FERRIS STATE UNIVERSITY

Database Standards

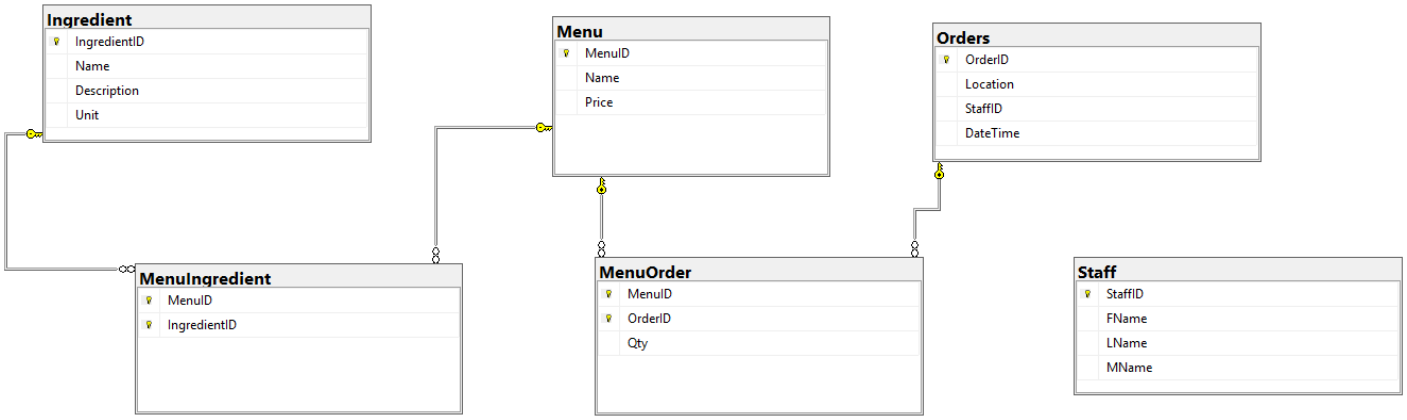
By. Jack Pfleghaar

Contents

Introduction

When working with databases, it is extremely important to follow set standards with the data and diagrams. Without set standards you may run into moments where you are struggling to work with a database due to naming, design, datatype standards, or more. We follow these standards with the concept of having as little problems as possible. In this document, I will be presenting an example database, breaking it down, and then explaining specific standards to follow within an organization.

Database Diagram



Data Dictionary

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
Menu					
	MenuID	Int	K	No	1
	Name	nvarchar(50)	I	No	Steak
	Price	Decimal(6, 2)	I	No	10.99

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
Ingredient					
	IngredientID	Int	K	No	1
	Name	nvarchar(50)	I	No	Cheese
	Description	Nvarchar(MAX)	I	Yes	American Cheese
	Unit	Nchar(10)	i	Yes	

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
MenuIngredient					
	MenuID	Int	K	No	1
	IngredientID	int	K	No	1

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
Orders					
	OrderID	Int	K	No	1
	Location	nvarchar(50)	I	No	Table 10
	StaffID	int	I	No	1
	DateTime	datetime	i	No	2025-11-03 18:42:15.377

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
MenuOrder					
	MenuID	Int	K	No	1
	OrderID	int	K	No	1
	Qty	Int	I	No	5

Table	Field	Datatype	Key/Index	Nullable?	Sample Data
Staff					
	StaffID	Int	K	No	1
	FName	nvarchar(50)	I	No	John
	LName	nvarchar(50)	I	No	Smith
	MName	nvarchar(50)	I	Yes	

VENDOR STANDARDS

Database vendor standards dictate what tools, products and providers are acceptable to use in an organization. It is important to not get sucked into the seduction of using the hot “tool of the month” that trade magazines tout as the greatest thing since canned beer. A manager needs to have an environment that is manageable, and that means keeping things as simple as possible. Every time you add a new tool or vendor into the mix, you often end up having to hire someone to become an expert in it. Right now, justifying new positions is not very easy to do.

A vendor standard might be something like all packaged software has to run on Oracle. If it doesn't run on Oracle, it can't be purchased. Another standard might be that all support tools that don't come from the database vendor have to come from a particular third party provider such as BMC.

Another standards, less common but frequently discussed, is that if a new vendor or tool is brought in, another tool or vendor has to be dropped. In the age of not being able to hire additional people, this may be more than just discussed in management meetings.

Vendor standards might also include consultants, training, or similar areas.

DESIGN STANDARDS

Design standards might include referential integrity techniques, index considerations and guidelines as to what goes into a database. Referential integrity is a mechanism that makes sure that the proper data exists in a database. For example, referential integrity would ensure that a customer exists before they can place an order. Referential integrity can be implemented by using something called a trigger or a constraint. Triggers are kind of like stored procedures. They check to make sure data is in tables – or, they may make sure all the data is cleaned up when a record is deleted. When only a portion of the data exists in a database, it is called an orphan record. Orphan records are the sign of a poorly designed database.

A constraint usually enforces referential integrity by key structures. Remember primary and foreign keys? (the diagram below should refresh your memory). Keys and indexes are two different objects in a database, but they often use the same fields. In the table below, CustomerNo is both a key and an index. LastName might be indexed, but it is not a key.

NAMING STANDARDS

Naming standards entail spelling, abbreviations, use of underscores, and capitalization. What I mean by spelling is are full words used or are they abbreviated. If they are abbreviated, how are they abbreviated? Some organizations put prefixes or suffixes on the database, table, field, or object names.

When you are creating naming standards (if they don't already exist), keep in mind that you don't want to have names that are exceedingly long. Remember, people have to type the name in the SQL statements and they won't want to type 30 characters for each field name in a table.

I'll explain what we did at Amway, then I'll tell you what I would do if I had the opportunity to do it over.First of all, no field name could exceed 18 characters because DB2 on the AS400 (now iSeries) would not allow longer names. We used UPPER CASE for everything and all names were separated with an underscore. We also had to prefix the table name to each field. Let me give you an example before I go further, because the naming standards were a little confusing due to some embedded meaning in names.

STUDENT_FIRST_NAME

Note that this is 19 characters, which violated our 18 character limit. Fields like name and code, which are commonly found at the end of a field name, were abbreviated. We called this type of field a class. Now we have:

STUDENT_FIRST_NM

To name databases, we used to letters that represented that system. For example, Marketing databases started with an MK. Finance databases started with FI. The third letter indicated what type of object it was. A 'B' meant database, a 'T' meant it was a table, and a 'G' meant it was a stored procedure. So, the first Marketing database was called MKB001. The second Marketing database was MK B002.A table in database MKB001 would be MKT001_CUSTOMER.The nice thing about this scheme is that if you see an object name, you know what it is. The disadvantage to this scheme is that if you didn't know the codes, it pretty much looked like a foreign language. You may also have wondered why we used a 'G' for a stored procedure. Well, it was because 'P' was already used for program and 'S' was used for screen. There weren't that many letters to choose from, none of which really fit, so since it was up to me to pick a letter and my name has 4 G's in the first seven letters, I decided it should be G. Impressive logic, right?

Getting back to abbreviations – I ordered a three-volume set of books that had just about any abbreviation you could think of in it. When a new word needed an abbreviation, we looked it up in the books. If it wasn't in the abbreviation books or the abbreviation violated our standards, we would come up with one. To make the abbreviation, we'd start deleting vowels, make sure the word was in the proper tense of form, and then publish it in the data dictionary. Proper tense or form means that we would standardize on one. For example: run, ran, running. We would always use run.

A data dictionary is used to hold information about all of the objects in your databases, as well as the database standards and abbreviations. A data dictionary for a table might look like this:

Table	Field	Datatype	Key/Index	Description
MKT_CUSTOMER				
	CUST_NO	Int	K/I	Customer number…
	CUST_FIRST_NM	Char(20)	I	Customer first name

	CUST_LAST_NM	Char(20)	I	Customer last name

This is very basic, but it gives you an idea about what a data dictionary might look like. If I were to redo the Amway standards, I'd change from UPPER case to lower case. I'd also get rid of underscores and capitalize the first letter of each word. Other than that, things were quite workable. So, CUST_NO would become CustNo.

DATATYPE STANDARDS

Datatypes determine what kind of information can be stored in a field, as well as the format. Each DBMS supports a variety of datatypes, but most DBMS's have some of the datatypes that are unique to it. Numeric data should be stored in a numeric data type – especially if mathematical operations may be executed on it. Common numeric datatypes include integer, money, decimal and float. Alphanumeric (character) data should be stored in datatypes that support it. In fact, you can't store alphanumeric data in numeric datatypes. An example of alphanumeric data would be a street address. Common character datatypes include character, variablecharacter - i.e., varchar, varchar(2) – text and memo. Dates should be stored in date datatypes.

The datatype standards that your organization might have would be things like standardizing on a particular datatype for a common field. For example, a person's first name is always char(20). Other standards might be that fixed datatypes or integers are preferred for indexes.

There are some datatypes that can be dangerous. A BLOB (binary large object) is something like a Microsoft Word document or a jpg. These fields can consume a lot of memory and can actually cause great difficulty for mainframes. Therefore a standard might be that BLOB or image datatypes can't be used unless certain standards are met.

Datatype standards are pretty basic, but as you can see, carelessness can have significant consequences.

PLATFORM STANDARDS

Platform standards are easy to understand, but their implications can be very far reaching. When I talk about database platforms, I mean the environment that they run on. Remember from class that we said SQL Server runs on Microsoft server operating systems usually on an Intel machine. Oracle runs there as well, but most of Oracle's robust installations are on the Unix/Linux platform. Similarly, DB2 is usually on a mainframe, iSeries or maybe a Unix box.

It is important that organizations don't extend themselves too much. Even multi-billion dollar corporations have finite resources and they can't support everything. Building interfaces between different databases on different platforms is one of the most difficult and complex tasks that IT departments face. Remember: Keep it simple. Or, as you may have heard somewhere, follow the KISS principle: Keep It Simple, Stupid!

Many organizations try to pick a preferred platform for their important systems. This is the platform where they have developed a high degree of expertise. If you have everyone trained in Oracle and Java, you don't bring in a new system written in Visual Basic with SQL Server without introducing a high degree of risk exposure. Keep it simple and leverage your strengths – even if your strengths are not the "tool of the month." One qualification

to this last statement is to make sure you don't go down with the ship. Keeping things with a dying technology and a vendor that is on shaky ground is an indication of poor planning.

SECURITY STANDARDS

Database security standards are extensive. Security is the topic of Week 5, so I'll only introduce some of those concepts. Remember in class that we created stored procedures. Stored procedures run faster than embedded sql, are stored on the database server to assist in impact analysis (for database changes), and are also a good security tool. Applications can be given access to stored procedures but not database tables. This limits security exposure.

Other standards might include who gets what level of authority, differences between a test and production environment, auditing, logging, and a variety of other things. Unfortunately, most security standards are reactive. By that I mean that they are put in place after a problem or crisis has occurred.

Database security standards vary to some degree based on the platform and vendor involved. You may recall that when you installed SQL Server that you could use Windows security or Mixed Mode. Windows security means that if you have a windows administrator login that you will have a SQL Server administrative login. Mixed mode means that you have to be given system administrator authority in SQL Server to be a system administrator. Think about the implications of this seemingly simple decision point. If you use Window security, a DBA can log into multiple SQL Servers without having to go into each one and establishing logins. Sounds good so far. However, all the LAN Administrators will also be able to log into these same database servers and access the data because they are also Windows administrators. Huge potential problem: a student intern in the LAN group can pull up the corporation's most sensitive data...I think you see my point.

SOFTWARE STANDARDS

There are a variety of software standards that have implications into the database environment. I've already mentioned my bias that databases are the most important part of IS (besides people). It doesn't matter how fast your networks are, if they don't have good data, there isn't much reason for the networks. Similarly, the important function of computer programs is to interface with data. Programs that don't access databases efficiently have limited functionality.

Some software standards might include naming conventions, but they may also cover what applications can and can't access – and how they perform the access. Software naming standards are often similar to database naming standards. If you call something StudentID in a database, you should probably call it something similar in the program.

Programs have variables that hold data similar to variables in a mathematical equation. In the equation 3 * y = 12, y is considered the variable. The goal is to solve for y to get the answer (divide 12 by 3 to get 4, which is the value of y). Similarly, a programmer defines fields to hold data for use in an algorithm or simply to move to a web page or report. Rather than calling the variable y, calling it studentid would be more descriptive for someone reading the program. Depending on the computer programming language used, there are different types of variables and they are handled differently. My preference was to put some sort of prefix or suffix on the variable so that I knew what kind of variable it was – and so that I didn't get it mixed up with the database field. So I might read StudentID from the Student table and move it to iStudentID in my program. The reason I

call it iStudentID is so that I know it contains integer data. In a different programming language, I might read StudentID from Student and move it to lStudentID (the prefix is a lower case L). The 'l' means that it is a local variable that is only used in the program I am in. In object oriented programming, local variables mean the variable can only be used in the current program, while a global variable can be passed between programs.

My purpose is not to begin a debate on proper variable naming in software programs. My purpose is to bring an awareness that these standards should exist, and they should be reasonable. One other point is that software standards should provide guidance with regard to what is proper and acceptable use within an organization, but not be so restrictive that the become a cumbersome burden. And oh yes, standards should be enforced!

To the earlier point about standards guiding how data can and can't be accessed...remember OLAP and OLTP? OLAP is online analytical processing, which involves decision support and generally inquiry only processing. OLTP is online transaction processing, which is the usual add/change/delete processing most people associate with computer systems. Data access standards might be that certain databases can only be accessed at certain times or via a particular method. Some companies require that all data access routines must be written by DBA staff. Again, my purpose is to make you aware of these concepts – not to give a list of data access commandments.

Summary

This policy has been created to ensure the robust standards of any SQL database in an organization. This document walked through an example database and then provided explanations into standards to be followed. By using these standards we can ensure the security and robustness of our SQL databases.