DBA SHOPTALK

Repositories, data dictionaries, and encyclopedias for SQL/DS

A few months ago, Thomas Ross, a subscriber of Database Programming & Design, *suggested we run an article on the topic of data dictionaries and repositories for IBM's SQL/DS. We responded enthusiastically, enlisting the expertise of R. G. Eaton, a developer in Kansas City, Missouri. If you have an issue or problem that you would like to see addressed in the pages of this magazine, feel free to follow Mr. Ross's example and give us a call!*

BY R. G. EATON

Seeing through the Smoke

JUSTIFIED CONFUSION and wonderment exists about the support of repositories, data dictionaries, and encyclopedias for the SQL/DS platform. In my role as an SQL/DS DBA, I was obliged to design and develop my own data dictionary. During the process, I developed an understanding of the various genre of development support databases. At their highest level of definition, repositories, encyclopedias, and data dictionaries are databases that support applications development. I may get some argument about this definition from business data modeling people, but hey, most of us are paid to deliver business systems. A business model is only valuable (to most of us, anyway) if it leads to a business solution. Thus, I refer to repositories, encyclopedias, and data dictionaries as applications-development support databases. For our purposes, let's agree that designing a business model is part of the applications-development process.

I classify these specialized databases differently. While the data dictionary is typically confined to storing specifications about a system's information needs, the repository includes specifications about the entire system and its environment. The repository includes information about the system's software release levels, source code, and operational instructions. It may also be used to integrate development tools. Encyclopedias fall in the middle. Most are used to support software development and exclude environment specifications.

When I think of the Repository/MVS product from IBM, I imagine an aeronautical engineer designing a jet to save his doomed company. The engineer creates the ultimate passenger jet—it's virtually a cruise ship with wings. As the engineer and most of the company's management proclaim that the customers will line up to order the new jet, a seasoned engineer offers one comment: "Will it fly?"

If you've followed the development of Repository/MVS, you can see the parallels. While a good idea, the product is having problems getting off the ground. Although it will support SQL/DS, no repository support exists for native SQL/DS in either VSE or VM flavors. However, many expect IBM to redirect its repository effort toward Unix and OS/2 platforms.

Another problem with Repository/MVS is accessing the information buried within the repository. The repository is a database. To access information within this database you must obtain an access tool such as those supplied by Arlington, Virginia-based Reltech Group Inc. and New York City-based Brownstone Solutions. Both vendors offer repositories and complimentary tools, and their tool support includes DB2 administration, applications-development impact analysis, and CASE interface.

According to Tom DePasquale, president of Reltech, "The role of the repository in the '90s will be to support a spectrum of DBMSs and CASE tools. With the advent of PC-based front ends for repositories, the requirement for the repository to utilize multiple DBMSs for data store is lessened, while the need to support multiple DBMSs (SQL/DS, Teradata, IMS, Sybase, and so on) is increased. Our direction will be to support SQL/DS from both a DB2 and client/server (open systems) perspective." I believe Tom is saying that it's more likely for repository support for SQL/DS to be LAN- or DB2-based than actually SQL/DS-based.

Encyclopedias are to programs as data dictionaries are to SQL/DS tables. That is, a data definition language (DDL) generator will use a data dictionary to provide specifications for code generation, while an encyclopedia stores the relevant information required by a program generator (such as a COBOL code generator) for code generation. Encyclopedias are necessary components of CASE tools. Some tools don't use encyclopedias—but that's another story. I like to think of an encyclopedia as a place to inventory application system components, or as a system inventory.

Mainframe-based central encyclopedias for CASE tools, such as Texas Instrument's Information Engineering Facility (IEF), offer support for SQL/DS installations. While the workbench component will generate program and DDL code for an SQL/DS target platform, the encyclopedia requires MVS and DB2. According to Kevin Green at Texas Instruments, "A

trend is developing toward server-based encyclopedias including an SQL/DS target environment." He adds, "A Unix-based approach may resolve some of the performance issues associated with other lower-based operating systems."

Green's scenario highlights the flexibility of CASE encyclopedia technology. When comparing encyclopedias, you should consider support for facilitating check-in and check-out of specific encyclopedia objects. If one developer is altering an SQL/DS table definition, it should be protected from other developers changing the same table.

WITH THIS DESCRIPTION of encyclopedias, I'd like to return to the repository description. The repository should be able to accommodate multiple encyclopedias. An SQL/DS table defined by one CASE tool is stored in its encyclopedia. When this encyclopedia is within the repository, the SQL/DS table definition should be accessible to a different CASE tool. According to Barry Brown, cofounder and managing director of Brownstone, "Our vision of a repository includes providing bridges between different CASE tool encyclopedias." He continues, "An application object (such as an entity type) created by one CASE tool should be available to other CASE tools. And the original object should be locked to ensure that competing CASE tool developers don't change the object."

CASE tools providing encyclopedia support include Analyzer from Cambridge, Massachusetts-based Bachman Information Systems, Application Development Workbench (ADW) from Atlanta-based KnowledgeWare, and IEF from Plano, Texas-based Texas Instruments. However, none of these products provide native SQL/DS central encyclopedia support.

The basic support for applications development is the data dictionary. Therefore, all encyclopedia and repository products provide data dictionary support of one form or another. The data dictionary is a good starting point for developers with minimal exposure to CASE, encyclopedias, or repositories since most encyclopedias can be populated from a data dictionary.

In a sense, SQL/DS provides a data dictionary with its systems catalogs. Using the system catalog as a data dictionary can expose the installation to some serious resource contention problems, however. In addition, the objects (tables, indexes, and DBspaces, for example) must exist before you can receive any value from the catalogs. This type of data dictionary environment is "reactive."

Products loosely fitting in the reactive data dictionary classification include: SQL/Master from IBM, SQL/DS Workbench from Houston-based CDB Software Inc., and DB/Reorganizer from Vienna, Virginia-based Relay Technologies (formerly VM Systems Group). I say "loosely" because these products actually use the system catalogs as a data dictionary. Therefore, the database objects must exist before these tools can be effective.

Proactive dictionaries let the developer or DBA enter information about the database objects into tables separate from the systems catalogs. Developer queries and DDL generation can be performed from the dictionary tables instead of the system catalogs, thereby reducing contention. Furthermore, proactive dictionaries enable the DBA to perform capacity planning and impact analysis before applications are deployed.

If you're using the reactive approach to database management, converting your operation to a proactive one can be fairly painless. The data dictionary can be initially populated from the existing system catalogs. Thereafter, all database objects must be created from DDL generated by tools that use the data dictionary. That is, characteristics about database objects are entered into the data dictionary, and tools use this information to generate the DDL.

Even though it's the better approach, I know of only one proactive data dictionary for SQL/DS. Silent DBA from Naples, Florida-based Allen Systems Group provides DDL generators that are supported by a data dictionary. Ira David, vice president of development for Allen Systems, revealed his company's view of data dictionary support for SQL/DS: "The ability to define objects within a self-contained dictionary, independent of both the application and the system catalog, has been recognized in the DB2 world for some time. In this mode, the information flow starts with data needs (as defined to the dictionary) and progresses to the structural needs (the object in the catalog via generated DDL). This approach allows for a more interactive and efficient definition process, the ability to analyze and report on objects, and the potential to study overall efficiencies directly from the data dictionary without the overhead of catalog object definition and alteration. Dictionary objects can also be defined while the original objects are in production, without any integrity problems or conflicts."

David continued by saying, "For a number of reasons, this functionality has not been heavily in demand within the SQL/DS arena. SQL/DS hasn't had the proliferation of products that we have seen in DB2. The products that have been released have generally been more basic tool kits. The Silent DBA addresses this need."

Deciding on the correct application support database depends upon your requirements. For many SQL/DS installations, the repository is quickly eliminated because of the MVS and DB2 requirement. Encyclopedias may be the best selection for CASE users since CASE tools usually offer their own encyclopedias. Not all CASE encyclopedias are interchangeable, so you will likely purchase the encyclopedia along with a CASE tool. If you aren't ready for CASE, a data dictionary can be a wise step in the transition. These products are good ways to organize your business's applications development. For the SQL/DS installation, it seems that the best available support is in the data dictionary class of support databases.

Again, these classifications are what I use to compare apples to apples. One vendor may have a "repository" while another has an "encyclopedia." Use these classifications as lenses to help you see through the smoke.

R. G. Eaton is a development team leader at Twentieth Century Services Inc. in Kansas City, Missouri.