DBMoto®

IBM DB2 for i Journals and Receivers

Version 9.0.8.5

Software Release Date: 5/31/2017

HiT Software, Inc.	T +1 408-345-4001
4040 Moorpark Ave	F +1 408-345-4899
Suite 221	info@hitsw.com
San Jose, CA 95117	www.hitsw.com

i/iSeries/AS400 System Journals and Receivers

If you are performing mirroring or synchronization with an DB2 system source table, you need to manage the journals and receivers for your source tables. Typically, your system administrator manages journals and receivers, but it is helpful to know a little about the operations involved.

A journal is a collector of modified data from "journaled" files. The modifications that occurred in the files are detailed and written in a receiver as a log of the operations performed on the physical file.

The terminology is a little misleading: a journal is not, as you would expect, the place where modifications are tracked, but only the reference to write them on a receiver. A receiver is the physical location where traced modifications are written.

When a journal is started, it must be associated with a receiver. Receivers are set to a defined size that can be configured to fit your needs. The receiver is "bound" to the journal.

A physical file cannot be associated with more than one journal at the same time, like a journal cannot be associated with more than one receiver at a time. However, the same journal can track information for many physical files. If you want to change the current journal/receiver setting for a file, you can stop the journaling for that file and associate the file with a different one, but not at the same time.

Logical files (view, indexes, etc.), as well as LOB data types, are not journaled. Because a file must be journaled to be replicated in mirroring or synchronization modes, logical files can only be replicated in refresh mode.

Journal and Receiver Names

Use receiver names in which the last 4 or 5 character of the name are digits, starting with 00001, like

```
DBRSJ00001
```

Then create a new receiver with a command like

```
CRTJRNRCV JRNRCV(DBRSWRK/DBRSJ00001) AUT(*ALL)
```

and the journal with the command

```
CRTJRN JRN(DBRSWRK/DBRSJ) JRNRCV(DBRSWRK/DBRSJ00001) AUT(*ALL)
```

Note the names of journal and receiver: the second the second is the same as the first with a "00001" suffix.

New for DBMoto 7: It is now possible to create and use journals with minimized entries with DBMoto:

```
CRTJRN JRN(DBRSWRK/DBRSJ) JRNRCV(DBRSWRK/DBRSJ00001) AUT(*ALL)MINENTDTA(*FLDBDY)
```

Adding the MINENTDTA(*FLDBDY) option to the CRTJRN command will decrease the size of journal entries as follows. The Minimized Entry Specific Data (MINENTDTA) parameter for an object type allows entries for that object type, in this case a database physical file, to be minimized. While *FILE, *DTAARA, and *FLDBDY values are allowed the MINENTDTA parameter in the CRTJRN command, DBMoto supports only *FLDBDY (field boundaries.) This means that the minimizing will occur on field boundaries. The entry specific data will be viewable and may be used for auditing purposes.

A journal configured with MINENTDTA(*FLDBDY) only saves values for changed columns: changes to primary keys are not logged in the journal. To force one or more columns to be saved in the journal even if not changed, create a journaled index on those columns. DBMoto requires primary key values to be present for each change, therefore it is necessary ry to create a journaled index on all the fields mapped to a target DBMoto key.

Below is an example of how to set a table that DBMoto can replicate. For the following table and index, assuming that HITTEST2/QSQJRN is the journal name:

```
create table hittest2.test1 (id integer, name char(20),address varchar(50), dbo
date)
```

```
create index hittest2.test1idx on hittest2.test1 (id)
```

you should journal the index to ensure that the ID values appear in the journal:

```
STRJRNAP FILE(HITTEST2/TEST1) JRN(HITTEST2/QSQJRN)
```

Finally, use the command

```
CHGJRN JRN(DBRSWRK/DBRSJ) JRNRCV(*GEN)
```

The option (*GEN) enables automatic naming for receivers based on last number + 1.

If you need to have "different" names for journal and receivers, you can set the automatic naming management using the following command:

```
CHGJRN JRN(DBRSWRK/DBRSJ) MNGRCV(*SYSTEM)
```

(which is automatically set when using the JRNRCV(*GEN) option)

Using automatic naming management, MNGRCV(*USER), for receivers is useful because when the current receiver ("attached" receiver) becomes full, the system will automatically unbind the attached receiver, then create and bind a new receiver with a name based on last number + 1, i.e.

DBRSJ00001 --> DBRSJ00002

Activating a Journal

```
STRJRNPF FILE(DBRSWRK/TableFILE) JRN(DBRSWRK/DBRSJ) IMAGES(*BOTH) OMTJRNE(*OPNCLO)
```

The IMAGES parameter can be set to either *BOTH or *AFTER. HiT Software recommends that you set the IMAGES parameter to *BOTH. This option saves the record's image in the log, before and after the update command, and is requested by DBMoto in order to correctly manage the record's primary key. If you choose to set the IMAGES parameter to *AFTER, you will need to modify the target table by adding a Relative Record Number (RRN) field and mapping it to the source table !RecordID field as follows:

1. Create an Int field on the target table and make it the primary key.
2. Create source and target connections for the replication.

3. Define the replication.

4. When mapping source and target fields, right click on the target table field you want to map the RRN to, and choose Map to Expression...

5. In the upper pane of the Expression Editor, type [!RecordID].

6. Click **OK** to save the expression.

Note: You should not set the IMAGES parameter to *AFTER if you are planning to perform synchronization (bi-directional mirroring). Also if the source table is reorganized, you need to run a refresh replication on the target table (to update RRN changes) before mirroring can proceed again.

Receiver Management

When a receiver is unbound for any reason (because it becomes full, manual management, system management), you can choose what to do with it between:

- Unbind it and let it remain on disk
- Unbind and delete it

This assumes that you have automated the receiver management as described before, and therefore excludes a manual intervention: you are giving the system the appropriate instructions in order to manage receivers by itself.

In order to allow the receiver to be unbound but not deleted, you need to run the following command on your journal object:

```
CHGJRN JRN(DBRSWRK/DBRSJ) DLTRCV(*NO)
```

The default is *YES. Using this setting, the system will unbind old receivers and keep them available on disk. The unbound receivers available on disk are called "online" (their status), while the unique and currently bound receiver is called "attached".

If you need to change the system management to delete old receivers when they're unbound, you need to run

```
CHGJRN JRN(DBRSWRK/DBRSJ) DLTRCV(*YES)
```

Sequence Number for Journal Entries

A receiver is the final container which keeps track of every operation (aka transaction) performed on the physical files with which they are associated via the journal. Every operation tracked is completely described and numbered with a list of details - date, time, kind of operation, etc. - and a numeric progressive value, named "Sequence number".

You can choose to set the sequence number for journal entries to continue among several receivers or to be reset at every receiver change. Run the command

```
CHGJRN JRN(xxx) JRNRCV(*GEN) SEQOPT(*CONT)
```

to allow your journal entry sequence number to continue in the subsequent receivers when the attached receiver is unbound.

Normally, when you change journal receivers, you continue the sequence number for journal entries. When the sequence number becomes very large, you should consider resetting the sequence to start the numbering at 1.

You can reset the sequence number only when all changes are forced to auxiliary storage for all journaled objects and commitment control is not active for the journal. Resetting the sequence number has no effect on how the new journal receiver is named.

If you use system journal-receiver management for a journal, the sequence number for the journal is reset to 1 whenever you restart the system or vary on the independent disk pool containing the journal. When you restart the system or vary on an independent disk pool, the system performs the change journal operation for every journal on the system or disk pool that specifies system journal-receiver management.

The operation that the system performs is equivalent to

```
CHGJRN JRN(xxx) JRNRCV(*GEN) SEQOPT(*RESET)
```

The sequence number is not reset if journal entries exist that are needed for commitment control IPL recovery.

The maximum sequence number is 2147483136. If you specified RCVSIZOPT(*MAXOPT1) or RCVSIZOPT(*MAXOPT2) for the journal that you attached the receiver to, then the maximum sequence number is 9999999999. If this number is reached, journaling stops for that journal.

When the size limit is reached (or a manual intervention occurs), a system-managed receiver is unbound. If the delete option is disabled, it becomes an "online" receiver, meaning it is not attached but still available.

Sequence Numbers and DBMoto Mirroring

DBMoto keeps all the information about the current status of the replication, including the journal entry number and corresponding receiver, in the metadata.

It is important to have the sequence number for journal entries to continue from receiver to receiver because DBMoto, when replicating in mirroring mode from the i/iSeries/AS400, performs comparisons between the last used (mirrored) journal entry and the current one. If the replicator detects a current value lower than the last managed value (stored in the metadata), it stops replicating and reports a message saying that a potential problem has occurred.

For this reason, it is important to keep old receivers on disk without deleting them (keep them "online".) DBMoto stores journal entry numbers so that if a change receiver (or more than one) occurs and the last managed entry is now in an unbound receiver but the receiver is still online, the data are retrieved and the mirroring process continue without problems. If, instead, the last managed journal entry is in a deleted receiver, a potential problem is signalled in the log because something unexpected has occurred, and transactions could be missing. DBMoto also stops replicating the source files associated with that journal, The information stored in an online receiver is still retrievable using simple i/iSeries/AS400 CL commands. DBMoto uses these commands. Deleted receivers cannot be used at all, so if they are not yet processed, the information they hold is lost – and, when mirroring, even a potential loss of information is a problem.

Even if a new receiver is created and bound, it does not help. If, for instance, the Data Replicator was stopped BEFORE it had processed all the journal entries in the old receiver, data may already be (potentially) missing. As a general rule, it is appropriate for the Data Replicator to signal an error and stop.

A contextual stop for DBMoto when an IPL is performed or backup operations are in place on the i/iSeries/AS400 is a very good rule. Journaling and backup at the same time can often lead to problems.

Last Updated on 6/1/17