



## ► Data Management: Theory and Practice

By Richard J. Cichelli, SCS President

You can divide data management tools into file and database management tools. File management solutions have their data structures internally specified within the programs that are used to maintain them. Their data structures can be arbitrarily complex.

Database tools externalize the definition of the data structures. They sacrifice the generality allowed by file-based technology for the ease of management that a more structured approach offers. All database management systems have a database definition language (DDL), a database manipulation language (DML) and a database access language (DAL). They also have a database engine to process DDL statements to set up databases, process DML statements to change the data in a database and process DAL statements to query and report from databases.

Some database tools support E.F. Codd's relational model of databases. These are called relational database management systems (RDBMS). Codd's model envisions data being stored in two-dimensional tables consisting of rows and columns. Each row of a table is an instance of the type of entity (a noun) the table contains. All columns have the same type of data. (Despite what you might have heard about "Flat Files vs. RDBMS," nothing is flatter than a relational database's two-dimensional tables.)

Codd had other rules for RDBMSs concerning record pointers (not allowed), atomic values and uniqueness (required), keys, etc. His rules are called "normalization" rules. Any DBMS that follows Codd's rules is, by definition, a relational database management system.

So all relational databases are flat, some databases (like those used in text archiving systems) aren't relational and picking good systems requires going beyond buzz-word compliance.

### Software Consulting Services, LLC

630 Selvaggio Drive, Suite 420  
Nazareth, PA 18064

Sales: 1-800-568-8006

Fax: 610-746-7900

E-mail: [sales@newspapersystems.com](mailto:sales@newspapersystems.com)

[www.newspapersystems.com](http://www.newspapersystems.com)

Codd's published work on RDBMSs dates back to 1969. (Edgar F. (Ted) Codd, A Relational Model of Data for Large Shared Data Banks CACM 13 (6) 1970.) He experimented with IBM's System/R, a testbed database that had a unified DDL, DML and DAL called Structured Query Language (SQL). Most of Codd's model can be expressed in SQL. Codd's research showed mathematically that a RDBMS could be used to do the same data processing tasks that any application using the file management approach could.

Relational databases are a good thing. Their principle advantage is the relational model. Codd had defined data storage and usage in a platform-independent way. In the sixties, if you wrote a data processing program on an IBM mainframe, you actually had to be concerned about the number of heads and the size of the sectors and cylinders on the disk drive. Moving an application to a drive with different characteristics often required changing applications programs. This was a terrible nuisance.

There were a few issues with the original SQL, like concurrency control and privileges, but these were eventually resolved. The one big and still remaining problem is that SQL doesn't handle the recursive data structures. These are needed in manufacturing systems (i.e., assemblies made of assemblies made of parts, etc.). To this day, you see Manufacturing Resource Planning Systems (MRPS) augmenting their RDBMSs with a Bill of Materials Processing system (BOMP). While Codd's original model supported recursion and thus could support BOMP, SQL didn't.

To give Codd proper credit, he noted the need for recursion and its lack in SQL. BOMP isn't the only context where SQL and lack of recursion hinders clear algorithm expression.

XML schemas inherently support nesting and are best processed as tree structures, something very foreign to SQL.

Codd's enduring contribution was that his relational model was mathematically based in set theory. His concepts defined database operations in terms



of accessing and modifying one or more rows (records) at once. (Traditional data processing had a one-record-at-a-time processing model.) Set operators of RDBMS are bigger verbs; they do more with a terser notation than prior data processing notations.

When picking systems, you have three database options. You could buy a proprietary database system like MS SQL, Oracle or Sybase. Each of these requires using a proprietary development language as the DAL and DML to go along with SQL. For MS SQL it is Visual Basic, for Oracle it's PL/SQL and for Sybase it's E-SQL. Each of these proprietary notations is completely incompatible with the others. An application built with these tools cannot move to another RDBMS without a rewrite. Some are completely platform-dependent, such as Microsoft's Visual Basic, which can only run on Microsoft's proprietary Windows operating systems.

Another alternative is an open-source database like MySQL or PostgreSQL.

There is a lot to be said in favor of Open Source Software (OSS) and Open Source Relational Database Management Systems (OSRDBMS). Reliability, high performance and commodity, rock-bottom pricing lead the list.

There is a level of compatibility that all of these share: it is their connectivity in a client-server architecture. This is called open database connectivity (ODBC). ODBC calls are a lower level notation than SQL statements. They are more explicit about specifying data management tasks than SQL but do the same processes. ODBC-compliant tools can modify the data in ODBC-compliant databases outside the application code.

All of these solutions presume you have a database administrator (DBA) to manage your database management system. In contrast to this approach are applications that use embedded database management systems (EDBMS), such as those built using development tools from Progress or Faircom. Packaged accounting applications and MRPS are often implemented with embedded database technology. The suppliers of these systems control how storage resources are utilized within their applications. They supply all the tools necessary to extract data from their systems. Usually they want to ensure that data is highly secure. They want using their applications to be the only feasible way data can be changed. Such applications usually include audit trails of all transactions.

SCS has chosen the embedded database approach for its advertising systems. We use Faircom's ([www.faircom.com](http://www.faircom.com)) c-tree on which to build a relational database management system. (Just like Oracle, MS-SQL, etc., c-tree uses the b-tree algorithm for record management.) We call our development tools "Spice." Faircom's c-tree is a part of Spice. We have a fully-paid, royalty-free license to Faircom's source code. SCS has ported all of Spice, including its c-tree component, to more than a dozen platforms.

Spice is an ODBC-compliant relational database management system engineered to support building publishing applications. It implements Codd's model, including recursion, so it even handles BOMP and XML easily. We have found it to be extremely reliable, fast, functional, portable and cost-effective.