Journaling at object creation on DB2 for iSeries, TIPS0604

Abstract

There is often a time gap between the time a file is created and the time some person or product turns on journaling protection for this newly created object. In some environments, that time gap represents a risk: What if an application adds data to this new object before journal protection is enabled?

For that reason, especially in environments that demand high availability, it is desirable to find a mechanism by which you can assure that no such time gap exists. What if you could assure that newly created objects, such as database files, had journaling enabled at creation?

Is that possible?

Yes!

This has always been the default behavior for SQL tables. Now you can have similar behavior for native DB2® files as well as for data areas and data queues.

Contents

Written by Peg Levering, Software Engineer
IBM® Systems & Technology Group
Development iSeries™ Journaling

Automatically starting journaling as objects are added to a library

Many users wish to have all of the objects in their production library journaled to assure proper recovery. Yet they find that this list of objects is not static. They might use software packages that constantly create new objects. These users often struggle trying to detect the arrival of the new objects so that journaling can be started for the objects before any new content (for example, database rows) flows into them. With Version 5 Release 4 Modification 0 (V5R4M0) of the i5/OS™ operating system, this task is now easier to accomplish.

Release V5R4M0 of i5/OS has some new functions to ensure that objects added to a library are journaled at creation. That is, they start out life in a journaled state and even their creation is recorded in the journal. To accomplish this, a special data area named QDFTJRN must be created in the library of interest. The data in this data area informs the operating system what journal to start journaling to, what objects to consider, and what operations to consider. Thereafter, when objects that are eligible to be journaled (object types of *FILE, *DTAARA, and *DTAQ) are added to the library, the operating system looks for this data area in the library and uses the data within it to decide whether journaling should be started for the object.

The layout and contents of the QDFTJRN data area

The information in the data area must be entered in a special format. The data area should be created as a character data area of at least 40 bytes. The following table illustrates the layout of the data area.

Offset	**Field**	**Description**	
1	Library Name	The library name for the journal. Left aligned. Entered in UPPER CASE.	
11	Journal Name	The name of the journal to which the objects are to be journaled. This name also must be entered in UPPER CASE.	
21	**Object type/operation pairs**		
	Length of field	**Field name**	**Description**

	10	Object Type	The object types to consider journaling. This field must be one of the following special values: • *FILE – just physical files • *DTAARA – just data areas • *DTAQ – just data queues • *ALL – all eligible object types
	10	Operation	The operations that should cause journaling to automatically start. If no value is specified then *CREATE is assumed. This field must be one of the following special values: • *ALLOPR – this option includes *CREATE, *MOVE and *RESTORE (*RSTOVRJRN is not included in *ALLOPR) • *CREATE – Create operations (including those on behalf of CRTDUPOBJ or CPYF) will cause automatic journaling. • *MOVE – Moving an object into the library will cause automatic journaling. • *RESTORE – Restoring an object to the library will cause automatic journaling. • *RSTOVRJRN - when restoring an object, override the usual start journal behavior (PTFs SI24505, SI24794, SI24812, and SI24864 are required for this support)

The last pair of 10-character fields (object type and operation) can be repeated to allow different options for different object types. When more than one object type/operation combination is specified in the data area, the first combination that covers an object and operation will be used.

Caution: If lower case is used for the library name, the library will not be found. Likewise, using lower case for the journal name will prevent the journal from being found and journaling will not be automatically started.

How about an example?

The following example shows how to set up the QDFTJRN data area to have journaling started to journal JRNL in library @JRNLIB for objects added to library PRODLIB.

Perhaps your application creates temporary work data areas in library QTEMP and moves them into the production library. You do not want to bother journaling these temporary work data areas. So your desire is to have the following customized behavior: Files automatically start journaling during any operation, data areas only automatically start journaling when they are created in the library, and data queues never automatically start journaling. To achieve this behavior, the QDFTJRN data area would be created as follows:

```
CRTDTAARA DTAARA(PRODLIB/QDFTJRN) TYPE(*CHAR) LEN(100)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (1 10)) VALUE(@JRNLIB)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (11 10)) VALUE(JRNL)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (21 10)) VALUE(*FILE)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (31 10)) VALUE(*ALLOPR)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (41 10)) VALUE(*DTAARA)
CHGDTAARA DTAARA(PRODLIB/QDFTJRN (51 10)) VALUE(*CREATE)
```

Assuring your users have sufficient authority to succeed

For journaling to successfully start in this automated fashion, all of the normal authority rules still apply. Because applications creating new objects are not only creating the object but also, under the covers, are performing a start journal operation, any user adding objects to the library must have sufficient authority to both the journal specified in the QDFTJRN data area and the journal's library. You might recall that start journal operations require both *OBJOPR and *OBJMGT authority to the journal and *EXECUTE authority to the journal's library. Do not worry that giving all your users

this much authority to the journal means that they can see information they should not. The secret to proper authority management is that you are really dealing with two objects: the journal to which the newly created object is journaled, and the underlying journal receiver into which the resulting object changes are recorded. Your users need not be given authority to both objects. You can limit how much information your users can access by excluding their access to the journal receivers.

Anything that can prevent automatic start journaling from working?

If things go wrong and journaling was not started as expected, you will get message CPI6954 (Object could not be journaled) in the job log of the job creating the object. Any of a variety of mistakes might have been made. Obviously journaling cannot be started if the designated journal does not exist. i5/OS could easily draw this conclusion if you entered what appears to be the wrong journal name and hence the designated journal cannot be found (perhaps you entered the name in lower case in the QDFTJRN data area). Another situation that will prevent journaling from automatically starting is if the designated journal is not operational. Perhaps the sequence numbers have been exhausted or the currently attached journal receiver has reached its maximum size. Any of the conditions that prevent journaling from starting will not, however, prevent the object from being added to the library. The act of creating the object and the act of starting journaling for this new object are independent. The first can succeed even if the second fails.

How about object moves rather than creates?

Maybe you have an application that you enhance occasionally. To test your enhancements you create your new objects in a test library and then move them to your production library when you are confident with your changes. What will happen when you move the objects to your production library? If you were journaling the object when it was in the test library, then it will remain journaled to that same journal even if the production library contains a QDFTJRN data area that indicates that journaling should be started to a different journal when moving objects of this type into the library. Only if the object is not journaled at the time of the move operation would the QDFTJRN data area information be used to start journaling.

How about restore time behavior using *RESTORE?

What if you test your new enhancements on a test system and restore them to your production library? Will journaling be started in this situation? If the object being restored was journaled on your test system, then the first attempt is always made to start journaling to the journal the object was journaled to when it was saved prior to considering any QDFTJRN data area that might exist in the library into which the object is being restored. Only if that journal or its library cannot be found will an attempt be made to journal the object to a journal that was specified in the QDFTJRN data area. If the save time journal is found, but is not healthy, then journaling is not started during the restore operation at all and message CPF3848 (Security or data format changes occurred) is sent as a diagnostic message to the job log. An error such as this will not fail the restore operation.

If the object being restored was not journaled on your test system, then any QDFTJRN data area found in the restore library would influence whether journaling is started during the restore operation. If the attempt to automatically start journaling on behalf of the QDFTJRN data area fails, the restore operation will still be successful.

What about restore time behavior using *RSTOVRJRN?

Perhaps you would like to save your production library, and restore it to another library on your production system to allow for testing new applications. Maybe you would like to be able to do this without having the objects in your test library journaled to your production journal but you do want them journaled. If this scenario sounds like something you would like to do then you might wish to check into the *RSTOVRJRN function.

The *RSTOVRJRN function was delivered for V5R4M0 via four PTFs. All four PTFs are required to provide the function. When using *RSTOVRJRN the journal that was used at the time the object was saved is not used during restore. Like *RESTORE, if an existing object is being restored over, then the journaling attributes of the object on the system are not changed. But when not restoring over an object, with *RSTOVRJRN, the state of the object at save time does not affect the journaling behavior during the restore operation. During the restore, only the QDFTJRN data area will influence whether or not journaling is started during the restore operation.

The required PTFs for this function are: SI24505, SI24794, SI24812, and SI24864.

Are there any concerns with using a new journal during restore?

The situation to consider when using the QDFTJRN data area to change journaling during the restore operation is the use of the Apply Journaled Changes (APYJRNCHG) command. Be aware that in order to use APYJRNCHG to apply changes, the object must be journaled to the same journal it was journaled to at save time. If there is a desire to use APYJRNCHG, and the restore operation started journaling the object to a different journal because of data in the QDFTJRN data area, then the object should be saved again after the restore. This newly saved object will be the object that can be used to apply any future journaled changes. Specifying *RSTOVRJRN in the QDFTJRN data area is one way that a new journal could get used during a restore.

What journal attributes will be used?

An ordinary start journal command request provides a set of choices for customizing the resulting journal behavior. Specifically, this is done via the images (IMAGES) and/or Journal entries to be omitted (OMTJRNE) parameters on the Start Journal Physical File (STRJRNPF) or Start Journal Object (STRJRNOBJ) commands. It is natural to ask which of these behaviors will ensue if you employ the QDFTJRN data area technique to start journaling. When journaling is automatically started for a file, the journal images value will be *BOTH and the entries to be omitted value will be *OPNCLO. When journaling is automatically started for a data area, the journal images value will be *AFTER. Neither of these choices is available when journaling data queues. If different journaling options are desired for files or data areas, the Change Journaled Object (CHGJRNOBJ) command can be used to change the journaling options. The following example changes the journaling attributes of a file to capture only the after images of the records.

```
CHGJRNOBJ OBJ((PRODLIB/CUSTMAST *FILE)) ATR(*IMAGES) IMAGES(*AFTER)
```

Can you journal SQL tables to a journal other than QSQJRN?

When creating tables via SQL, the operating system has always looked for journal QSQJRN in the library into which the table is being created and started journaling to that journal. If you want to have a different journal employed as the default for new SQL tables, a QDFTJRN data area can be created in the library (or Schema) to direct the operating system to start journaling the table to a journal other than QSQJRN, perhaps even a journal in another library.

Why journal objects at creation?

By using the QDFTJRN data area to journal objects at creation there will never be a time when your nightly save does not find all of the objects in your library journaled. This allows for recovery of your objects to another point in time via the Apply Journaled Changes (APYJRNCHG) command.

Also, having the objects start journaling at creation ensures that the very first change made to the object is recorded in the journal. This might be important to you if you have auditors who want to see every change, or if you are replicating your object to another system via the changes recorded in the journal.