There is a movement that has been taking place for quite some time to persuade the IBM i community to use SQL DDL instead of the old standby DDS when creating files. SQL DDL refers to the SQL data *definition* language which is used in conjunction with the SQL data *manipulation* language. Data definition statements include CREATE, ALTER, and DROP. There are some sound reasons for consideration of IBM's suggestion to move towards DDL.

First and foremost, the main reason to move to DDL is the future IBM is planning involves enhancements to DDL, not DDS. Similarly, SQL is universally accepted and the learning curve should be fairly minor. Other reasons to make the move are more substantive. The differences in DDS and DDL with respect to accessing data with logical files, indexes, and views are also at the heart of this discussion.

As you know with DDS a logical is nothing more than a source member with a key list for retrieving; viewing, joining, or updating, records in a particular order. You may already be familiar with the difficulties in creating complex join and combining physical views with logicals. Unlike DDS, DDL allows creating views over other views, the equivalent of logicals over other logicals, if you will.  In addition, DDL will allow creation of a column called the identity column that will keep a unique number that automatically increments by definition with no intervention required. SQL tables have had a unique row-id column for years so it's nice to finally get something similar on the System i. DDL also allows creating views over resultant fields, selecting data based on the WHERE clause and even more complex nested SELECT statements. Often, newly created DDL files can be accessed by existing RPG IV with native I/O requiring no changes to the program.

For a quick refresher here's an example of creating a new table, a view, and an index:

**CREATE TABLE EMP001 AS SELECT \* FROM EMPLOYEE WHERE ACT='1' and LOC = '001'**

Fields can be explicitly defined by the CREATE statement which can also use a field reference file. Fields can be defined as primary keys, variable character fields, null or not null, dates, Long fields, currency, yes/no, embedded objects, or hyperlinks.

A view represents the results of a query:

**CREATE VIEW EMPV001 AS SELECT \* FROM EMPLOYEE WHERE ACT='1' and STATPROV='KY'**

An index is similar to a logical file in that it becomes a separate object once created and allows retrieval based on key fields. The first index example allows duplicate records (same empno) while the second does not.

**CREATE INDEX EMPI001 on EMP001 (empno)**

**CREATE UNIQUE INDEX EMPI002 on EMP001 (empno)**

Performance appears to be another area that favors DDL over DDS, albeit only slightly and on a case-by-case basis. Retrofitting your programs to use DDL will sometimes mean changing native I/O to embedded SQL. Retrofitting can be automated but at some expense. Of course it will be expensive to devote a lot of programmer time to manually retrofit SQL into legacy code.

IBM provides an API, namely QSQGNDDL, to generate an SQL DDL source member from an existing object and also allows generating DDL using the Operations Navigator. From the iSeries Navigator select the appropriate connection, open Databases, open the System i folder, open Schemas, open the appropriate library, find the Table object and right-click and select "generate SQL...". The output can be sent to an SQL editor window or to the System i source file. Note that this will work even for objects without source code. Once the DDL is created in the source library the file is created with the command RUNSQLSTM, instead of CRTPF, CRTLF.

Currently I think it would be prudent creating new files with DDL. However, doing a major rewrite should only be considered after careful consideration of the scope of such a task. There is a free diagnostic reporting tool from one company, [Xcase](http://www.xcaseforsystemi.com/), which also happens to sell a tool for retrofitting DDS to DDL.

Conclusion:

The "worth it or not" discussion rages on but there can be no debate that the direction IBM is taking (for now) is to move toward standardization with SQL DDL when creating files. The "good ole" days of DDS are numbered. Now is the perfect time to brush up on your basic SQL skills but be wary of jumping into a large retrofitting project. Rather, review the methods that IBM provides and determine if the time required is worth the benefit. You can easily get started by creating your new logicals using SQL DDL. DDL and DDS can co-exist.

I am sure there will be those that claim "If it ain't broke...." but if we all thought like that we might still be sitting in caves rubbing rocks and sticks together. Hmmmmm... on second thought, maybe, Nah... that wouldn't be much fun.