October 1998

DB2 update

Preparing for DB2 Version 6

By Craig S. Mullins

IBM recently announced DB2 Version 6 at the International DB2 User Group in May of 1998. As part of this announcement IBM is renaming the product, to keep consistency across its DB2 product line, to DB2 Universal Database for OS/390. IBM plans a June 1999 general availability date for DB2 V6.

This new release includes many new and exciting features including large objects, triggers, user-defined functions, user-defined data types, and more. However, the item that is most important for every DB2 shop to understand is what is *not* in DB2 V6. For the first time IBM is removing features from DB2 and you will need to prepare your DB2 subsystems for V6 by removing the soon-to-be-unsupported features from your installation.

Let's examine each of the features that will be removed as of DB2

V6.

Type 1 Indexes
There are two types of indexes available to DB2: type 1 and type 2. Type 2 indexes were introduced with DB2 Version 4 and should be the standard index type implemented in your shop. Most organizations already favor creating type 2 indexes over type 1 indexes because they provide the following benefits:

- Eliminates index locking (the predominant cause of contention in most pre-V4 DB2 applications)
- Type 2 indexes do not use index subpages
- Type 2 indexes are the only type supported for ASCII encoded tables
- Many newer DB2 features can not be used unless Type 2 indexes are used; these features include row level locking, data sharing, full partition independence, uncommitted reads, UNIQUE WHERE NOT NULL, and CPU and Sysplex parallelism

As of DB2 V6 type 1 indexes will no longer be supported by DB2. All of your shops indexes must be type 2 before migrating to DB2 V6. It is wise to begin this migration as soon as possible because of the benefits outlined above. If you are on DB2 V3 or an earlier release you cannot implement type 2 indexes because they are not supported. In that case you should move to DB2 V4 or a later release as soon as possible to begin migrating your indexes to type 2 in preparation for DB2 V6.

To find all type 1 indexes issues the following SQL statement:

```
SELECT CREATOR, NAME
FROM SYSIBM.SYSINDEXES
WHERE INDEXTYPE = ' ';
```

Shared Read Only Data
Shared read only data (SROD) was provided as a new feature of DB2 in Version 2.3. SROD provided a way for the same DB2 database to be read by multiple DB2 subsystems without implementing distributed data or Sysplex data sharing. However, the shared object must be started ACCESS(RO) and all data access is read only. When the data needs to be updated only one of the subsystems, the one marked as the owner, can update the data.

SROD is complex to implement, limited in functionality, and not frequently implemented. As of DB2 V6, SROD support is removed. In order to support SROD-like functionality you will need to convert to data distribution or data sharing.

To find all SROD databases issue the following SQL statement:

```
SELECT NAME, BPOOL, ROSHARE
FROM SYSIBM.SYSDATABASE
WHERE ROSHARE IN ('O', 'R');
```

RECOVER INDEX
Through DB2 V5, the RECOVER INDEX utility is used to recreate indexes from current data. RECOVER INDEX scans the table on which the index is based and regenerates the index based on the actual data. Indexes are always recovered from actual table data, not from image copy and log data.

With DB2 V6, the function of the RECOVER INDEX utility changes. Instead of rebuilding indexes from the current data, RECOVER INDEX will actually recover the index by reading an image copy of the index data set. So, with DB2 V6 you can use the COPY utility to make backups of DB2 indexes and the RECOVER utility to restore them.

To provide equivalent functionality for recreating an index from the current data, IBM provides a new utility called REBUILD INDEX. The REBUILD INDEX utility works exactly like RECOVER INDEX used to function.

Organizations should begin changing all of their current RECOVER INDEX jobs to use REBUILD INDEX syntax instead. The REBUILD INDEX syntax is available in DB2 V5 and V4 (with PTF PQ09842) and will work exactly like RECOVER INDEX. After you migrate to DB2 V6, the RECOVER INDEX utility will cease to function if the proper index backup copies are not available to use during recovery.

Host Variables Without Colons
All DB2 programmers know that host variables used in SQL statements in a program should be preceded by a colon. So, if a host variable is named HV it should be coded in the SQL statement as :HV. However, most programmers do not know that through V5, DB2 programs tolerate host variables that are not preceded by a colon. This "feature" ends as of DB2 V6.

The reasoning from IBM for eliminating this feature is that it is getting too difficult for DB2 to differentiate host variables from SQL. This is due to the rising complexity of SQL and the new features being added to DB2. As such, for DB2 V6 and onward, all host variable must be prefixed with a colon.

This change should not impact many programs because most organizations have DB2 standards that dictate all host variables will begin with a colon. However, because DB2 has tolerated host variables without a colon through DB2 V5, you should inspect all DB2 SQL statements in application programs to ensure compliance prior to migrating to DB2 V6.

Dataset Passwords

A little-used feature of DB2 is the ability to provide security via dataset passwords. Using the DSETPASS key word of the CREATE TABLESPACE and CREATE INDEX statement, it is possible to password protect DB2 datasets.

This feature disappears with DB2 V6. If you need to protect your DB2 datasets outside of DB2 security you can use RACF, ACF2, Top Secret, or whatever security package you have installed at your site to accomplish this.

To find datasets that are password protected using DSETPASS, issue the following SQL statement:

```
SELECT 'INDEX ', CREATOR, NAME
FROM SYSIBM.SYSINDEXES
WHERE DSETPASS <> '        '
UNION ALL
SELECT 'TSPACE', DBNAME, NAME
FROM SYSIBM.SYSTABLESPACE
WHERE DSETPASS <> '        '
```

Stored Procedure Registration
After coding a stored procedure, you must register information about that stored procedures in the DB2 system catalog. This process is in sharp contrast to the manner in which other database objects are recorded in the system catalog. Typically, when an object is created, DB2 automatically stores the metadata description of that object in the appropriate DB2 Catalog tables. For example, to create a new table the CREATE TABLE statement is issued and DB2 automatically records the information in multiple system catalog tables (SYSIBM.SYSTABLES, SYSIBM.SYSCOLUMNS, and possibly SYSIBM.SYSFIELDS). Because stored procedures are not created within DB2, nor are they created using DDL, the database administrator must use SQL INSERT statements to populate the SYSIBM.SYSPROCEDURES system catalog table with the meta-data for the stored procedure.

The following SQL provides an example of an INSERT to register a stored procedure:

```
INSERT INTO SYSIBM.SYSPROCEDURES
  (PROCEDURE, AUTHID, LUNAME, LOADMOD, LINKAGE,
   COLLID, LANGUAGE, ASUTIME, STAYRESIDENT,
   IBMREQD, RUNOPTS, PARMLIST, RESULT_SETS,
   WLM_ENV, PGM_TYPE, EXTERNAL_SECURITY,
   COMMIT_ON_RETURN)
 VALUES
  ('PROCNAME', ' ', ' ', 'LOADNAME', ' ',
   'COLL0001', 'COBOL', 0, 'Y',
   'N', ' ', 'NAME CHAR(20) INOUT', 1,
   ' ', 'M', 'N', 'N');
```

This SQL statement registers a stored procedure written in COBOL and named PROCNAME with a load module named LOADNAME.

It uses a package with a collection ID of COLL0001. Any location can execute this procedure. The program stays resident and uses the DB2 SPAS (not Workload Manager), and no limit is set on the amount of time it can execute before being canceled. Furthermore, the stored procedure uses one input/output parameter, and the parameter cannot be null.

This method of registering stored procedures changes in DB2 V6. Instead of the INSERT statement CREATE and ALTER statements are provided for registering stored procedures to the DB2 system catalog. Additionally, a new catalog table named SYSIBM.SYSROUTINES replaces SYSIBM.SYSPROCEDURES. This new table will store information on triggers, user-defined functions, and stored procedures. The metadata for all of these "routines" will be provided to the system catalog by means of DDL statements.

A number of organizations have implemented processes for creating and updating stored procedures that includes registration. These processes will need to be modified for DB2 V6. Additionally, if your organization uses a third party tool to register or change stored procedure information be sure that it will be changed to support the new DB2 V6 DDL syntax.

Other Concerns — DB2 private protocol distributed data
Distributed data support was added to DB2 as of V2.2. At that point in time, IBM had not yet formulated its DRDA framework. In DB2 V2.2, distributed unit of work (DUW) capability was provided solely through a private protocol which did not support any industry standards. As of DB2 V3, both the private protocol DUW and full DRDA DUW is supported. Private protocol is also referred to as system-directed access and DRDA protocol is also referred to as

application-directed access.

Application-directed data access is the more powerful of the two options. With application-directed access, explicit connections are required. Furthermore, application-directed distributed access conforms to the DRDA standard.

But, DB2 also provides system-directed access to distributed DB2 data. The system-directed access is less flexible than application-directed access because:

- it does not use the open, DRDA protocol, but instead uses a DB2 only, private protocol
- it is viable for DB2-to-DB2 distribution only
- connections can not be explicitly requested, but are implicitly performed when distributed requests are initiated

Although system-directed access does not conform to DRDA, it does provide the same levels of distributed support as application-directed access: remote request, RUW, and DUW. System-directed access is requested using three part table names.

As of DB2 V6, three part names can be used with DRDA. This provides static SQL support for distributed requests using three part names. DB2 private protocol distribution is still available with DB2 V6, but IBM has indicated that it will be removed in a future release. As such, consider migrating away from DB2 private protocol distribution as of V6.

Synopsis
Version 6 is the first release of DB2 to take features out of the product. As such, organizations must understand what is being

removed, know how to provide similar functionality with other DB2 features, and develop a plan to migrate away from the non-supported features. It is not too early to begin planning and migration now. The sooner you remove the old technology, the sooner you can move to the latest and greatest version of DB2 when it is available.

From DB2 Update (Xephon), October 1998.

Home.