Differentiate(Relationships)
BV6	Apply(Cardinality) → Differentiate(Cardinality)
BV7	Understand(Identifiers) → Differentiate(Identifiers)
BV8	Understand(Cardinality) → Apply(Cardinality)
BV9	Understand(Attributes) → Organize(Attributes)
BV10	Understand(Entity classes) → Organize(Entity classes)
BV11	Understand(Relationships) → Organize(Relationships)
BV12	Apply(Cardinality) → Organize(Cardinality)

Table 16: Bloom's Taxonomy Verification Hypotheses

H#	Hypothesis description
RC1	Understand(Relationships)→ Understand(Cardinality)
RC2	Understand(Entity classes) → Understand(Relationships)
RC3	Understand(Attributes) → Understand (Identifiers)
RC4	Understand(Relationships) → Understand(Foreign keys)
RC5	Differentiate(Entity classes) → Differentiate(Relationships)

Table 17: Relationship Among Constructs Hypotheses

3.3. Performance Framework

3.3.1. Overview

This section describes the Performance Framework (PF) developed. We first describe the dimensions of the PF. Then, we use PF to classify the types of errors. Then, we calculate the severity of each type of error.

Performance in semantic data modeling can be measured in different ways including model completeness, accuracy and time to completion (Bock and Yager 2001, Gemino and Wand 2004, Hull and King 1987, Topi and Ramesh 2004, Wand and Weber 2002). Time, as a measure of performance, is irrelevant since the objective of this research understanding expertise-related errors. Therefore, we evaluate performance in terms of completeness and accuracy: the number of errors in the semantic data schema. The relationship between performance and errors is inversely proportional: the fewer the errors, the better the performance.

3.3.2. Designing Performance Framework

Figure 5 provides an overview of PF. The errors are classified based on two dimensions: Semantic Quality and Representation. The Semantic Quality corresponds to a pre-existing framework to evaluate quality in semantic data modeling (Lindland, et al. 1994). The Representation dimension is added to provide a finer granularity to the types of errors and enable more accurate predictions based on the designer's position in the MEF. The two dimensions of the framework, Semantic Quality and Representation, are described in the next two subsections.

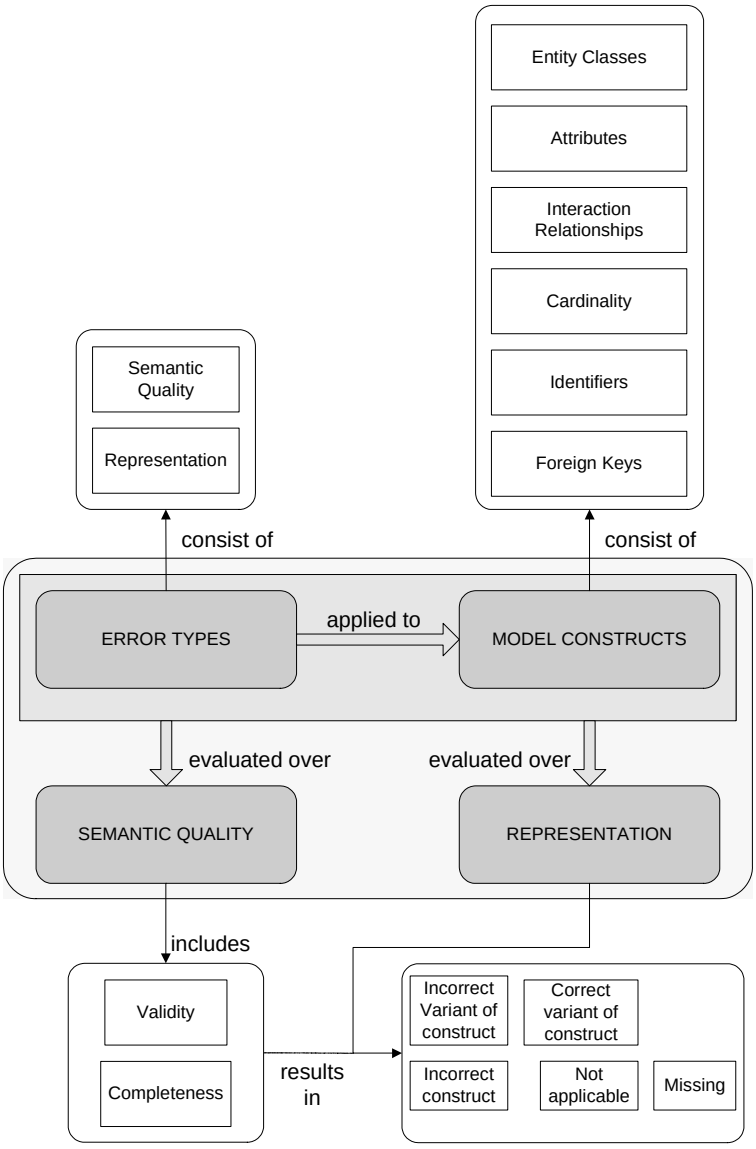

Figure 5: Overview of Performance Framework

3.3.2.1. Semantic Quality

There are three kinds of quality that can be evaluated for semantic data schemas: syntactic, semantic and pragmatic (Lindland, et al. 1994, Moody and Shanks 1994). Syntactic quality is the degree to which a model conforms to the modeling language's grammar rules. Semantic quality is the degree to which the model accurately captures

user requirements. Syntactic quality can be ensured by keeping a reference sheet listing the modeling grammar rules or by using a parser. Pragmatic quality is the degree to which all stakeholders understand the model. This research evaluates semantic quality of models developed. Pragmatic quality is only meaningful if the semantic quality is high. If the model does not capture the user requirements, the clarity of expression by the stakeholders is irrelevant. Therefore, first the designer should master creating models of high semantic quality with one audience, i.e., the database / system designer or implementer. Once she can create models with high semantic quality, then the pragmatic quality can be improved on. Semantic quality has two dimensions: Validity and Completeness (Lindland, et al. 1994). A valid schema implies that the statements made in the schema are correct and relevant to capturing the user requirements; i.e., the ER diagram does not have anything that is not in the user requirements. A complete schema implies that the schema correctly and relevantly captures all the user requirements; i.e., the ER diagram doesn't miss anything from the user requirements. To enable higher precision in predicting errors, both expertise and performance are examined at a finer granularity. The unit of analysis is a modular component of the requirements that that can be converted to an equivalent construct. For each modular component of the requirements, the designer could have either captured the information correctly (correct), could have missed capturing the information (missing), or captured the requirements incorrectly (incorrect). In addition, the designer could have captured something that was not in the user requirements (extra).

	Terminology used in previous research (Lindland, et al. 1994)	Terminology used in this research
Semantic Quality Equivalence	Complete and Valid	Correct
	Not complete	Missing
	Not valid, but complete	Extra
	Not valid and not complete	Incorrect

Table 18: Equivalence in notations for Semantic Quality

3.3.2.2. Representation

The representation dimension is added to provide a finer granularity of the relation between a correct solution and the designer's solution for the same modular component of the requirements. The possible values that representation can take for a modular component of the requirements depend on the Semantic Quality. If the designer's solution is correct for a particular component, the only valid value for Representation is "correct variant of semantic data modeling construct". If the solution for a particular component is missing, then the representation can take "missing". If the solution for a component is incorrect, then the valid value for representation is either "incorrect variant of semantic data modeling construct" or "incorrect semantic data modeling construct". Within the incorrect variant of semantic data modeling construct category, there are a number of subcategories that detail the representation of the error for the given situation. If the solution contains an extra semantic data modeling construct, i.e., the semantic data modeling construct is not present in the requirements or the semantic data modeling construct corresponds to irrelevant information in the requirements, then the valid value for the representation is "Not applicable".

Semantic Quality	Representation
Correct	Correct variant of semantic data modeling construct
Missing	Missing, No corresponding semantic data modeling construct
Extra	Not applicable
Incorrect	Incorrect modeling construct or Incorrect variant of modeling construct

Table 19: Valid Values for Representation for Different Values for Semantic Quality

3.3.3. Resulting Performance Framework

The previous section (section 3.3.2) described the overview of PF. This section describes the different levels of PF. The types of errors that novices make were developed based on discussions with various experts in the fields. The errors are then classified using the PF. Therefore, errors for each semantic data modeling construct are classified into missing, incorrectly captured or extra. The incorrectly captured errors are then further classified according to the representation dimension based on surface characteristics of the errors.

The framework is designed to be comprehensive (all major errors are included), and unambiguous (an error can be placed in exactly one category). The errors are broadly categorized based on the data modeling construct and semantic quality (missing, incorrect and extra). Errors within the incorrect dimension of semantic quality are further categorized based on representation. The categories errors are listed below

Semantic Quality	Representation	Explanation
Missing		A strong entity class in the requirements is not captured in the semantic data schema
Incorrect	Marked as weak:	A strong entity class is regarded as dependent on other entity classes for its existence and marked as weak
	Combined.	The value of separating entity classes is missed and multiple distinct strong entity classes are combined
Extra		A strong entity class that was not described in the requirements is captured in the semantic data schema

Table 20: Categories of Strong Entity Class Errors

Semantic Quality	Representation (level 1)	Representation (level 2)	Explanation
Missing			A relationship described in the requirements is not captured anywhere in the semantic data schema
Incorrect	Relationship errors affecting degree: Marked as an attribute		A relationship described in the requirements is not captured in the semantic data schema. Instead, the identifier of one of the entity classes is placed on another to attempt to capture the relationship. This error is similar to the 'Strong entity classes combined' error since it also involves incorrectly associating attributes with an entity class
	Relationship errors affecting degree: ternary as a binary:	Involving all strong entity classes	A ternary relationship among strong entity classes is incorrectly marked as a binary relationship
		Involving at least one weak entity class	One of the necessary entity classes for the relationship is missing from a ternary relationship involving at least one weak entity class
	Relationship errors affecting degree: binary as ternary	Entity class added is redundant	An extra, redundant entity, like a subclass, superclass or weak entity class is added to the relationship
		Entity class added is irrelevant, but not redundant	An extra, irrelevant entity class is added to the relationship

Incorrect	Relationship errors not affecting degree	Relationships combined:	Although the degree of the relationship is correct, the semantics are incorrect. This error would result in combining one or more relationships among the same set of entity classes, as well as weak entity classes associated with the relationships.
		Substituted entity class that shares a identifier	One of the participating entity classes in the relationship is replaced by an associated superclass or subclass or weak entity class
		Substituted entity class with an irrelevant entity class	One of the participating entity classes is replaced by an entity class that is irrelevant to the relationship, based on the requirements
Extra		A relationship that was not described in the requirements is captured in the semantic data schema	

Table 21: Categories of Relationship Errors

Semantic Quality	Explanation
Missing	A attribute described in the requirements is not captured anywhere in the semantic data schema
Extra	Extra information that was not stated in the requirements is captured

Table 22: Categories of Attribute Errors

Semantic Quality	Representation	Explanation
Missing		A weak entity class described in the requirements is not captured anywhere in the semantic data schema
Incorrect	Marked as attribute(s)	The attributes of the weak entity class are associated with one of the participating strong entity classes instead of the relationship. This error is similar to the 'Strong entity classes combined' and 'Relationship as an attribute' errors since all of those errors involve incorrectly associating attributes with an entity class
	Marked as strong	The weak entity class is mistakenly regarded as independent of other entity classes and is marked strong
	Marked as subclass.	A weak entity class is marked as a subclass. Although weak entity classes and subclasses "borrow" the identifier from another entity class (similar representation), the semantics of the two relationships are very different. Therefore, one cannot be substituted for the other
Extra		A weak entity class that was not described in the requirements is captured in the semantic data schema

Table 23: Categories of Weak Entity Class Errors

Semantic Quality	Explanation
Missing	Cardinality is not marked
Incorrect	The semantics for the cardinality is ambiguous / incorrect

Table 24: Categories of Cardinality Errors

Semantic Quality	Explanation
Missing	Identifier is not marked
Incorrect	The semantics for the identifier is incorrect

Table 25: Categories of Identifier Errors

The next section will describe, with examples, the significance of each error. Depending on the significance of the errors, the order of elimination of errors can be determined.

3.3.4. Cost of Errors

3.3.4.1. Overview

In this section, the cost for each type of error is estimated. The objective of calculating the cost of errors is to decide the priority of eliminating that error. We use a systematic approach to calculate the cost of each error type. First, we identify the criteria of good database design. Then, we evaluate the degree to which each error violates each criterion. Based on the evaluation, we classify the cost for each error as low, medium or high.

The criteria for a good database design include that the design: (a) accurately represents the real-world structure; (b) avoids redundant storage of data items; (c) provides efficient access to data; and (d) supports the maintenance of data integrity over time (Batini, et al. 1992, Hoffer, et al. 2006, Moody 1998, Moody and Shanks 1994). These evaluation criteria are used to estimate the seriousness of each error. Since the above criteria are specified at the logical stage of database design, we convert the conceptual schema to a logical schema before evaluating the cost.

3.3.4.2. Approach

To evaluate the implications of each error, a mini-case in the real estate domain is developed. To assess the cost of an error, we compare logical schemas corresponding to

(a) the "correct" conceptual diagram and (b) a conceptual diagram that contains the error. Then, we calculate the significance of the error based on the above criteria.

As can be seen in the conceptual schema, the design captures information on Buyers, Sellers, Realtors, Properties and the relationships among them. In addition, the design also captures additional information that supports the buying and selling processes like Appraisals, Public Schools, Communities, and Promotion Events.

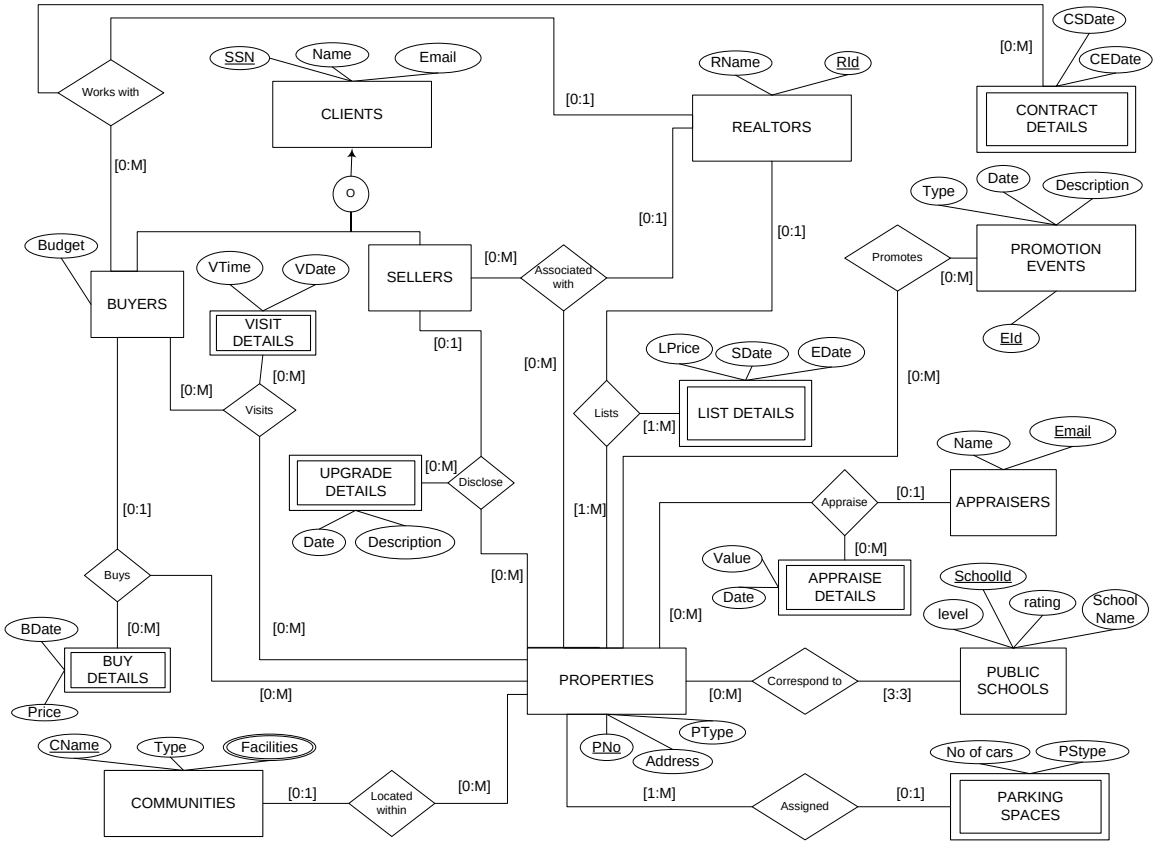

Figure 6: Correct ER Schema

The logical design corresponding to the above conceptual schema is shown below.

Properties(PNo, Address, PType, PSnoOfCars, PSType, *CName*)

Foreign key (CName) references Communities(CName)

Buyers(BSSN, BName, BEmail, Budget)

Sellers(SSSN, SName, SEmail)

Realtors(RId, RName)

Appraisers(AEmail, AName)

Promotion_Events(EId, Type, Date, Description)

Communities(CName, Type)

Public_Schools(SchoolId, SchoolName, level, rating)

Community_Facilities(*CName*, Facilities)

Foreign key (CName) references Communities(CName)

Buys(*BSSN*, *PNo*, BDate, Price)

Foreign key (BSSN) references Buyers(BSSN)

Foreign key (PNo) references Properties(PNo)

Visits(*BSSN*, *PNo*, VDate, VTime)

Foreign key (BSSN) references Buyers(BSSN)

Foreign key (PNo) references Properties(PNo)

List_Details(*RId*, *PNo*, LPrice, SDate, EDate)

Foreign key (RId) references Realtors(RId)

Foreign key (PNo) references Properties(PNo)

Contract_Details(*BSSN*, *RId*, CSDate, CEDate)

Foreign key (BSSN) references Buyers(BSSN)

Foreign key (RId) references Realtors(RId)

Appraise_Details(*AEmail*, *PNo*, Value, Date)

Foreign key (AEmail) references Appraisers(AEmail)

Foreign key (PNo) references Properties(PNo)

Associated_With(*SSSN, RId, PNo*)

Foreign key (SSSN) references Sellers(SSSN)

Foreign key (RId) references Realtors(RId)

Foreign key (PNo) references Properties(PNo)

SchoolsCorrespondToProperties(*PNo, SchoolId1, SchoolId2, SchoolId3*)

Foreign key (PNo) references Properties(PNo)

Foreign key (SchoolId1) references Public_Schools (SchoolId)

Foreign key (SchoolId2) references Public_Schools (SchoolId)

Foreign key (SchoolId3) references Public_Schools (SchoolId)

Upgrade_details(*PNo, SSSN*, Date, Description)

Foreign key (PNo) references Properties(PNo)

Foreign key (SSSN) references Sellers(SSSN)

Promotes(*EId, PNo*)

Foreign key (EId) references Promotion_Events (EId)

Foreign key (PNo) references Properties(PNo)

Table 26: Logical Design Corresponding To Correct ER Schema

Based on PF (section 3.3.3), errors are classified, based on the modeling construct that is incorrectly modeled, into strong entity class errors, relationship errors, weak entity class errors and cardinality errors. Our mini-case includes examples of each error in the PF and we use the mini-case to evaluate the cost of the error.

3.3.4.3. Strong entity class errors

As described previously, there are three major categories of strong entity class errors: missing, incorrectly modeled, and extra. The following subsections illustrate, with examples, the impact of each category of strong entity class errors.

Missing Strong entity class

To illustrate Missing strong entity class error, the *Buyers* entity class is deleted. If the designer does not see the relevance of *Buyers*, information about buyer activities like visiting properties, buying properties and the associating with realtors will appear irrelevant. This design results in an incomplete representation of the real world.

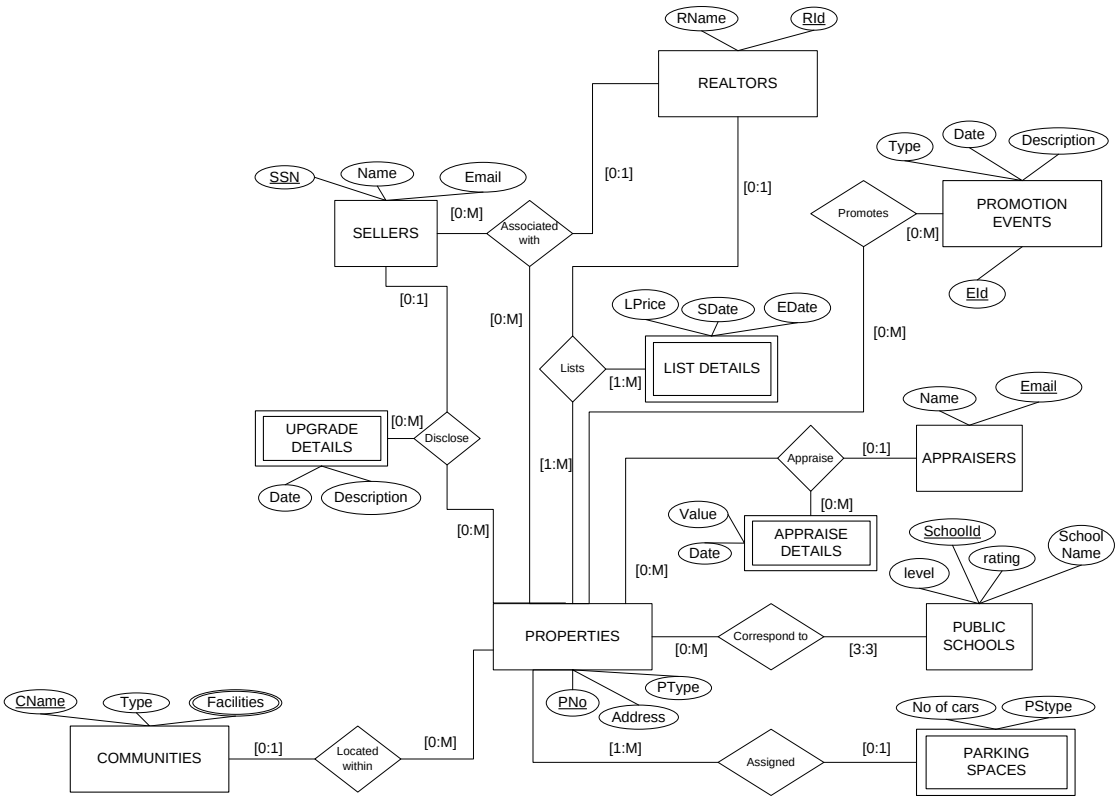

Figure 7: Illustrating Missing Strong Entity Class Error

To evaluate the impact of the missing strong entity class error more concretely, the conceptual design with the missing strong entity class is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two

designs are shown below (Table 27). The logical design shows that the correct design has four tables not present in the incorrect design. The error of missing information at the conceptual design stage is not detected and corrected at the logical design stage. Using the evaluation criteria, the design does not accurately represent the real-world structure for Buyers and hence does not support efficient access to data about Buyers. In addition, it cannot provide efficient access to the data (since it does not exist). Thus, the cost of missing entity class is high.

Correct design	**Missing strong entity class**
Buyers(BSSN, BName, BEmail, Budget) Buys(*BSSN, PNo*, BDate, Price) Visits(*BSSN*, *PNo*, VDate, VTime) Contract_Details(*BSSN, RId*, CSDate, CEDate)	*no table because associated strong entity class is missing*

Table 27: Logical Design comparison for missing strong entity class

Incorrectly Modeled Strong Entity Class

Within incorrectly modeled strong entity class, there are two types of errors based on the representation of the error: Strong entity class marked as weak and multiple strong entity classes combined.

Strong Entity Class Marked As Weak

To illustrate incorrectly modeled strong entity class: strong entity class marked as weak error, Public Schools is marked weak. By marking it weak, the designer implies that information about Public Schools is not relevant unless it is associated with a property. So, it affects the minimum cardinality for the *Correspond to* relationship slightly. Based on these two observations, it appears that the two problems caused by this

error are not accurately representing the real world structure and inability to enforce data integrity constraints over time.

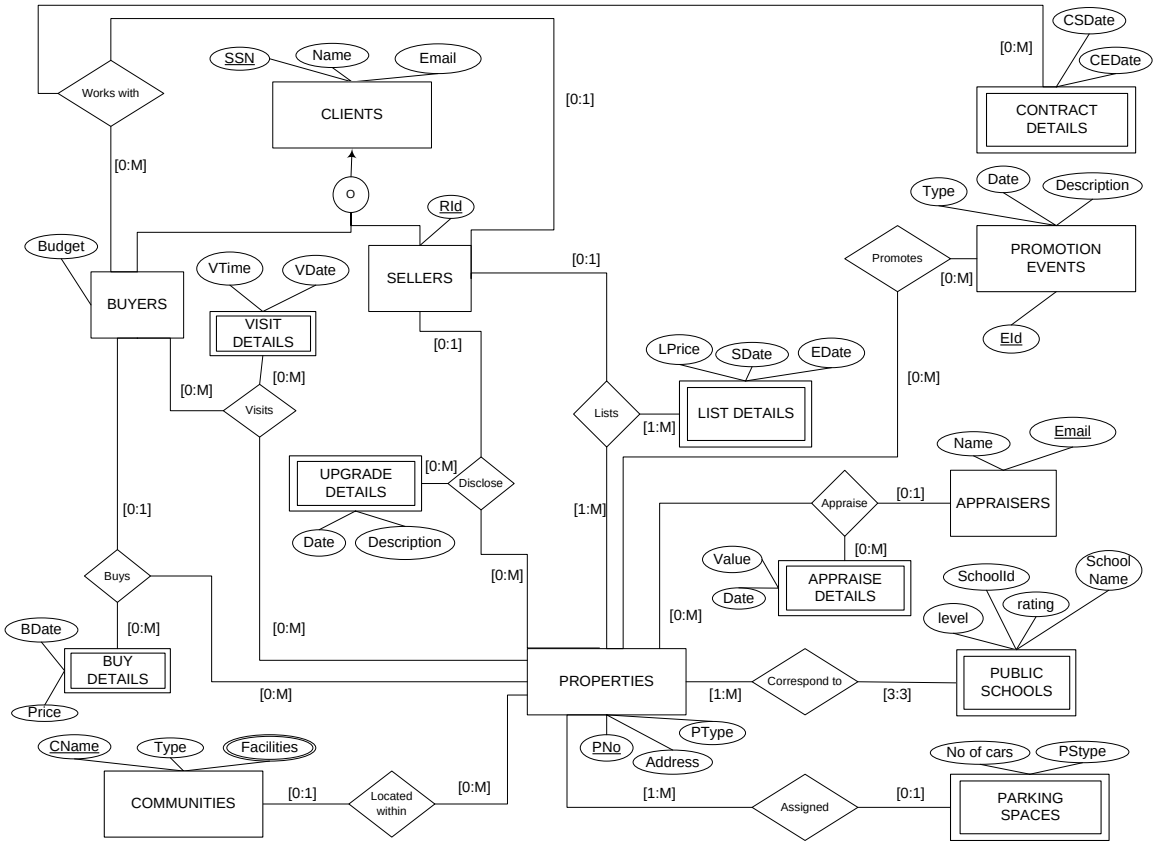

Figure 8: Illustrating Incorrectly Modeled Strong Entity Class Error

To evaluate the impact of the error of marking a strong entity class as a weak entity class more concretely, the conceptual design with the error is converted to a logical design. Each property is associated with three schools—an elementary school, a middle school and a high school. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 28).

Correct design	Strong entity class as weak
Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*) Public_Schools(SchoolId, SchoolName, level, rating) SchoolsCorrespondToProperties (*PNo, SchoolId1, SchoolId2, SchoolId3*)	Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*, SchoolId1, SchoolName1, level1, rating1, SchoolId2, SchoolName2, level2, rating2, SchoolId3, SchoolName3, level3, rating3)

Table 28: Logical Design Comparison For Illustrating Strong Entity Class Marked As Weak Error

The correct design has three tables while the incorrect design has the same fields, but it combines the three tables into one. This results in all four evaluation criteria being violated. By not correctly capturing the real world structure of Public Schools as a strong entity class, it results in many problems. For example, it results in redundant storage of information. For example, let us assume that the school with SchoolId 2 corresponds to Sahuaro High School, at the high school level and a rating of Good. Let us assume that there are two properties in the area on the market. In the first (correct) design, only the SchoolId would be repeated for both schools. In the second (incorrect: strong entity class as weak) design, the SchoolId, SchoolName, level and rating would be repeated for both schools. This would result in the storage of a lot of redundant information. If a potential buyer wanted to list which schools had a Good rating at the High school level, he could retrieve it using a simple query in the correct design. However, in the incorrect design, first a list of distinct schools and their ratings at the high school level would have to be retrieved before the results could be filtered based on the value of the rating. This makes the retrieval of information inefficient. Another problem would be the maintenance of

data integrity. Every time a new property is inserted, it needs to be associated with each of the following: an elementary school, a middle school and a high school. In the correct design, this only involves identifying the appropriate SchoolId. However, in the incorrect design, this involves typing in the SchoolId, the SchoolName, the level and the rating. Any errors in entering any of the school information will be hard to detect. This would result in problems in maintaining data integrity across records. Thus, the cost of marking a strong entity class as weak is high.

Strong Entity Classes Combined

To illustrate incorrectly modeled strong entity class: multiple strong entity classes combined error, sellers and realtors are combined. By combining two entity classes, the designer implies that there is no real distinction between the information captured by each. In this example, it implies that the realtor sells the property, so it is not important to separate the realtor from the seller. To illustrate this understanding, any relationships in the correct diagram linking the realtor and the seller will be lost. Therefore, the main problem caused by this error seems to be not being able to accurately represent the relationship between a realtor and a seller.

To evaluate the impact of the error of combining strong entity classes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 29). There are two tables that are missing in the Strong Entity Classes Combined design. The Realtors table is missing and the Associated with table is

missing because it was redundant at the conceptual modeling phase if Realtors is not a distinct entity class. The other difference is the content of the Sellers table: the seller name and email actually represent the realtor's name and email in the second design. Also, if the same realtor sold the property at two different times for different sellers, it is hard to distinguish what upgrades each seller made. It is important to identify the upgrades made by the most recent seller to understand some of the variances in the price. For example, the price of the house would be different if the current seller bought the house a year ago and added a new room to the house, vs. if the previous seller had added a new room and the current seller did nothing to the house since he bought it a year ago.

Correct design	**Strong Entity Classes Combined**
Sellers(SSSN, SName, SEmail) Realtors(RId, RName) Associated_With(*SSSN, RId, PNo*) Upgrade_details(*PNo, SSSN*, Date, Description)	Sellers(RId, SName, SEmail) Upgrade_details(*PNo, RId*, Date, Description)

Table 29: Logical Design Comparison For Illustrating Strong Entity Classes Combined Error

Thus, the major problems with combining strong entity classes are that information is lost and incorrect information is captured. Therefore, when strong entity classes are combined, the design does not capture the real world structure and the design does not provide efficient access to data because the underlying data is captured incorrectly. Thus, the cost of combining strong entity classes is high.

Extra strong entity class

To illustrate Extra entity class, the miniworld (Arizona Realtors) and branch information are included. The resulting conceptual schema is shown below. By adding the miniworld as an entity class, the designer does not realize the scope of the design. This would have the ripple effect of connecting the extra miniworld entity class to all the existing entity class because the miniworld / system / company is connected to everything within it by a Contained Within type of relationship. Including information about branches extends the scope of the initial design because it assumes that there are multiple branches each with multiple realtors. If there are multiple branches, a realtor would want to find out who works at another branch to refer a client moving to a new location.

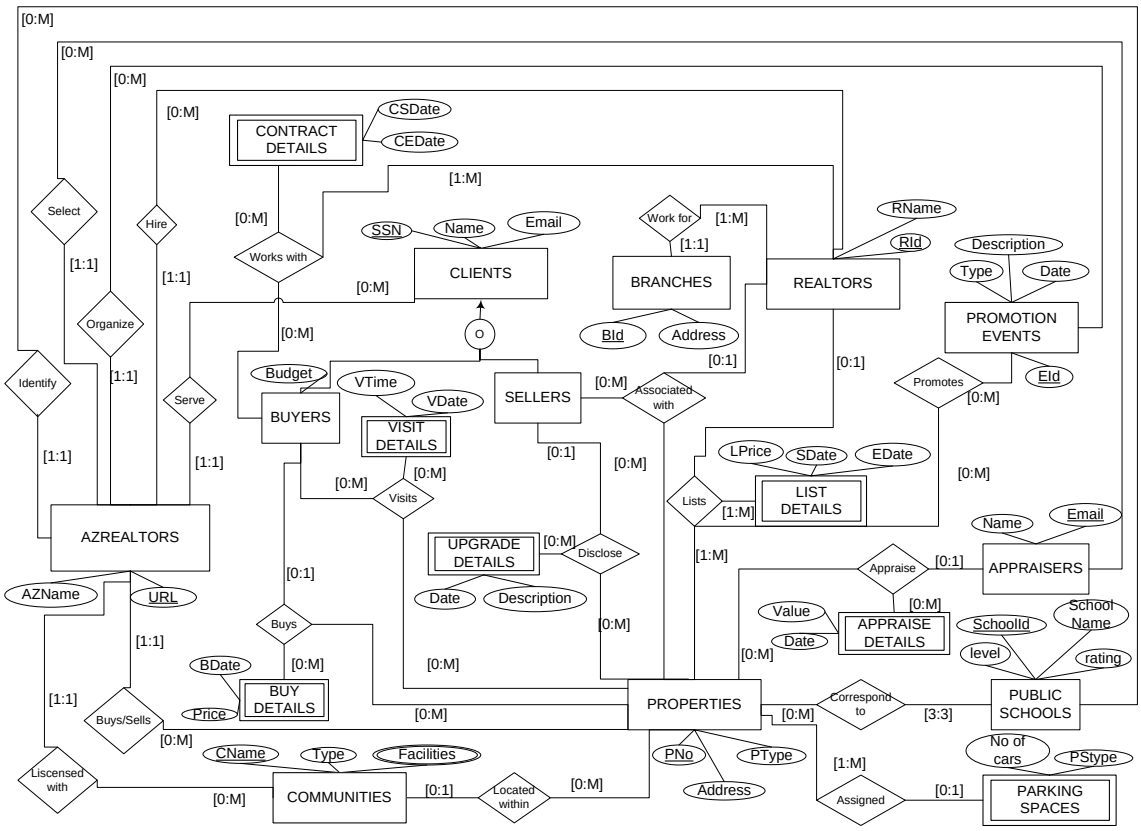

Figure 9: Illustrating Extra Strong Entity Class Error

Therefore, the main problem caused by this error is extra information being represented. To evaluate the impact of the error of extra strong entity classes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 30).

Correct design	**Extra Strong entity class**
Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*) Buyers(BSSN, BName, BEmail, Budget) Sellers(SSSN, SName, SEmail) Realtors(RId, RName) Appraisers(AEmail, AName) Promotion_Events(EId, Type, Date, Description) Communities(CName, Type) Public_Schools(SchoolId, SchoolName, level, rating)	AZRealtors(URL, AZName) Branches(BId, BAddress) Properties(PNo, Address, PType, PSNo_cars, PSType, CName, *URL*) Buyers(BSSN, BName, BEmail, Budget, *URL*) Sellers(SSSN, SName, SEmail, *URL*) Realtors(RId, RName, *URL*, BId) Appraisers(AEmail, AName, *URL*) Promotion_Events(EId, Type, Date, Description, *URL*) Communities(CName, Type, *URL*) Public_Schools(SchoolId, SchoolName, level, rating, *URL*)

Table 30: Logical Design Comparison For Illustrating Extra Strong Entity Class Error

There are two tables that are extra in the Extra Strong Entity class design: The AZRealtors table and the Branches table. In addition, all tables corresponding to strong entity classes, i.e., Properties, Buyers, Sellers, Realtors, Public Schools, Communities, Promotion Events and Appraisers have an extra attribute URL which links each of the

tables to the AZRealtors table. The Realtors table has an extra attribute BId which links Realtors to Branches. Having the same value of URL repeated for all those entity classes is redundant, serves no purpose and just takes up space. Based on our evaluation criteria, the incorrect design has a lot of redundant storage of items, i.e., the design is not valid since it contains information not in user requirements. It does not affect the efficiency with which the data can be accessed and does not affect the integrity of data. So, the seriousness of the error will be determined by how much redundant data is stored. In the case of Branches, since there is only one Branch, and it only affects the relationship with realtors, the amount of redundancy is low. However, in the case of the miniworld, it adds a redundant field to a number of strong entity classes. Therefore, there is a lot of redundancy. In addition in case of the miniworld, it is a misunderstanding of the semantics of what an entity class represents because the designer did not understand the scope of the application.

Thus, the cost of an extra entity class varies: it can be low when it is based on an additional constraint imposed by the designer (in the case of branches) or it can be high when it affects a large number of tables (in the case of miniworld).

Summary

The costs of the errors are summarized below (Table 31). The errors are grouped by major types of errors (missing, incorrect and extra)

	Low Cost	Medium Cost	High Cost
Missing			Missing
Incorrect			Strong as Weak Combine Strong Entity classes
Extra	Extra (if not miniworld)		Extra (if miniworld)

Table 31: Cost of Different Strong Entity Class Errors

3.3.4.4. Relationship errors

Errors in relationships have effects on decisions that can be made. However, unlike errors with entity classes, relationship errors do not have a ripple effect on other strong entity classes and relationships. There are three major categories of relationship errors based on validity and completeness: missing relationships, incorrectly modeled relationships, extra relationships. The following subsections illustrate with examples the impact of each of the subcategories of relationship errors.

Missing Relationship

To illustrate missing relationship, the Lists relationship is deleted. The conceptual schema for the two different representations of missing relationship error is shown below (Figure 10).

The main problem caused by this error is missing information. To evaluate the impact of the error of totally missing a relationship more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 32).

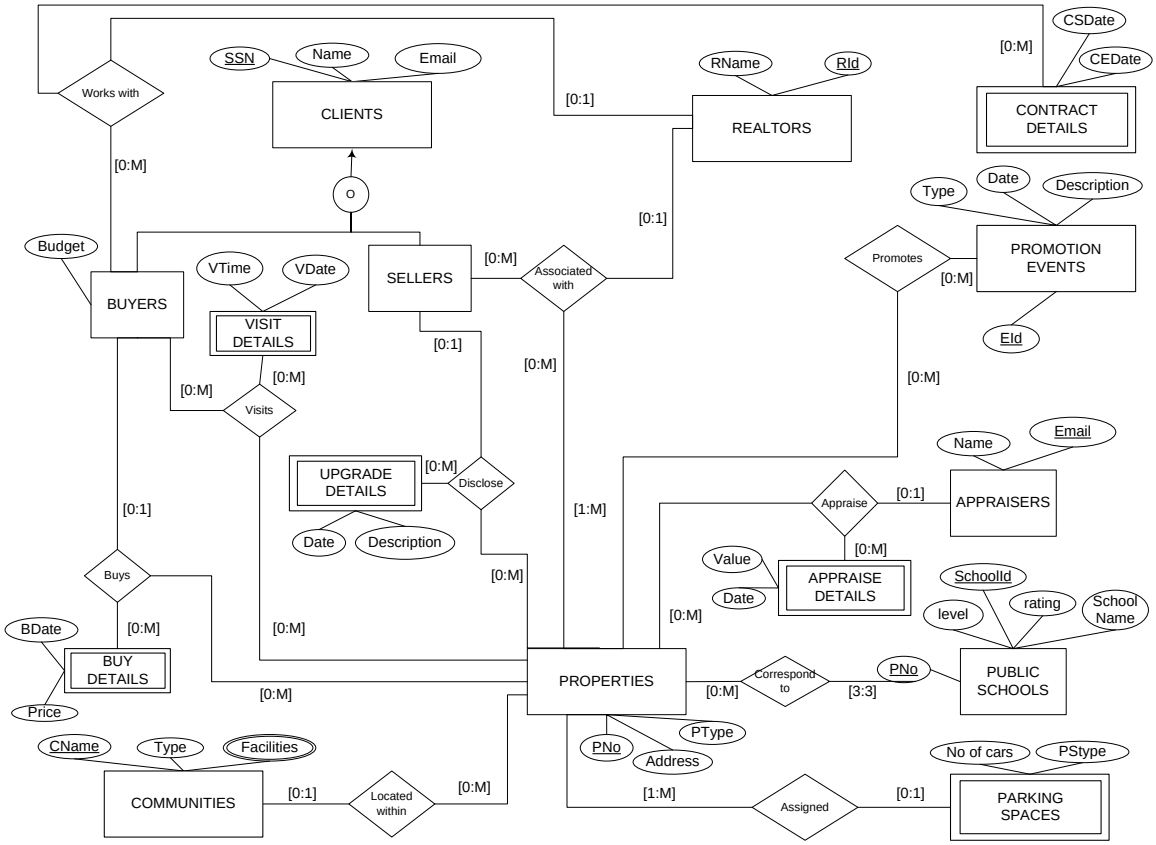

Figure 10: Illustrating Missing Relationship Error

Correct design	**Relationship Totally Missing**
List_Details(*RId*, *PNo*, LPrice, SDate, EDate)	*no table because associated relationship is missing*

Table 32: Logical Design Comparison for Illustrating Missing Relationship Error

The correct logical design has a List Details table that the incorrect design does not. Based on the evaluation criteria, the design does not accurately represent the real-world structure for Listings and hence does not support efficient access to properties currently on the market. This results in losing information about the start and end date for the listing and the price that the property was listed at. It also results in losing information

about the listing realtor for the property. Therefore, there is no way for prospective buyers to know whether they would be interested in the house, i.e., there is no way to answer questions like is the property still available and is the price within their budget? Information about the earliest start date provides indications on how desirable the property is; for example, if it has not sold for over two months, there may be some problems with the property. Information about the end date provides indications on whether the property is still available. If the end date has passed, the seller may have taken the property off the market or may be in negotiations with a potential buyer. Therefore, missing a relationship, though it results in missing just one table, has significant implications. Thus, the cost of missing a relationship is high.

Incorrectly Modeled Relationship

Within incorrectly modeled relationship, there are several types of errors based on the representation of the error.

Relationship As An Attribute

To illustrate a relationship represented as an attribute, the Corresponds to relationship between Public Schools and Properties is deleted. Instead, the primary key of properties (PNo) is added to Public Schools to attempt to capture the association. The conceptual schema for the relationship represented as an attribute is shown above (Figure 11).

The main problem caused by this error is missing information. To evaluate the impact of the error of representing a relationship as an attribute more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any

errors (Table 26). Only the differences between the two designs are shown below (Table 33).

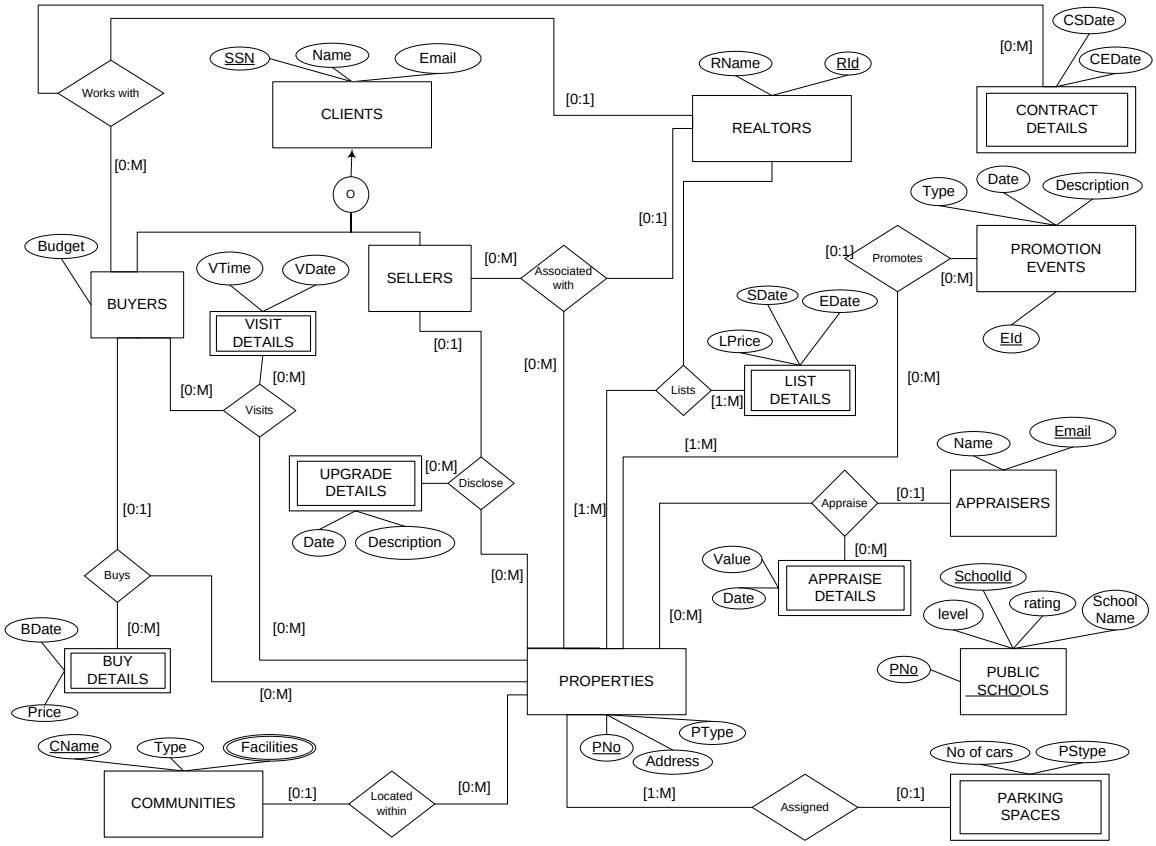

Figure 11: Illustrating Relationship As An Attribute Error

Correct design	Relationship As An Attribute
SchoolsCorrespondToProperties(*PNo*, *SchoolId1*, *SchoolId2*, *SchoolId3*) Public_Schools(SchoolId, SchoolName, level, rating)	Public_Schools(SchoolId, SchoolName, level, rating, *PNo*)

Table 33: Logical Design Comparison For Illustrating Relationship As An Attribute Error

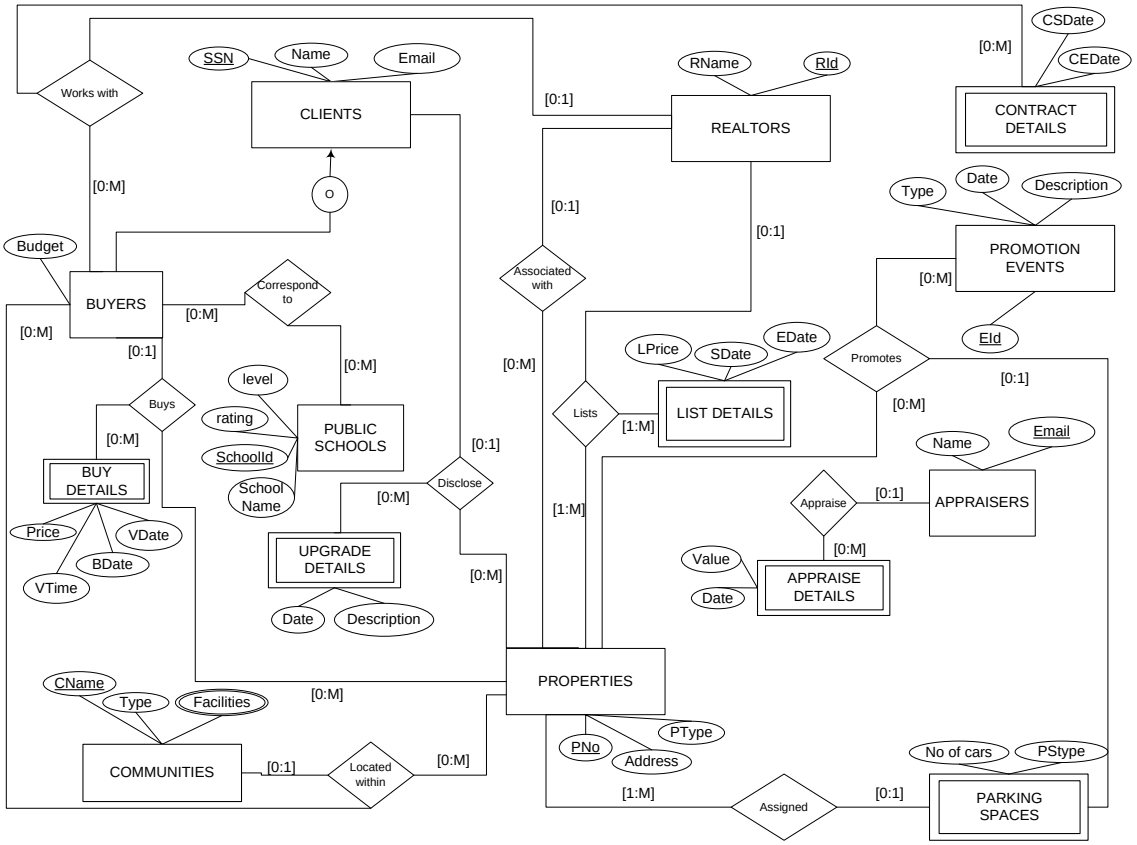

Figure 12: Illustrating Incorrectly Modeled Relationship Errors

The correct logical design has a SchoolsCorrespondToProperties table while the incorrect design doe not. Instead, the incorrect design includes a PNo field within the Public_Schools table. Based on the evaluation criteria, the design results in redundant storage of items and does not support the data integrity over time. The Public_Schools table is not in second normal form because SchoolName, level and rating depend only on SchoolId. If the designer had chosen PNo as the primary key instead of SchoolId and PNo, the table would not be in third normal form because of transitive dependencies. One of the problems of the incorrect design is that SchoolName, level and rating is redundantly stored for every Property. Another implication of the table not being in at

least third normal form is that there will be insertion, updation and deletion anomalies. This will result in data integrity problems. Thus, the cost of representing a relationship as an attribute is high.

Errors affecting degree: Ternary as a Binary

Two types of Ternary as binary errors are illustrated: one where all the entity classes involved are strong and the second where there are two strong and one weak entity class. The two errors are distinguished because the expertise levels predicting them are different.

Ternary as a Binary (Not Involving Weak Entity Classes)

To illustrate ternary relationship with all strong entity classes as a binary relationship, the Associated with relationship is modified to be between Realtors and Properties instead of between Sellers, Realtors and Properties. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is the inability to link Sellers with Realtors for the Properties they are selling. To evaluate the impact of the error of incorrectly capturing the ternary relationship as a binary more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 34).

The correct logical design has SSSN linking the Sellers table to Properties and Realtors table that the incorrect design only links Realtors and Properties. Based on the evaluation criteria, the design does not accurately represent the real-world structure for

the association and hence does not support efficient access to who is selling the property or who a seller has chosen as a realtor. This results in losing information about the link between the seller and the properties being sold. Without this link, it is impossible to determine the seller for a property and which realtor was chosen. Therefore, marking a ternary as a binary, though it results in missing one attribute in one table, has implications. Thus, the cost of ternary as a binary (involving all strong entity classes) relationship is medium.

Correct design	**Ternary as a binary (Not Involving Weak Entity Classes) Relationship**
Associated_With(*SSSN, RId, PNo*)	Associated_With(*RId, PNo*)

Table 34: Logical Design Comparison For Illustrating Ternary As A Binary Relationship Error (Not Involving Weak Entity Classes)

Ternary as a Binary (Involving Weak Entity Classes)

To illustrate ternary relationship with all strong entity classes as a binary relationship, the Appraise relationship is modified to be between Appraisers and Appraise Details instead of between Appraisers, Properties and Appraise Details. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is the inability to link Properties with Appraiser and Appraise details. To evaluate the impact of the error of incorrectly capturing the ternary relationship as a binary more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 35).

Correct design	Ternary as a binary (Involving Weak Entity Classes) Relationship
Appraise_Details(*AEmail, PNo*, Value, Date)	Appraise_Details(*AEmail*, Value, Date)

Table 35: Logical Design Comparison For Illustrating Ternary As A Binary Relationship Error (Involving Weak Entity Classes)

The correct logical design has PNo linking the Properties to Appraisers and identifying the value while the incorrect design only links Appraisers to the value of the appraisal. Based on the evaluation criteria, the design does not accurately represent the real-world structure for the appraisal and hence does not support efficient access to what property the appraisal was for. This results in useless information capture. Knowing the range of values that the appraiser came up with is useless without knowing the property under consideration. Without this link to properties, it is impossible to determine the property for which the appraisal was done. Therefore, marking a ternary as a binary, though it results in missing one attribute in one table, has implications. Thus, the cost of ternary as a binary (involving weak entity class) relationship is medium.

Errors affecting degree: Binary as a Ternary

Two types of binary as ternary errors are illustrated: one where the entity class added does not add any new information and the second where the entity class added changes the structure of the resulting relationship table.

Binary as a Ternary (Added Redundant Entity Classes)

To illustrate binary relationship as a ternary relationship, the Promotes relationship is modified to be between Promotion events and Properties is modified to include Parking

Spaces. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

To evaluate the impact of the error of incorrectly capturing the binary relationship as a ternary more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 36). There are no differences because the Parking Spaces entity class gets merged with the Properties entity class into the Properties table. If a separate table was created for the weak entity class, it may have resulted in an extra attribute in the table for the relationship.

Correct design	**Binary as ternary (Added Redundant Entity classes) Relationship**
no tables because the logical designs are equivalent	

Table 36: Logical Design Comparison For Illustrating Binary As A Ternary Relationship Error (Added Redundant Entity Classes)

Based on the evaluation criteria, the design does not result in any problems at the logical design in this case. However, it could result in extra attributes in some cases. Thus, the cost of marking a binary as a ternary (including redundant entity classes) is low.

Binary as a Ternary (Added Other Entity Class)

To illustrate binary relationship as a ternary relationship when irrelevant entity classes are added, the Located within relationship between Properties and Communities is

modified to include Buyers. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is the redundant storage of Buyer information for each Located within entry. A property will be located within a particular community irrespective of who buys the property. A buyer might choose a particular property because of the community it is within, but the buyer cannot select the community and the property independently. To evaluate the impact of the error of incorrectly capturing the binary relationship as a ternary more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 37).

Correct design	**Ternary as a binary (Not Involving Weak Entity Classes) Relationship**
Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*)	Properties(PNo, Address, PType, PSNo_cars, PSType) Located_Within(*PNo, CName, BSSN*)

Table 37: Logical Design Comparison For Illustrating Binary As A Ternary Relationship Error (Adding Other Entity Classes)

The correct logical design captures the Located within relationship by extending the Properties table to include CName. The incorrect design captures the Located within relationship with a separate table that includes PNo, CName and BSSN, i.e., it links Properties, Communities and Buyers. Based on the evaluation criteria, the design does not accurately represent the real-world structure for the Located Within table and hence results in redundant storage of data and does not support efficient access of connecting

communities to properties. Since joins is the most expensive operation in a database, and the incorrect design results in joining 3 tables instead of combining two tables into one, it will be inefficient. Therefore, marking a binary as a ternary, though it results in a couple of extra attributes in one table, has implications for redundancy and efficiency. Thus, the cost of marking a binary as a ternary relationship is medium.

Errors not affecting degree

Three types of errors not affecting degree are illustrated: one where multiple relationships among the same set of entity classes are combined, one where one of the entity classes in a relationship is substituted with another entity class that shares the same primary key and one where the entity class substituted has no obvious link with the one removed.

Combining Relationships Among the Same Set of Entity Classes

To illustrate the combining relationships among the same set of entity classes error, the Buys and the Visits relationships between Buyers and Properties were combined. This resulted in the weak entity class coming out of the relationship having the attributes of Buys and Properties. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is that it does not accurately represent the real world structure. It would result in constraining the values of some of the attributes in the weak entity class representing the buys and visits information. To evaluate the impact of the error of incorrectly capturing the binary relationship as a ternary more concretely, the conceptual design with the error is converted to a logical design. This logical design is

then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 38).

Since the cardinality of the relationship is 1:M:M, the primary key for the table corresponding to the weak entity class buys will be PNo together with the partial key for the weak entity class. If the partial key chosen for the combined relationship was BDate, i.e., the partial key for Buys, the resulting table would be restrictive. The combined relationship table would be able to store one visit instance for each purchase, i.e., the date the buyer who bought the property visited it. This would provide information about the time to convert a visit into a sale which would provide information on how quickly properties are selling. The combined relationship structure loses information on what properties a buyer visited, and when. However, it would not achieve the intended purpose of improving recommendations.

Correct design	**Combining Relationships Among the Same Set of Entity Classes**
Buys(*BSSN, PNo*, BDate, Price) Visits(*BSSN, PNo*, VDate, VTime)	Buys(*BSSN, PNo*, BDate, Price, VDate, VTime)

Table 38: Logical Design Comparison For Illustrating Combining Relationships Among The Same Set Of Entity Classes Error

Therefore, combining relationships among the same set of entity classes, though it maintains the same set of attributes does not accurately represent the real world structure and loses a lot of information. Thus, the cost of combining relationships is high.

Substituting An Entity Class That Shares the Primary Key

To illustrate substituting an entity class with another that shares the primary key, the disclose relationship between Sellers and Properties is modified to be between Clients and Properties. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is the inability to constrain the people making the upgrades to sellers. The inability to enforce the constraint will allow the system to store disclosures from people other than the seller. To evaluate the impact of the error of incorrectly capturing the substituting an entity class that shares the primary key more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 39).

Correct design	**Substituting An Entity Class That Shares the Primary Key**
Buyers(BSSN, BName, BEmail, Budget) Sellers(SSSN, SName, SEmail) Upgrade_details(*PNo, SSSN*, Date, Description)	Buyers(*BSSN*, Budget) Clients(SSN, Name, Email) Upgrade_details(*PNo, SSN*, Date, Description)

Table 39: Logical Design Comparison For Illustrating Substituting An Entity Class That Shares The Primary Key Error

The correct logical design links the Properties and Sellers in the Upgrade_details table while the incorrect design links Clients with Properties in the Upgrade_details table. Also, in the incorrect design, there is no need to separate sellers from clients because sellers has no unique relationships or attributes. The incorrect design has less redundant

information because the name and email are not repeated for buyers who are also sellers. However, the incorrect design makes it impossible to identify sellers. Since the underlying data of who the sellers are is not captured, it cannot be retrieved. Based on the evaluation criteria, the design does not accurately represent the real-world structure for Sellers and the disclose relationship. This results inability to represent necessary information about sellers and it also results in inability to constrain who can participate in the disclose relationship. This results in information loss. Therefore, substituting an entity class that shares the same primary key has implications in losing critical information. Thus, the cost of substituting an entity class that shares the same primary key in a relationship is high.

Substituting An Entity Class With An Irrelevant Entity Class

To illustrate substituting an entity class in a relationship with an irrelevant entity class, the SchoolsCorrespondToProperties relationship is modified to be between Schools and Buyers instead of between Schools and Properties. The conceptual schema illustrating this incorrect relationship error is shown above (Figure 12).

The main problem caused by this error is the loss of real world structure. The correct relationship captures information about which schools are associated with a property. The incorrect relationship captures which schools the Buyers considers suitable. A Buyer may find multiple schools satisfactory and then choose a property that best satisfies other needs (budget, number of rooms, age of the house, etc.). However, if the information about the association between schools and properties is not captured, the buyer will not be able to narrow down the selection based on the schools. So, the two relationships

capture very different pieces of information. To evaluate the impact of the error of substituting an entity class in a relationship with an irrelevant entity class, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 40).

Correct design	**Substituting An Entity Class With An Irrelevant Entity Class**
SchoolsCorrespondToProperties(*PNo, SchoolId1, SchoolId2, SchoolId3*)	SchoolsBuyersInterestedIn (*BSSN, SchoolId*)

Table 40: Logical Design Comparison For Illustrating Substituting An Entity Class With An Irrelevant Entity Class Error

The correct logical design has the Properties and Public Schools tables linked through PNo and SchoolId. Based on the evaluation criteria, the incorrect design does not accurately represent the real-world structure for the linking between Schools and Properties. The relationship SchoolsBuyersInterestedIn captures a different piece of information. There is no way to extract which schools correspond to what properties from the relationship. This results in loss of information. If the design does not reflect the user needs and hence the real world structure, it is a bad design. Therefore, substituting an entity class with an irrelevant entity class, captures irrelevant information and has implications of losing information. Thus, the cost of substituting an entity class with an irrelevant entity class in a relationship is high.

Extra Relationship

To illustrate Extra relationship, the Recommend relationship between Realtors and Appraisers is included. The resulting conceptual schema is shown below. By adding the Recommend relationship, the designer is including information not stated in the requirements. This Recommended relationship will help track which appraisers were recommended by which realtor. Based on the relationship, it may be possible to infer which appraisers are more reliable: An appraiser recommended by multiple realtors is probably more reliable than one who has not been recommended at all. However, there is a fine line between feature creep and including information based on knowledge of the domain.

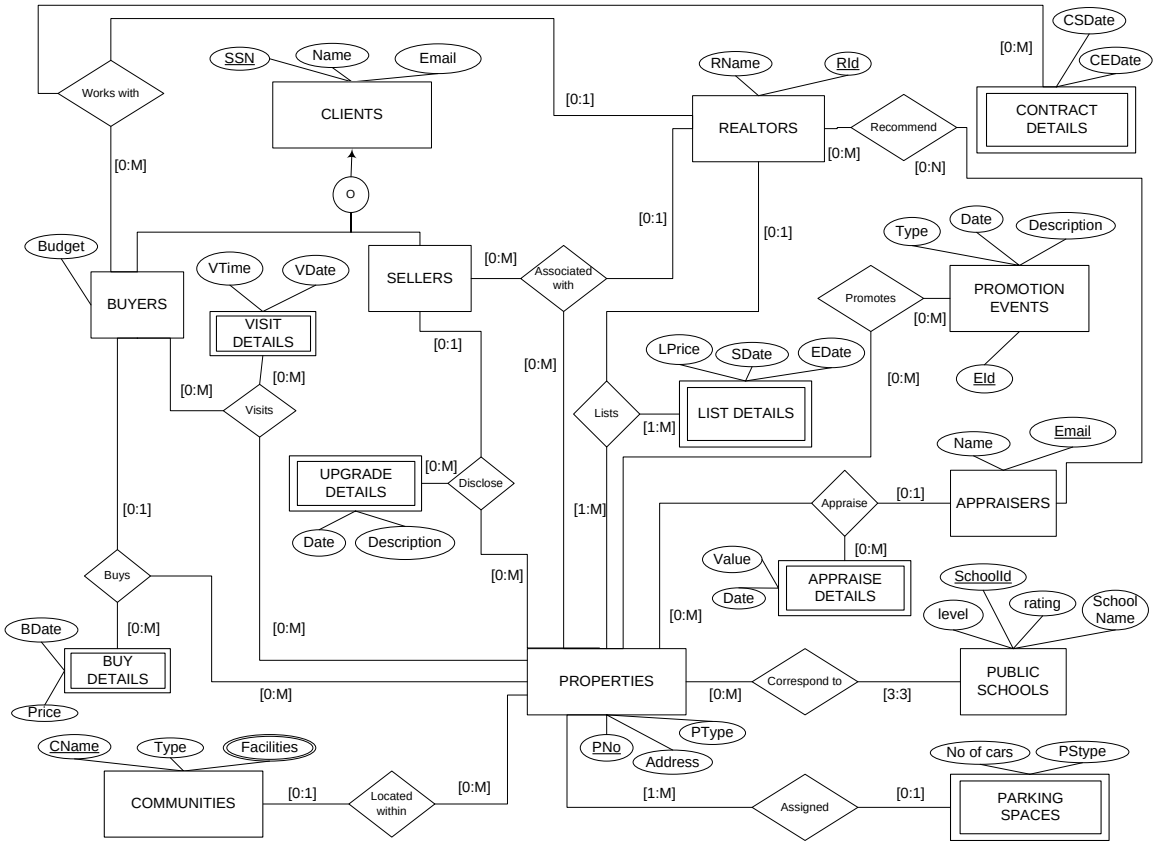

Figure 13: Illustrating Extra Relationship Error

Therefore, the main problem caused by this error is extra information being represented. To evaluate the impact of the error of extra relationship more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 41).

Correct design	Extra Relationship
No table	Recommend (*RId, AEmail*)

Table 41: Logical Design Comparison for Illustrating Extra Relationship Error

There is an extra table in the Extra Relationship design: The Recommend table linking Realtors and Appraisers. Based on our evaluation criteria, the incorrect design has some irrelevant information, i.e., the design is not valid since it contains information not in user requirements. It does not affect the efficiency with which the data can be accessed and does not affect the integrity of data. So, the seriousness of the error will be determined by how much redundant data is stored. Since there is only one table that is affected, the cost of an extra relationship is low.

Summary

The costs of the relationship errors are summarized below (Table 42). The errors are grouped by major types of errors (missing, incorrect and extra)

	Low Cost	Medium Cost	High Cost
Missing			Missing
Incorrect	Binary as Ternary (included entity class that shares an identifier)	Ternary as Binary Binary as Ternary (included entity class that does not share an identifier)	Relationship as an Attribute Non degree(Combine Relationships) Non degree (Substituted entity class that shares an identifier) Non degree (Substituted entity class with an irrelevant entity class)
Extra	Extra		

Table 42: Cost of Different Interaction Relationship Errors

3.3.4.5. Weak entity class errors

There are three major categories of weak entity class errors based on completeness and validity: missing strong entity class, incorrectly modeled strong entity class, extra strong entity class. The following subsections illustrate with examples the impact of each of the subcategories of weak entity class errors.

Missing Weak entity class

Within missing weak entity class errors, there are two types of errors based on the representation of the error: Weak entity class totally missing and Weak entity class as an attribute.

Weak entity class Totally Missing

To illustrate weak entity class totally missing error, the *Upgrade Details* entity class is deleted. This design results in an incomplete representation of the real world. With the relationship *Disclose*, the linkage between Sellers and Properties exists, i.e., it is possible to determine who the Property belongs to. However, without Upgrade Details, information on what upgrades the seller made and when is lost. The Upgrade_details captures the list of significant upgrades. This allows a potential buyer to have a better understanding on why the property is significantly more expensive than expected or evaluate what aspects may need to be upgraded in the new future.

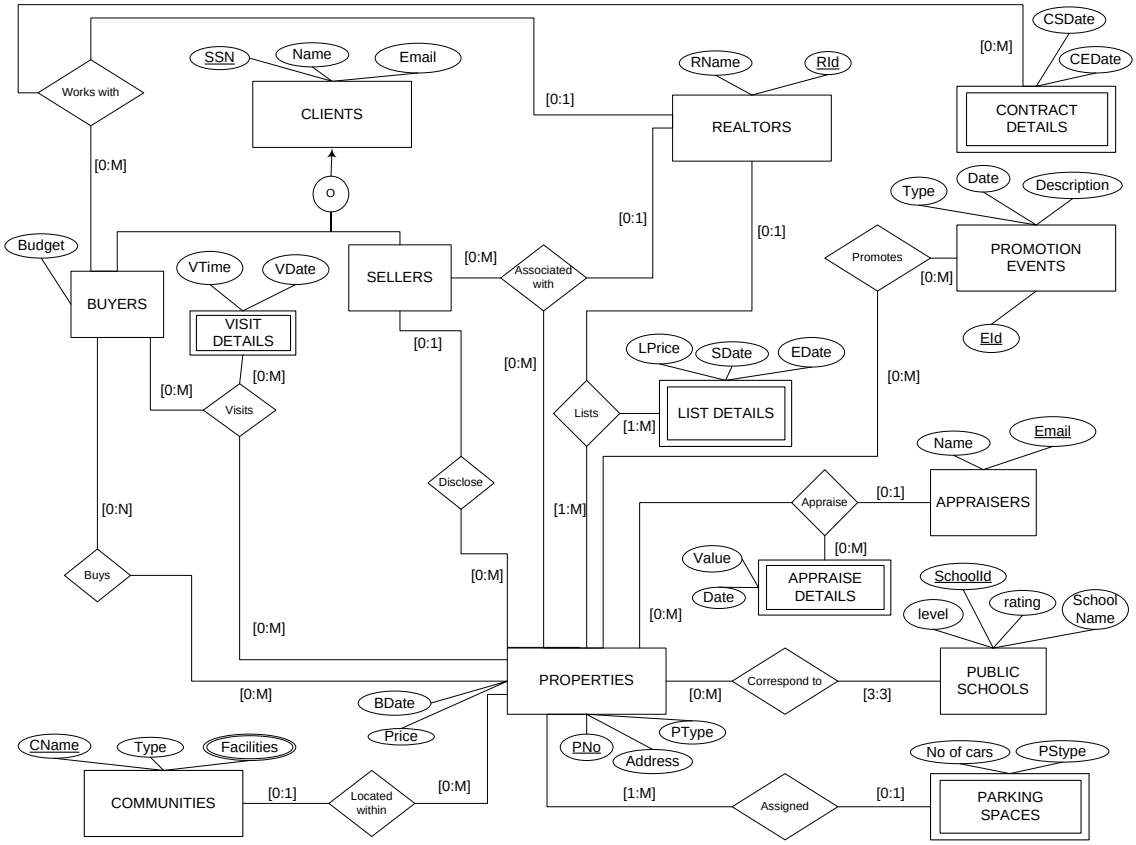

Figure 14: Illustrating Missing Weak Entity Class Error

To evaluate the impact of the missing weak entity class error more concretely, the conceptual design with the missing weak entity class is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 43).

Correct design	Missing weak entity class
Upgrade_details(*PNo, SSSN*, Date, Description)	*no table because associated weak entity class is missing*

Table 43: Logical Design Comparison For Missing Weak Entity Class

The logical design shows that the correct design has one table not present in the incorrect design. The error of missing information at the conceptual design stage is not found and corrected at the logical design stage. Using the evaluation criteria, the design does not accurately represent the real-world structure for *Upgrade Details* and hence does not support efficient access to data about *Upgrade Details*. Thus, the cost of missing a weak entity class is medium.

Weak Entity Class As An Attribute

To illustrate a weak entity class represented as an attribute, the attributes of the Buy details weak entity class is combined with Properties. By combining the weak entity class Buy Details with the strong entity class Properties, the designer implies that the attributes of the weak entity class belong to one of the strong entity classes and are not shared by the participating strong entity classes from which the weak entity class is derived from. In this example, it implies that BDate and Price are a characteristic of Properties, not shared

between the interaction of Buyers and Properties. If BDate and Price are assigned to Properties, any history of purchases is lost. Therefore, in the incorrect design where a weak entity class is represented as attribute(s) of a strong entity class, history of previous purchases will be lost. Therefore, the main problem caused by this error is losing information by not being able to represent multiple purchase details.

To evaluate the impact of the error of combining strong entity classes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 44). Two tables are affected by the incorrect design: Properties and Buys. The attributes contained in both the designs are the same. However, the attributes BDate and Price are associated with the Buys table in the correct design and are associated with the Properties table in the incorrect design. The incorrect design would store the list of buyers who purchased properties. However, if the house has been purchased multiple times, it cannot determine the current owner. Also, the incorrect design cannot keep track of the history of the purchases like when it was bought, and at what price.

Correct design	**Weak entity class as an attribute**
Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*) Buys(*BSSN, PNo*, BDate, Price)	Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*, BDate, Price) Buys(*BSSN*, *PNo*)

Table 44: Logical Design Comparison For Illustrating Weak Entity Class As An Attribute Error

Thus, the major problem with capturing a weak entity class as an attribute is that information is lost. Therefore, when strong entity classes are combined, the design does not capture the real world structure and the design does not provide efficient access to data because the underlying data is captured incorrectly. Thus, the cost of marking a weak entity class as an attribute is medium.

Incorrectly Modeled Weak entity Class

Within incorrectly modeled weak entity class, there are two types of errors based on the representation of the error: Weak entity class marked as strong and weak entity class marked as a subclass.

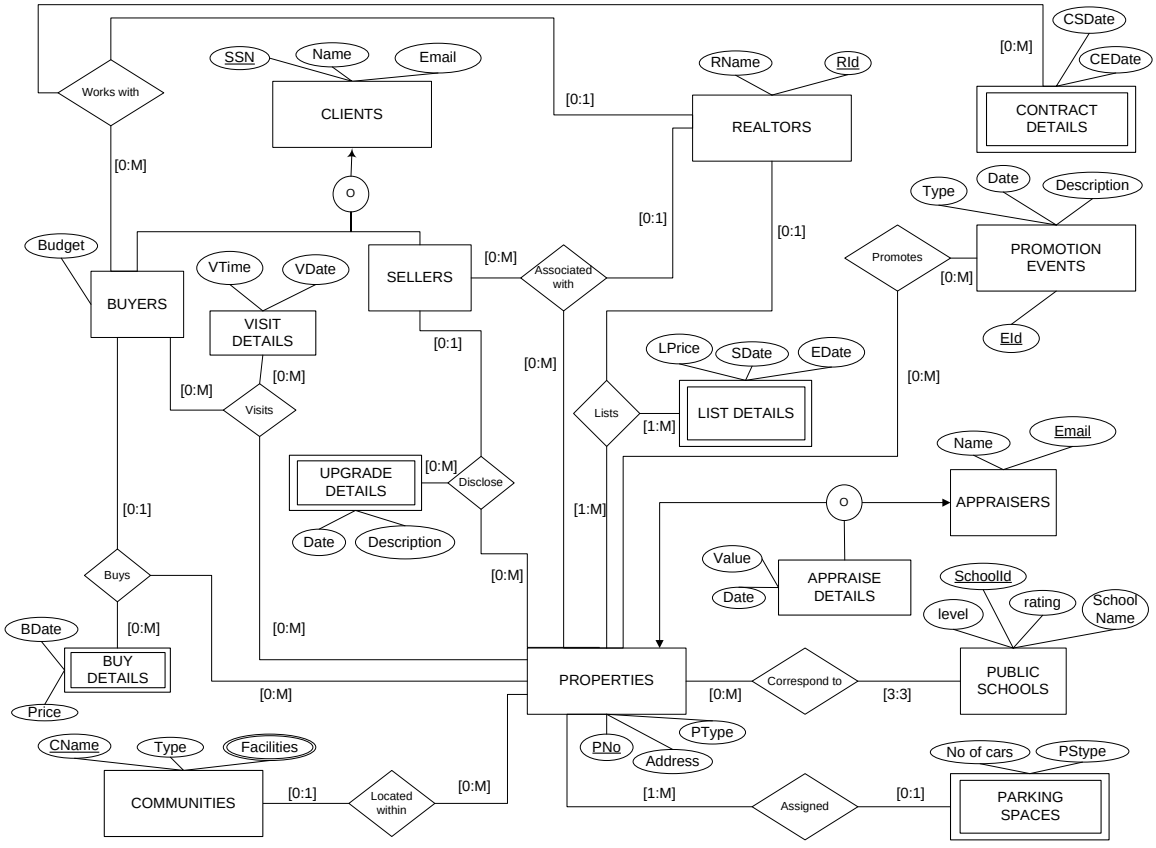

Figure 15: Illustrating Incorrectly Modeled Weak Entity Class Error

Weak Entity Class Marked As Strong

To illustrate incorrectly modeled weak entity class: weak entity class marked as strong error, Visit Details s is marked as strong. By marking it strong, the designer implies that Visit Details stands alone. It is not dependent on the relationship between Buyers and Properties. Therefore, the problem caused by this error is not accurately representing the real world structure.

To evaluate the impact of the error of marking a weak entity class as a strong entity class more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 45).

Correct design	**Weak entity class as strong**
Visits(*BSSN, PNo*, VDate, VTime)	Visits_Details(VNo, VDate, VTime) Visits(*BSSN, PNo*, *VNo*)

Table 45: Logical Design Comparison for Illustrating Weak entity Class Marked As Strong Error

The correct design has one table while the incorrect design has two. This results in three evaluation criteria being violated. By not correctly capturing the real world structure of Visits as a weak entity class, it results in many problems. First, it changes the real-world structure of how the data is stored. In addition, data integrity cannot be enforced, because the constraint that Visits is dependent on Buyers and Properties is lost. Also, the design creates inefficiencies because the incorrect design will involve two joins

instead of one to access information on which buyer visited which property on what date. Thus, the cost of marking a weak entity class as strong is high.

Weak entity Class Marked as Subclass

To illustrate weak entity class incorrectly modeled as a subclass: weak entity class "Appraise Details" is marked as a subclass of properties and Appraisers. By marking a weak entity class as a subclass, the designer implies that the weak entity class is a "type of" strong entity class. In this example, it implies that Appraise Details is a type of Appraiser and Property. However, Appraise_Details is not a type of Appraiser nor is it a type of Property. So, the real world semantics are incorrectly captured.

To evaluate the impact of the error of combining weak entity classes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 46). The difference between the two designs is the primary key. The correct design has PNo and Date as the primary key while the incorrect design has AEmail. This is because weak entity classes can have a partial key to define instances if the maximum cardinality marked between it and the relationship is many. However, since a subclass is a "type of" parent class, the primary key of any of the parent classes should be sufficient to identify instances of the subclass. This constraint will only allow the system to enter one Appraisal for an Appraiser. Therefore, if an Appraiser is hired a second time the information about the appraisal cannot be stored in the system.

Correct design	Weak entity Class as a subclass
Appraise_Details(*AEmail, PNo*, Value, Date)	Appraise_Details(*AEmail, PNo*, Value, Date)

Table 46: Logical Design Comparison For Illustrating Weak Entity Class As A Subclass Error

Thus, the major problem with capturing a weak entity class as a subclass is that information is lost. Therefore, when a weak entity class is captured as a subclass, the design does not capture the real world structure. Thus, the cost of capturing a weak entity class as a subclass is high.

Extra weak entity class

To illustrate extra weak entity class, the Associate details is included. The resulting conceptual schema is shown below. By adding the Associate details as a weak entity class, the designer adds extra information not explicitly stated in the design.

Therefore, the main problem caused by this error is extra information being represented. To evaluate the impact of the error of extra weak entity classes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 47).

The only difference is that the incorrect design has a few extra attributes. Based on the evaluation criteria, the incorrect design contains information not in user requirements. It does not affect the efficiency with which the data can be accessed, and does not affect

the integrity of data. The cost of an extra weak entity class is low because only one table is affected and no information is lost.

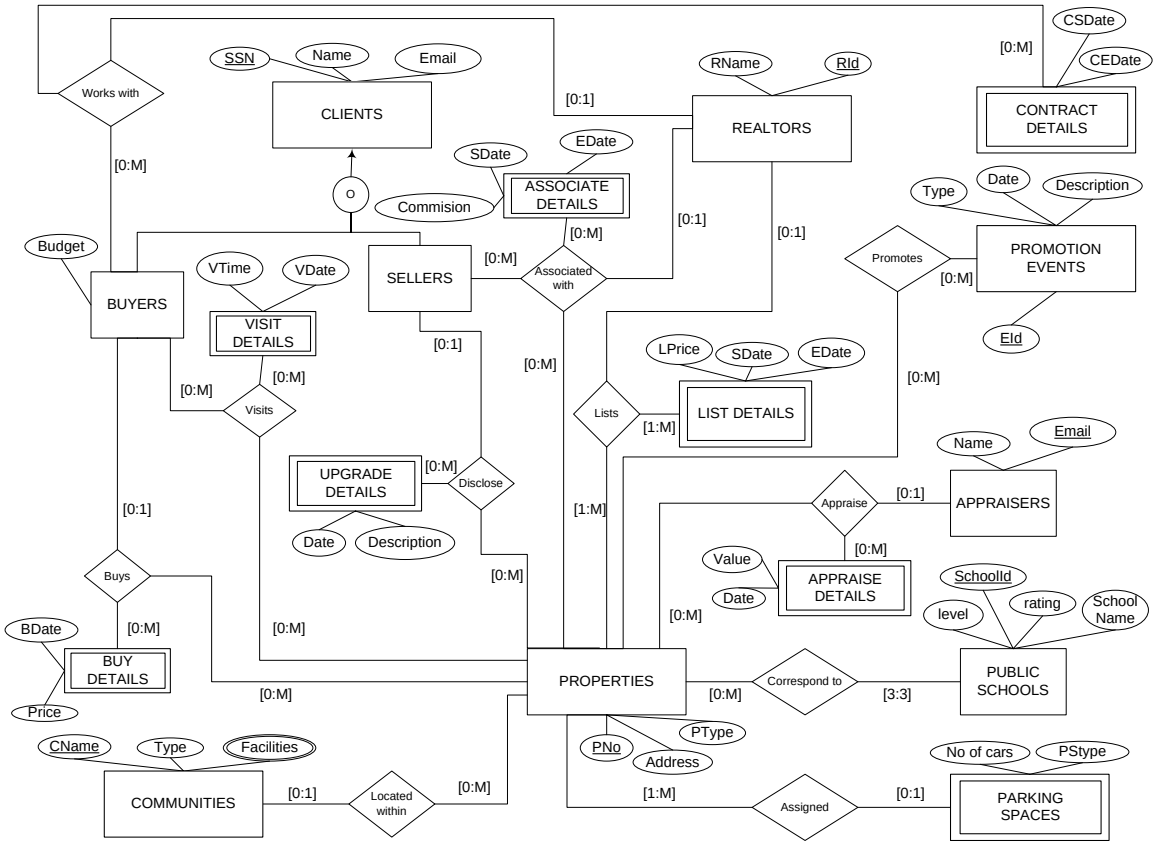

Figure 16: Illustrating Extra Weak Entity Class Error

Correct design	Extra Weak entity class
Associated_With(*SSSN, RId, PNo*)	Associated_With(*SSSN, RId, PNo*, AEDate, ASDate, ACommission)

Table 47: Logical Design Comparison For Illustrating Extra Weak Entity Class Error

Summary

The costs of weak entity class errors are summarized below (Table 48). The errors are grouped by major types of errors (missing, incorrect and extra)

	Low Cost	Medium Cost	High Cost
Missing		Missing	
Incorrect			Weak as strong Weak entity class as subclass
Extra	Extra		

Table 48: Cost of Different Weak Entity Class Errors

3.3.4.6. Attribute errors

There are two major categories of attribute errors based on completeness and validity: missing attributes, and extra attributes. The following subsections illustrate with examples the impact of each of the subcategories of attribute errors.

Missing Attribute

Within missing attribute errors, there is one type of error based on the representation of the error: Attribute totally missing.

Attribute Totally Missing

To illustrate attribute totally missing error, the *Budget* attribute is deleted. The resulting conceptual schema is shown below (Figure 17). This design results in an incomplete representation of the real world. Information on the Buyer's budget is lost. Therefore, the problem caused by this error is not accurately representing the real world structure.

To evaluate the impact of the error of totally missing an attribute more concretely, the conceptual design with the error is converted to a logical design. This logical design is

then compared to the logical design of the conversion of the conceptual design without any errors (Table 26).

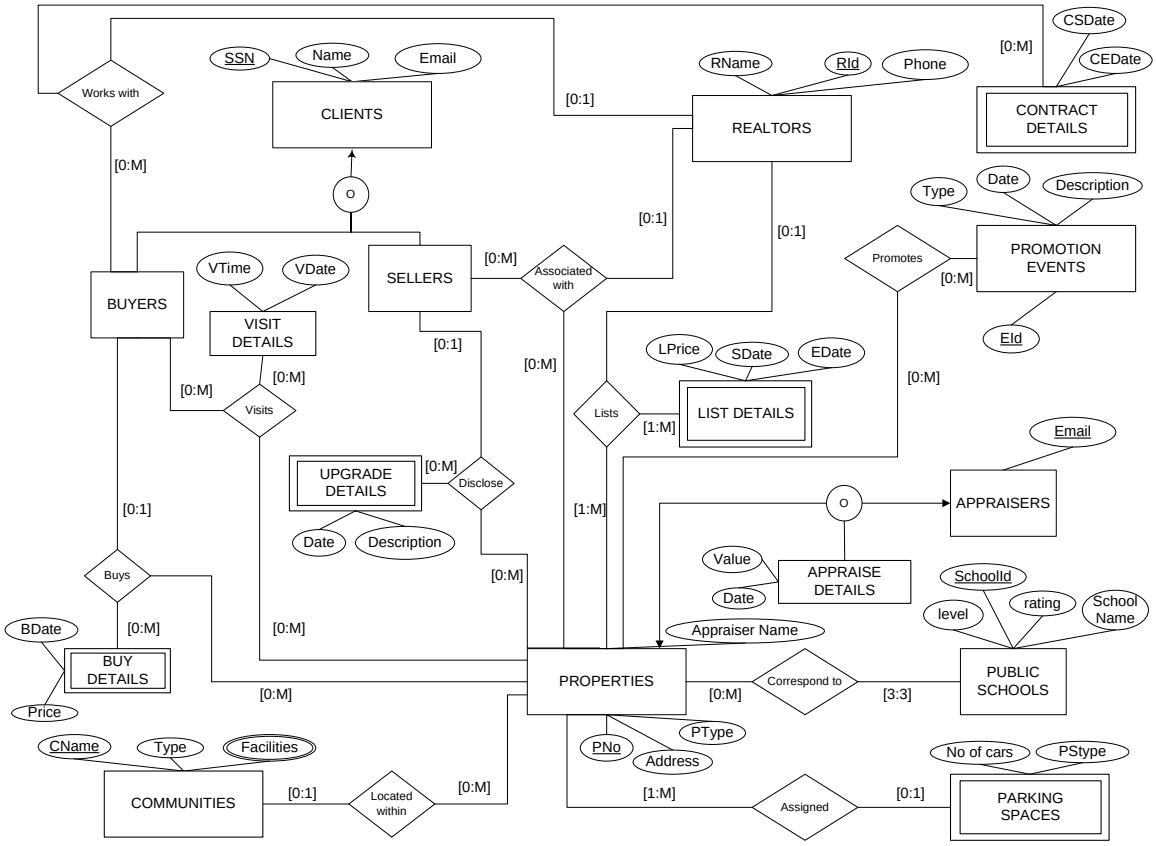

Figure 17: Illustrating Attribute Errors

Correct design	Attribute Totally Missing
Buyers(BSSN, BName, BEmail, Budget)	Buyers(BSSN, BName, BEmail)

Table 49: Logical Design Comparison for Illustrating Attribute Totally Missing Error

The only the differences between the two designs are shown above (Table 49). The correct design has an additional attribute compared to the incorrect design. This error results in one evaluation criterion being violated: the design does not capture the real

world structure of Buyers. The cost of the totally missing attribute is medium because it results in losing information.

Incorrectly Modeled Attribute

The only incorrectly modeled attribute error for attributes is associating attribute with wrong entity class. To illustrate this incorrectly modeled attribute error, Appraiser *Name* is associated with Properties instead of Appraisers. By associating Name with Properties, the designer implies that Appraiser Name is a characteristic of Property. Therefore, the problem caused by this error is not accurately representing the real world structure.

To evaluate the impact of the error of associating an attribute with the wrong entity class more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 50).

Correct design	Extra Attribute
Properties(PNo, Address, PType, PSNo_cars, PSType, *CName*) Appraisers(AEmail, AName)	Properties(PNo, AName, Address, PType, PSNo_cars, PSType, *CName*) Appraisers(AEmail)

Table 50: Logical Design Comparison For Illustrating Incorrectly Modeled Attribute Error

Two tables are affected: Properties and Appraisers. The AName attribute is associated with Appraisers in the correct design and associated with Properties in the incorrect design. This results in all four evaluation criteria being violated. By not correctly capturing the real world structure of AName as being associated with Appraisers, it

results in many problems. First, it changes the real-world structure of how the data is stored. It also restricts a property to have one Appraiser across time. In addition, data integrity cannot be enforced, because the constraint that Appraiser Email (primary key of Appraisers) determines Appraiser Name is lost. In the incorrect design, to retrieve Appraiser Name, the Properties that an Appraiser has appraised needs to be accessed. The extra joins required to access Appraiser Name creates inefficiencies. Also, Appraiser Name will be repeatedly stored or every Property that the Appraiser appraised. This repeated storage results in redundant data storage. Thus, the cost of an incorrectly modeled attribute is high.

Extra Attribute

To illustrate extra attribute, *Phone* is included for Realtors. The resulting conceptual schema is shown above (Figure 17). By adding the Phone as an attribute, the designer adds extra information not explicitly stated in the design.

Therefore, the main problem caused by this error is extra information being represented. To evaluate the impact of the error of extra attributes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 51).

Correct design	Extra Attribute
Realtors(RId, RName)	Realtors(RId, RName, Phone)

Table 51: Logical Design Comparison For Illustrating Extra Attribute Error

The only difference is that the incorrect design has an extra attribute. Based on our evaluation criteria, the incorrect design contains information not in user requirements. It does not affect the efficiency with which the data can be accessed, and does not affect the integrity of data. The cost of an extra attribute is low because only one table is affected and no information is lost. This error was ignored because novices are sometimes encouraged to add extra attributes as needed.

Summary

The costs of the attribute errors are summarized in below (Table 52). The errors are grouped by major types of errors (missing, incorrect and extra). As discussed previously, the error of including extra attributes is ignored. Since it is a low cost error, ignoring the error will not have serious implications on design.

	Low Cost	**Medium Cost**	**High Cost**
Missing		Missing	
Incorrect			Associate Attribute With Wrong Entity Class
Extra	Extra		

Table 52: Cost of Different Attribute Errors

3.3.4.7. Cardinality errors

There are two major categories of cardinality errors based on completeness and validity: missing cardinality, and incorrect cardinality. The following subsections illustrate with examples the impact of each of the subcategories of cardinality errors.

Missing Cardinality

Within missing cardinality errors, there is one type of error based on the representation of the error: Cardinality totally missing.

Cardinality Totally Missing

To illustrate cardinality totally missing error, the cardinality for the Buys relationship is deleted. The resulting conceptual schema is shown below (Figure 18). The exact semantics of the Buys relationship is lost. Therefore, the problem caused by this error is not accurately representing the real world structure.

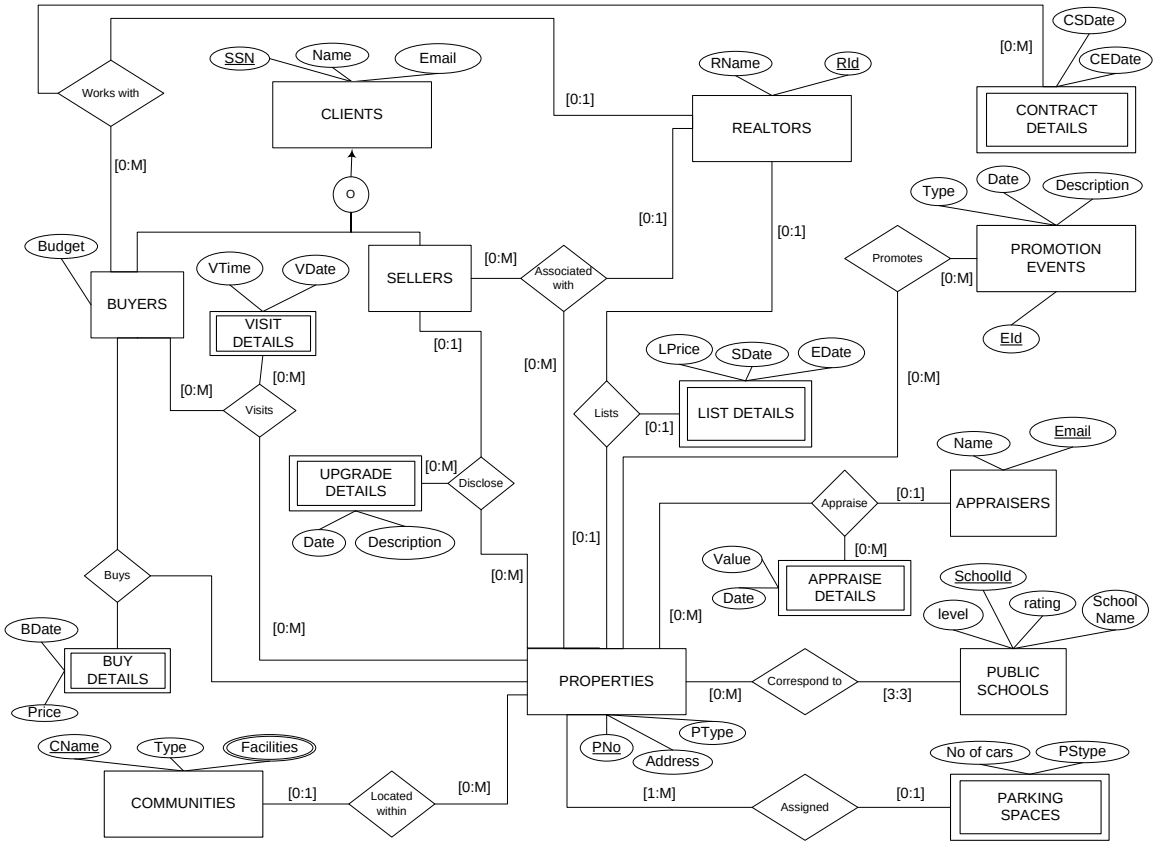

Figure 18: Illustrating Cardinality Errors

Correct design	Cardinality Totally Missing
Buys(*BSSN, PNo*, BDate, Price)	Buys(*BSSN, PNo*, BDate, Price)

Table 53: Logical Design Comparison for Illustrating Cardinality Totally Missing Error

To evaluate the impact of the error of totally missing cardinality more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26).

This error will be detected when the conceptual design is converted to a logical design. However, detection does not ensure correction. If the correct cardinality is determined, then the cost of the error is negligible. However, if cardinality is incorrectly determined, then the cost can be high (see 0) Thus, the cost of missing cardinality can be negligible to high.

Incorrect Cardinality

To illustrate incorrect cardinality, the cardinality for the *lists* relationship is modified from a maximum of M (many) to a maximum of 1 on the lines that connect properties and List details to the relationship. The resulting conceptual schema is shown above (Figure 18). This error results in changing the semantics to imply that a relationship a 1:M:N (one to many to many) to a 1:1:1 (one to one to one). Changing the cardinality restricts a given realtor to list a maximum of one property.

The main problem caused by this error is not accurately representing real-world structure. To evaluate the impact of the error of extra attributes more concretely, the

conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 51).

Correct design	Incorrect Cardinality
List_Details(*RId, PNo*, LPrice, SDate, EDate)	List_Details(*RId, PNo*, LPrice, SDate, EDate)

Table 54: Logical Design Comparison For Illustrating Incorrect Cardinality Error

Both tables have the same attributes. The only difference between the two tables is the primary keys. The correct design has RId, PNo and SDate together as the primary key while the incorrect design only has RId as the primary key. This will create extra constraints that are not intended and not desired. For example, having RId as the primary key will only allow one property to be listed with one realtor. This constraint violates the business constraint that a realtor lists many properties for sale. In addition, there would be no data integrity checks to ensure that the same property is not listed multiple times with the same start date but different end dates and prices. Therefore, the main problems caused by this error are not accurately representing real-world structure, and inability to maintain data integrity over time. Thus, the cost of incorrect cardinality is high.

Summary

The costs of different cardinality errors are summarized below (Table 55). The errors are grouped by major types of errors (missing and incorrect)

	Low Cost	Medium Cost	High Cost
Missing	Missing		
Incorrect			Incorrect

Table 55: Cost of Different Cardinality Errors

3.3.4.8. Identifier errors

There are two major categories of cardinality errors based on completeness and validity: missing cardinality, and incorrect cardinality. The following subsections illustrate with examples the impact of each of the subcategories of cardinality errors.

Missing Identifiers

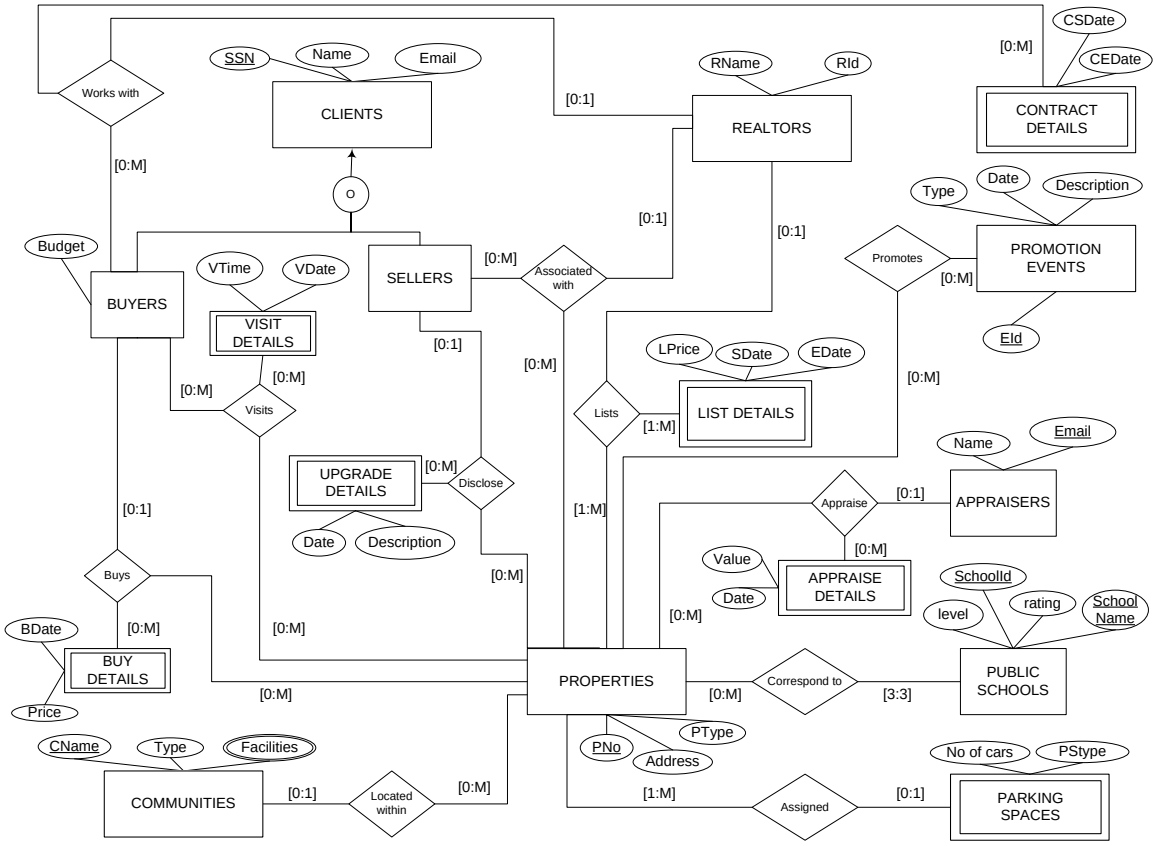

Figure 19: Illustrating Identifier Errors

There is only one type of error based on the representation of the error: Identifier totally missing.

Identifier Totally Missing

To illustrate identifier totally missing error, the identifier for Realtors (RId) is not marked. The resulting conceptual schema is shown below (Figure 19). The problem caused by this error is not accurately representing the real world structure.

Correct design	Identifier Totally Missing
Realtors(RId, RName)	Realtors(RId, RName)

Table 56: Logical Design Comparison for Illustrating Identifier Totally Missing Error

To evaluate the impact of the error of totally missing cardinality more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26).

This error will be detected when the conceptual design is converted to a logical design. The detection will generally result in correction. If the correct identifier is determined, then the cost of the error is negligible. Thus, the cost of missing identifier will be negligible.

Incorrect Identifier

To illustrate incorrect cardinality, the identifier for the *Public Schools* entity class is modified from SchoolId to SchoolId and SchoolName. The resulting conceptual schema is shown above (Figure 19). This changes the data constraints to imply that SchoolId and

SchoolName together uniquely identify Public Schools; i.e., SchoolId or SchoolName alone is not sufficient. However, in most cases, it is a notation issue. The designer intended to indicate that SchoolId and SchoolName can serve as Identifiers on their own. However, it is possible that the designer blindly converts the schema to the logical design in which case both attributes will jointly serve as the primary key.

The main problem caused by this error is the potential of not enforcing data integrity. To evaluate the impact of the error of extra attributes more concretely, the conceptual design with the error is converted to a logical design. This logical design is then compared to the logical design of the conversion of the conceptual design without any errors (Table 26). Only the differences between the two designs are shown below (Table 57).

Correct design	Incorrect Identifier
Public_Schools(SchoolId, SchoolName, level, rating)	Public_Schools(SchoolId, SchoolName, level, rating)

Table 57: Logical Design Comparison For Illustrating Incorrect Identifier Error

Both tables have the same attributes. The only difference between the two tables is the primary keys. The correct design has SchoolId as the primary key while the incorrect design has SchoolId and SchoolName jointly as the primary key. Since SchoolId and SchoolName have a one-one correspondence, this will not be a problem in most cases. However, it will allow the same school to be entered multiple times accidentally with different variants in the name. Therefore, the main problem caused by this error is

inability to maintain data integrity over time. Thus, the cost of incorrect cardinality is low to medium.

Summary

The costs of identifier errors are summarized in the table below (Table 58). The errors are grouped by construct and major types of errors (missing and incorrect)

	Low Cost	**Medium Cost**	**High Cost**
Missing	Missing		
Incorrect	Incorrect		

Table 58: Cost of Different Identifier Errors

3.4. Performance Prediction Model

The objective of this research is to predict the type of errors made based on the modeling expertise (Figure 20). The reasoning behind the objective is that if the correspondences are known, given the designer's expertise level, the types of errors that will be made can be predicted. ERM predicts that as a designer's expertise increases, the designer's performance will improve. If the level of expertise is known, a designer can be provided with training modules that address only the levels of expertise that she needs. Further, some aspects of the expertise levels are hierarchical; i.e., it is necessary to attain the lower levels of expertise before the higher levels can be attained. Therefore, the research will also be able to suggest the order of presentation of the learning modules for the designer. The order of training will also be influenced by the seriousness of the errors and the relationship between expertise levels and errors.

3.4.1. Developing Hypotheses

The hypotheses test whether being at a particular expertise level helps reduce errors. Since the hypotheses are directional, i.e., increasing expertise should reduce the errors, all the hypotheses specified will be tested as one sided tests.

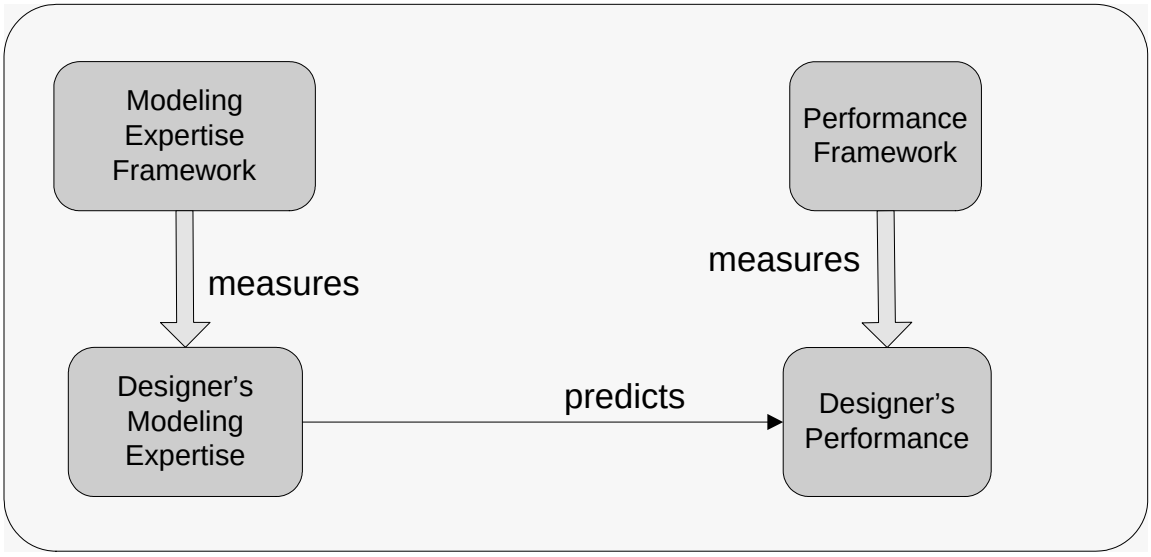

Figure 20: Performance Prediction model

There are levels to modeling expertise. At the Understand level, the novice understands the meaning of each data modeling construct. It is necessary for the novice to be at least the Understand level before any errors can be predicted. If the novice is not at the Understand level, i.e., the novice interprets the data modeling constructs in a different way, then it is impossible to evaluate what the schema drawn represents; the schema could be drawn according to another modeling grammar. If so, the errors will not be predicted by MEF for semantic data modeling and the novice's data should be excluded from further analysis.

3.4.1.1. Errors Predicted if Not at the Apply Level

At the Apply level, the novice knows how to apply the procedure to determine cardinality. Cardinality adds semantics to relationships indicating how the entity classes in a relationship are related. If a novice is not at the Apply level for cardinality, she will make errors in determining cardinality as shown in Table 59.

Construct Name	Errors predicted if not at Apply
Cardinality	Incorrect cardinality

Table 59: Errors Predicted if Not at the Apply Level

Error Prediction Hypothesis 1: If a novice is not at the Apply level for cardinality, she will incorrectly determine cardinality through the semantic data schema

3.4.1.2. Errors Predicted if Not at the Differentiate sublevel of Analyze

At the Differentiate sublevel of Analyze, the novice knows how to distinguish relevant and irrelevant information in the specifications. Irrelevant information should not be captured at all by the conceptual schema, while relevant information should be captured. The Differentiate sublevel of Analyze helps predict whether irrelevant information will be captured in the semantic data schema or relevant information will be missing as shown in Table 60.

Irrelevant information includes organization information (miniworld), process and constraint information. If the novice cannot identify irrelevant information for entity classes, she will include the organization information or system information as entity classes. Similarly, if a novice cannot identify process information as irrelevant for

relationships, she will capture irrelevant processes as relationships or add extra entity classes to a relationship.

If the novice cannot identify relevant information for entity classes, she will miss relevant entity classes. If a novice cannot recognize relevant information for relationships, she will miss relationships altogether. Another reason for missing relevant entity classes or relationships or attributes is missing the information in the requirements. Due to limitations of resources, this research could not measure attention of participants to control for this variable. Therefore, the likelihood of statistical significance for this type of error is low.

Hypotheses	**Construct Name**	**Errors predicted if not at Differentiate$_{Relevant}$**	**Errors predicted if not at Differentiate$_{Irrelevant}$,**
EP2	Entity class	Miss entity classes	Extra entity classes
EP3	Interaction relationship	Miss relationship	Extra relationship Incorrect(Binary as Ternary)
EP4	Foreign key		Miss relationship Incorrect(Replace Entity class with foreign key)
EP5	Attribute	Miss attribute	
EP6	Cardinality	Miss cardinality	
EP7	Identifier	Miss identifier	

Table 60: Errors Predicted if Not at the Differentiate sublevel

Error Prediction Hypothesis 2: If a novice is not at the Differentiate sublevel within Analyze for entity classes, she will (a) add extra irrelevant entity classes and / or (b) miss relevant entity classes.

This can be separated into the following hypotheses:

Error Prediction Hypothesis 2a: If a novice is not at the Differentiate sublevel (relevant) within Analyze for entity classes, she will miss relevant entity classes.

Error Prediction Hypothesis 2b: If a novice is not at the Differentiate sublevel (irrelevant) within Analyze for entity classes, she will add extra irrelevant entity classes.

Error Prediction Hypothesis 3: If a novice is not at the Differentiate sublevel within Analyze for interaction relationships, she will (a) add extra irrelevant interaction relationships and / or (b) miss relevant interaction relationships and / or (c) will incorrectly capture binary relationships as ternary relationships

Since identifying relevant and irrelevant pieces of information are independent, the hypotheses can be split into the following hypotheses:

Error Prediction Hypothesis 3a: If a novice is not at the Differentiate sublevel (relevant) within Analyze for interaction relationships, she will miss relevant interaction relationships

Error Prediction Hypothesis 3b: If a novice is not at the Differentiate sublevel (irrelevant) within Analyze for interaction relationships, she will (a) add extra irrelevant interaction relationships and / or (b) will incorrectly include irrelevant entity classes in relationships

Error Prediction Hypothesis 4: If a novice is not at the Differentiate sublevel within Analyze for foreign keys, she will (a) include foreign keys redundantly (in addition to capturing the relationship) in the semantic data schema and / or (b) replace relevant interaction relationships or an entity class in an interaction relationship by foreign keys

Error Prediction Hypothesis 5: If a novice is not at the Differentiate sublevel within Analyze for attributes, she will miss marking attributes.

Error Prediction Hypothesis 6: If a novice is not at the Differentiate sublevel within Analyze for cardinality, she will miss marking cardinality.

Error Prediction Hypothesis 7: If a novice is not at the Differentiate sublevel within Analyze for identifiers, she will miss marking identifiers.

3.4.1.3. Errors Predicted if Not at the Organize Level of Analyze

At the Organize sublevel of analyze, the novice knows how to select the correct variant of the data modeling construct within the context of the conceptual schema. To be able to select the correct variant of the data modeling construct, the novice needs to be able to select information relevant to the data modeling construct and organize it appropriately. If a novice is not at the organize level, she will make a number of errors as shown in Table 61. Evaluating incorrect identifiers was difficult because some learners had been instructed to create an artificial "Id" identifier for all strong entity classes. Therefore, we did not include hypotheses associated with the organize level for identifiers.

Hypotheses	Construct Name	Errors predicted if not at $Organize_{Grouping}$	Errors predicted if not at $Organize_{Iype}$
EP8	Attribute	Incorrect(Associate attribute with wrong entity class)	
EP9	Entity class	Incorrect(Combine entity classes)	Incorrect(Strong entity class as weak) Incorrect(Weak entity class as strong)
EP10	Interaction relationship	Incorrect(Ternary as two binary)	
EP11	Cardinality	Incorrect(Ternary cardinality)	

Table 61: Errors Predicted if Not at the Organize Level

When the novice is at the Organize sublevel for attributes, she will correctly associate attributes with the appropriate strong entity class.

Error Prediction Hypothesis 8: If a novice is not at the Organize sublevel within Analyze for attributes, she will incorrectly associate attributes of one strong entity classes with another strong entity class.

When the novice is at the Organize sublevel for entity classes, she will be able to correctly group entity classes and will be able to correctly select between the two types of entity classes—strong and weak entity classes. There are three different variants of correctly selecting between strong and weak entity classes: strong and strong, strong and weak, weak and weak.

Error Prediction Hypothesis 9

Error Prediction Hypothesis 9a: If a novice is not at the Organize sublevel within Analyze for entity classes, she will incorrectly mark strong entity classes as weak.

Error Prediction Hypothesis 9b: If a novice is not at the Organize sublevel within Analyze for entity classes, she will incorrectly associate attributes of a strong / weak entity classes with another strong / weak entity class; i.e., she will incorrectly combine entity classes.

When the novice is at the Organize sublevel for interaction relationships, she will be able to correctly organize shared information among entity classes and choose the correct degree for the relationships.

Error Prediction Hypothesis 10: If a novice is not at the Organize sublevel within Analyze for interaction relationships, she will incorrectly capture ternary relationships as two binary relationships.

When the novice is at the Organize sublevel for cardinality, she will be able to correctly organize cardinality for the relationships.

Error Prediction Hypothesis 11: If a novice is not at the Organize sublevel within Analyze for cardinality, she will incorrectly capture cardinality for relationships.

3.4.2. Control Variables

There are other variables that could possibly contribute to the performance. All the variables described in this section were measured as control variables, i.e., to check whether other variables are influencing the performance levels.

Domain Familiarity influences some aspects of designer performance. However, domain familiarity does not impact syntactic and semantic based comprehension tasks. it

only impacts only problem solving tasks because of cognitive fit (Khatri, et al. 2006). This research limits its applicability on errors in creating semantic data schemas. This research is primarily focused on the semantics. Since the there is a direct relationship between the problem representation and the task requirements, Application domain knowledge will have no effect on the errors made. However, if there is a difference in performance among novices with high application domain knowledge and low application domain knowledge, the prediction of errors based on expertise levels will need to be conducted separately among those with high and low domain knowledge.

Error Prediction Hypothesis 12: A novice who has high domain knowledge in real estate will perform as well as a novice who has low domain knowledge.

Error Prediction Hypothesis 13: A novice who has a lot of prior experience with Object Oriented programming will perform as well as a novice who has little or no prior experience.

Error Prediction Hypothesis 14: A novice who has prior experience with ER modeling will perform better than a novice who doesn't have prior experience with ER modeling In addition, a novice who has prior experience with ER modeling should be at a higher expertise level than a novice who doesn't.

Error Prediction Hypothesis 15: Familiarity with UML will not impact performance in semantic data modeling.

Different learning styles may result in different biases and errors. For example, since semantic data modeling is a visual technique, would novices who are visual learners have an advantage over those who are not? Therefore, this research measured looked at

different learning styles and how they might potentially impact performance levels. There are many different learning style models in existence. Prominent models include Visual, Auditory, Kinesthetic (VAK) family of models and the Myers-Briggs Type Indicator (MBTI) (Myers and McCaulley 1985). The dimensions of the VAK family of models seemed more relevant in the semantic data modeling context than the dimensions in MBTI. Among the choices within the VAK family of models, 'Index of Learning Styles' (Felder and Silverman 1988, Felder and Spurlin 2005) was chosen because it has been previously established as valid and reliable (Felder and Spurlin 2005). Two of the dimensions in the Index of Learning Styles were particularly interesting: the visual-verbal dimension and the sensing-intuitive dimension. Since semantic data modeling is a visual technique, would novices who are visual learners outperform novices who are verbal learners? Also, since semantic data modeling involves abstractions, would novices who are intuitive learners outperform novices who are sensing learners?

Scale	Definition
Visual-Verbal	Visual learners remember best what they see while Verbal learners get more out of words
Sensing-Intuitive	Sensing learners like learning facts, while intuitive learners often prefer discovering possibilities and relationships.

Table 62: Different Types of Learners

Error Prediction Hypothesis 16: Novices who are visual learners will perform better (make fewer errors) than novices who are verbal learners.

Error Prediction Hypothesis 17: Novices who are intuitive learners will perform better (make fewer errors) than novices who are sensing learners.

While observing the learners create semantic data models, it was noted that learners tended to use three main approaches. One group of learners appeared to convert the information in one sentence before moving to the next. A second group of learners would first create entity classes and identify attributes, and then find relationships among the entity classes. The third group of learners would identify entity classes, then identify relationships and then mark the attributes for the whole schema. It is anticipated that learners that identify relationships before attributes are more likely to make process related errors than learners that identify relationships after identifying attributes. For example, the buys relationship should not be a four way relationship between Buyers, Realtors, Sellers and Properties just because they all participate in the transaction.

Also, some learners appeared to determine cardinality as soon as they identified the relationship while others determined the cardinality at the end. Again, it is anticipated that learners that identify cardinality at the end are more likely to make process errors than those that identify it as soon as they identify relationships

Error Prediction Hypothesis 18: Novices that identify attributes at the end are more likely to make process related errors than learners that identify attributes before relationships.

Error Prediction Hypothesis 19: Novices that identify cardinality at the end are more likely to make process related errors than learners that identify cardinality immediately after relationships.

3.4.3. Summary

This section described hypotheses that predicted the different types of errors made in constructing ER schemas. In particular, the hypotheses described the relationship of modeling expertise and errors as well as the relationship between control variables and errors. These hypotheses are summarized in Table 63. Of these hypotheses, hypotheses EP2a, EP3a, EP5a are not expected to be significant because the major cause of the Miss modeling ER modeling construct error is thought to be caused by an unmeasured variable: missing information in the requirements.

H#	Hypothesis description
EP1	*Not* Apply(Cardinality)→ Incorrect (Cardinality)
EP2a	*Not* Differentiate $_{\text{Relevant}}$ (Entity class)→ Miss (Entity class)
EP2b	*Not* Differentiate $_{\text{Irrelevant}}$ (Entity class)→ Extra (Entity class)
EP3a	*Not* Differentiate $_{\text{Relevant}}$ (Relationship)→ Miss (Relationship)
EP3b	*Not* Differentiate $_{\text{Irrelevant}}$ (Relationship)→ Extra (Relationship) *Not* Differentiate $_{\text{Irrelevant}}$ (Relationship)→ Error (Relationship: Binary as Ternary)
EP4	*Not* Differentiate $_{\text{Irrelevant}}$ (Foreign key) → Miss(Relationship) / Error(Relationship: Replace Entity class with Foreign key)
EP5a	*Not* Differentiate $_{\text{Relevant}}$ (Attribute) → Miss (Attribute)
EP5b	*Not* Differentiate $_{\text{Irrelevant}}$ (Attribute)→ Extra (Attribute)
EP6	*Not* Differentiate $_{\text{Relevant}}$ (Cardinality) → Miss(Cardinality)
EP7	*Not* Differentiate $_{\text{Relevant}}$ (Identifier) → Miss(Identifier)
EP8	*Not* Organize $_{\text{Grouping}}$(Attribute) → Error(Attribute: associate attribute with wrong entity class error)

EP9a	*Not* Organize $_{Type}$(Entity class) → Error(Entity class: strong-weak distinction error)
EP9b	*Not* Organize $_{Grouping}$ (Entity class) → Error(Entity class: combine entity classes)
EP10	*Not* Organize $_{Grouping}$ (Relationship) → Error(Ternary as two binary)
EP11	*Not* Organize $_{Grouping}$ (Cardinality) → Error(Cardinality)
EP12	Knowledge(Application Domain) → No impact on error
EP13	Knowledge(Objected Oriented Programming) → No impact on error
EP14	Prior Experience(ER modeling) → Impacts Errors
EP15	Familiarity(UML) → No impact on error
EP16	Type of learner (Visual vs. Verbal) → Impacts Errors
EP17	Type of learner (Intuitive vs. Sensing) → Impacts Errors
EP18	Process of Identifying Attributes → Impacts Errors
EP19	Process of Identifying Cardinality → Impacts Errors

Table 63: Error Prediction Hypotheses Summary

3.5. Summary

In this chapter, we developed the Modeling Expertise Framework, the Performance Framework and described three categories of hypotheses: Bloom's taxonomy verification hypotheses, Relationship Among Construct hypotheses and Error Prediction hypotheses. Bloom's taxonomy verification hypotheses test the hierarchical nature of the Bloom's taxonomy's cognitive processes within the MEF. The Relationship Among Construct hypotheses explore potential intuitive relationships within the MEF. The Error Prediction hypotheses predict Performance based on the position in the MEF. We also calculated the cost of each error.

The next chapter discusses the operationalization of the hypotheses and the coding methodology. The following chapters discuss the analysis, implications of the results and conclusions.

CHAPTER 4: RESEARCH METHODOLOGY

The research objective is to understand why novices make expertise-related errors in semantic data modeling. The previous chapter developed the model, constructs, and hypotheses. The goal of this chapter is to operationalize these constructs so that the hypotheses can be tested. A survey methodology was used in this research to gather quantitative data, and therefore, sampling and data collection are also going to be described.

After the constructs were identified, a paper-based survey was developed. The survey had two main parts: a questionnaire and a task of creating an ER schema. The questionnaire measured the independent and control variables. The task of creating an ER schema contained a description for requirements and its purpose was to measure the dependent variable, i.e., different types of errors. The following sections discuss the definitions of constructs, the corresponding items in the survey, the process of developing the instrument and the recruitment of participants.

4.1. Study Materials

A paper based questionnaire was developed to measure the expertise in ER modeling. A requirements description was created to measure the performance levels. This section describes the questions used to measure expertise in ER modeling and the task for creating the ER schema.

Since the research involved building a new instrument, it was especially important to examine its validity. The revised Bloom's taxonomy (RBT) details the range of expertise

levels. RBT also provides guidelines for designing instruments to measure attainment of the expertise levels. Using the guidelines, questions that measured the important aspects of proficiency were designed. The questions were then reviewed by a group of specialists in semantic data modeling for content validity. Input from dissertation committee members was also solicited and implemented. A limited version of the instrument was pre-tested on twenty undergraduate students. Based on feedback, the instrument was further refined and extended. Attention was given to the format of questions, order of questions and answer choices to minimize their effects on the task (Bettman and Zins 1979, Case and Swanson 2003, Haladyna 1999). Variables like mood (Constantine and Lockwood 2006) and lighting (Baron, et al. 1992) were ignored because they would impact the independent, control and dependent variables similarly. The full version was then pre-tested on four students who had been exposed to semantic data modeling. No major problems were reported.

4.1.1. Tasks to Measure Expertise Levels

The objective of this research is to understand the expertise-related reasons behind errors. To achieve our objective, the range of expertise levels related to creating semantic database schemas need to be measured. The expertise levels were operationalized by designing questions using guidelines in the Revised Bloom's Taxonomy. Thus, tasks were designed to measure attainment of each of the cognitive processes from Understand to Analyze within the Conceptual knowledge dimension for each of the constructs: entity classes, interaction relationships, attributes, primary keys, cardinality, foreign keys. The following subsections explain, with examples, how each Cognitive Process was

operationalized. All tasks used a relatively familiar environment that had not been used in class discussions. A recognizable environment was chosen to minimize unfamiliarity of the domain as a source of difficulty (Khatri, et al. 2004). The selection of an unseen environment minimized the possibility that learners relied solely on memory and hence was able to measure higher levels of expertise.

4.1.1.1. Understand Level

The first relevant expertise level is *Understand*. According to RBT, learners are at the Understand level, if they have a basic grasp of the instructional material, whether the material is oral, written or graphical. There are seven different ways to measure the Understand level. The seven processes are Interpreting, Exemplifying, Classifying, Summarizing, Inferring, Comparing and Explaining. Of the seven, Exemplify is the most relevant process for our research. In the rest of this section, each cognitive process is described. Then, the applicability of the different cognitive processes to semantic data modeling is discussed and our choices are justified.

Interpreting: This cognitive process occurs when the learner is able to represent the information presented in an alternative form. Describing in words, the information provided in a semantic data schema, is an example of Interpreting. *Exemplifying*: This cognitive process occurs when the learner is able to provide an example of a general concept. Providing examples of each of the constructs is an illustration of Exemplifying. *Classifying*: This cognitive process occurs when the learner recognizes that an example belongs to a particular category. Categorizing the different components of a semantic data schema is an example of Classifying. *Summarizing*: This cognitive process occurs

when the learner can provide a précis of the general theme. Summarizing is not applicable in semantic data modeling because the definitions are short and do not need further summarization. *Inferring*: This cognitive process occurs when the learner makes logical conclusions based on facts presented. Inferring is not applicable because the definitions of the constructs are presented explicitly; they do not have to be inferred. *Comparing*: This cognitive process occurs when the learner detects correspondences between ideas in different situations. Comparing is not applicable since the definitions of the constructs are explicit and the objective at the Understand level is only to know the definition of the constructs. *Explaining*: This cognitive process occurs when the learner determines correspondences between effects and their causes. Explaining is not applicable in semantic data modeling because there are no cause effect relationships to be studied.

For semantic data modeling, at the Understand level, learners should comprehend the definition of each of the constructs. As discussed above, there are multiple applicable cognitive processes that fall within the Understand category. Of all the cognitive processes applicable at the Understand level, Exemplifying seemed the most appropriate because it can be measured independently of knowing a particular notation for ER modeling. In the other applicable cognitive processes, it is possible to appear to be at the Understand level because they remember the notation, even though they are really not at the Understand level.

To measure the Exemplifying cognitive process, the task questions asked the learner to provide multiple examples of each of the constructs in a familiar, but previously

unseen environment, a burger chain. This design minimized the possibility that the learners relied on memory of what was discussed in class (Remember level), and forced them to be at the Understand level to provide appropriate examples. If at least half the examples for a construct were correct, then the learner was said to have achieved Understand level for that construct.

Cognitive process	Definition and Items
Exemplifying	Finding a specific example that illustrates a concept 1. Give 3 examples of [construct name] in a *burger chain environment*. 2. Explain how foreign keys are represented in an ER diagram

Table 64: Measuring the Understand Level

4.1.1.2. Apply Level

The second expertise level to be measured is Apply. According to RBT, learners are at the Apply level, if they can perform a procedure in a particular context. There are two different cognitive processes that can be used to measure the Apply level. The cognitive processes are *Executing* and *Implementing*. In the rest of this section both cognitive processes will be briefly described. Then, the cognitive processes selected for this research will be outlined and justified.

In *Executing*, the learner is able to apply the procedure to a familiar task. In *Implementing*, the learner is able to select the appropriate procedure and apply it to an unfamiliar task. For semantic data modeling, there is a set procedure for determining two of the constructs—cardinality and identifiers. For each construct, there is only one applicable procedure and the task type is familiar. Therefore, only the *Executing* cognitive process is applicable.

To measure the *Executing* cognitive process, the questions asked the learner to apply the procedure for reading the cardinality (and identifiers) in multiple situations. If the learner recognizes that cardinality constrains how many entity instances are related through a relationship more than half the time, then the learner is said to have achieved the Apply level for cardinality. If the learner recognizes that the identifier uniquely distinguishes each entity instance more than half the time, then the learner is said to have achieved Apply level for identifiers.

Cognitive process	Definition and Items
Executing	Apply procedure to a familiar task 1. Determine [construct] in the following situation 2. For [construct instance], describe in 1-2 sentences what information is being captured

Table 65: Measuring the Apply Level

4.1.1.3. Analyze Level

The third level to be measured is Analyze. According to RBT, learners are at the Analyze level, if they can divide the material into constituent parts and know how the pieces relate to one another to achieve a particular purpose. There are three different cognitive processes that can be used to measure the Analyze level. The cognitive processes are *Differentiating, Organizing* and *Attributing*. In the rest of this section each of the cognitive processes will be briefly described. Then, the cognitive processes selected for this research will be outlined and justified.

In *Differentiating*, the learner is able to distinguish relevant and irrelevant information. For example, information relating to the *mini-world* or *processes* cannot be represented in the conceptual data model and hence should be recognized as irrelevant. In

Organizing, the learner is able to select the correct construct and place it correctly within the overall structure. In *Attributing*, the learner is able to identify the intent underlying the presented material. Attributing is not relevant in measuring expertise in semantic modeling constructs since the intent for creating a semantic data model is constant, i.e., to create a database that captures the user requirements.

To measure the Differentiating cognitive process, the task questions asked the learner to categorize presented information material into irrelevant or relevant. To measure the Organizing cognitive process, the learner is asked to identify the appropriate construct to model information. Multiple questions measured attainment of the organize level for different variants of each construct.

Cognitive process	Definition and Items
Differentiate	Distinguish relevant from irrelevant parts 1. Given a case description: Classify information in the following description into "can be modeled" and "cannot be modeled" by an ER diagram 2. Given a case description and a sample semantic schema representation: Are there any missing instances of [construct] in the diagram?
Organize	Determine how the pieces fit together 1. Given a case description: Which construct (and variant) is most appropriate 2. Given a case description and sample semantic schema: Evaluate whether there is any error in the diagram for this [construct] in capturing the requirements

Table 66: Measuring the Analyze Level

4.1.2. Task to Measure Performance

To measure the Performance or Errors, a requirements description for a real estate case was provided. The domain was chosen because it was unseen (had not been used in class discussions), it was realistic and easy for business school students to relate to.

4.1.3. Tasks to Measure Other Variables

In addition to measuring expertise levels and performance, other variables like individual learning styles, domain expertise, prior experience, demographic information were also measured. The detail and purpose behind measuring these variables is included below.

4.1.3.1. Prior Database and Domain Familiarity

Prior experience in domain, conceptual modeling, databases or related concepts could influence performance. They are measured using single item questions. A Likert scale was used for all the items. The scale and items are shown below

Scale	Items
Domain familiarity	How familiar are you with the real estate domain?
Prior Database familiarity	How much do you know about databases?
Prior ER modeling familiarity	What is your familiarity with ER modeling?
Prior CASE tool familiarity	What is your familiarity with CASE tools (like ERWin)?
Prior Object-oriented programming familiarity	What is your familiarity with Object Oriented programming (OO)?
Prior UML familiarity	What is your familiarity with UML?

Table 67: Items measuring Database and Domain Familiarity

4.1.3.2. ER modeling approach

Different approaches to modeling may result in different biases and errors. Therefore, two items were included to capture variances in converting requirements to a conceptual schema.

Scale	Items
Process of creating ER model	Which of the following most closely resembles the order of converting the requirements to an ER model: "Read the whole description, identify entity classes, then add the attributes for entity classes and then identify relationships," "Read the whole description, identify the entity classes, then identify the relationships, then add attributes for both entity classes and relationships," "Read a sentence and convert information about entity classes, attributes and relationships contained in it before moving on to the next sentence," "Other (please explain)"
Process of determining cardinality	Which of the following most closely approximates when you determine cardinality: "Determine cardinality for a relationship immediately after identifying the relationship," "Determine cardinality at the end, after identifying all the relationships," "Other (please explain)"

Table 68: Items Measuring Process Of Creating ER Model

4.1.3.3. Learning Styles

Different learning styles may result in different biases and cause errors. For example, since ER modeling is a visual task, visual learners may make fewer errors than verbal learners. Therefore, this research measured learning styles to control for learning styles, using an established instrument 'Index of Learning Styles' (Felder and Silverman 1988, Felder and Spurlin 2005).

Scale	Sample Items
Visual-Verbal	1. I remember best a. what I see. b. what I hear.
Sensing-Intuitive	1. I am more likely to be considered a. careful about the details of my work. b. creative about how to do my work.

Table 69: Items Measuring Learning Styles

4.1.3.4. Demographic Information

Demographic information was measured using single item questions. The scale and items are shown below

Scale	Items
Gender	What is your gender? Male, Female
Age	What is your age group? 18-20 yrs, 21-24 yrs, 25-30 yrs, 31 yrs or over
Class	What is your Class Level? Junior, Senior, Other, please specify
Work experience	How many years of full time professional work experience do you have? 0 months, 1 – 6 months, 7 – 11 months, 1 – 2 years, 3 – 5 years, Greater than 5 years
Major	Your major is _____

Table 70: Items Measuring Demographic Information

4.2. Sampling and Data Collection

The population of interest is novice analysts, who have completed their database design training in university. The participants were recruited from a pool of undergraduate business school students enrolled in database design classes in business schools. The study was conducted near the end of the semester to allow students to gain some proficiency with semantic data modeling and to understand how semantic modeling fits into the overall database design lifecycle. This also ensured that the students received the usual amount of learning, feedback and error-correction they normally would have over the course of a semester (i.e., the errors recorded in our test were not caused by an

induced insufficient level of database training). Naturally, we (the researcher) were not involved in the design or the instruction of the database course (i.e., the course was taught as it normally would have been) and thus the results of our laboratory study was not biased on that account. Each of the database design classes was the only one offered in the curriculum. Participation was voluntary. Students were offered $10 for participating in the study. If they were willing to think aloud while they performed the task and be recorded, they were offered an extra monetary incentive of $10. Data was gathered over two semesters to achieve a sufficient sample size of fifty one. A follow up interview was done, in the second semester, to gather qualitative data about the errors and the reasons behind them.

4.3. Summary

This chapter explained the methodology used in this research study. We provided a detailed description of the constructs, their development or their sources. The key variables in our research are expertise, errors and control variables like learning style, prior experience, domain familiarity and ER modeling approach. We provided a detailed description of our methodology for applying the Revised Bloom's Taxonomy to measure expertise levels. Since expertise has not been examined as a construct in the past, it was important to document the process to allow subsequent research to replicate our work. We used a carefully designed and pre-tested questionnaire for collecting data. The analysis and results are discussed in the next chapter.

CHAPTER 5: DATA ANALYSIS AND RESULTS

Chapter 3 described the overall research model and how the hypotheses were developed. Chapter 4 described the methodology for designing and collecting the data. This chapter describes the results of the data analysis (sections 5.1 and 5.2) and uses both the qualitative and quantitative data gathered to discuss the significance of our results (section 5.3). Section 5.5 applies the results in previous sections to evaluate the cost of different expertise levels. The next chapter looks at the implications and limitations of our research.

5.1. Initial Analysis

The data for the items was coded by two raters. Inter-rater reliability was calculated using Cohen's kappa. Inter-rater was found to be at least 0.8 for all variables. Any differences in coding between the raters were examined and resolved.

5.1.1. Demographic Information

The participants were 51 undergraduate business school students enrolled in database design classes in business schools of US Universities. 90% of the participants were between 21 and 30 years of age; 85% were male; 80% had some full time work experience.

5.2. Detailed Analysis

Each of the hypotheses was examined in turn. We start by examining Bloom's taxonomy verification (BV) hypotheses (section 5.2.1) which test the hierarchical nature of the Revised Bloom's taxonomy for ER modeling. The results for the BV hypotheses

are summarized in Table 71. Then, we examine associations among the different ER modeling constructs using the Relationships among Constructs (RC) hypotheses in section 5.2.2. The results of the RC hypotheses are summarized in Table 84. In section 5.2.3 we discuss how expertise levels predict errors in the Error Prediction hypotheses, as summarized in Table 86.

5.2.1. Bloom's Taxonomy Verification Hypotheses

The analysis and discussion for verifying each of the hypotheses for the hierarchical nature of Bloom's taxonomy are given below.

H#	Hypothesis description	Result
BV1	Remember(Foreign keys)→ Understand(Foreign keys)	Unverified
BV2	Understand(Foreign keys) → Differentiate(Foreign keys)	Supported
BV3	Understand(Entity classes) → Differentiate(Entity classes)	Unverified
BV4	Understand(Attributes) → Differentiate(Attributes)	Unverified
BV5	Understand(Relationships) → Differentiate(Relationships)	Unverified
BV6	Apply(Cardinality) → Differentiate(Cardinality)	Unverified
BV7	Understand(Identifiers) → Differentiate(Identifiers)	Unverified
BV8	Understand(Cardinality) → Apply(Cardinality)	Unverified
BV9	Understand(Attributes) → Organize(Attributes)	Unverified
BV10	Understand(Entity classes) → Organize(Entity classes)	Unverified
BV11	Understand(Relationships) → Organize(Relationships)	Unverified
BV12	Apply(Cardinality) → Organize(Cardinality)	Unverified

Table 71: Bloom's Taxonomy Verification Hypotheses Summary

Bloom's taxonomy verification hypothesis 1: A novice should know what foreign keys are before she understands that foreign keys cannot be used.

Data for five novices for at least one of the variables was missing. For the remaining 46, all novices remembered what foreign keys were. Therefore, the hypothesis could not be verified because of the limited range of knowledge of novices along the remember dimension. Not all novices who were at the remember level were also at the Understand

level. Fifteen novices who were at the remember level were not at the Understand level while thirty one novices who were at the remember level were at the Understand level. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	46

a. No statistics are computed because "Remember level for foreign key attained?" is a constant.

Table 72: Bloom's Taxonomy Verification Hypothesis 1: Remember Before Understand For Foreign Keys

Bloom's taxonomy verification hypothesis 2: A novice should be at the Understand level for foreign keys before she is at the Differentiate sublevel for foreign keys.

Data was missing for three novices for at least one of the variables. For the remaining forty eight, fifteen novices did not understand foreign keys, while thirty three did. Six out of the fifteen novices not at the Understand level were not at the Differentiate sublevel. Thirty out of the thirty three novices who were at the Understand level for foreign keys were also at the Differentiate sublevel. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.018).

This implies that it is necessary to be at the Understand level to be at the Differentiate sublevel for foreign keys. Since our prediction is that the Understand level is necessary, but not sufficient, we expect that (a) if the novice is at the Understand level, she may or may not be at the Differentiate sublevel and (b) if the novice is not at the Understand level, she should not be at the Differentiate sublevel. Therefore, we do not calculate any directional measures of association.

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	6.467[b]	1	.011		
Continuity Correction[a]	4.597	1	.032		
Likelihood Ratio	6.031	1	.014		
Fisher's Exact Test				.018	.018
Linear-by-Linear Association	6.332	1	.012		
N of Valid Cases	48				

a. Computed only for a 2x2 table
b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 2.81.

Table 73: Bloom's Taxonomy Verification Hypothesis 2: Understand Before Differentiate For Foreign Keys

Bloom's taxonomy verification hypothesis 3: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel for entity classes.

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. However, that not all novices who were at the Understand level were also at the Differentiate sublevel for entity classes. Fifteen novices who were at the Understand level were not at the Differentiate sublevel while thirty five novices who were at the Understand level were also at the Differentiate sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	50

a. No statistics are computed because "Understand level for entity class" is a constant.

Table 74: Bloom's Taxonomy Verification Hypothesis 3: Understand Before Differentiate For Entity classes

This hypothesis can be further separated into the following two hypotheses:

Bloom's taxonomy verification hypothesis 3a: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel (relevant) for entity classes.

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. The Differentiate sublevel (relevant) indicates whether a novice can identify relevant information for entity classes. Not all novices who were at the Understand level were also at the Differentiate sublevel (relevant) for entity classes. Two novices were not at the Differentiate sublevel (relevant) for entity classes, while forty eight novices were.

Bloom's taxonomy verification hypothesis 3b: A novice should be at the Understand level for entity classes before she is at the Differentiate sublevel (irrelevant) for entity classes.

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. The Differentiate sublevel (irrelevant) indicates whether a novice can identify information irrelevant for modeling entity classes. Not all novices who were at the Understand level were also at the Differentiate sublevel (irrelevant) for entity classes. Thirteen novices were not at the Differentiate sublevel (irrelevant) for entity classes, while thirty seven novices were.

Bloom's taxonomy verification hypothesis 4: A novice should be at the Understand level for attributes before she is at the Differentiate sublevel (relevant) for attributes.

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for attributes. Therefore, the hypothesis could not be verified. The Differentiate sublevel (relevant) indicates whether a novice can identify relevant information for attributes. Not all novices who were at the Understand level were also at the Differentiate sublevel for attributes. Five novices who were at the Understand level were not at the Differentiate sublevel while forty five novices who were at the Understand level were also at the Differentiate sublevel. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	50

a. No statistics are computed because Understand level for attribute is a constant.

Table 75: Bloom's Taxonomy Verification Hypothesis 4: Understand Before Differentiate For Attributes

Bloom's taxonomy verification hypothesis 5: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel for interaction relationships.

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for relationships. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Differentiate sublevel for relationships. Thirty six novices who were at the Understand level were not at the Differentiate sublevel while fourteen novices who were

at the Understand level were also at the Differentiate sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	50

a. No statistics are computed because "Understand level for relationships" is a constant.

Table 76: Bloom's Taxonomy Verification Hypothesis 5: Understand Before Differentiate For Relationships

This hypothesis can be further separated into the following two hypotheses:

Bloom's taxonomy verification hypothesis 5a: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel (relevant) for interaction relationships.

Data was missing for two novices for at least one of the variables. Of the remaining novices, all fifty were at the Understand level for relationships. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Differentiate sublevel (relevant) for relationships. The Differentiate sublevel (relevant) indicates whether a novice can identify relevant information for relationships. Nineteen novices who were at the Understand level were not at the Differentiate sublevel while thirty one five novices who were at the Understand level were also at the Differentiate sublevel.

Bloom's taxonomy verification hypothesis 5b: A novice should be at the Understand level for interaction relationships before she is at the Differentiate sublevel (irrelevant) for interaction relationships.

All fifty one novices were at the Understand level for relationships, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Differentiate sublevel (irrelevant) for relationships. Therefore, the hypothesis could not be verified. The Differentiate sublevel (relevant) indicates whether a novice can identify relevant information for relationships. Twenty one novices who were at the Understand level were not at the Differentiate sublevel while twenty nine novices who were at the Understand level were also at the Differentiate sublevel.

Bloom's taxonomy verification hypothesis 6: A novice should be at the Apply level for cardinality before she is at the Differentiate sublevel for cardinality.

All fifty one novices were at the Apply level for cardinality. Therefore, the hypothesis could not be verified. Twelve novices who were at the Understand level were not at the Differentiate sublevel while thirty nine novices who were at the Understand level were also at the Differentiate sublevel. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because “Apply level for cardinality” is a constant.

Table 77: Bloom’s Taxonomy Verification Hypothesis 6: Apply Before Differentiate For Cardinality

Bloom's taxonomy verification hypothesis 7: A novice should be at the Understand level for identifiers before she is at the Differentiate sublevel for identifiers.

All fifty one novices were at the Understand level for identifiers. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Differentiate sublevel for identifiers. Six novices who were at the Understand

level were not at the Differentiate sublevel while forty five novices who were at the Understand level were also at the Differentiate sublevel. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because "Understand level for identifier" is a constant.

Table 78: Bloom's Taxonomy Verification Hypothesis 7: Understand Before Differentiate For Identifiers

Bloom's taxonomy verification hypothesis 8: A novice should be at the Understand level for cardinality before she is at the Apply level for cardinality.

All fifty one novices were at the Understand and the Apply levels for cardinality. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because "Understand level for cardinality" is a constant.

Table 79: Bloom's Taxonomy Verification Hypothesis 8: Understand Before Differentiate For Cardinality

Bloom's taxonomy verification hypothesis 9: A novice should be at the Understand level for attributes before she can correctly associate attributes with entity classes (Organize sublevel).

Data was missing for one novice for at least one of the variables. For the remaining fifty novices, all novices were at the Understand level for attributes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Organize sublevel for attributes. One novice who was at the Understand level

were not at the Organize sublevel while fifty novices who were at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	50

a. No statistics are computed because "Understand level for attributes" is a constant.

Table 80: Bloom's Taxonomy Verification Hypothesis 9: Understand Before Organize For Attributes

Bloom's taxonomy verification hypothesis 10: A novice should be at the Understand level for entity classes before she is at the Organize sublevel for entity classes.

All fifty one novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Organize sublevel for entity classes. Fifty novices who were at the Understand level were not at the Organize sublevel while one novice who was at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because "Understand level for entity class" is a constant.

Table 81: Bloom's Taxonomy Verification Hypothesis 10: Apply Before Organize For Entity Classes

This hypothesis can be further separated into the following two hypotheses:

Bloom's taxonomy verification hypothesis 10a: A novice should be at the Understand level for entity classes before she can correctly choose between strong and weak entity classes (Organize sublevel).

All fifty one novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level could correctly choose between strong and weak entity classes, i.e., were also at the Organize sublevel for entity classes. Thirty eight novices who were at the Understand level were not at the Organize sublevel while thirteen novices who were at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Bloom's taxonomy verification hypothesis 10b: A novice should be at the Understand level for entity classes before she can correctly decide when to combine and when to separate entity classes (Organize sublevel).

All fifty one novices were at the Understand level for entity classes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level could correctly decide when to combine / separate entity classes, i.e., were also at the Organize sublevel for entity classes. Thirty six novices who were at the Understand level were not at the Organize sublevel while fifteen novices who were at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Bloom's taxonomy verification hypothesis 11: A novice should be at the Understand level for relationships before she can correctly organize information shared by entity classes into appropriate relationships (Organize sublevel).

All fifty one novices were at the Understand level for interaction relationships. Therefore, the hypothesis could not be verified. Not all novices who were at the

Understand level could correctly organize information shared by entity classes into relationships, i.e., were also at the Organize sublevel for relationships. Twenty two novices who were at the Understand level were not at the Organize sublevel while twenty nine novices who were at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because "Understand level for relationship" is a constant.

Table 82: Bloom's Taxonomy Verification Hypothesis 11: Apply Before Organize For Relationships

Bloom's taxonomy verification hypothesis 12: A novice should be at the Apply level for cardinality before she can correctly organize cardinality for a relationship (Organize sublevel).

	Value
Pearson Chi-Square	.[a]
N of Valid Cases	51

a. No statistics are computed because "Apply level for cardinality" is a constant.

Table 83: Bloom's Taxonomy Verification Hypothesis 12: Apply Before Organize For Cardinality

All fifty one novices were at the Understand level for entity classes and relationships and attributes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Organize sublevel for relationships. Twenty two novices who were at the Understand level were not at the Organize sublevel while twenty nine novices who were at the Understand level were also at the Organize sublevel. So,

this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

This hypothesis can be separated into the following two hypotheses:

Bloom's taxonomy verification hypothesis 12a: A novice should be at the Apply level for cardinality before she can correctly organize cardinality for a binary relationship (Organize sublevel).

All fifty one novices were at the Understand level for entity classes and relationships and attributes. Therefore, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Organize sublevel for binary relationships. Six novices who were at the Understand level were not at the Organize sublevel while forty five novices who were at the Understand level were also at the Organize sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Bloom's taxonomy verification hypothesis 12b: A novice should be at the Apply level for cardinality before she can correctly organize cardinality for a ternary relationship (Organize sublevel).

Since all fifty one novices were at the Understand level for entity classes and relationships and attributes, the hypothesis could not be verified. Not all novices who were at the Understand level were also at the Organize sublevel for ternary relationships. Twenty two novices who were at the Understand level were not at the Organize sublevel while twenty nine novices who were at the Understand level were also at the Organize

sublevel. So, this hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

5.2.2. Relationship Among Constructs Hypotheses

The analysis and discussion for verifying each of the hypotheses for the hypothesized relationship among the constructs are given below.

H#	Hypothesis description	Result
RC1	Understand(Relationships)→ Understand(Cardinality)	Unverified
RC2	Understand(Entity classes) → Understand(Relationships)	Unverified
RC3	Understand(Attributes) → Understand (Identifiers)	Unverified
RC4	Understand(Relationships) → Understand(Foreign keys)	Unverified
RC5	Differentiate(Entity classes) → Differentiate(Relationships)	Supported

Table 84: Relationship Among Constructs Hypotheses Summary

Relationship among constructs hypothesis 1: A novice should be at the Understand level for interaction relationships before she is at the Understand level for cardinality.

All novices were at the Understand level for interaction relationships and for cardinality. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Relationship among constructs hypothesis 2: A novice should be at the Understand level for entity classes before she is at the Understand level for relationships.

All novices were at the Understand level for interaction relationships and for entity classes. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Relationship among constructs hypothesis 3: A novice should be at the Understand level for attributes before she is at the Understand level for identifiers.

All novices were at the Understand level for attributes and for identifiers. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Relationship among constructs hypothesis 4: A novice should be at the Understand level for interaction relationships before she is at the Understand level for foreign keys.

All novices were at the Understand level for interaction relationships and for foreign keys. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels.

Relationship among constructs hypothesis 5: A novice should be at the Differentiate sublevel for entity classes before she is at the Differentiate sublevel for interaction relationships.

Data was missing for one novice for at least one of the variables. For the remaining fifty, fifteen novices were not at the Differentiate sublevel for entity classes, while thirty six were at the Differentiate sublevel. All fifteen novices not at the Differentiate sublevel for entity classes were not at the Differentiate sublevel for interaction relationships. Fourteen out of the thirty six novices who were at the Differentiate sublevel for entity classes were also at the Differentiate sublevel for interaction relationships. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.004).

	Value	Df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	8.333[b]	1	.004		
Continuity Correction[a]	6.467	1	.011		
Likelihood Ratio	12.185	1	.000		
Fisher's Exact Test				.004	.002
Linear-by-Linear Association	8.167	1	.004		
N of Valid Cases	50				

a. Computed only for a 2x2 table
b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 4.20.

Table 85: Relationship Among Constructs Hypothesis: Differentiate for Entity Classes Before Differentiate For Relationships

This implies that it is necessary to be at the Differentiate sublevel for entity classes to be at the Differentiate sublevel for interaction relationships. Since our prediction is that the Differentiate sublevel for entity classes is necessary, but not sufficient, we expect that (a) if the novice is not at the Differentiate sublevel for entity classes, she may or may not be at the Differentiate sublevel for interaction relationships and (b) if the novice is not at the Differentiate sublevel for entity classes, she should not be at the Differentiate sublevel for interaction relationships. Therefore, we do not calculate any directional measures of association.

5.2.3. Error Prediction Hypotheses

The analysis and discussion for verifying each of the hypotheses for the error prediction are given below. The frequency of each error (or group of errors) predicted by a hypothesis is recorded to get an estimate of the frequency of errors in the sample.

H#	Hypothesis description	Result
EP1	*Not* Apply(Cardinality)→ Incorrect (Cardinality)	Unverified
EP2a	*Not* Differentiate $_{Relevant}$ (Entity class)→ Miss (Entity class)	Unverified
EP2b	*Not* Differentiate $_{Irrelevant}$ (Entity class)→ Extra (Entity class)	Supported
EP3a	*Not* Differentiate $_{Relevant}$ (Relationship)→ Miss (Relationship)	Not supported
EP3b	*Not* Differentiate $_{Irrelevant}$ (Relationship)→ Extra (Relationship) *Not* Differentiate $_{Irrelevant}$ (Relationship)→ Error (Relationship: Binary as Ternary)	Supported Not supported
EP4	*Not* Differentiate $_{Irrelevant}$ (Foreign key) → Miss(Relationship) / Error(Relationship: Replace Entity class with Foreign key)	Supported
EP5	*Not* Differentiate $_{Relevant}$ (Attribute) → Miss (Attribute)	Not supported
EP6	*Not* Differentiate $_{Relevant}$ (Cardinality) → Miss(Cardinality)	Not supported
EP7	*Not* Differentiate $_{Relevant}$ (Identifier) → Miss(Identifier)	Not supported
EP8	*Not* Organize $_{Grouping}$(Attribute) → Error(Attribute: associate attribute with wrong entity class error)	Unverified
EP9a	*Not* Organize $_{Type}$(Entity class) → Error(Entity class: strong-weak distinction error)	Supported
EP9b	*Not* Organize $_{Grouping}$ (Entity class) → Error(Entity class: combine entity classes)	Not supported
EP10	*Not* Organize $_{Grouping}$ (Relationship) → Error(Ternary as two binary)	Supported
EP11	*Not* Organize $_{Grouping}$ (Cardinality) → Error(Cardinality)	Not supported
EP12	Knowledge(Application Domain) → No impact on error	Supported
EP13	Knowledge(Objected Oriented Programming) → No impact on error	Supported
EP14	Prior Experience(ER modeling) → Impacts Errors	Not supported
EP15	Familiarity(UML) → No impact on error	Supported
EP16	Type of learner (Visual vs. Verbal) → Impacts Errors	Not supported
EP17	Type of learner (Intuitive vs. Sensing) → Impacts Errors	Not supported
EP18	Process of Identifying Attributes → Impacts Errors	Not supported
EP19	Process of Identifying Cardinality → Impacts Errors	Not supported

Table 86: Error Prediction Hypotheses Summary

Error Prediction Hypothesis 1: If a novice is not at the Apply level for cardinality, she will incorrectly determine cardinality through the ER schema

The frequency of incorrect cardinality in this sample was low (0.17). All novices were at the Apply level. Therefore, this hypothesis could not be verified.

Error Prediction Hypothesis 2: If a novice is not at the Differentiate sublevel within Analyze for entity classes, she will (a) add extra irrelevant entity classes and / or (b) miss relevant entity classes.

This can be examined in further detail as:

Error Prediction Hypothesis 2a: If a novice is not at the Differentiate sublevel (relevant) within Analyze for entity classes, she will miss relevant entity classes.

The frequency of missing relevant entity classes in this sample was low (zero). Data was missing for one novice for at least one of the variables. Of the remaining fifty novices, forty eight were at the Differentiate sublevel (relevant) and two were not. None of the fifty novices made the error. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels and errors.

Error Prediction Hypothesis 2b: If a novice is not at the Differentiate sublevel (irrelevant) within Analyze for entity classes, she will add extra irrelevant entity classes.

The frequency of adding extra entity classes in this sample was low (0.16). Data was missing for one novice for at least one of the variables. For the remaining fifty novices, thirteen novices were not at the Differentiate sublevel for entity classes, while thirty seven were at the Differentiate sublevel. Of the thirteen novices were not at the Differentiate sublevel, six made the error of adding extra irrelevant entity classes and seven did not. Of the thirty seven novices who were at the Differentiate sublevel for entity classes, thirty five did not make the error while two did. Since one of the cells had

an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value for one sided test 0.002).

In addition, we calculated Lambda (λ). It was 0 for the Differentiate sublevel (irrelevant) predicting the error of adding extra irrelevant entity classes. It was 0.31 for the error of adding the extra irrelevant entity class predicting being at the Differentiate sublevel. This implies that we make 31% fewer errors predicting whether the novice is at the Differentiate sublevel (irrelevant) for entity classes given that we know whether she added irrelevant entity classes than we would if we did not know whether she made the error.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	11.885[b]	1	.001		
Continuity Correction[a]	9.046	1	.003		
Likelihood Ratio	10.461	1	.001		
Fisher's Exact Test				.002	.002
Linear-by-Linear Association	11.647	1	.001		
N of Valid Cases	50				

a. Computed only for a 2x2 table

b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 2. 08.

Table 87: Error Prediction Hypothesis 2: Not at Differentiate for Entity Classes Results in Extra Entity Class error

Error Prediction Hypothesis 3: If a novice is not at the Differentiate sublevel within Analyze for interaction relationships, she will (a) add extra irrelevant interaction relationships and / or (b) miss relevant interaction relationships and / or (c) will incorrectly capture binary relationships as ternary relationships

This can be examined in further detail as:

Error Prediction Hypothesis 3a: If a novice is not at the Differentiate sublevel (relevant) within Analyze for interaction relationships, she will miss relevant interaction relationships

The frequency of missing interaction relationships in this sample was low (0.2). Data was missing for one novice for at least one of the variables. For the remaining fifty novices, nineteen novices were not at the Differentiate sublevel for entity classes, while thirty one were at the Differentiate sublevel. Of the nineteen novices not at the Differentiate sublevel, three made the error of adding extra irrelevant relationships and sixteen did not. Of the thirty one novices who were at the Differentiate sublevel for relationships, twenty four did not make the error while seven did. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value for one sided test is 0.421).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.340[b]	1	.560		
Continuity Correction[a]	.048	1	.827		
Likelihood Ratio	.348	1	.555		
Fisher's Exact Test				.722	.421
Linear-by-Linear Association	.333	1	.564		
N of Valid Cases	50				

a. Computed only for a 2x2 table

b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 3. 80.

Table 88: Error Prediction Hypothesis 3a: Not at Differentiate for Relationships Results in Missing Relationship

Error Prediction Hypothesis 3b: If a novice is not at the Differentiate sublevel (irrelevant) within Analyze for interaction relationships, she will (a) add extra irrelevant interaction relationships and / or (b) will incorrectly include irrelevant entity classes in relationships

For (a) add extra relationships: The frequency of adding extra interaction relationships in this sample was low (0.08). Data was missing for one novice for at least one of the variables. For the remaining fifty novices, twenty one novices were not at the Differentiate sublevel for relationships, while twenty nine were at the Differentiate sublevel. Of the twenty one novices not at the Differentiate sublevel, four made the error of adding extra irrelevant relationships and seventeen did not. Of the twenty nine novices who were at the Differentiate sublevel for relationships, all twenty nine did not make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.026).

In addition, we calculated λ. It was 0 for the Differentiate sublevel (relevant) for interaction relationships predicting the error of missing interaction relationships. It was 0.19 for the error of missing interaction relationships predicting being at the Differentiate sublevel. This implies that we make 19% fewer errors predicting whether the novice is at the Differentiate sublevel (relevant) for relationships given that we know whether she made an error of missing interaction relationships than we would if we did not know whether she made the error.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	6.004[b]	1	.014		
Continuity Correction[a]	3.695	1	.055		
Likelihood Ratio	7.427	1	.006		
Fisher's Exact Test				.026	.026
Linear-by-Linear Association	5.884	1	.015		
N of Valid Cases	50				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1. 68.

Table 89: Error Prediction Hypothesis 3b: Not at Differentiate for Relationships Results in Extra Relationship

For (b) include irrelevant entity classes in relationships: The error would include adding an irrelevant entity class in a binary to make it a ternary relationship and substituting an entity class with a wrong entity class in a relationship.

The frequency of irrelevant entity classes in interaction relationships in this sample was low (0.16). Data was missing for one novice for at least one of the variables. For the remaining fifty novices, twenty one novices were not at the Differentiate sublevel for relationships, while twenty nine were at the Differentiate sublevel. Of the twenty one novices not at the Differentiate sublevel, three made the error of adding irrelevant entity classes to relationships and eighteen did not. Of the twenty nine novices who were at the Differentiate sublevel for relationships, five made the error of adding irrelevant entity classes while twenty four did not make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.549).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.079[b]	1	.778		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.080	1	.777		
Fisher's Exact Test				1.000	.549
Linear-by-Linear Association	.078	1	.781		
N of Valid Cases	50				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 3. 36.

Table 90: Error Prediction Hypothesis 3b: Not at Differentiate for Relationships Results in Extra Entity Class in Relationships

Error Prediction Hypothesis 4: If a novice is not at the Differentiate sublevel within Analyze for foreign keys, she will (a) include foreign keys redundantly (in addition to capturing the relationship) in the ER schema and / or (b) replace relevant interaction relationships or an entity class in an interaction relationship by foreign keys

The frequency of including foreign keys in this sample was medium (0.47). Data was missing for two novices for at least two of the variables. For the remaining forty nine novices, nine novices were not at the Differentiate sublevel for entity classes, while forty were at the Differentiate sublevel. Of the nine novices not at the Differentiate sublevel, seven made the error of adding extra irrelevant relationships and two did not. Of the forty novices who were at the Differentiate sublevel for foreign keys, sixteen made the error and twenty four did not make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.045).

In addition, we calculated λ. It was 0.22 for the Differentiate sublevel (relevant) for foreign keys predicting the error of including foreign keys irrelevantly or in place of relationships. It was 0 for the error of including foreign keys irrelevantly or in place of relationships predicting being at the Differentiate sublevel. This implies that we make 22% fewer errors predicting whether the novice made the error of including foreign keys irrelevantly or in place of relationships given that we know whether she is at the Differentiate sublevel (irrelevant) for foreign keys than we would if we did not know whether she was at the Differentiate sublevel.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	4.210[b]	1	.040		
Continuity Correction[a]	2.830	1	.093		
Likelihood Ratio	4.369	1	.037		
Fisher's Exact Test				.064	.045
Linear-by-Linear Association	4.124	1	.042		
N of Valid Cases	49				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 4. 22.

Table 91: Error Prediction Hypothesis 4: Not At Differentiate For Foreign Keys Results In Including Foreign Keys Redundantly / In Place Of Entity Class(es) In A Relationship

Error Prediction Hypothesis 5: If a novice is not at the Differentiate sublevel within Analyze for attributes, she will miss marking attributes.

The frequency of missing attributes in this sample was low (0.06). Data was missing for one novice for at least one of the variables. For the remaining fifty novices, five novices were not at the Differentiate sublevel for attributes, while forty five were at the

Differentiate sublevel. Of the five novices not at the Differentiate sublevel, none made the error of adding extra attributes and five did not. Of the forty five novices who were at the Differentiate sublevel for attributes, forty two did not make the error while three novices did make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.724).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.355[b]	1	.552		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.653	1	.419		
Fisher's Exact Test				1.000	.724
Linear-by-Linear Association	.348	1	.556		
N of Valid Cases	50				

a. Computed only for a 2x2 table

b. 3 cells (75.0%) have expected count less than 5. The minimum expected count is .30.

Table 92: Error Prediction Hypothesis 5: Not at Differentiate for Attributes Results in Missing Attributes

Error Prediction Hypothesis 6: If a novice is not at the Differentiate sublevel within Analyze for cardinality, she will miss marking cardinality.

The frequency of missing cardinality in this sample was low (0.12). Of the fifty one novices, twelve novices were not at the Differentiate sublevel for cardinality, while thirty nine were at the Differentiate sublevel. Of the twelve novices not at the Differentiate sublevel, one made the error of adding extra irrelevant relationships and eleven did not. Of the thirty nine novices who were at the Differentiate sublevel for attributes, thirty five

did not make the error while five novices did make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.189).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.477[b]	1	.224		
Continuity Correction[a]	.778	1	.378		
Likelihood Ratio	1.467	1	.226		
Fisher's Exact Test				.320	.189
Linear-by-Linear Association	1.448	1	.229		
N of Valid Cases	51				

a. Computed only for a 2x2 table

b. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5. 18.

Table 93: Error Prediction Hypothesis 5: Not at Differentiate for Cardinality Results in Missing Cardinality

Error Prediction Hypothesis 7: If a novice is not at the Differentiate sublevel within Analyze for identifiers, she will miss marking identifiers.

The frequency of missing identifiers in this sample was medium (0.32). Of the fifty one novices, six novices were not at the Differentiate sublevel for identifiers, while forty five were at the Differentiate sublevel. Of the six novices not at the Differentiate sublevel, two made the error of missing identifiers and four did not. Of the forty five novices who were at the Differentiate sublevel for identifiers, thirty one did not make the error while fourteen novices did make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.621).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.012[b]	1	.912		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.012	1	.913		
Fisher's Exact Test				1.000	.621
Linear-by-Linear Association	.012	1	.913		
N of Valid Cases	51				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1. 88.

Table 94: Error Prediction Hypothesis 5: Not at Differentiate for Identifiers Results in Missing Identifiers

Error Prediction Hypothesis 8: If a novice is not at the Organize sublevel within Analyze for attributes, she will incorrectly associate attributes of one strong entity classes with another strong entity class.

No one made the error of associating attributes (non identifier attributes) of one strong entity class with another. Therefore, the hypothesis could not be verified. This hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels and errors.

Error Prediction Hypothesis 9a: If a novice is not at the Organize sublevel within Analyze for entity classes, she will incorrectly mark strong entity classes as weak and weak entity classes as strong.

The frequency of incorrectly marking weak entity classes as strong in this sample was low (0.18). Data was missing for five novices for at least one of the variables. For the remaining forty six novices, seven novices were not at the Organize sublevel for entity

classes (strong-weak distinction), while thirty nine were at the Organize sublevel. Of the seven novices not at the Organize sublevel, five made the error of incorrectly marking strong entity classes as weak and two did not. Of the thirty nine novices who were at the Organize sublevel for entity classes, thirty two did not make the error while seven novices did make the error. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.009).

None of the novices made the error of marking strong entity classes as weak. Therefore, the second part of the hypothesis could not be verified. This part of the hypothesis could be explored in the future by sampling among novices with a wider range of expertise levels and errors.

In addition, we calculated λ. It was 0.25 for the organize level for entity classes predicting the error of marking weak entity classes as strong. It was 0 for the error of weak entity class as strong predicting being at the organize level. This implies that we make 25% fewer errors predicting whether the novice made the error of marking weak entity classes as strong given that we know whether she is at the organize level for entity classes than we would know if we did not know whether she was at the organize level.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	8.803[b]	1	.003		
Continuity Correction[a]	6.248	1	.012		
Likelihood Ratio	7.721	1	.005		
Fisher's Exact Test				.009	.009
Linear-by-Linear Association	8.612	1	.003		
N of Valid Cases	46				

a. Computed only for a 2x2 table

b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 1. 83.

Table 95: Error Prediction Hypothesis 9a: Not at Organize for Entity Classes Results in Distinguishing Strong and Weak Entity Classes

Error Prediction Hypothesis 9b: If a novice is not at the Organize sublevel within Analyze for entity classes, she will incorrectly associate attributes of a strong / weak entity classes with another strong / weak entity class; i.e., she will incorrectly combine entity classes.

The frequency of incorrectly combining entity classes in this sample was medium (0.37). Of the fifty one novices, twelve novices were not at the Organize sublevel for entity classes, while thirty nine were at the Organize sublevel. Of the twelve novices not at the Organize sublevel, six made the error of combining information across entity classes and six did not. Of the thirty nine novices who were at the Organize sublevel for entity classes, twenty six did not make the error while thirteen novices did make the error. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.239).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.090[b]	1	.296		
Continuity Correction[a]	.494	1	.482		
Likelihood Ratio	1.067	1	.302		
Fisher's Exact Test				.325	.239
Linear-by-Linear Association	1.069	1	.301		
N of Valid Cases	51				

a. Computed only for a 2x2 table
b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 4.47.

Table 96: Error Prediction Hypothesis 9b: Not at Organize for Entity Classes Results in Combining Entity Classes

Error Prediction Hypothesis 10: If a novice is not at the Organize sublevel within Analyze for interaction relationships, she will incorrectly capture ternary relationships as two binary relationships.

The frequency of ternary relationships as two binary relationships in this sample was low (0.24). Of the fifty one novices, twenty two novices were not at the Organize sublevel for relationships, while twenty seven were at the Organize sublevel. Of the twenty two novices not at the Organize sublevel, ten made the error of incorrectly marking strong entity classes as weak and twelve did not. Of the twenty nine novices who were at the Organize sublevel for entity classes, twenty seven did not make the error while two novices did make the error. Since none of the cells had an expected value of less than 5, the Pearson Chi square test was used. The test was significant at the 0.05 level (p value=0.001).

In addition, we calculated λ. It was 0 for the organize level for interaction relationships predicting the error of ternary relationships as two binary relationships. It was 0.36 for the error of ternary relationships as two binary relationships predicting being

at the organize level. This implies that we make 36% fewer errors predicting whether the novice is at the organize level given that we know whether she has made the error of ternary relationships as two binary relationships than we would know if we did not know whether she made the error.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	10.337[b]	1	.001		
Continuity Correction[a]	8.305	1	.004		
Likelihood Ratio	10.779	1	.001		
Fisher's Exact Test				.002	.002
Linear-by-Linear Association	10.134	1	.001		
N of Valid Cases	51				

a. Computed only for a 2x2 table

b. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5. 18.

Table 97: Error Prediction Hypothesis 10: Not at Organize for Relationships Results in Capturing Ternary Relationships as Binary Relationships

Error Prediction Hypothesis 11: If a novice is not at the Organize sublevel within Analyze for cardinality, she will incorrectly capture cardinality for relationships.

The frequency of incorrect cardinality in this sample was low (0.17). Errors in cardinality for a relationship can only be measured if the novice organizes the relationship correctly. Therefore, data was unavailable for thirty one novices. For the remaining twenty novices, fourteen novices were not at the Organize sublevel for cardinality, while six were at the Organize sublevel. Of the fourteen novices not at the Organize sublevel, one made the error of incorrect cardinality and thirteen did not. Of the six novices who were at the Organize sublevel for cardinality, two did not make the error

while four novices did make the error. Since three of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.014) because the direction of the test was opposite to what was anticipated. The hypothesis predicted that the proportion of novices at the Organize sublevel for cardinality making the errors should be less than the proportion of novices not at the Organize sublevel.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	7.937[b]	1	.005		
Continuity Correction[a]	5.079	1	.024		
Likelihood Ratio	7.650	1	.006		
Fisher's Exact Test				.014	.014
Linear-by-Linear Association	7.540	1	.006		
N of Valid Cases	20				

a. Computed only for a 2x2 table

b. 3 cells (75.0%) have expected count less than 5. The minimum expected count is 1. 50.

Table 98: Error Prediction Hypothesis 11: Not at Organize for Cardinality Results in Errors in Cardinality

Error Prediction Hypothesis 12: A novice who has high domain knowledge in real estate will perform as well as a novice who has low domain knowledge.

Data was missing for two novices for at least one of the variables. For the remaining forty nine novices, thirty three had medium domain knowledge, so were excluded from this hypothesis. Of the remaining seventeen, fourteen had low domain familiarity and three had high domain familiarity. Of the fourteen novices with low domain familiarity, five did not get any relationships correct and nine got at least one relationship correct. Of

the three novices with high domain familiarity, two got at least one relationship correct and one did not get any relationships correct. Since three of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. As expected, the test was not significant at the 0.05 level (p value=1.). Therefore, this prediction was supported.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.074[b]	1	.785		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.076	1	.783		
Fisher's Exact Test				1.000	.639
Linear-by-Linear Association	.073	1	.788		
N of Valid Cases	49				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1. 22.

Table 99: Error Prediction Hypothesis 12: Application Domain Familiarity Does Not Result in Better Performance

Error Prediction Hypothesis 13: A novice who has a moderate to high of prior experience with Object Oriented programming will perform as well as a novice who has little or no prior experience.

Data was missing for five novices for at least one of the variables. For the remaining forty six novices, ten novices had low expertise for object oriented programming, while thirty six had moderate to high expertise for object oriented programming. Of the ten novices with low expertise for object oriented programming five novices did not get any relationships correct and five got at least one relationship correct. Of the thirty six

novices with moderate to high expertise for object oriented programming thirteen novices did not get any relationships correct and twenty three got at least one relationship correct. Since one of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.48). Therefore, this prediction was supported.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.634[b]	1	.426		
Continuity Correction[a]	.185	1	.667		
Likelihood Ratio	.623	1	.430		
Fisher's Exact Test				.480	.330
Linear-by-Linear Association	.620	1	.431		
N of Valid Cases	46				

a. Computed only for a 2x2 table

b. 1 cells (25.0%) have expected count less than 5. The minimum expected count is 3. 91.

Table 100: Error Prediction Hypothesis 13: Prior Experience with Object Oriented programming Does Not Result in Better Performance

Error Prediction Hypothesis 14: A novice who has prior experience with ER modeling will perform better than a novice who doesn't have prior experience with ER modeling In addition, a novice who has prior experience with ER modeling should be at a higher expertise level than a novice who doesn't.

Data was missing for five novices for at least one of the variables. For the remaining forty six novices, thirty eight novices had low prior experience in ER modeling, while eight had some experience in ER modeling. No novices had extensive prior experience in ER modeling. Of the thirty eight novices with low expertise for ER modeling, sixteen did

not get any relationships correct while twenty two novices got at least one relationship correct. Of the eight novices with some ER modeling prior experience, two did not get any relationships correct while six got at least one relationship correct. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.34).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.812[b]	1	.368		
Continuity Correction[a]	.252	1	.615		
Likelihood Ratio	.853	1	.356		
Fisher's Exact Test				.453	.314
Linear-by-Linear Association	.794	1	.373		
N of Valid Cases	46				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 3. 13.

Table 101: Error Prediction Hypothesis 14: Object Oriented Familiarity Does Not Result in Better Performance

Error Prediction Hypothesis 15: Familiarity with UML will not impact performance in ER modeling.

Data was missing for five novices for at least one of the variables. For the remaining forty six novices, thirty one novices had low familiarity for UML, while fifteen had moderate familiarity for UML. None of the novices had high familiarity for UML. Of the thirty one novices with low familiarity for UML thirteen novices did not get any relationships correct and eighteen got at least one relationship correct. Of the fifteen novices with moderate familiarity for UML five novices did not get any relationships

correct and ten got at least one relationship correct. Since none of the cells had an expected value of less than 5, the Pearson's Chi square test was used. The test was not significant at the 0.05 level (p value=0.575). Therefore, this prediction was supported.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.314[b]	1	.575		
Continuity Correction[a]	.057	1	.812		
Likelihood Ratio	.318	1	.573		
Fisher's Exact Test				.749	.409
Linear-by-Linear Association	.307	1	.579		
N of Valid Cases	46				

a. Computed only for a 2x2 table

b. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5. 87.

Table 102: Error Prediction Hypothesis 15: UML Familiarity Does Not Result in Better Performance

Error Prediction Hypothesis 16: Novices who are visual learners will perform better (make fewer errors) than novices who are verbal learners.

Fifteen novices were balanced, i.e., neither visual nor verbal learners. Of the remaining thirty six novices, seventeen learners were visual learners, while nineteen were verbal learners. Of the seventeen visual learners, seven not get any relationships correct and eleven got at least one relationship correct. Of the nineteen verbal learners, eleven did not get any relationships correct and eight got at least one relationship correct. Since none of the cells had an expected value of less than 5, the Pearson's Chi square test was used. The test was not significant at the 0.05 level (p value=0.575).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.839[b]	1	.175		
Continuity Correction[a]	1.044	1	.307		
Likelihood Ratio	1.857	1	.173		
Fisher's Exact Test				.202	.154
Linear-by-Linear Association	1.788	1	.181		
N of Valid Cases	36				

a. Computed only for a 2x2 table

b. 0 cells (.0%) have expected count less than 5. The minimum expected count is 8. 03.

Table 103: Error Prediction Hypothesis 16: Visual Learners Perform Better Than Verbal Learners

Error Prediction Hypothesis 17: Novices who are Intuitive learners will perform better (make fewer errors) than novices who are sensing learners.

Eighteen novices were balanced, i.e., neither intuitive nor sensing learners. Of the remaining thirty three novices, fifteen learners were sensing learners, while eighteen were intuitive learners. Of the fifteen sensing learners, five not get any relationships correct and ten got at least one relationship correct. Of the eighteen intuitive learners, seven did not get any relationships correct and eleven got at least one relationship correct. Since none of the cells had an expected value of less than 5, the Pearson's Chi square test was used. The test was not significant at the 0.05 level (p value=0.741).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.109[b]	1	.741		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.109	1	.741		
Fisher's Exact Test				1.000	.514
Linear-by-Linear Association	.106	1	.745		
N of Valid Cases	33				

a. Computed only for a 2x2 table

b. 0 cells (.0%) have expected count less than 5. The minimum expected count is 5. 45.

Table 104: Error Prediction Hypothesis 17: Intuitive Learners Perform Better Than Sensing Learners

Error Prediction Hypothesis 18: Novices that identify attributes at the end are more likely to make process related errors than novices that identify attributes before relationships.

Data was missing for thirty two novices for at least one of the variables. For the remaining nineteen novices, thirteen novices identified attributes before relationships and six novices identified attributes at the end. Of the thirteen novices who identified attributes before relationships, ten novices did not make process related errors and three did. Of the six novices who identified attributes at the end, three novices made process related errors and three did not. Since three of the cells had an expected value of less than 5, Fisher's exact test was used over the Pearson's Chi square test. The test was not significant at the 0.05 level (p value=0.257).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	1.377[b]	1	.241		
Continuity Correction[a]	.413	1	.520		
Likelihood Ratio	1.336	1	.248		
Fisher's Exact Test				.320	.257
Linear-by-Linear Association	1.305	1	.253		
N of Valid Cases	19				

a. Computed only for a 2x2 table

b. 3 cells (75.0%) have expected count less than 5. The minimum expected count is 1. 89.

Table 105: Error Prediction Hypothesis 18: Identifying Attributes at the End Result in More Process Related Errors

Error Prediction Hypothesis 19: Novices that identify cardinality at the end are more likely to make process related errors than novices that identify cardinality immediately after relationships.

Data was missing for six novices for at least one of the variables. For the remaining forty five novices, five novices identified cardinality immediately after relationships, while forty identified cardinality at the end. Of the five novices who identified cardinality immediately after relationships, three made no process related error while two did. Of the forty novices who identified cardinality at the end, twenty four novices made no process related error while sixteen did. Since two of the cells had an expected value of less than 5, Fisher's exact test was used instead of the Pearson's Chi square test. The test was not significant at the 0.05 level (p value=0.675).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	.000[b]	1	1.000		
Continuity Correction[a]	.000	1	1.000		
Likelihood Ratio	.000	1	1.000		
Fisher's Exact Test				1.000	.675
Linear-by-Linear Association	.000	1	1.000		
N of Valid Cases	45				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 2. 00.

Table 106: Error Prediction Hypothesis 18: Identifying Attributes at the End Result in More Process Related Errors

5.3. Discussion

The results below discuss the significance of the results using both quantitative and qualitative data.

5.3.1. Bloom's Taxonomy verification hypotheses

All novices were at least at the Understand level for all the ER modeling constructs. To statistically verify the hypotheses, neither variable can be a constant. Since most of the Bloom's taxonomy hypotheses involved the Understand level, and it was a constant, they could not be verified. The only hypothesis that could be tested was BV2 which predicted that the Understand level for foreign keys is necessary to be at the Differentiate sublevel for foreign keys. This hypothesis was significant which indicates that it is necessary to be at the Understand level to be at the Differentiate sublevel for foreign keys. Future work can explore verifying Bloom's Taxonomy for a wide range of expertise levels.

The primary objective in this study was to use expertise levels to predict errors. It was necessary that novices be at least at the Understand level, to allow the prediction of errors. Also, our motivation was fuelled by the inability of most novices to grasp ER modeling fully at after taking the required database design class in the curriculum. Therefore, novices were invited to participate near the end of the semester after they had had time to learn ER modeling.

5.3.2. Relationship among constructs hypotheses

All the novices were at least at the Understand level for all the ER modeling constructs which implies that the understand variable was a constant. Therefore, most of the relationship among constructs could not be verified. The only hypothesis not which did not have at least one variable as a constant was RC5 which states that it is necessary to be at the Differentiate sublevel for entity classes to be at the Differentiate sublevel for interaction relationships. This hypothesis was significant which indicates that it is necessary to be at the Differentiate sublevel for entity classes to be at the Differentiate sublevel for relationships. Future studies can explore verifying the relationships among the other constructs.

5.3.3. Error Prediction Hypotheses

Some hypotheses (EP1, EP2a, EP5b and EP8) could not be verified either because all novices had attained the corresponding expertise level or that error was not made by any of the novices in the study. This section discusses the error prediction hypotheses to test which expertise levels significantly predicted errors. If the errors were not significantly predicted, we discuss possible reasons why, using qualitative data gathered.

5.3.3.1. Differentiate Expertise Level Error Prediction Hypotheses

Hypotheses EP2b through EP7 deal with predicting errors associated with the Differentiate expertise level for ER modeling constructs. There are two aspects to the Differentiate sublevel: one is the ability to identify relevant information for a conceptual schema while the other is to identify irrelevant information, i.e., information that cannot be captured by the conceptual schema.

Post Hoc Analysis for better understanding the relationship between Differentiate (Relevant) sublevel and the error 'Missing ER modeling constructs':

The following hypotheses fall in this category: EP2a, EP3a, EP5a, EP6 and EP7. As discussed in the Research Model chapter, the relationships between Differentiate (Relevant) sublevel and the error 'Missing ER modeling constructs' were expected to be marginally significant, if at all. This is because a major reason for the error is missing the information in the requirements. It was not possible to control for the factor of missing information in requirements in this study. A reliable way to track the factor of missing the information in the requirements is through eye tracking mechanisms (Liversedge and Findlay 2000 , Richardson and Spivey 2004). Future work can incorporate eye tracking to better predict missing capturing information in the conceptual schema.

Post Hoc Analysis for better understanding the relationship between Differentiate (Irrelevant) sublevel and the error 'Extra ER modeling constructs':

The hypotheses that fall into the category between Differentiate (Irrelevant) sublevel and the error 'Extra ER modeling constructs' are EP2b, EP3b (first part), and EP5b. Of these, EP2b, and EP3b (first part) were supported. The frequency of the *extra entity class*

error, predicted by hypotheses EP2b was 0.16. The frequency of the *extra interaction relationship* error, predicted by hypotheses EP3b was 0.08. Since the frequency of error in the sample was so low, we do not expect the directional measures of association to be significant predicting the error, given the Differentiate sublevel. However, it will be interesting to explore the association from the error to the expertise level. For low frequency errors, it may make sense, from a pragmatic standpoint to measure errors in a controlled setting and provide novices who make the error the training on the associated expertise level. For hypothesis EP2b, i.e., *Not* Differentiate $_{\text{Irrelevant}}$ (Entity class)→ Extra (Entity class), λ was 0 from expertise level to error, as expected, but 0.31 from error to expertise. This implies that we make 31% fewer errors predicting whether the novice is at the Differentiate sublevel (irrelevant) for entity classes given that we know whether she added irrelevant entity classes than we would if we did not know whether she made the error. For hypothesis EP3b, *Not* Differentiate $_{\text{Irrelevant}}$ (Relationship)→ Extra (Relationship), λ was 0 from expertise level to error, as expected, but 0.19 from error to expertise. This implies that we make 19% fewer errors predicting whether the novice is at the Differentiate sublevel (relevant) for relationships given that we know whether she made an error of missing interaction relationships than we would if we did not know whether she made the error. EP5b was unverified because none of the novices in the sample made the error.

Post Hoc Analysis for better understanding the relationship between Differentiate (Irrelevant) sublevel and other errors:

The hypotheses that fall into this category are EP3b (second part) and EP4. Hypothesis 3b includes a prediction of not being at the Differentiate sublevel for relationships to making the error of a binary relationship as a ternary and Hypothesis 4 includes a prediction of not being at the Differentiate sublevel for foreign keys to replacing an entity class in a relationship or the whole relationship by using a foreign keys.

Hypothesis 4 is significant. The frequency of the errors predicted by hypotheses EP4, i.e., missing a relationship or replacing an entity class in a relationship with a foreign key was 0.47. Since the frequency of error in the sample was medium, we expect the directional measures of association to be significant predicting the error, given the Differentiate sublevel. For hypothesis EP4, i.e., *Not* Differentiate $_{\text{Irrelevant}}$ (Foreign key) → Miss(Relationship) / Error(Relationship: Replace Entity class with Foreign key), λ was 0.22 from expertise level to error, and 0 from error to expertise. This implies that we make 22% fewer errors predicting whether the novice made the error of including foreign keys irrelevantly or in place of relationships given that we know whether she is at the Differentiate sublevel (irrelevant) for foreign keys than we would if we did not know whether she was at the Differentiate sublevel.

Hypothesis 3b (second part) predicts that not being at the Differentiate sublevel (irrelevant) for relationships will result in adding irrelevant information to relationships to make them ternary (higher order). A possible explanation for this part not being

significant is the high granularity of measuring irrelevant information. There are two levels to Differentiate sublevel. The first level is differentiating what is relevant and irrelevant information in the context of what can be captured by the conceptual schema (as described in this research). The second level is differentiating what is relevant and irrelevant for a particular ER modeling construct: only a subset of the information relevant to be captured by conceptual schema is relevant for a particular construct. This second level of Differentiate was not identified by RBT identified, and hence not included as one of our expertise levels. Since it is necessary to select the information relevant to the construct to be able to organize the information, the second level of Differentiate is interlinked with the Organize sublevel. A future study will need to measure this second level of measuring information irrelevant to the construct.

5.3.3.2. Organize Expertise level Error Prediction hypotheses

Hypotheses EP8 through EP11 deal with predicting errors associated with the organize expertise level for ER modeling constructs. One hypothesis (EP8) was unverified, two hypotheses (EP9a and EP10) were supported and two hypotheses (EP9b and EP11) were not supported.

Post Hoc Analysis for better understanding the relationship between Organize sublevel for entity class and the error 'combining entity classes':

Hypotheses EP9b, ' *Not* Organize $_{\text{Grouping}}$ (Entity class) → Error(Entity class: combine entity classes)', was not significant. This result was surprising at first. So, the first step taken was to examine the variable Organize $_{\text{Grouping}}$ (Entity class) at a finer level of granularity. So, it was divided into three variables, based on the types of entity classes:

Organize $_{Grouping}$ (Strong Entity class), Organize $_{Grouping}$ (Weak Entity class) and Organize $_{Grouping}$ (Across types of Entity classes). The errors were similarly divided into Error(Entity class: combining strong), Error(Entity class: combining weak), Error(Entity class: combining strong and weak). The data was then reanalyzed and the results are shown below.

Post Hoc Hypothesis 1: If a novice is not at the Organize sublevel within Analyze for strong entity classes, she will incorrectly combine strong entity classes.

The frequency of incorrectly combining strong entity classes in this sample was zero (0.0). Therefore, the hypothesis could not be verified because this error was not committed.

Post Hoc Hypothesis 2: If a novice is not at the Organize sublevel within Analyze for weak entity classes, she will incorrectly combine weak entity classes.

The frequency of incorrectly combining entity classes in this sample was low (0.12). Of the fifty one novices, thirty eight novices were not at the Organize sublevel for weak entity classes, while thirteen were at the Organize sublevel. Of the thirty eight novices not at the Organize sublevel, six made the error of combining information across weak entity classes and thirty two did not. Of the thirteen novices who were at the Organize sublevel for weak entity classes, thirteen did not make the error while no one did make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was not significant at the 0.05 level (p value=0.153).

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	2.326[b]	1	.127		
Continuity Correction[a]	1.054	1	.305		
Likelihood Ratio	3.797	1	.051		
Fisher's Exact Test				.318	.153
Linear-by-Linear Association	2.281	1	.131		
N of Valid Cases	51				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1. 53.

Table 107: Post Hoc Hypothesis 1: Not at Organize for Relationships Results in Combining Weak Entity Classes

The protocol analysis shed more light on this matter: possibly the variable was unintentionally at the 'evaluate semantic quality' level (instead of analyze). The method to measure the Organize sublevel for the independent variable required the novice to restructure the diagram to correct the error and demonstrate she had attained the Organize sublevel (the only question that did so). The talk aloud protocol and post task interview explaining why they performed the tasks the way they did, two novices who did not make the error, but appeared not to be at the Analyze level for relationships indicated that the "erroneous design" didn't correspond to the way they knew. However, they weren't sure whether the "erroneous design" could be an alternative solution (since they had learnt there were many different ways to model the same situation). This result could indicate that the novice hadn't achieved the Evaluate cognitive process (to evaluate semantic quality), but they may have reached the Analyze level. Since the above analysis was done

post hoc, it will need to be confirmed in future studies. Since the organize level for weak entity classes was not measured, this hypothesis remains unverified.

Post Hoc Hypothesis 3: If a novice is not at the Organize sublevel within Analyze across types of entity classes, she will incorrectly associate attributes of weak entity classes with strong entity classes.

The frequency of incorrectly combining strong and weak entity classes in this sample was low (0.078). Of the fifty one novices, eighteen novices were not at the Organize sublevel across types of entity classes, while thirty three were at the Organize sublevel. Of the eighteen novices not at the Organize sublevel, four made the error of combining information across strong and weak entity classes and fourteen did not. Of the thirty three novices who were at the Organize sublevel for entity classes, thirty three did not make the error while no novice did make the error. Since two of the cells had an expected value of less than 5, the Fisher's exact test was used over the Pearson Chi square test. The test was significant at the 0.05 level (p value=0.012).

Since the frequency of error in the sample was so low (7.8%), we do not expect the directional measures of association to be significant predicting the error, given the organize level. However, it will be interesting to explore the association from the error to the expertise level. For low frequency errors, it may make sense, from a pragmatic standpoint to measure errors in a controlled setting and provide novices who make the error the training on the associated expertise level. For hypothesis EP2b, i.e., *Not* Organize $_{\text{Grouping x Type}}$ (Entity class)→ Error (Combine strong and weak entity classes), λ was 0 from expertise level to error, as expected, but 0.22 from error to expertise. This

implies that we make 22% fewer errors predicting whether the novice is at the organize level (Grouping X Type) for entity classes given that we know whether she combined entity classes than we would if we did not know whether she made the error.

Chi-Square Tests

	Value	df	Asymp. Sig. (2-sided)	Exact Sig. (2-sided)	Exact Sig. (1-sided)
Pearson Chi-Square	7.957[b]	1	.005		
Continuity Correction[a]	5.180	1	.023		
Likelihood Ratio	8.973	1	.003		
Fisher's Exact Test				.012	.012
Linear-by-Linear Association	7.801	1	.005		
N of Valid Cases	51				

a. Computed only for a 2x2 table

b. 2 cells (50.0%) have expected count less than 5. The minimum expected count is 1. 41.

Table 108: Error Prediction Hypothesis 18: Not at Organize for Entity Classes-Relationships Results in Combining Strong and Weak Entity Classes

Post Hoc Analysis for better understanding the relationship between Organize sublevel for cardinality and the errors in cardinality:

Hypotheses EP11 deals with when Organize sublevel for cardinality predicts errors in cardinality. The finding was extremely surprising because it indicated that novices at the Organize sublevel of cardinality performed significantly worse than novices not at the Organize sublevel. As mentioned earlier, errors in cardinality could only be determined if the novice had at least one relationship captured correctly. So, this limited the sample size to twenty. Only one relationship in the final ER schema could be Organize sublevel for cardinality was measured by the ability to calculate cardinality for an entity rather than the entity class. There are other aspects of organize for cardinality that was not measured

like the ability to consistently mark cardinality at a point in time, the distinction among the different types of cardinality like projection and co-occurrence (Ram and Khatri 2005). On going back to the data, it was observed that the difference in calculating cardinality at the entity vs. the entity class level would have made a difference for only one relationship. It is possible that the novices who were not at the Organize sublevel made an error in that particular relationship and hence the results ended up being significant. The Organize sublevel of cardinality with its implications on errors needs to be explored in further detail in the future.

5.3.4. Other Variable Error Prediction Hypotheses

Hypotheses EP12 through ER17 looked at the impact of possible confounding variables on the errors made. None of the variables explained any of the errors. This provided further support that the expertise levels help explain the errors.

Hypotheses EP18 through ER19 were included to examine whether the process of creating an ER schema would have an impact on the errors made. This was intended to see whether a revised analysis was needed to examine the process of creating the conceptual schema to better explain errors. However, none of the process related variables were significant.

5.4. Summary of Results

This chapter provides a statistical analysis and interpretation of the hypotheses using chi square tests.

Most Bloom's taxonomy verification hypotheses and Relations among constructs hypotheses were unverified because of the limited range of expertise level among

participants. Only one Bloom's taxonomy verification hypothesis was supported. Similarly, only one Relationship among constructs hypothesis was supported. Future work can explore verifying Bloom's Taxonomy (and Relations Among Constructs) hypotheses for a wider range of expertise levels.

The results suggest that the control variables did not have any impact on explaining errors. Since the control variables did not influence the errors made, we can conclude that expertise levels did impact errors. As hypothesized, the following modeling expertise levels are significant predictors for errors: Differentiate sublevel (Irrelevant) as well as Organize (Type) for Entity class, as well as Organize (Grouping) for Relationships. Three of hypotheses are unverified because no one made the error / everyone was at that modeling expertise level. As anticipated, the Differentiate sublevel (Relevant) was not a significant predictor of errors. Three of the hypotheses predicted were not supported. The following modeling expertise levels did not have significant predictors some errors: Differentiate (Irrelevant) for Relationships, Organize (grouping) for Entity classes and cardinality. Explanations for why those hypotheses were not supported have been proposed based on post hoc analysis of the data and qualitative data (using verbal reports).

5.5. Cost of Each Expertise Level

In this section, the errors are grouped based on the expertise level that predicted them. We estimated the costs for each error in 3.3.4. Now, we calculate the cost of each expertise level based on the cost of errors associated with that that level. The error prediction hypotheses (section 5.2.3) fall into three categories: supported, unverified

(insufficient distribution of errors or expertise levels across subjects), and not supported. If a hypothesis is not supported, we cannot estimate the cost of the expertise level. If a hypothesis is unverified, we discuss the cost of the expertise level, based on the associated errors, but emphasize that the link needs be explored in the future before training is developed. If a hypothesis is verified, then, we can estimate the cost of the expertise level, based on the errors made.

5.5.1. Entity class

5.5.1.1. Strong Entity Class

There are four types of errors for strong entity classes: missing, extra, strong entity class as weak, and combine strong entity classes. The expertise levels that are relevant to predicting errors for strong entity classes are: $Differentiate_{Relevant}$, $Differentiate_{Irrelevant}$, $Organize_{Grouping}$ and $Organize_{Type}$. The resulting hypotheses are shown in Table 109.

	Low Cost	**Medium Cost**	**High Cost**
Differentiate Relevant			Unverified[Missing]
Differentiate Irrelevant	**Supported[Extra (if not miniworld)]**		**Supported [Extra (if miniworld)]**
Organize Grouping			Unverified [Combine Strong Entity classes]
Organize Type			Unverified [Strong as Weak]

Table 109: Costs of Different Expertise Levels For Strong Entity Classes

All the hypotheses associated with strong entity classes were supported / unverified which indicates that the Differentiate and Organize levels cannot be ignored. Moreover, since the these expertise levels have high cost errors associated with them, it is important to ensure the novices achieve Differentiate and Organize levels for strong entity classes.

5.5.1.2. Weak Entity Class

	Low Cost	Medium Cost	High Cost
Differentiate **Relevant**		Unverified [Missing]	
Organize $_{\text{Type}}$			**Supported [Weak as strong]** Unverified [Weak entity class as subclass]
Organize **Grouping**			**Supported [Combine Strong and Weak Entity classes]** Unverified [Combine Weak Entity classes]
Differentiate **Irrelevant**	**Supported [Extra]**		

Table 110: Cost of Different Expertise Levels For Weak Entity Classes

There are six types of errors involving weak entity classes: missing, extra, weak entity class as strong, weak entity class as subclass, combine weak entity classes, combine weak and strong entity classes. The expertise levels that are relevant to predicting errors for weak entity classes are: $\text{Differentiate}_{\text{Relevant}}$, $\text{Differentiate}_{\text{Irrelevant}}$, $\text{Organize}_{\text{Grouping}}$ and $\text{Organize}_{\text{Type}}$. The resulting hypotheses are shown in Table 110. All

the hypotheses associated with weak entity classes were supported / unverified which indicates that the Differentiate and Organize levels cannot be ignored. Moreover, since $Organize_{Grouping}$ and $Organize_{Type}$ have high cost errors associated with them, it is important to ensure the novices achieve Organize levels for weak entity classes. $Differentiate_{Irrelevant}$, has low cost errors associated with the level, so it low priority for training for novices. $Differentiate_{Relevant}$, has medium cost error that is potentially associated to it. If the hypothesis is verified, in the future, it is can be addressed after the high cost expertise levels are corrected.

5.5.2. Attribute

There are two types of errors involving attributes: missing, and associate attribute with wrong entity classes. The expertise levels that are relevant to predicting errors for attributes are: $Differentiate_{Relevant}$, and $Organize_{Grouping}$. The resulting hypotheses are shown. The hypotheses associated with expertise levels for attributes, as shown in Table 111, were unverified or not supported. The $Organize_{Grouping}$ potentially (if the hypothesis is verified) has a high cost error associated with it, and therefore it is important to test this hypotheses in the future, If significant, training needs to be designed to ensure the novices achieve Organize levels for attributes.

	Low Cost	Medium Cost	High Cost
Differentiate **Irrelevant**		Not Supported [Missing]	
Organize$_{Grouping}$			Unverified [Associate Attribute With Wrong Entity Class]

Table 111: Cost of Different Expertise Levels For Attributes

5.5.3. Relationship

There are three expertise levels relevant to predicting errors for interaction relationships: Differentiate$_{Relevant}$, Differentiate$_{Irrelevant}$, and Organize$_{Grouping}$. The four types of errors that can be predicted by these expertise levels are missing, extra, binary as ternary and ternary as binary. The resulting hypotheses are shown in Table 112. Hypotheses associated with two expertise levels are significant, i.e., Differentiate$_{Irrelevant}$, and Organize$_{Grouping}$. The hypothesis connecting Organize $_{Grouping}$ for relationships to the errors is supported. The cost of marking a ternary relationship as a binary relationship is medium, therefore, the benefits of moving to Organize $_{Grouping}$ level for relationships is medium. The cost of an extra relationship error is low, therefore, the benefits of moving to Differentiate $_{Irrelevant}$ level for relationships is low. Hence, the costs of the errors associated with these levels are low to medium. Therefore, eliminating the errors is not high priority. Training can be designed to eliminate expertise levels corresponding to medium to low cost errors once the high priority errors are eliminated.

The hypothesis connecting Differentiate $_{Irrelevant}$ for relationships to the error of including an extra entity class in a relationship is not supported. The cost of adding an

entity class to a relationship error ranges from low to medium, therefore, other potential explanations for the errors should be sought. To address this issue, as discussed earlier (section 5.3.3), this research suggests examining the Differentiate $_{Irrelevant}$ at a finer granularity.

The hypothesis connecting Differentiate $_{Relevant}$ for relationships to the errors is not supported. The cost of missing a relationship error is high, therefore, other potential explanations for the errors should be sought. As discussed in the previous chapter, this research suggests exploring eye tracking in conjunction with Differentiate $_{Relevant}$.

	Low Cost	**Medium Cost**	**High Cost**
Differentiate $_{Relevant}$			Not Supported [Missing]
Organize $_{Grouping}$		**Supported [Ternary as Binary]**	
Differentiate $_{Irrelevant}$	Not Supported [Binary as Ternary (included entity class that shares an identifier)]	Not Supported [Binary as Ternary (included entity class that does not share an identifier)]	
	Supported [Extra]		

Table 112: Costs of Different Expertise Levels For Relationships

5.5.4. Cardinality

There are three expertise levels relevant to predicting errors for cardinality: Apply, Differentiate$_{Relevant}$, and Organize$_{Grouping}$. The two types of errors that can be predicted by these expertise levels are missing, and incorrect. The resulting hypotheses are shown in

Table 113. Two of the hypotheses associated with cardinality expertise levels are not supported. One hypothesis is unverified.

Moreover, since Apply is potentially associated with a high cost error, it is important to ensure the novices achieve Apply levels for cardinality.

The hypothesis connecting Differentiate $_{\text{Relevant}}$ for cardinality to the errors is not supported. The cost of missing cardinality varies from low to high, therefore, other potential explanations for the errors should be sought. As discussed in the previous chapter, this research suggests exploring eye tracking in conjunction with Differentiate$_{\text{Relevant}}$.

	Low Cost	**Medium Cost**	**High Cost**
Apply			Unverified [Incorrect]
Differentiate $_{\text{Relevant}}$	Not supported [Missing]		
Organize$_{\text{Grouping}}$			Not supported [Incorrect]

Table 113: Costs of Different Expertise Levels For Cardinality

The hypothesis connecting Organize $_{\text{Grouping}}$ for cardinality to the errors is not supported. The cost of the associated error, incorrectly marking cardinality is high, therefore possible explanations for the error should be examined in further detail in the

future. A larger sample size and measuring expertise levels for cardinality at a finer granularity is recommended.

5.5.5. Identifier

There is one expertise levels relevant to predicting errors for identifiers, $Differentiate_{Relevant,}$. The only error that can be predicted by this expertise level is missing. The resulting hypotheses are shown in Table 114. Evaluating incorrect identifiers was difficult because some learners had been instructed to create an artificial "Id" primary key for all strong entity classes. Therefore, it was not included in the analysis of this research. Since the cost of the error is low to medium, it should be explored in future research.

The hypothesis connecting $Differentiate_{Relevant}$ for identifiers to the errors is not supported. The cost of missing identifiers is low because it will be detected at implementation time. Potential explanations for the errors should be sought. As discussed in the previous chapter, this research suggests exploring eye tracking in conjunction with $Differentiate_{Relevant}$..

	Low Cost	**Medium Cost**	**High Cost**
$Differentiate_{Relevant}$	Missing		

Table 114: Costs of Different Expertise Levels for Identifiers

5.5.6. Foreign Key

There is one expertise levels relevant to predicting errors for foreign keys, $Differentiate_{Irrelevant,}$. The errors that can be predicted by this expertise levels are

redundantly including foreign keys, replacing relationship with attribute and replacing an entity class in a relationship with an attribute. The resulting hypotheses are shown in Table 115. The hypothesis connecting Differentiate $_{Irrelevant}$ for foreign keys to the errors is supported. The cost of including a foreign key redundantly in addition to a relationship is low. However, the cost of including a foreign key in place of a relationship or in place of an entity class in a relationship is high. Therefore, the benefits of moving to Differentiate $_{Irrelevant}$ level for foreign keys ranges from low to high.

	Low Benefit	**Medium Benefit**	**High Benefit**
Differentiate $_{Irrelevant}$	Supported [Foreign key redundantly included]		Supported [Replace entity class in relationship with an attribute]

Table 115: Costs of Different Expertise Levels For Foreign Keys

5.5.7. Summary

If the hypotheses for a given expertise level are supported, it is expected that the errors predicted by the expertise level will reduce if the novice is moved up to that level.

Based on this section, the cost of the different expertise levels is estimated. Training for high cost errors should be provided before training for low cost errors. The frequencies of each type of error in the real-world needs to be estimated to best design the order of training.

For example, in this sample the highest frequency of errors corresponds to the errors predicted by the Differentiate $_{\text{Irrelevant}}$ (Foreign Keys). Differentiate $_{\text{Irrelevant}}$ (Foreign Keys) is a high cost expertise level. Therefore, it is a high priority expertise level.

CHAPTER 6: CONCLUSION AND FUTURE RESEARCH

This chapter has a dual purpose. The first purpose is to summarize the research contributions (Section 6.1). The second is to discuss future research (Section 6.2).

This research focused, within the ER modeling domain, on the impact of designer expertise on performance. At first glance, expertise is an irrelevant construct. By now, it is hoped that the reader is convinced that expertise is a useful construct for predicting errors. Also, that there are more topics related to this topic that still need to be explored. A multi-methodology approach was used to theorize and justify (i) Designer Expertise, and Designer Performance (constructs), (ii) Bloom's taxonomy hierarchy model, relationship among data modeling constructs model, performance prediction model (models), and (iii) and generated responses to surveys and schemas (instantiations) as recommended by March and Smith for natural science research (March and Smith 1995). A combination of prior research, logic(Caws 1969), creativity and experience (Bechtel 1988) were used to generate the models.

6.1. Contributions

The role of designer expertise is an issue that has been neglected in research on conceptual modeling. Motivating designer expertise as a construct and describing how the construct can be measured, using Bloom's Taxonomy is one contribution. There are two main reasons to capture designer expertise. The first is insufficient expertise does not always result in errors, especially for misunderstandings in cardinality (Ram and Currim 2005). The second is that the expertise predicts errors while errors do not predict

expertise. Therefore, it is not possible to derive what aspect of expertise is lacking without measuring it explicitly.

The comprehensive classification of errors is another contribution. Although previous research examined errors made in database design (Antony and Batra 2002, Batra 1993, Batra and Antony 1994, Batra and Antony 2001, Batra and Davis 1989, Batra, et al. 1990, Batra and Marakas 1995, Batra and Sein 1994, Batra and Wishart 2004, Fessakis, et al. 2005, Gemino and Wand 2004, Liao and Palvia 2000, Shoval and Shiran 1997, Sinha and Vessey 1999, Topi and Ramesh 2004), no formal classification of errors has been made. Previous research has not systematically estimated the costs for different types of errors. Classifying errors helps examine errors at a finer granularity, and identify patterns among errors. Classification also enables more specific predictions between comprehension levels and errors made.

The performance prediction model is another contribution. Previous research looks at what errors are made. However, prior research does not look at *why* errors were made. Using Bloom's Taxonomy, the different comprehension levels for ER modeling were determined (Bloom's taxonomy hierarchy model). Then, the impact of comprehension level on the performance of creating conceptual schemas was predicted and tested. If the comprehension level that causes the error is known, a tutoring system can be built in the future to eliminate the error.

Knowing the levels of data modeling expertise, and how they link to different errors would aid in developing strategies in the future to help novices grasp the material better,

and reduce errors committed. There is a balance that needs to be struck between attaining expertise and the cost of attaining that expertise. To optimize that balance, this research also estimates the benefit for each level of modeling expertise. Expertise required also depends on the complexity and implications of the real world data modeling task that the novice will be undertaking. If the task complexity and implications are known, the training can be designed to ensure an acceptable performance rather than attempting to move all novices to the highest level of modeling expertise.

6.2. Future Research

Future work is categorized into two areas: framework extension and application, and framework evaluation.

6.2.1. Framework Extension and Application

Framework extension refers to research that refines existing constructs / models (to deal with EER modeling constructs, for example) or adds new branches of related research (for example, applying the methodology to different areas, like Networking).

To improve external validity and reliability of the constructs in the performance prediction model, the methodology should be replicated for novices under different situations. For example, in this research participants were invited to participate near the end of a class involving database design. Hence, all participants who were at the Understand level for most of the constructs like entity classes, attributes, relationships, etc. This restricted the study's ability to verify the hierarchy of Bloom's Taxonomy. Therefore, future research should look at a wider range of novices to verify Bloom's Taxonomy. In addition, longitudinal studies should be undertaken to observe how

participants move through Bloom's taxonomy and how that impacts errors. In addition, future research should use the methodology of this research to different areas like data mining, networking, etc.

To extend this research, it would be useful to perform additional studies involving real-world database analysts to test the performance model. While sample sizes in such studies are likely to be small, valuable insights in the terms of practical applicability could be gained.

It would be useful to explore the source of errors not explained by this research. For example, this research was not able to explain errors that involved missing requirements in the ER schema altogether. Future research can use eye tracking mechanisms to help explain the errors that involved missing capturing requirements in the ER schema.

It was discovered that a few aspects in the Expertise framework need to be measured at a finer granularity during the course of the study. Future research can fine tune the measurement of some of the expertise levels to help better predict performance.

Also, this research was restricted to a subset of ER modeling constructs. Future studies should expand the research to comprehensively cover the range of EER constructs.

6.2.2. Framework Evaluation

Evaluation of the framework includes further case studies and experiments to test the framework. In a sense it seeks to complement the natural science aspect of this

dissertation with design science research. In particular, future research in the design science area needs to address how to build an information system to help learners achieve different levels of expertise.

Some work already exists in designing intelligent tutoring systems to help train learners. The performance prediction model helps determine why the error is made. Using the performance prediction model, a tutoring system can be built in the future to eliminate the error. This helps tailor the training based on the learner's current level of knowledge and provide training on an as needed basis. In addition, Bloom's taxonomy hierarchy model can help determine the ordering of training modules. If the implications of the application to be built are known, the tutoring system can decide what level of expertise the learner needs to achieve before starting on the project. Future work needs to build such a tutoring system to enable learners to achieve higher levels of expertise and eliminate errors related to lack of expertise.

Future laboratory / field studies would benefit from the participation of consultants, who frequently encounter badly designed systems. Consultants can provide insights on how training can be best developed to help reduce errors in the future.

In addition, future research should look at the cognitive processes involved (Fitzgerald, et al. 2005, Srinivasan and Te'eni 1995) and mental representations created (Fix, et al. 1993)in the data modeling process. Cognitive science is potentially a very useful reference discipline for gaining a deeper understanding why people do things in a particular way (Siau 1999).

REFERENCES

[1] Agarwal, R., Sinha, A. P., and Tanniru, M., "The role of prior experience and task characteristics in object-oriented modeling: an empirical study," *International Journal of Human-Computer Studies*, vol. 45 (6), pp. 639-667, 1996.

[2] Ahrens, J. D. and Sankar, C. S., "Tailoring Database Training for End Users," *MIS Quarterly*, vol. 17 (4), pp. 419-439, 1993.

[3] Allwood, C. M., "Error Detection Processes in Statistical Problem Solving," *Cognitive Science*, vol. 8 (4), pp. 413-437, 1984.

[4] Anderson, J. R., Boyle, C. F., and Reiser, B. J., "Intelligent Tutoring Systems," *Science*, vol. 228 (4698), pp. 456-462, 1985.

[5] Anderson, J. R., Corbett, A. T., Koedinger, K. R., and Pelletier, R., "Cognitive tutors: Lessons learned," *The Journal of Learning Sciences*, vol. 4 (2), pp. 67-207., 1995.

[6] Anderson, L. W., Krathwohl, D. R., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., Raths, J., and Wittrock, M. C., *A Taxonomy for Learning,Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. New York: Allyn & Bacon, 2001.

[7] Antony, S. R. and Batra, D., "CODASYS: a consulting tool for novice database designers," *SIGMIS Database*, vol. 33 (3), pp. 54-68, 2002.

[8] ASEP, *Coaching Youth Tennis*, Third ed, 2002.

[9] Baars, B. J., *Experimental Slips and Human Error: Exploring the Architecture of Volition*: Plenum Publishing Corporation, 1992.

[10] Baron, R. A., Rea, M. S., and Daniels, S. G., "Effects of indoor lighting (illuminance and spectral distribution) on the performance of cognitive tasks and interpersonal behaviors: The potential mediating role of positive affect," *Motivation and Emotion*, vol. 16 (1), pp. 1-33, 1992.

[11] Batini, C., Ceri, S., and Navathe, S. B., *Conceptual Database Design: An Entity-Relationship Approach*: The Benjamin/Cummings Publishing Company, 1992.

[12] Batra, D., "A Framework for Studying Human Error Behavior in Conceptual Database Modeling," *Information and Management*, vol. 25 (3), pp. 121-131, 1993.

[13] Batra, D. and Antony, S. R., "Novice errors in conceptual database design," *European Journal of Information Systems*, vol. 3 (1), pp. 57-69, 1994.

[14] Batra, D. and Antony, S. R., "Consulting support during conceptual database design in the presence of redundancy in requirements specifications: an empirical study," *International Journal Of Human-Computer Studies*, vol. 54 (1), pp. 25-51, 2001.

[15] Batra, D. and Davis, J., "A study of conceptual data modeling in database design: similarities and differences between expert and novice designers," in *Proceedings of the Proceedings of the 10th International Conference on Information Systems (ICIS-89)*, Boston, MA, pp. 91-100, 1989.

[16] Batra, D. and Davis, J. G., "Conceptual Data Modelling in Database Design: Similarities and Differences between Expert and Novice Designers," *International Journal of Man-Machine Studies*, vol. 37 (1), pp. 83-101, 1992.

[17] Batra, D., Hoffer, J. A., and Bostrom, R. P., "Comparing representations with relational and EER models," *Communications of the ACM*, vol. 33 (2), pp. 126-139, 1990.

[18] Batra, D. and Kirs, P. J., "The quality of data representations developed by nonexpert designers: An experimental study," *Journal of Database Management*, vol. 4 (4), pp. 17-29, 1993.

[19] Batra, D. and Marakas, G. M., "Conceptual data modelling in theory and practice," *European Journal of Information Systems*, vol. 4 (3), pp. 185 - 193, 1995.

[20] Batra, D. and Sein, M. K., "Improving conceptual database design through feedback," *International Journal of Human-Computer Studies*, vol. 40 (4), pp. 653-676, 1994.

[21] Batra, D. and Wishart, N. A., "Comparing a rule-based approach with a pattern-based approach at different levels of complexity of conceptual data modelling tasks," *International Journal of Human-Computer Studies*, vol. 61 (4), pp. 397-419, 2004.

[22] Beach, L. R. and Mitchell, T. R., "A Contingency Model for the Selection of Decision Strategies," *Academy of Management Review* vol. 3 (3), pp. 439-449, 1978.

[23] Bechtel, W., *Philosophy of Science: An Overview for Cognitive Science*. Baltimore: Lawrence Erlbaum, 1988.

[24] Bell, B. G., Gardner, M. K., and Woltz, D. J., "Individual differences in undetected errors in skilled cognitive performance," *Learning and Individual Differences*, vol. 9 (1), pp. 43-61, 1997.

[25] Bettman, J. R. and Zins, M. A., "Information Format and Choice Task Effects in Decision Making," *The Journal of Consumer Research*, vol. 6 (2), pp. 141-153, 1979.

[26] Bloom, B., Krathwohl, D. R., and Masia, B. B., *Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain.* New York: Longmans, Green, 1956.

[27] Bock, D. B. and Yager, S. E., "Improving entity relationship modeling accuracy with novice data modelers," vol. 42, 2001, pp. 69-75.

[28] Boehm, B. W. and Papaccio, P. N., "Understanding and controlling software costs," *Software Engineering, IEEE Transactions on*, vol. 14 (10), pp. 1462-1477, 1988.

[29] Bostrom, R. P., Olfman, L., and Sein, M. K., "End-user Computing: A Research Framework for Investigating the Training/Learning Process," in *Human Factors in MIS*, J. Carey, Ed. Norwood, NJ: Ablex Publishing Corp., 1988, pp. 221-250.

[30] Bouzeghoub, M., Gardarin, G., and Métais, E., "Database Design Tools: An Expert System Approach," in *Proceedings of the Proceedings of the 11th International Conference on Very Large Data Bases*, Stockholm, Sweden, pp. 82-95, 1985.

[31] Brosey, M. and Shneiderman, B., "Two experimental comparisons of relational and hierarchical database models," *International Journal Of Man-Machine Studies*, vol. 10, pp. 625-637, 1978.

[32] Browne, G. J. and Ramesh, V., "Improving information requirements determination: a cognitive perspective," *Information and Management*, vol. 39 (8), pp. 625 - 645, 2002.

[33] Burton-Jones, A. and Weber, R., "Understanding relationships with attributes in entity-relationship diagrams," in *Proceedings of the Proceedings of the 20th international conference on Information Systems (ICIS-99)*, Charlotte, North Carolina, United States, pp. 214 - 228, 1999.

[34] Cambell, D., "Entity-relationship modeling: one style suits all," *Data Base*, vol. 23 (5), pp. 12-18, 1992.

[35] Case, S. M. and Swanson, D. B. Constructing Written Test Questions For the Basic and Clinical Sciences. 2003 [last accessed November 18, 2004 2004]; 3rd (Revised):[Available from: http://www.nbme.org/PDF/ItemWriting_2003/2003IWGwhole.pdf.

[36] Caws, P., "The Structure of Discovery," *Science*, vol. 166 (3911), pp. 1375–1380, 1969.

[37] Chi, M. T. H., Siler, S. A., and Jeong, H., "Can Tutors Monitor Students' Understanding Accurately?," vol. 22, 2004, pp. 363-387.

[38] Constantine, L. L. and Lockwood, L. A. D., *Software for Use: A Practical Guide to the Models and Methods of Usage-centered Design*. New York, NY: ACM Press, 2006.

[39] Crawford, C. M. and Brown, E., "Focusing Upon Higher Order Thinking Skills: WebQuests and the Learner-Centered Mathematical Learning Environment.," Texas 2002.

[40] Date, C. J., *An Introduction to Database Systems*, Seventh ed: Addison Wesley Longman, 2000.

[41] Dey, D., Storey, V. C., and Barron, T. M., "Improving database design through the analysis of relationships," *ACM Transactions on Database Systems (TODS)*, vol. 24 (4), pp. 453 - 486, 1999.

[42] Dickie, L. O., "Approach to Learning, the Cognitive Demands of Assessment, and Achievement in Physics," *Canadian Journal of Higher Education*, vol. 33 (1), pp. 87-111, 2003.

[43] Dreyfus, H. and Dreyfus, S., *Mind Over Machine: The power of human intuition and expertise in the era of the computer*: Free Press, 1986.

[44] Ebel, R. L. and Frisbie, D. A., *Essentials of educational measurement*. Englewood Cliffs, N.J.: Prentice-Hall, 1986.

[45] Elmasri, R. and Navathe, S. B., *Fundamentals of Database Systems*, Fourth ed: Addison Wesley, 2003.

[46] Evans, J. S. B. T., *Bias in Human Reasoning: Causes and Consequences*: Lawrence Erlbaum Ltd., 1989.

[47] Felder, R. M. and Silverman, L. K., "Learning and Teaching Styles in Engineering Education," *Engr. Education*, vol. 78 (7), pp. 674-681, 1988.

[48] Felder, R. M. and Spurlin, J. E., "Applications, Reliability, and Validity of the Index of Learning Styles," *Intl. Journal of Engineering Education*, vol. 21 (1), pp. 103-112, 2005.

[49] Fessakis, G., Dimitracopoulou, A., and Komis, V., "Improving database design teaching in secondary education: Action Research implementation for documentation of didactic requirements and strategies," *Computers in Human Behavior*, vol. 21 (2), pp. 159 -194, 2005.

[50] Fitzgerald, S., Simon, B., and Thomas, L., "Strategies that students use to trace code: an analysis based in grounded theory," in *Proceedings of the Proceedings of the 2005 international workshop on Computing education research*, Seattle, WA, USA, pp. 69-80, 2005.

[51] Fix, V., Wiedenbeck, S., and Scholtz, J., "Mental representations of programs by novices and experts," in *Proceedings of the Proceedings of the SIGCHI conference on Human factors in computing systems*, Amsterdam, The Netherlands, pp. 74-79, 1993.

[52] Gagne, R. M., Briggs, L. J., and Wager, W. W., *Instructional Design*, 3rd ed. New York: Holt, Rinehart and Winston Inc, 1988.

[53] Gemino, A. and Wand, Y., "A framework for empirical evaluation of conceptual modeling techniques," *Requirements Engineering*, vol. 9 (4), pp. 248 - 260, 2004.

[54] Gorgone, J. T., Davis, G. B., Valacich, J. S., Topi, H., Feinstein, D. L., and Herbert E. Longenecker, J., "IS 2002, Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems," ACM, AIS, AITP 2002.

[55] Haladyna, T. M., *Developing and Validating Multiple-Choice Test Items*, 2nd ed. Mahwah, NJ: Lawrence Erlbaum Associates, 1999.

[56] Hardgrave, B. C. and Dalal, N. P., "Comparing object-oriented and extended-entity-relationship data models," *Journal of Database Management*, vol. 6 (3), pp. 15-21, 1995.

[57] Hoffer, J. A., Prescott, M., and McFadden, F., *Modern Database Management*, Eighth ed: Prentice Hall, 2006.

[58] Hull, R. and King, R., "Semantic database modeling: survey, applications, and research issues," *ACM Computing Surveys*, vol. 19 (3), pp. 201 - 260, 1987.

[59] Jarvenpaa, S. L. and Machesky, J. J., "Data analysis and learning: an experimental study of data modeling tools," *International Journal Of Man-Machine Studies*, vol. 31 (4), pp. 367-391, 1989.

[60] Jenkins, A. M., *MIS design variables and decision making performance: A simulation experiment*: UMI Research Press, 1983.

[61] Jolliffe, F. R. and Ponsford, R. A., "Classification and comparison of mathematics examinations methods based on Bloom's taxonomy," *International Journal of Mathematical Education in Science and Technology*, vol. 20, pp. 677-688, 1989.

[62] Kantowitz, B. and Sorkin, R. D., *Human Factors: Understanding People-System Relationships*. New York: Wiley, 1983.

[63] Katz, I. R. and Anderson, J. R., "Debugging: An Analysis of Bug-Location Strategies," *Human-Computer Interaction*, vol. 3 (4), pp. 351-399, 1987-1988.

[64] Khatri, V., Ramesh, V., Vessey, I., Clay, P., and Park, S.-J., "Does Application Domain Knowledge Really Matter? Exploring Its Role in Comprehension of Conceptual Schemas," Indiana University, Bloomington TR143-1, April 2004.

[65] Khatri, V., Vessey, I., Ramesh, V., Clay, P., and Park, S.-J., "Understanding Conceptual Schemas: Exploring the Role of Application and IS Domain Knowledge " *Information Systems Research* vol. 17 (1), pp. 81-99, 2006.

[66] Kim, J., Lerch, F. J., and Simon, H. A., "Internal representation and rule development in object-oriented design," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 2 (4), pp. 357-390, 1995.

[67] Kim, Y.-G. and March, S. T., "Comparing data modeling formalisms," *Communications of the ACM*, vol. 38 (6), pp. 103 - 115, 1995.

[68] Krathwohl, D. R., "Reflections on the taxonomy: Its past, present and future," in *Bloom's taxonomy: A forty year retrospective, Ninety-third Yearbook of the National Society for the Study of Education*, N. S. f. t. S. o. Education, Ed. Chicago: University of Chicago Press, 1994, pp. 181-202.

[69] Krathwohl, D. R., "A Revision of Bloom's Taxonomy: An Overview," *Theory into Practice*, vol. 41 (4), pp. 212 - 218, 2002.

[70] Liao, C. and Palvia, P. C., "The impact of data models and task complexity on end-user performance: an experimental investigation," *International Journal Of Human-Computer Studies*, vol. 52 (5), pp. 831-845, 2000.

[71] Lindland, O. I., Sindre, G., and Solvberg, A., "Understanding quality in conceptual modeling," *Software, IEEE*, vol. 11 (2), pp. 42-49, 1994.

[72] Liversedge, S. P. and Findlay, J. M., "Saccadic Eye Movements and Cognition," *Trends in Cognitive Sciences*, vol. 4 (1), 2000

[73] Lloyd-Williams, M. and Beynon-Davies, P., "Expert systems for database design: A comparative review," *Artificial Intelligence Review*, vol. 6 (3), pp. 263-283, 1992a.

[74] Lloyd-Williams, M. and Beynon-Davies, P., "Knowledge Based CASE tools for Database Design," in *CASE: Current Practice, Future Prospects*. New York, NY: John Wiley & Sons, Inc., 1992b, pp. 205-222.

[75] March, S. T. and Smith, G. F., "Design and Natural Science Research on Information Technology," *Decision Support Systems* vol. 15 (4), pp. 251-266, 1995.

[76] McConnell, S., *Rapid Development*. Redmond, Washington: Microsoft Press, 1996.

[77] Moody, D. L., "Metrics for Evaluating the Quality of Entity Relationship Models," in *Proceedings of the 17th International Conference on Conceptual Modeling*, Singapore, pp. 211-225, 1998.

[78] Moody, D. L. and Shanks, G. G., "What Makes a Good Data Model? Evaluating the Quality of Entity Relationship Models," in *Proceedings of the Thirteenth International Conference on the Entity-Relationship Approach*, Manchester, U.K., pp. 94-111, 1994.

[79] Morris, B. H., "The Beauty of Geometry," *Mathematics Teaching in the Middle School*, vol. 9 (7), pp. 358-61, 2004.

[80] Murphy, G. L. and Medin, D. L., "The Role of Theories in Conceptual Coherence," *Psychological Review*, vol. 92 (3), pp. 289-316, 1985.

[81] Murphy, K. R. and Cleveland, J., *Understanding Performance Appraisal: Social, Organizational, and Goal-Based Perspectives*: Sage Publications Inc, 1995.

[82] Myers, I. and McCaulley, M. H., *MBTI® Manual*. Palo Alto: Consulting Psychologists Press, 1985.

[83] Nelson, R. R., "Educational Needs as Perceived by IS and End-User Personnel: A Survey of Knowledge and Skill Requirements," *MIS Quarterly*, vol. 15 (4), pp. 503-525, 1991.

[84] Noah, S. A. and Lloyd-Williams, M., "A Selective Review of Knowledge-Based Approaches to Database Design," *Information Research*, vol. 1 (2), pp. 2-8, 1995.

[85] Panko, R. R. Human Error Website. 1997 [last accessed 02-19-2005 2005]; Available from: http://panko.cba.hawaii.edu/HumanErr/.

[86] Panko, R. R., "What We Know About Spreadsheet Errors," *Journal of End User Computing*, vol. 10 (2), pp. 15-21, 1998.

[87] Pazzani, M. J., "Influence of Prior Knowledge and Concept Acquisition: Experimental and Computational Results," *Journal of experimental psychology: Learning, memory, and cognition* vol. 17 (3), pp. 416-432, 1991.

[88] Penrose, J. M. and Seiford, L. M., "Microcomputer Users' Preferences for Software Documentation: An Analysis," *Journal of Technical Writing and Communication*, vol. 18 (4), pp. 355-366, 1988.

[89] Ram, S. and Currim, S., "Why do novices make errors in conceptual modeling of databases?," in *Proceedings of the Fourth Symposium on Research in Systems Analysis and Design (AIS SIGSAND)*, Cincinnati, Ohio, 2005.

[90] Ram, S. and Khatri, V., "A Comprehensive Framework for Modeling Set-based Business Rules during Conceptual Database Design," *Information Systems*, vol. 30 (2), pp. 89 - 118, 2005.

[91] Ramesh, V. and Browne, G. J., "Expressing Causal Relationships in Conceptual Database Schemas," *Journal of Systems and Software*, vol. 45 (3), pp. 225-232, 1999.

[92] Rasmussen, J., *Information Processing and Human-machine Interaction: an Approach to Cognitive Engineering*, vol. 12. New York: Elsevier Science Ltd, 1986.

[93] Raths, J., "Improving instruction," *Theory into Practice*, vol. 41 (4), pp. 233 - 237, 2002.

[94] Reason, J., *Human Error*. Cambridge, UK: Cambridge University Press, 1990.

[95] Richardson, D. and Spivey, M., "Eye tracking: Research areas and applications," in *Encyclopedia of Biomaterials and Biomedical Engineering* G. Wnek and G. Bowlin, Eds. New York: Marcel Dekker, Inc, 2004 pp. 573-582

[96] Rush, G., "A Fast way to define system requirements," in *Computerworld*, vol. 19, 1985, pp. 11 - 16.

[97] Schank, R., Collins, G., and Hunter, L., "Transcending Inductive Category Formation in Learning," *The Behavioral and Brain Sciences*, vol. 9 (4), pp. 639-686, 1986.

[98] Schmidt, F. L. and Hunter, J. E., "Select on Intelligence," in *The Blackwell Handbook of Principles of Organizational Behavior* E. A. Locke, Ed., 2004, pp. 4.

[99] Schutz, P. A., Drogosz, L. M., White, V. E., and Distefano, C., "Prior Knowledge, Attitude, and Strategy Use in an Introduction to Statistics Course," *Learning and Individual Differences*, vol. 10 (4), pp. 291-308, 1998.

[100] Sein, M. K., Bostrom, R. P., and Olfman, L., "Rethinking End-User Training Strategy: Applying a Hierarchical Knowledge-Level Model," *Journal of End User Computing*, vol. 11 (1), pp. 32-39, 1999.

[101] Shoval, P. and Even-Chaime, M., "Database schema design: an experimental comparison between normalization and information analysis," *SIGMIS Database*, vol. 18 (3), pp. 30-39, 1987.

[102] Shoval, P. and Shiran, S., "Entity-relationship and object-oriented data modeling — An experimental comparison of design quality," *Data & Knowledge Engineering*, vol. 21 (3), pp. 297-315, 1997.

[103] Siau, K., "Information modeling and method engineering," *Journal of Database Management*, vol. 10 (4), pp. 44-50, 1999.

[104] Siau, K., Wand, Y., and Benbasat, I., "A Psychological Study on the Use of Relationship Concept--Some Preliminary Findings," in *Proceedings of the Proceedings of the 7th International Conference on Advanced Information Systems Engineering*, pp. 341-354, 1995.

[105] Siau, K. L., *Empirical studies in information modeling*, University of British Columbia, 1996.

[106] Sinha, A. P. and Vessey, I., "An empirical investigation of entity-based and object-oriented data modeling: a development life cycle approach," in

Proceedings of the 20th International Conference on Information Systems (ICIS-99), Charlotte, North Carolina, pp. 229-244, 1999.

[107] Spohrer, J. C. and Soloway, E., "Novice mistakes: are the folk wisdoms correct?," *Communications of the ACM*, vol. 29 (7), pp. 624-632, 1986a.

[108] Spohrer, J. G. and Soloway, E., "Analyzing the high frequency bugs in novice programs " in *Proceedings of the first workshop on empirical studies of programmers on Empirical studies of programmers*, Washington, D.C., United States pp. 230 - 251, 1986b.

[109] Srinivasan, A. and Te'eni, D., "Modeling as constrained problem solving: an empirical study of the data modeling process," *Management Science*, vol. 41 (3), pp. 419-434, 1995.

[110] Storey, V. C., "A Selective Survey of the Use of Artificial Intelligence for Database Design Systems " in *Proceedings of the Data & Knowledge Engineering* Amsterdam, The Netherlands, The Netherlands pp. 61 - 102 1993.

[111] Storey, V. C. and Goldstein, R. C., "Knowledge-based approaches to database design," *MIS Quarterly*, vol. 17 (1), pp. 25 - 46, 1993.

[112] Suraweera, P., *An Intelligent Teaching System for Database Modeling*, Computer Science, University of Canterbury, 2001.

[113] Suraweera, P. and Mitrovic, A., "KERMIT: A Constraint-Based Tutor for Database Modeling," in *Proceedings of the Proceedings of the Sixth International Conference on Intelligent Tutoring Systems (ITS 2002)*, Biarritz, France and San Sebastian, Spain, pp. 377-387, 2002.

[114] Topi, H. and Ramesh, V., "Toward an Extended Framework for Human Factors Research on Data Modeling," in *Advanced Topics in Database Research*, vol. 3, K. Siau, Ed.: Idea Group, 2004, pp. 188-217.

[115] Trochim, W. M. K., *The Research Methods Knowledge Base*, Second ed. Cincinnati, Ohio: Atomic Dog Publishing, 1999.

[116] Vidakovic, D., Bevis, J., and Alexander, M. Bloom's Taxonomy in Developing Assessment Items. Journal of Online Mathematics and its Applications 2003 September, 2003 [last accessed March 13, 2005 3]; Available from: http://www.joma.org/mathDL/4/?pa=content&sa=viewDocument&nodeId=504.

[117] Wand, Y. and Weber, R., "Research Commentary: Information Systems and Conceptual Modeling - A Research Agenda," *Information Systems Research*, vol. 13 (4), pp. 363-376, 2002.

[118] Weber, R., "Are attributes entities? A study of database designers' memory structures," *Information Systems Research*, vol. 7 (2), pp. 137-162, 1996.

[119] Wisniewski, E. J., "Prior knowledge and functionally relevant features in concept learning," *Journal of experimental psychology: Learning, memory, and cognition*, vol. 21 (2), pp. 449-468, 1995.

[120] Zwass, V., *Management Information Systems*. Dubuque, IA: Wm. C. Brown, 1992.