Normalizing Database Normalization Definitions In AIS Text Books

Ting J. (TJ) Wang, Governors State University, USA
David K. Dennis, Otterbein University, USA

ABSTRACT

Due to the abstract nature of the definitions for normal forms, over the years the interpretations of the definitions published in the textbooks, both MIS and AIS disciplines, have been differentiated and even deviated from its original form. The concept of deviation from the original form is a phenomenon that linguists call "semantic drift." The most noticeable deviations are on first and second normal forms (i.e., 1NF and 2NF). Their definitions range from "atomic attribute" to "removing repeating group" for 1NF and from "functional dependency" to "removing partial dependency" in addition to being 1NF for 2NF. The purpose of this paper is to compare definitions of first, second, and third normal forms from the textbooks with those of the earlier forms and to identify shortfalls if there are any.

Keywords: Relational Database; Database Normalization; Normalization Definitions

INTRODUCTION

The purpose of database normalization is to increase the integrity of data stored in the database. Its goal is to reduce data redundancy and eliminate data inconsistency in the database in order to avoid insertion, update, and deletion anomalies (i.e., increase data integrity). Its importance has been advocated by E.F. Codd and noted in computer science literature in the 1970s.

Since the introduction of the concept of "normal form" in relational database by E.F. Codd, there have been many revisions and enhancements to what he originally described as a preferred way in his 1970's paper (Codd, 1972; Bernstein, 1976; Beeri, Bernstein, and Goodman, 1978; Kent, 1983). In computer science discipline, definitions of normal forms are very theoretical and conceptual. However, in MIS and AIS disciplines, in order to help students to comprehend the concept, definitions have been simplified and sometimes overshadowed by sample data. Over the years, these definitions have been changed and deviated from its earlier form defined in computer science discipline. We will review the way a well-intended effort to help students learn the database normalization rules has actually changed the meaning of Codd's rules and thus departed from its goal.

FIRST NORMAL FORM (1NF)

There are at least three types of definitions (and a combination of these three) of the first normal form in the textbooks today. The first two types of definitions relate to the term "repeating groups".

First Definition for 1NF

The term "repeating groups" refers to a table containing duplicate rows of a same record. Table I-a below is an example showing what these textbooks meant of the term "repeating groups". Table I-a shows that each invoice (i.e., Invoice Number) is allowed to have more than one row in the table.

Table I-a: Multiple Rows for the Same Invoice

Invoice Number	Invoice Date	Items Purchased	…
1125	March 1, 2009	G50367	…
		J32984	…
1126	March 2, 2009	H84976	…
		G36748	…
…	…	…	…

The first type of definition for 1NF defines that a table is in first normal form when there are no empty cells in the table, referring to the empty cells in Invoice Number and Invoice Date data fields/columns in Table I-a. As a result, under this definition, the above table is in 1NF when we populate/fill out the data in the empty cells in Invoice Number and Invoice Date data fields as shown below in Table I-b (see in Gelinas and Dull, 2012; Bagranoff, Simkin, and Norman, 2010).

Table I-b: First Norm Form with Multiple Rows Filled

Invoice Number	Invoice Date	Items Purchased	…
1125	March 1, 2009	G50367	…
1125	March 1, 2009	J32984	…
1126	March 2, 2009	H84976	…
1126	March 2, 2009	G36748	…

Shortfalls of First Definition for 1NF

Many authors, and even product support websites such as Microsoft, misinterpret the meaning of "repeating groups". Repeating group, in its original meaning, implies that a column can accommodate multiple values (Pascal, 2005; Sen, 2009). Authors using this type of definition for 1NF imply that "repeating group" contains an entry/transaction occupying more than one row of recording space in a table and then indicates that the solution is to populate the empty cells, which actually does nothing to the problem of repeating groups as they have defined. Actually, its focus is on the empty cells which have nothing to do with "repeating group", columns or rows of the entity.

First of all, this type of definition for 1NF is inconsistent with the original meaning of "repeating group". Secondly, the solution they suggested (i.e., populating the cells) is totally useless because it does not remove the repeating group which they have defined. Thirdly, probably not the last, the repeating group problem, as they have defined, will be automatically resolved in the 2NF; i.e., functional dependencies. We will discuss this later in 2NF.

Second Definition for 1NF

The second type of definition for 1NF defines that a table is in first normal form when there is no repeating group of data relating to its key attribute(s). Repeating group is defined here as multiple values existing for an attribute in a specific record (see Table II-a). The solution suggests that the table is in 1NF by separating the repeating group of data from the original key attribute(s); e.g., one stores invoice-related information and the other item-related information (with Items Purchased being the key), using a foreign key to link them as shown below in Table II-b (Hall, 2011; Bodnar and Hopwood, 2010) or placing the repeating data with a copy of the original key attribute(s) in the same table using both Invoice Number and Items Purchased attributes as a composite primary key as seen in Table II-c (Hurt, 2010).

Table II-a: Multiple Items of Data in an Attribute

Invoice Number	Invoice Date	Items Purchased	…
1125	March 1, 2009	G50367, J32984	…
1126	March 2, 2009	H84976, G36748	…

Table II-b: A Separate Relation (table) with Invoice Number as Link

Invoice Number	Invoice Date	…
1125	March 1, 2009	…
1126	March 2, 2009	…

[Items Purchased]	[Invoice Number]	…
G50367	1125	…
J32984	1125	…
G36748	1126	…

Table II-c: Repeating Groups of Data Extended into Rows

Invoice Number	Invoice Date	Items Purchased	…
1125	March 1, 2009	G50367	…
1125	March 1, 2009	J32984	…
1126	March 2, 2009	H84976	…
1126	March 2, 2009	G36748	…

Shortfalls of Second Definition for 1NF

Authors following this definition may interpret the meaning of repeating groups consistently with the original meaning, but their solution is unnecessary and out of the order in the normalization process. Database normalization involves a process of applying progressive normal forms to the tables in a database, which means that these norm forms are dependent on each other with a strict progressive relationship (i.e., 1NF, 2NF, 3NF, … by such a sequence). The solution suggested here involves with the original key attribute(s) in 1NF. Actually, the key concept is embedded under the concept of functional dependencies, which will be introduced in 2NF. As a result, the solution violates the progressive requirement in the normalization by using the key concept in 1 NF.

The solutions to the "repeating groups" problem have been documented in the literature as (1) entering appropriate data in the empty columns of rows containing the repeating data and (2) placing the repeating data, along with a copy of the original key attribute(s), in a separate relation (i.e., Table II-b) or in the same relation (i.e., Table II-c). The first approach introduces more redundancy into the original table and the second approach advances further into the normalization process (Connolly and Begg, 2010).

Third Definition for 1NF

The third type of definition for 1NF specifies that a table is in first normal form when there are atomic, indivisible, or simple-valued attributes (Codd, 1970; Silberschatz, Korth, and Sudarshan, 2002; Ricardo, 2004; Connolly and Begg, 2010).

An attribute may be a simple-valued, single-valued, composite, or multi-valued attribute. Definitions for these terms can be found in most MIS textbooks. For example, a simple-valued attribute is an attribute which cannot be subdivided (e.g., age, sex). A single-valued attribute is an attribute that can have only a single value (e.g., Social Security Number, Date, etc.). A single-valued attribute may not be a simple attribute because a single-valued may be subdivided (e.g., first three digits and last four digits of the Social Security Number). A composite attribute is an attribute that can be further subdivided to yield addition attributes (e.g., address into street, city, state, and zip code). A multi-valued attribute is an attribute that can have many values (e.g., a person has many phone numbers).

Codd (1970) defined a relation as an n-ary relation R of having the following properties: 1) each row represents an n-tuple of R, 2) the ordering of rows is immaterial, 3) all rows are distinct, 4) the ordering of columns is significant, and 5) the significance of each column is partially conveyed by labeling it with the name of the corresponding domain. Codd also mentioned related concepts, such as active domain, primary key, foreign key, and non-simple domain. His work, and later work by others, in the normal forms has placed emphasis on the domains (i.e., attributes). As a result, the third kind of definition for the first normal form seen here is consistent with Codd's earlier form of 1NF.

In the 70's and 80's, the concept of atomicity was used consistently in the definition of 1NF. One of the primary reasons that authors walked away from this concept is because of its abstract in nature and vagueness.

Atomicity implies that data are in its smallest unit; i.e., data cannot be decomposed further into smaller pieces. However, sometimes it is really difficult to determine whether data are in its atomic stage. For example, data with a data type of "text" (or characters) can always be decomposed to a string of letters (Sen, 2009).

Problems with First and Second Types of Definitions for 1NF

The AIS textbooks that we have surveyed (see Appendix A) use either the first or second type of definitions and solutions for 1NF. The first type of definition totally ignores problems with composite and single-valued attributes and the "filling out" solution used is invalid and useless, which actually makes the table more redundant (Connolly and Begg, 2010). On the other hand, the solution for the second type of definition not only ignores problems with composite and single-valued attributes, but also violates the progressive requirement of the normalization distracting the emphasis from the fundamental work that 1NF is supposed to do.

The conventional definition of first normal form places an emphasis on the attributes, not on the tuples, making sure that each and every attribute in a table contains only a simple value. It leaves the relationships among attributes to the second normal form (discussed next). The second type of definition for 1NF actually includes a relationship component that is not necessary because the second normal form will address them. However, like the first type of definition, the second type ignores composite and single-valued attributes as well. For example, the following tables will pass the first normal form under the first and second type of definitions. Note that Table III contains a composite attribute with a number of simple-valued attributes in it and Table IV is a single-valued but composite attribute.

Table III: Composite Attribute in Address Column

Customer No	Name	Address	…
10233	ABC	1 Main St., Georgetown, DC 20001	…
10244	CBS	One Happy St., Foodtown, DC 20004	…

Table IV: Single-Valued and Composite Attribute in Employee No

Employee No	Year Started	Title	…
ACCT-0267	1962	Assistant Manager	…
FINC-0315	2003	Director	…

Since a composite attribute contains data value that can be subdivided, it may be of interest to the users on these individual subdivided values. For example, the users may have interest in having information regarding customers with addresses in certain city, state, and/or zip code. If so, the composite attribute will make the programming of separating these sub-values very difficult, if not impossible. In addition, there will be data redundancy in the table (i.e., same city name, same state name, and same zip code). As a result, it is more likely to be subject to data inconsistency.

Single-Valued and Composite Attributes

The single-valued attribute, like the one shown in Table IV above, will permit certain data redundancy and will also create potential problems down the road. For example, Employee No is likely to be set up as the key in the table with the first four digits being the department initial, which the employee works for, and last four the employee individual identification number in that department. As a result, these departmental initials will be redundant and is more likely to be subject to inconsistency. In addition, the employees may switch positions to different departments within the company, requiring changes to employees' Employee No. In that case, it is very difficult for the database to keep track of the employees' records when employees have different employee numbers (Silberschatz, Korth, and Sudarshan, 2002).

An Exception to Composite Attributes with a Standardized Input Format and Functions

Although 1NF focuses on the atomic nature of the attributes, there is an exception. For example, "Date" is also a single-valued and composite attribute since it consists of Day, Month, and Year data values. However, we do not need to separate them in the storage as long as we have no frequent uses for these data values individually. We can store them in a standardized "date" format (i.e., system data type) so that we can identify them easily whenever we need to access them individually. For that we can use the date-related functions in the database management system. In sum, we do not need to split the single-valued and composite attributes as long as there are standardized input formats to capture them and functions to extract them in the database management system that we will be using.

SECOND NORMAL FORM (2NF)

There are at least three types of definitions for 2NF published in textbooks. The first type defines that a table is in second norm form if there are no partial dependencies (Hurt, 2010). The second type defines that a table is in second norm form if it is in a first norm form and has no partial dependencies (Gelinas and Dull, 2012; Hall, 2011), while the third type defines that a table is in second norm form if it is in a first norm form and has fully functional dependencies (Bagranoff, Simkin, and Norman, 2010; Heagy, 2005). There are other types of definitions for 2NF which deal with the situation of the data in the table from their first norm form solution (Bodnar and Hopwood, 2010).

Shortfalls of First Definition for 2NF

Using the example from Table I-b; i.e., the first definition for 1NF, the primary key in the table will be both the Invoice Number and Items Purchased attributes together. As a result, the first type of definition for 2NF (i.e., no partial dependencies) will help develop the table into at least two 2NF tables seen in Table II-b. However, the first type of definition for 2NF may have no applications on the second type of definition for 1NF when repeating group of data may have already been separated into two tables, as shown in Table II-b. The Invoice table in Table II-b has only a single-attribute key with no partial key since they were separated in 1NF (as seen in Hall, 2011).

The normalization rules are designed to prevent data anomalies and data inconsistencies for all situations. The normalization process applies a series of progressive normal forms over an existing schema in order to bring its components to compliance (Kent, 1983). As a result, the first type of definition for 2NF is not valid because it does not include the progressiveness requirement; i.e., being in 1NF has to be in the definition of second norm form. For example, Tables III and IV will pass these definitions of being 1NF and 2NF tables. However, there are data redundancy and possibly data inconsistency in the tables.

Table V below shows another example of failures which the first type of definition for 2NF may encounter. In this example, assume that members make an annual donation and *Member Number* is the key. Since the key consists of only one attribute, there is no partial dependency application here. However, *Member Number* has a 1-to-many relationship with Donation$ overtime. As a result, *Donation$* is not a dependent of *Member Number*. Using the first kind of definition for 1NF and the first kind of definition for 2NF, this table would be defined as a second norm form table by some novices who have no knowledge about functional dependency (the meaning of a primary key, since the focus of the definition is on "partial dependencies") or by someone who fails to see the relationship between Member Number and Donation$. With this table structure, members' donations will be overwritten every year. As a result, there are no ways that we can see a history of members' donation records.

Table V: Functional Dependencies Between Member No and Donation$

Member Number	**Membership Since**	**Donation$**	**Year**	**…**
82390	March 31, 2001	250	2011	…
62581	April 2, 2004	125	2011	…

Shortfalls of Second Definition for 2NF

The second type of definition for 2NF works only when it follows the first type of definition in first normal form in which the table is assumed to contain a composite key due to its repeating groups of data. In other words, this definition helps prevent data anomalies and data redundancies by separating these composite key attributes into different tables where each of the key attributes can have functional dependencies over their non-key attribute. However, this type of definition does not work on tables with a single-attribute key, e.g., tables from the second type of definition in 1NF when tables were separated. That is, 1NF tables with a single key attribute are automatically 2NF tables under this definition because there is no partial dependency (i.e., partial dependency only exists when the key is a composite key with non-key attributes depending on one or more, but not all, of the composite key attributes). As a result, this type of definition becomes useless when the second kind of definition for 1NF is followed.

Third Definition for 2NF

The third type of definition for 2NF follows the conventional approach in examining the dependent relationships between its key attribute(s) and non-key attributes. It requires an establishment of a primary key and then examines the fully functional dependencies in a table (Bagranoff, Simkin, and Norman, 2010; Heagy, 2005). In Table V, a composite key, which we might have established for the table, consists of Member Number and Year (academic or fiscal year depends on their membership due basis). However, with a thorough examination, we discovered that there are partial dependencies in the table. For example, the attribute "Membership Since" depends only on "Member Number", but not on "Year". As a result, we end up with Table VI-a and VI-b where these two tables are in 2NF; i.e., with all non-key attributes in Table VI-a dependents of "Member Number" and in Table VI-b, dependents of the composite key "Member Number" and "Year".

Table VI-a: Functional Dependency between Membership since and Member Number

Member Number	Membership Since	…
82390	March 31, 2001	…
62581	April 2, 2004	…

Table VI-b: Functional Dependency between Donation$ and "Member Number and Year

Member Number	Year	Donation$	…
82390	2003	200	
62581	2003	100	
82390	2004	250	…
62581	2004	125	…

THIRD NORMAL FORM (3NF)

There are less discrepancies in the definition for third normal form, except for the progressive requirement. That is, a table must already be in a 2NF. The additional requirement in 3NF is that the table contains no transitive dependencies. There are two types of definitions for 3NF seen in textbooks - one defines that a table is in 3NF when there are no transitive dependencies (Bodnar and Hopwood, 2010), and the other defines that a table is in 3NF if the table is in 2NF and contains no transitive dependencies (Bagranoff, Simkin, and Norman, 2010; Gelina and Dull, 2008; Hall, 2011; Heagy, 2005).

Shortfalls of First Definition for 3NF

The first type of definition is invalid since it ignores the progressive requirement. The authors also suggest that there are less discrepancies in AIS textbook definitions of the third normal form because there is less concentration on third normal form material on the part of instructors.

CONCLUSION

The conventional method from computer science discipline focuses on atomic (or simple-valued) attributes in 1NF, the relationship between the primary key and non-key attributes in 2NF (or fully functional dependencies), and the relationship among non-key attributes in 3NF (or no transitive dependencies). Since these definitions are provided in a very conceptual and theoretical way, it makes them more difficult for the students to learn and comprehend. In addition, from the textbooks' point of view, it will not be feasible to provide examples for every possibility in the data variations. Furthermore, authors of the textbooks tend to follow the example they came up with along with the development of these definitions. For the textbooks that we have surveyed, all of those who cover database normalization (i.e., 1NF, 2NG, and 3NF) use only one example to carry out the discussions of the first three normal forms. As a result, their definitions are example dependent. That is, their definitions ONLY work in certain situations similar to the example they provided.

Due to the abstractions in the definitions, many MIS scholars have come up with alternative techniques involving algorithms and procedures to replace the conceptual approach with a procedural approach of database normalization (Diederich and Milton, 1988; Rosenthal and Reiner, 1994; Kung and Tung, 2006). Many MIS and AIS scholars have included sample data in order to help students visualize the changes in table structures during the normalization process. However, over the years, some of these textbooks have switched the focus from attributes to tuples (i.e., rows/records).

As a result, some of the definitions are changed; that is, they deviate from earlier forms. They do not produce corresponding benefits; i.e., reducing data anomalies, data redundancies, and data inconsistencies. These definitions only work in certain situations; for example, 1) when there are no single-valued and no composite attributes, 2) when tables have been constructed with semi-1NF and/or semi-2NF requirements, 3) when a composite key exists, and/or 4) when tables have been constructed with semi-2NF and/or semi-3NF requirements. Students and educators need to recognize what may be a natural process of deviation from earlier forms due to semantic drift. Appendix A shows the definitions from the textbooks surveyed.

The authors see indications of inevitable change in definition and in application (even when the definition does not change). Subsequent examination of this "semantic drift" from the Codd rules for database normalization may reveal that this process is inevitable and fundamental to any set of rules that require the parsing of meaning that business and accounting data contain into what is, in effect, strict categories or attributes. This issue encourages speculations that the problem of semantic drift is not just one of accounting information systems textbook writers trying to database normalization with easier to understand realistic examples. If in fact this is the case, this characteristic of database normalization of attribute classification has significant implication for accounting educators and beyond that for software and database developers.

This paper has shown that most of the first normal form definitions published in the textbooks ignore the concept of atomicity of attributes, especially single-valued and composite attributes, and tend to lose the rigorousness of the progressive requirement between first normal form and third normal form. This limits its applicability to certain situations only, such as the examples they have provided in the textbooks.

Since there are a variety of definitions for the first three normal forms available in the literature and textbooks, it becomes confusing sometimes. Some of the definitions from CIS and MIS textbooks are listed in Appendix B.

Many authors choose to skip the individual definition by focusing on overall requirements for each table (Cerci and Gottlob, 1986). As a result, we list them as follows:

- Every attribute in a table must consist of only simple-valued data; i.e., indivisible attribute. That is, there is no composite or single-valued attribute if possible.
- There are no multi-valued/repeating groups of attributes in a table.
- In each table, there must be a primary key; that is, a single attribute or a combination that uniquely determines the non-key attributes in that table.

- If there is a composite key in a table, there are no partial dependencies allowed.
- There are no transitive dependencies in a table.

AUTHOR INFORMATION

TJ Wang, Ph.D. is an associate professor in the College of Business and Public Administration at Governors State University. He holds a B.S., M.B.A., and Ph.D., all in accounting, from Rutgers University, the State University of New Jersey. He has written papers related to relational database and won the Best Paper awards in the AIS Educator Annual Conference in 2002, 2004 and 2005. He has published papers in the *American Journal of Business Education, Journal of College Teaching and Learning, and Review of Business Information Systems*. He has taught at Robert Morris University, University of Wisconsin-Milwaukee, Bellevue Community College, Rutgers University and New Jersey Institute of Technology.

David K. Dennis, Ph.D. C.P.A. is a Professor in the Department of Business, Accounting and Economics at Otterbein College. He holds a B.S. B.A. from Ohio State University, a M.B.A. from Wright State University, and a Ph.D. from the University of Cincinnati, all in accounting. He has been a long time advocate for improved and innovative teaching and learning models, practices in the classroom and in accounting education in general. As a member of the Ohio Region of the AAA he has participated in panels on Innovations in Accounting Education. His current interest is in the application of L. Dee Fink's model of "significant learning." Recently he has organized and directed "Ohio Master Teachers" panels at the AAA Ohio and Annual AAA meetings.

REFERENCES

1. Bagranoff, Nancy A., Mark G. Simkin, and Carolyn Strand Norman. *Core Concepts of Accounting Information Systems*. 11th Edition. John Wiley and Sons, Hoboken: NJ 2010.
2. Beeri, Catriel, Philip A. Bernstein, and Nathan Goodman. 1978. "A Sophisticate's Introduction to Database Normalization Theory." In Proceeding of the 4th International Conference on Very Large Data Bases (West Berlin).
3. Bernstein, Philip A. 1976. "Synthesizing Third-Normal-Form Relations from Functional Dependencies." *ACM Transactions on Database Systems*, Vol. 1, No. 4, pp. 277-298.
4. Bodnar, George H., and William S. Hopwood. *Accounting Information Systems*. 10th Edition. Prentice Hall, Upper Saddle River: NJ 2010.
5. Ceri, Stefano, and Georg Gottlob. 1986. "Normalization of Relations and Prolog." *Communications of the ACM*, Vol. 29, No. 6, pp. 524-544.
6. Codd, Edgar Frank. 1970. "A Relational Model of Data for Large Shared Data Banks" *Communications of the ACM*, Vol. 13, No. 6, pp. 377-387.
7. Codd, Edgar Frank. 1972. "Further Normalization of the Database Relational Model." In *Data Base Systems Courant Computer Science Symposia*, Vol. 6, R. Rustin Ed. Prentice-Hall, Englewood Cliffs, NJ.
8. Connolly, Thomas M., and Carolyn E. Begg. *Database Systems*. 5th Edition. Addison-Wesley, New York: NY 2010.
9. Diederich, J. and J. Milton. 1988. "New Methods and Fast Algorithms for Database Normalization." *ACM Transactions on Database Systems*, Vol. 13, No. 3, pp. 339-365.
10. Dunn, Cheryl L., J. Owen Cherrington, and Anita S. Hollander. *Enterprise Information Systems: A Pattern-Based Approach.* 3rd Edition. McGraw-Hill, New York: NY 2005.
11. Gelinas, Jr., Ulric J., and Richard B. Dull. *Accounting Information Systems*. 9th Edition. South-Western, Mason: OH 2012.
12. Hall, James A. *Accounting Information Systems*. 7th Edition. Thomson, Mason: OH 2011.
13. Heagy, Cynthia D. *Accounting Information Systems: A Practitioner Emphasis*. 5th Edition. Thomson, Mason: OH 2005.
14. Hurt, Robert L. *Accounting Information Systems: Basic Concepts and Current Issues*. 2nd Edition. McGraw-Hill, New York: NY 2010.
15. Jones, Frederick L., and Dasaratha V. Rama. *Accounting Information Systems-A Business Process Approach.* 2nd Edition. Thomson South-Western, Canada 2006.

16. Kent, William. 1983. "A Simple Guide to Five Normal Forms in Relational Database Theory." *Communications of the ACM*, Vol. 26, No. 2, pp. 120-125.
17. Kung, Hsiang-Jui, and Hui-Lien Tung. 2006. "An Alternative Approach to Teaching Database Normalization: A Simple Algorithm and An Interactive e-Learning Tool." *Journal of Information Systems Education*, Vol. 17 (Fall), Issue 3, pp. 315-325.
18. Pascal, Fabian. 2005. "What First Normal Form Means Not" Online. http://www.dbdebunk.com/page/page/629796.htm
19. Ricardo, Catherine M. Databases Illuminated. Jones and Bartlett, Sudbury: MA 2004.
20. Rob, Peter, and Carlos Coronel. *Database Systems: Design, Implementation, and Management*. 5th Edition. Course Technology, Boston: MA 2002.
21. Romney, Marshall B., and Paul John Steinbart. *Accounting Information Systems*. 10th Edition. Prentice Hall, Upper Saddle River: NJ 2008.
22. Rosenthal, A., and D. Reiner. 1994. "Tools and Transactions – Rigorous and Otherwise – for Practical Database Design." *ACM Transactions on Database Systems*, Vol. 19, No. 2, pp. 167-211.
23. Sen, Anith. 2009. "Facts and Fallacies about First Normal Form". Online. http://www.simple-talk.com/sql/learn-sql-server/facts-and-fallacies-about-first-normal-form/
24. Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. 4th Edition. McGraw-Hill, New York: NY 2002.
25. Vaassen, Eddy. *Accounting Information Systems-A Managerial Approach*. John Wiley and Sons, Hoboken: NJ 2004.

APPENDIX A: DEFINITIONS OF RELATIONAL DATABASE NORMAL FORMS PUBLISHED IN AIS TEXTBOOKS

AIS Textbook	Definitions		
	1NF	**2NF**	**3NF**
Bagranoff, Simkin, and Norman (pp. 155-157, 11th Edition, 2010)	All attributes are well defined (i.e., a flat file; populating null cells)—1st Type Definition	1NF and all the data items in each record depend on the record's primary record key—3rd Type Definition	2NF and contains no transitive dependencies—2nd Type Definition
Bodnar and Hopwood (pp. 459-460, 10th Edition, 2010)	No repeating groups (i.e., duplicate rows)—2nd Type Definition	No key is allowed to determine a nonkey	No nonkey can determine another nonkey—1st Type Definition
Gelinas & Dull (pp. 154-160, 9th Edition, 2012)	No repeating groups (i.e., duplicate rows; populating null cells) —1st Type Definition	1NF and has no partial dependencies—2nd Type Definition	2NF and has no transitive dependencies—2nd Type Definition
Hall (pp. 439-443, 7th, 2011)	Removing repeating groups (in the attributes; populating null cells)—2nd Type Definition	1NF and removing partial dependencies—2nd Type Definition	2NF and removing transitive dependencies—2nd Type Definition
Heagy (pp. 11-7 ~ 11-11, 2005)	No multi-valued or repeating elements (separate tables)— 2nd Type Definition	1NF and all nonkey data element must depend on the entire primary key—3rd Type Definition	2NF and has no transitive dependencies—2nd Type Definition
Hurt (pp. 116-118, 2nd, 2010)	Eliminates repeating groups (in the attributes)— 2nd Type Definition	Eliminates redundant data—1st Type Definition	Eliminates columns not dependent on the primary key
Dunn, Cherrington, and Hollander, 2005 Jones & Rama, 2006 Romney & Steinbart, 2008 Vaassen, 2004	No coverage	No coverage	No coverage

APPENDIX B. SAMPLE DEFINITIONS OF NORMAL FORMS FROM CIS AND MIS TEXTBOOKS

1NF

"A relation whose domains are all simple can be represented in storage by a two-dimensional column-homogeneous array of the kind discussed above" (p. 381, Codd, 1970).

"A domain is atomic if elements of the domain are considered to be indivisible units". (Silberschatz, Korth, and Sudarshan, 2002)

"A relation in which the interaction of each row and column contains one and only one value" (Connolly and Begg, 2010)

A relation is in first normal form if, and only if, every attribute is single-valued for each tuple. The domains of the attributes of the relation are atomic (Ricardo, 2004).

The term first normal form describes the tabular format in which all the key attributes are defined, there are no repeating groups in the table (in other words, each row/column intersection can contain one and only one value, not a set of values, and all attributes are dependent on the primary key (Rob and Coronel, 2002)

2NF

A relation that is first normal form and every non-primary key attribute is fully functionally dependent on the primary key (Connolly and Begg, 2010).

A relation is in second normal form if, and only if, it is in first normal form and all the non-key attributes are fully functionally dependent on the key (Ricardo, 2004).

A table is in second normal form if it is in 1NF and it includes no partial dependencies; that is, no attribute is dependent on only a portion of the primary key (Rob and Coronel, 2002).

3NF

A relation that is in first and second normal form and in which no non-primary-key attribute is transitively dependent on the primary key (Connolly and Begg, 2010).

A relation is in third normal form if, whenever a non-trivial functional dependency $X \rightarrow A$ exists, then either X is a superkey or A is a member of some candidate key (Ricardo, 2004).

The definition of 3NF is the original one developed by Codd. It is sufficient for relations that have a single candidate key, but was found to be deficient when there are multiple candidate keys that are composite and overlapping (Ricardo, 2004).

A table is in third normal form if it is in 2NF and it contains no transitive dependencies (Rob and Coronel, 2002)

NOTES

