

The Data Administration Newsletter (TDAN.com)

[Robert S. Seiner - Publisher](#)

[Main Menu](#)

[New Articles](#)

[Article Archive](#)

[Special Features](#)

[What's New](#)

[Contact the Publisher](#)

EXPENSIVE LESSONS FROM THE PAST: PROFIT FROM THEM

Written by Michael M. Gorman - Whitemarsh Information Systems Corp.

[\[Return to the Article Archive\]](#)

Published in TDAN.com July 2005

1.0 Metadata Management

To be successful within a Data Management environment, the enterprise must have quality data management that recognizes, embraces, and manages data on all its levels. A four level approach was created by the National Institute of Standards and Technology in the early 1980s. It was created to support the architecture of metadata repositories. Generally, as shown in Figure 1, there are four levels to data: the fundamental level, the repository definition level, the repository level, and the application level. Across the columns from this same figure, there are three sets of pairs: application level pair, repository level pair, and repository definition pair. Within the cells of these rows and columns are the metadata contents. All the levels and pairs must be properly managed to avoid the data standardization failures. The repository and system supporting this failed effort is described further in Section 2.

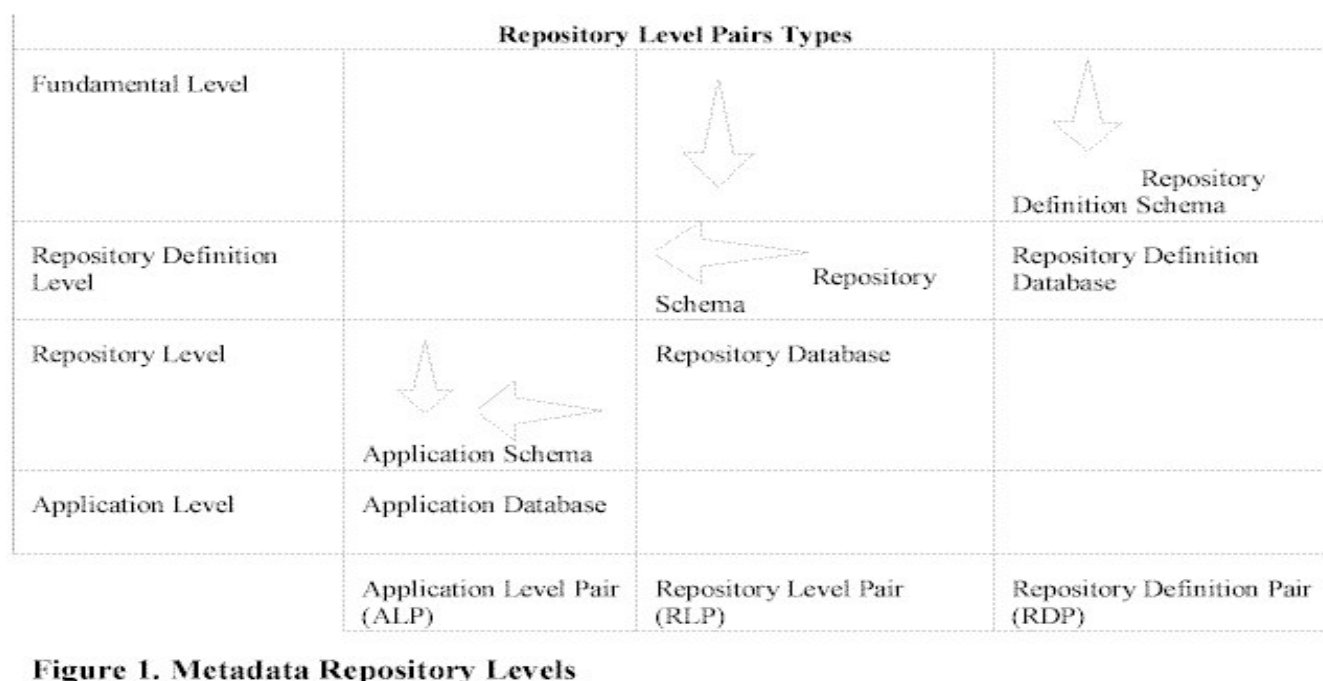




Figure 1. Metadata Repository Levels

Figure 2 illustrates value sets corresponding to Figure 1 to help explain these levels and pairs. A database application (Application Level Pair) consists of value sets from the application level and the database schema from the repository level that represents the metadata for those value sets. In this example, the DBMS schema has a database table with three columns. The database table is Employee, and the three columns are Employee Name, Employee ID, and Department Name. In this case, the value set for Employee is "P. Shaw," with an Employee Identifier of: "525-88-2876," and the Employee's Department is "SQL Development."

In this example of an Application Level Pair, the metadata names are at the Repository Level and the data values are at the Application Level. Together, these two levels comprise the Application Level Pair.

The metadata names that are at the Repository Level come from the Repository Level pair. That is, there is a software system, called a metadata repository, that has its own set of metadata types ((T)able and (C)olumn), and for these types, their own instances: T=employee, C=emp_name, C=emp_id, and C=dept name. To populate a database that resides as an application level pair, the metadata repository is queried for the tables and columns. This is represented in Figure 3 by the horizontal arrow between the Repository Level Pair and the Application Level Pair. Because the metadata repository exists, then any database that employs the employee table can retrieve its table and column set from the metadata repository. This enables a level of data standardization that would not be practically possible if no metadata repository existed.

Instances Example for Repository Level Pairs			
Fundamental Level			Fundamental Types: ENTITY ATTRIBUTE RELATIONSHIP
Repository Definition Level		Meta Types: TABLE (T) COLUMN (C)	Fundamental Entity Type Instances: table column

Repository Level	T: Employee COL:EMP_NAME COL:EMP_ID COL:DEPTNAME	Meta Type Instances: T: employee C: emp_name C: emp_id C: deptname	
Application Level	DBMS Schema Instances: P. Shaw 525-88-2876 SQL Development		
	Application Level Pair (ALP)	Repository Level Pair (RLP)	Repository Definition Pair (RDP)

Figure 2. Data instances for the levels and application pairs.

The metadata types, that is, table (T) and column (C) come from the repository definition pair. Very sophisticated metadata repository environment can accomplish these definition with reasonable effort. In this example, the three fundamental types are Entity, Attribute, and Relationship. In this specific example, the two fundamental entity types that have been defined are Table and Column. For each, there are attributes, like the table's name, the author of the table, and the definition of the table. In this example, there are only the two entity type instances: Table and Column. These instances are, in turn, read from the Repository Definition pair into the Repository Level pair and, within the Repository Level Pair become its types. In robust metadata repositories there may be hundreds of fundamental entity types such as DBMS, Business Rule, Information System, SQL View, Mission, Organization, Function, and Business event.

In summary, there are three sets of pairs--repository definition, repository level, and application level--are employed to fully define metadata repository environments.

These three pairs are critical to a well-ordered data management environment because these enable an enterprise to restrict its fundamental entity types, which, in turn, can be used to restrict the metadata that can be defined, which, in turn, can be used restrict the database application's tables, columns and all other metadata objects.

Other than the intellectual discipline of attempting to bring order out of data chaos, are there any practical applications and benefits from this strategy? The answer is emphatically, yes. The following problems, endemic to many IT environments, are addressed through sophisticated metadata environments:

- Stove pipe systems, which are typified by 1) conflicting semantics, 2) non-shared data, and 3) costly, inefficient change mechanisms exist if the metadata management environment consists only of **Application Level Pairs**.

- Stove pipe systems are prevented when their application level pairs are based on **Repository Level Pairs**. Further, because of repository level pairs, application level pair systems can have integrated metadata.
- Meta data repositories can only be build through use of pre accomplished **Repository Definition Pairs**.

The repository level pairs are created through sophisticated IT environments that can employ repository definition pairs to quickly and effectively develop the metadata repository environment. This is all much more important than just elegant IT. A 1995 study and report was created by a large enterprise on the state of its Information Resource Management. The following is a list of the information resource management's weaknesses that could have either been completely avoided or severely reduced if there had been a comprehensive metadata management program in place. Each item on the list points to the metadata repository level pair that would have prevented or severely reduced the weakness.

Current IRM program is poorly defined with conflicting policy and procedures (Repository Level Pair)

Technology solutions implemented prior to process improvements (Application Level Pair)

Current IRM program does not measure customer needs well (Application Level Pair)

Benefits from sound investments in IRM need to be articulated (Repository Level Pair)

Lack of standards for creation, collection, accessibility, storage, retrieval, protection, and destruction of electronic information (Repository Level Pair)

Many stove-piped and incompatible systems require costly changes to share information and to interoperate (Application Level Pair)

Systems and communication infrastructure based on old costly and inefficient technologies which impede the decisions maker's access to information (Application Level Pair)

Lack of data standardization which results in inconsistent data, difficult integration, and costly software development and maintenance. (Repository Level Pair)

It should be noted that only when a higher pair (e.g., Repository Level pair) exists can the lower level pair be properly managed (e.g., Application Level pair). Even though none of the IRM weaknesses are attributed to the Repository Definition Pair, this pair not only facilitates the Repository Level Pair, it also provides control and integrity.

2.0 Data Element Standardization (DES) Program

The DES approach was supported by a repository that was only a repository level pair (see Figures 1 and 2). Not only was it restricted in pairs, it was also severely restricted in scope to just data elements. Thus, the DES repository was not able to manage, for example, DBMS, Business Rule, Information System, SQL View, Mission, Organization, Function, and Business Event, which, along with many other types of metadata, are essential for a well ordered metadata environment. Further, because the DES repository only consists of a Repository Level Pair, it was unable to either manage or control instances from the Application Level Pair. Because of these severe restrictions, DES

clients, that is, designers, developers and end-users were unable to accomplish proper metadata management. In short, the effort was doomed from day one.

A systems engineering consulting organization performed an internal study around 1994 that showed the following problems with the DES approach:

- A fundamentally flawed data standardization model
- No accommodation for enterprise wide data architectures
- Multiple implementation technologies
- Central standardization and maintenance authority

Another, and completely independent organization also did a study of the data element standardization approach in 1994. Their overall conclusions were that:

There are poor data administration practices
The enterprise has not determined its corporate data requirements
Data element standardization procedures are ineffective
The DES repository system does not support data administration goals

There was a response on all these points. While fundamentally the audited organization concurred and expressed firm resolve to eliminate these problems, none were.

The sections that follow outline the key deficiencies in the DES effort from first assessment study and concludes with the characteristics of what must exist for data standardization to be successful, that is, having the definitive set of the right data at the right time to the right person with an unambiguous and universally accepted meaning. In today's enterprise's standardization cannot be an option, and further, it is the essential first ingredient towards achieving for a data sharing environment.

2.1 Flawed Data Standardization Model

The DES data standardization model and its procedures focused almost exclusively on data elements, which, because of the way they were engineered, caused the data elements to be database table columns. The DES engineering model that caused this was its name construction process. That is, <prime word> <modifier> and <class word>. Prime word was, by DES convention the "table name;" hence, all data elements became database table columns. The simple consequence of this was that DES would be complete only after all the data elements were standardized. But because the data elements were actually database table columns, that would then be, never.

The DES naming standard required the use of a single class word, for example, identifier, code, or text. As an example, if a data element were to be the unique data record selector for a database table, the class word would be "Identifier." How, however would that square with the data element, Social Security Number? To adhere to the rules, Social Security Number would have to be changed to Social Security Identifier. But that too was a problem because the Social Security Number is really a set of three numeric codes. Because this "problem" became to complicated to deal with, an exception to the DES naming standards was granted.

As another example, consider Telephone Number. This too had to be granted a naming standard exception because it a) sometimes was not a number, b) was really an identifier, c) was really a

compound data element consisting of well defined individual parts, d) the parts in common had discrete sets of valid values, and e) it was not the kind of number that could be combined with other numbers through mathematical operations.

Slowly but surely, because of the fundamental engineering problem of placing all the semantics of the data element into its name, the quantity of exceptions grew and grew. Exceptions, however, should be “engineered” out of the design from the very beginning. In addition, the overall utility of a data element, given that almost all its semantics were embedded in its name, is severely compromised. How, for example, would you find all identifiers if the data element is named Social Security Number? Or how would you know that a Social Security Number is a compound data element consisting of three codes? All of this semantic meaning was lost due to DES engineering failures. The engineering design also did not address derived data elements such as Final Monthly Balance, or compound data elements such as Federal Stock Numbers.

The DES repository only contains data element metadata. It does not contain data model metadata or any other metadata such as business rules or other the IT architecture products. That missing metadata, all essential for a complete environment, had to be created and managed in other CASE/repository tools. These other CASE/repository tools were not integrated with the DES. In short, the DES and all the other CASE/repository tools were just another set of stove-pipes. Had the enterprise implemented the late 1980s ANSI Information Resource Dictionary System standard through any of its several implementations, as was recommended by the auditors in 1994, this single-level and single meta object (data element) problem could have been avoided.

For existing systems, to conform to the DES approach, a report from its Repository Level Pair would be produced and then database application builders would make sure that the database column names conformed exactly to the DES data element names. For new systems, a data element proposal package was created and submitted to a data element standards group for approval. Until approved, nothing could proceed. This “don’t proceed” requirement was impossible to meet, monitor, or maintain because the need for new databases across the entire enterprise greatly overwhelmed the ability of the data element standardization team to review and approve each and every database schema creation or change. In short, DES failed by its own engineering and processes.

The DES effort also never became a critical component in any system’s development methodology because the DES data element proposal package creators had to wait months for their data elements to be approved. Many never were. Consequently, after new data element proposal packages were created and submitted, a “box” was checked, and then systems development efforts just continued. Waiting was not an option if schedules were to be met.

2.2 No Accommodation for Enterprise Wide Data Architectures

The DES engineering approach also did not address all the different data model architectures such as reference data, enterprise resource planning (ERP) packages, data warehouses and the like. Under all these different data architecture classes, core data elements proliferate under slightly different names. For example, there could be use of specific data elements associated with an Inventory Item, such as Beginning Balance Inventory Item, Ending Balance Inventory Item, Inventory Item Reorder Point, or Inventory Item Average Quantity On Hand. All these variations are minor, but important, variations. Each, under the DES approach, had to be submitted and standardized as a separate data element, which only really addressed the data element’s name and definition. Other issues such as languages, value domains, security and privacy, business rules and the like were neither addressed nor accommodated.

2.3 Multiple Implementation Technologies

The DES approach had no way of handling the different requirements and/or capabilities of implementation environments such as database management systems, programming languages, and data types, precision and scale. Some DBMSs allow names up to 128 characters while others restrict the name to 32 characters. Are all to be restricted to the smallest name length? What about abbreviations? How are they managed? What about numeric precision? In different DBMSs, an integer number can have different precisions. How are different but equivalent value domains handled? A very simple example is gender, that is, 1 is male, M is male, and Male is male. Are all variations stored and managed? Are they mapped one to the other? None of these problems were handled by the DES approach.

2.4 Centralized Data Standardization Authority

The very fact that there was a centralized approach to data standardization was, in itself, a fundamental problem. There was the never ending increase in the quantity of database columns that then had to be standardized. For example, if a new database had 200 tables and each table had 15 columns, then instantly there had to be 3,000 new data elements standardized. If each data element required 15 minutes of effort then it would take about 4 staff months to standardize one database. Two of the enterprises largest business units have well over 125,000 discrete systems. If there are only 100,000 databases across all these systems then it would take about 33,000 staff years to accomplish the data standardization effort. Not possible.

3.0 The Enterprise's New Data Sharing Program

As a consequence of DES failing it was decided to take an inverse approach. The existing data standardization approach was canceled and replaced with a new data sharing directive. This new approach does not really address problems cited in the auditors' studies. Rather, it avoids dealing with them altogether by pushing them down onto the business units to accomplish. A problem ignored is a problem solved?

From Section 2, the problems were:

- A fundamentally flawed data standardization model
- No accommodation for enterprise wide data architectures
- Multiple implementation technologies
- Central standardization and maintenance authority

The new data sharing approach resolves these in the following way. First, there is no data standardization model. Second, there is no strategy for enterprise wide data architectures. Third, the need for or the ability of have multiple implementation technologies is not addressed. And, fourth, there is clearly no central data standardization or maintenance authority.

In the place DES, the new directive simply states that data is an essential enabler and shall be made visible, accessible, and understandable to any potential user as early as possible in the life cycle to support mission objectives. That data assets shall be made:

Visible by creating and associating metadata ("tagging"), including discovery metadata.

Accessible by making data available in shared spaces

Understandable by publishing associated semantic and structural metadata in a federated metadata registry.

And, to enable trust, data assets shall have an authoritative source. Data interoperability shall be supported by making data assets understandable and by enabling business and mission processes to be reused where possible.

The policy also states that semantic and structural agreements for data sharing shall be promoted through communities of interest (COIs)). And finally, that data sharing concepts and practices shall be incorporated into education and awareness training and appropriate enterprise processes. The directive however gives zero guidance on how any of these policies are to be accomplished.

In addition to the policy statements the new approach contain a definition of data asset. It states that a data asset is any entity that is comprised of data. For example, a database is a data asset that is comprised of data records. A data asset may be a system or application output file, database, document, or web page. A data asset also includes a service that may be provided to access data from an application. For example, a service that returns individual records from a database would be a data asset. Similarly, a web site that returns data in response to specific queries (e.g., www.weather.com) would be a data asset. A human, system, or application may create a data asset. A careful review of this definition leads one to the conclusion that virtually anything that acquires, stores, and/or produces data could be a data asset. That means that all data within the enterprise is required to be: visible, accessible, understandable, trusted, and interoperable.

Under the new approach, 100% of data standardization is the responsibility of the business units, and within and across them, the Communities of Interest. If data standardization is not accomplished in a manner that ensures understanding-based data interoperability then the users are just plain "out of luck." The new directive merely provides the requirement for data to be prepared such that it can be shared. The criterion level at which data can be shared, that is because it has the right characteristics, i.e., understood, trusted, visible, etc., is completely undefined. Worse, sharable data to one person may be completely unintelligible to another. The new directive and its data management supports offer no guidance on these issues.

While it is laudable to have quality goals and objectives, it is also critical that policies and procedures be put into place that can actually accomplish the work. It's one thing to declare that a problem has to be solved, and it's quite another to show the way. The first without the second is just empty rhetoric. For the new program to have any chance of success, it must be supplemented with policies and procedures that are thoroughly engineered, scalable, and must be both efficient and effective. None were provided and evidently, none are being developed.

Part of the new approach is to attach XML tags to data elements (really database table columns) within the context of existing automated information systems, and to then post these information system based XML tags to a metadata registry. That, however, is a far worse alternative than to standardize prior to tagging and posting (see: <http://www.tdan.com/i021hy02.htm> and also <http://www.tdan.com/i030hy01.htm>). That is because the just tag and post approach it institutionalizes all the mismatches in names, semantics, granularity, value domains and precision across all the columns of tables of the databases throughout the enterprise. Every XML schema is the schematic of an information exchange. If an application system has 100 windows and each window has 2 browses, then there needs to be 200 XML schemas for just that one information system. If there are 150,000 information systems between just the two largest enterprise business units, then there needs to be 30,000,000 XML schemas to serve their needs. In short, each XML schema then becomes an expression of a data standard. This too, like DES, is an impossible to create, monitor, and maintain approach.

Alternatively, if there are 100,000 databases, and each of the database's schema is standardized then there are 100 times fewer components to standardize. Further, if every column in a database is

really a reuse of an ISO 11179 data element, and if industry wisdom holds true, that there are fewer than 10,000 ISO 11179 data elements in all of the enterprise, then through smart, well engineered data standardization, manifest through sophisticated metadata repositories, then all columns in every database can have their names, definitions, value domains and data types automatically generated. Flowing from that would then be automatic standardization of all XML tags and, through metadata integration with information systems, the automatic generation of XML schemas.

If there is to be understanding-based interoperability, then there must be agreement among the members of the community of interest as to the shared information's semantics, value domains and meanings, granularity, precision, and any common business rules that transform data. That consensus is called data standardization.

Consequently, not only is there is no rational alternative to data standardization, the requirement for data standardization cannot be wished away or ignored. The more that it is wished away and ignored, the greater and more urgent it is to address and solve.

It is a fundamental requirement for shared information environments whether they are manual, message based, through databases, or through XML exchanges. Further, the quantity of information systems, and the quantity of types, formats and meanings of data are proliferating almost without control.

Every desktop with MS/Access is a source of a database. So too is every Excel generated spread-sheet. To exclude these and all other information systems and database development environments from the IT environment is not possible because such an exclusion could never be enforced. And if it could be enforced, then individual creativity and initiative would be unacceptably restricted. So, rather than a policy of exclusion, an environment of smart, well engineered data standardization must be established and co-exist with end-user development environments such that, together, they enhance and facilitate understanding-based data creation and use environments.

4.0 Lessons Learned

To be successful in understanding-based data interoperability data standardization plays an essential part. Studying the DES efforts provide valuable lessons learned, which are that any data standardization effort, no matter how small must:

- Properly focus on the data concepts such that the lowest level column is easily and automatically named, defined, and abbreviated.
- Accommodate enterprise wide data architectures such that data contexts are immediately apparent without compromising data identification, selection, and melding.
- Accommodate multiple implementation technologies such that regardless of the rules, a data concept are immediately obvious and relatable to all other data that espouse the same concept.
- Allow front-line project staff to create and maintain their own names under the guidance, facilitation, and work enhancing tools and techniques provided by a data standardization organization.

As to the first success factor, focusing on data concepts versus the lowest level column, the ISO 11179 has a formal paradigm for data elements. This standard is not only critical to the execution to smart, well engineered data standardization, it was also created well after the failure of the DES became apparent, and, based on its construction, it is clear that this ISO 11179 standard

incorporated many of the lessons learned.”

Key components of this standard are concepts/objects, conceptual value domains, data element concepts, and value domains. Because of the elegant engineering of this standard, most if not all of the DES engineering errors vanish. In ISO 11179, a data element concept is built from concepts and conceptual value domains. From the data element concepts, various data elements that may differ only in the precise value domain or in some other minor difference, are able to be created. Thereafter, the ISO 11179 data elements, which are essentially, independent business fact semantic templates, are able to be employed within different contexts, that is, as attributes of entities, columns of tables, or DBMS columns of DBMS tables.

After the semantics of an ISO 11179 data element has been infused into an entity's attribute or a table's column, each such specialization can have its own more localized name, and even more localized modifiers and value domains. While these are indeed different attributes, columns, or DBMS columns, they remain derived from their ancestor ISO 11179 data element. Thus, the essential semantic core of each attribute or column is the ISO 11179 data element.

Because of this strategy, there can be a real finite quantity of data elements, and the process of completing data element standardization can actually be accomplished in a practical quantity of time and resources. This standardization and closure process does not have to start from scratch. That is because the existing DES database of DES data elements can be mined to discover the ISO 11179 data elements. Once done, it is likely that about 90% of the ISO 11179 data elements will then have been identified and defined.

As to the second success factor, the ability to extend data standardization databases such as data warehouses, this approach is quite easy if there is a layered metadata management approach. Such an approach would have the ISO 11179 data element layer as the top layer, which would then provide standardization to a data model template layer. These two layers would then enable the creation of new databases from all these pre-standardized parts. It's simply a matter of tagging the data model templates and bringing them under the domain of a database schema. Creating relationships among these preassembled parts is equally simple. The key feature of this multi-layered is that as new data models are created they are automatically mapped to their generator data model templates, which, in turn are already mapped to their ISO 11179 data elements. In short, manufacturing new databases under different architectures is equivalent to “running a production line.”

As to the third success factor, accommodating multiple technologies, this is accomplished as a consequence of being able to generate physical data models required for different DBMSs from the logical data models in the same manner as described above. Mappings are automatic.

As to the fourth success factor, allowing for the front-line project staff to create and maintain their own names without dealing with an onerous, time consuming data standardization authority, this multi-layered approach is specifically designed to have a full set of semantics within meta attributes for every column. Because of this, and because of mapping, there is no requirement that names be severely managed.

This multi-layered approach is also especially designed to handle legacy and database maintenance environments wherein reverse engineering and database modifications are the standard operating procedures rather than new development. The multi-layered approach supports automatic naming, automatic definitions, value domain management, and the like, which is ideal for top-down new developments, the bottom-up approach is fully supported.

Metadata repositories that exist in federations can support metadata export and import and then

harmonization within the upper levels of pre-defined sets of 11179 data elements and data model templates. It is especially because of these upper layers that different legacy environments can be related one to the other.

Again, for a modern data management environment to be successful, that is, to have the definitive set of the right data at the right time to the right person with an unambiguous and universally accepted meaning, data standardization and value set cannot be an option. It is the essential first ingredient for data sharing environments that can then be easily adapted to support Net-Centric environment.

The key test for success is that it must be easier to do data standardization "right" than it is to do it "wrong." Unless and until that is achieved data standardization will be accomplished only after all documentation is complete. And that essentially means: never.

*Michael M. Gorman, President of Whitemarsh Information Systems Corporation, has been involved in database and DBMS for almost **35** years. Mr. Gorman has been the Secretary of the ANSI Database Languages Committee, X3H2 for **25** years. X3H2 standardizes SQL. A full list of Whitemarsh's clients and products can be found on the web site, www.wiscorp.com. The goal of the web site, WisWeb, is to make data management books, courses, methodologies, software, and metrics available to the database community through electronic publishing and downloading. WisWeb memberships are very reasonable and are designed for the individual, the ISD organization, universities/colleges, and professional training organizations.*

[\[Return to the Article Archive\]](#)

The Data Administration Newsletter (TDAN.com)

Robert S. Seiner - Publisher - rseiner@tdan.com

© Copyright 1997-2005 - The Data Administration Newsletter (TDAN.com) - [Disclaimer](#)