Technical Document

Document No:	**11-01001**
Document Title:	**Frequently Asked Questions About SQL Packages**
Category:	**Hints, Tips & FAQ**
Functional Area:	**SQL**
OS/400 Release:	

Document Description:

What are SQL packages?

SQL packages are permanent objects used to store information related to prepared SQL statements. They are used by ODBC support when the extended dynamic box is checked on a data source. They are also used by applications which use the QSQPRCED (SQL Process Extended Dynamic) API interface. The SQL packages created by ODBC and QSQPRCED are called extended dynamic SQL packages and are the subject of the following questions and answers. DRDA also uses SQL package objects but they are considerably different in behaviour and are not covered below.

When do SQL packages get created?

In the case of ODBC (Client Access R440), the existence of the package is checked when the client application first executes any SQL Statement. If the package does not exist, it is created at that time (even though it may not yet contain any SQL statements). In the case of QSQPRCED, creation of the package occurs when the application calls QSQPRCED specifying function '1'.

How are SQL packages named?

SQL packages for ODBC are named by taking the application name specified in the data source configuration and appending three letters which are an encoded set of the package configuration attributes. For QSQPRCED, the name of the package is provided by the application.

What library does the SQL package go into? Is there a preferred library for SQL packages?

For ODBC, this is part of the data source configuration. For QSQPRCED, the library is provided by the application. There is not really a preferred library for SQL packages. There are not any functional or performance differences based on the containing library. This choice should be made strictly on ease of management for the application.

What are the advantages of using SQL packages?

Because SQL packages are a shared resource, the information built when a statement is prepared is available to all the users of the package. This saves processing time, especially in an environment when many users are using the same or similar statements. Because SQL packages are permanent, this information is also saved across job initiation/termination and across IPLs. In fact, SQL packages can be saved and restored on other systems. By comparison, dynamic SQL requires that each user go through the prepare processing for a particular statement and this must be done every time the user starts the application.

SQL packages also allow the system to accumulate statistical information about the SQL statements which result in better decisions about how long to keep cursors open internally and how to best process the data needed for the query. As above, this information is shared across users and retained for future use. In the case of dynamic SQL, this information must be re-learned by every job and every user.

Do all prepared statements go into an SQL package?

Only certain **SQL** statements can be prepared. They are:

ALTER TABLE	CREATE PROCEDURE	LOCK TABLE
CALL	CREATE VIEW	RENAME
COMMENT ON	DELETE	REVOKE
COMMIT	DROP	ROLLBACK
CREATE COLLECTION	GRANT	SELECT
CREATE INDEX	INSERT	SET TRANSACTION
CREATE TABLE	LABEL ON	UPDATE

In the case of ODBC applications, Client Access makes some further restrictions about the statements that can go into a package. An SQL statement goes into the package only if one of the following is true: (This Information is subject to change without prior notice):

- The statement contains parameter markers. (Substitution variables specified when the statement is run.)
- It is an INSERT with subselect, for example INSERT INTO table1 SELECT FROM table2 WHERE ...
- It is a DECLARE PROCEDURE.
- It is a positioned UPDATE or DELETE.

QSQPRCED packages do not have these restrictions and will contain any of the statements listed above when they are prepared using function '2' or '9'.

How can I tell what statements are in an SQL package?

The **`PRTSQLINF`** command can be used to produce a formatted report showing the SQL statement and information about the access plan used to access the data.

Can the same statement appear multiple times in the same SQL package?

Every PREPARE operation checks to see if there is already a prepared statement with all of the same statement text and attributes. If there is, a new statement name entry (about 80 bytes) is allocated but it just points to the corresponding duplicate information already in the package.

How can I tell if the SQL package is being used?

The database monitor can be used to log information about SQL processing on the system. It includes the name of the package in the SQL summary records. The following statement from interactive SQL will show the package, the SQL operation, and the statement text.

```
SELECT qqc103, qqc21, qq1000 from <db monitor file>
```

When using ODBC, you can also check for the warning or error message, **Extended Dynamic has been disabled,** to determine if ODBC was unable to use an SQL package. This message is generated when the first SQL statement is prepared. It will be returned as a warning, an error, or not returned at all depending on the settings in the ODBC data source (R440 Unusable **package** setting). Note that the message is not logged in the job log.

What data is stored in an SQL package?
The SQL package contains all the necessary information to execute the prepared statement. This includes registry of the statement name, the statement text, the internal parse tree for the statement, definitions of all the tables and fields involved in the statement, and the query access plan needed to access the tables at run time.

How big is an SQL package?
Prior to V4R3, SQL packages were limited to 16MB. Furthermore, ODBC packages had additional restrictions on the number of statements allowed. In V4R3 and later, the maximum size of a package was increased to 16,384 statements or approximately 500MB, whichever comes first. Note that the SQL package must be created on V4R3 to be capable of this new larger size -- they are not converted automatically. If you are upgrading to V4R3, extended dynamic SQL packages should be deleted if you wish to take advantage of the larger size.

Are there any performance considerations for the new, bigger packages?
Prior to V4R3, statement name and statement text searches of an SQL package were implemented by a relatively slow linear search. The V4R3 enhancements for bigger packages also include support for hash tables which improves performance of searches by either statement name or searches by statement text.

Is there an optimal size for bigger packages?
Because of the hash tables, there are no significant performance impacts for using relatively large packages.

Are there any plans to support bigger packages in releases prior to V4R3?
No. The code changes were pervasive and cannot be delivered via PTFs.

What happens when SQL packages get full?
In the case of ODBC, Client Access code detects the package full condition (SQL0904, reason code 7) and switches over to dynamic SQL for newly prepared statements. This may have some negative performance consequences as indicated above. Statements that were previously prepared in the SQL package continue to be used.

With OS/400 R430 and later, the ODBC "clear package if package size is" setting is ignored. Packages are never cleared and there is no way to override the internal size limit of the package (unless the data source is set to "Use" rather than "Use/Add").

Prior to R430 the ODBC data source configuration offers to clear the SQL package when it is full. This setting is of limited use. The ODBC setting actually specifies a specific number of statements at which the existing package should be cleared. An estimate is made as to what number of statements is equivalent to a full package. Also, this check is only done at connect time. It is still possible for an application to connect, start using (and adding to) the package, then reach the limit. In this case the SQL0904 message will still be generated when the limit is reached. The package is not cleared until the next time a job connects and uses it.

In the case of QSQPRCED, an SQL code of -904 is returned in the SQLCA data structure and the application must decide how to proceed. V4R3 is much easier to manage due to the larger allowable size.

Are there other times when an SQL package can become unusable?
SQL packages have some attributes that are stored at the package level and must be compatible with the application. For example, ODBC SQL packages allow specification of a default collection for unqualified table names. If the SQL package already exists and its default collection does not match that of the client

application, the package will not be used and the user will use dynamic SQL. The same thing can happen with CCSID support.

When should I delete SQL packages?
Packages must be deleted when the underlying metadata for statements stored in the package has been changed. If a table, view, procedure or other SQL object is altered the information in the package is not updated. The package must be deleted to pick up the new information.

Prior to V4R3, packages may need to be deleted when they have reached their size limits. In V4R3, there is considerably less need to delete SQL packages but it is still reasonable to delete them when you have made substantial changes to the database (which could necessitate access plan rebuilds) and whenever you are advised to do so by appropriate system or application service personnel. Because extended dynamic SQL packages are re-created when the application is run, there is little harm in deleting them.

How do I delete SQL packages?
Before deleting SQL packages, you must first end all applications using the package. R420 and later (via SA83273) introduced package locking to prevent deletion while a package is in use. With older releases, deleting a package while it is in use will cause a high severity error, ending the job. To delete a specific SQL package, you can use the **DLTSQLPKG** command. To locate and delete SQL packages, you can use the **WRKOBJ** command and select option 4 to delete them. The command syntax is as follows:

```
WRKOBJ OBJ(*ALL/*ALL) OBJTYPE(*SQLPKG)
```

Only extended dynamic **SQL** packages are automatically re-created by the application. It is important that you do not delete IBM-supplied SQL packages: packages in QSYS; packages whose names start with Q; DRDA packages. If you are unsure about whether a specific **SQL package** is used by DRDA, use the **PRTSQLINF** command and look at the first page of the report. DRDA packages will contain an RDB keyword entry identifying the relational database for DRDA.

If either system package QSYS/QSQLPKG2 or QSYS/QSQXDPKG are accidentally deleted, they must be restored from a backup from any system at the same release. The QIWS/QZDAPKG package can be re-created by restarting the database server. If a backup of these objects are not available, they may be downloaded from the following ftp server directory:

ftp://testcase.software.ibm.com/as400/fromibm/ApiSamples

This directory has a file, INDEX.TXT which describes the contents of this directory, including these save files.

Why would service personnel recommend deleting SQL packages? Are they corrupted?
The most common reason for deleting packages is a change in the underlying metadata for statements stored in the package. If a table, view, procedure or other SQL object is altered the information in the package is not updated. This can lead to a variety of errors including missing fields, incorrect field or parameter descriptions and so on.

Deletion is also recommended because a package has become unusable (see above) or because of possible conflicts between an older DECLARE PROCEDURE (stored in the package) and a CREATE PROCEDURE definition (stored in the system catalogue). Deleting them is one way of ensuring they are primed with the most current and useful set of prepared statements.

SQL package corruption is actually not a common occurrence and there is rarely a need to delete them, especially with the V4R3 enhancements. Prior to V4R3, fragmentation within the SQL package was often an issue which could be resolved by deletion. If a system is at V4R3 but has SQL packages created in previous releases, they should be deleted so they will be re-created to enable growth to the larger size.

Why can't I delete a package with a name of _CSETUP (or other nonstandard system object names)?

The **`DLTSQLPKG`** command only accepts SQL package names which conform to OS/400 system naming conventions. These conventions exclude special characters in the first position. To delete the package, refer to the instructions in APAR SA59057 or contact your support provider for assistance.

What are the QSQLPKG and QSQXDPKG packages in QSYS for? Can they be deleted?

The QSWLPKG and QSQXDPKG packages in QSYS are system packages used as models to create application packages. Do NOT delete these packages. If they are accidentally deleted, they will need to be restored from backup. There may also be several system SQL packages in library QGPL that begin with a 'Q'. If these are deleted, they will be recreated when the jobs using them are restarted.