



## Job Accounting and the Retrieve Journal Entries API

[iPro Developer](#)

[Carsten Flensburg](#)

Carsten Flensburg

Fri, 12/21/2012 - 6:00am

APIs by Example

[Click here](#) to download the code bundle.

To report code errors, email [iPro Developer](#)

If you want to monitor and register activity, resource allocation, and security - or audit-related events on IBM i, many different options exist to do so, including exit points, history logs, system message queues, and system journals. All these facilities are readily available on your system at no extra cost beyond the effort of filtering and presenting the relevant information. Because an IBM i system running a full production workload can generate a heap of information, gleanning valuable knowledge and the control needed to properly manage your IBM i system might require an API programmer's skills.

In this APIs by Example, I show you how to use the Retrieve Journal Entries (QjoRetrieveJournalEntries) API to immediately extract and select job resource information from the system accounting journal QACGJRN. The point of this exercise is partly to demonstrate the QjoRetrieveJournalEntries API and partly to provide an example of how to make extracted system information accessible and useful via a CL command interface and a User Interface Manager (UIM) list panel. To that end, I've written the new Analyze Job Accounting Journal (ANZJOBACG) command. (See "Find Out More," below, for resources that cover other methods of accessing job resource information, as well as resources for extracting information from system journals by using APIs or exit programs.)

### Setting Up Job Accounting

To set up job accounting, perform the following steps, and enter the specified commands by using a command-line interface:

1. Create a journal receiver. I recommend that you give the receiver name a digit suffix (e.g., ACGJRN0001) so that additional receivers (e.g., ACGJRN0002, ACGJRN0003) can be created with the CHGJRN JRNRCV(\*GEN) command:

```
CRTJRNRCV JRNRCV (USERLIB/ACGJRN0001)
```

2. Create the job accounting journal and name it QSYS/QACGJRN. You also must have the authority to add objects to the QSYS library. The journal receiver should be the same as the receiver you created in step 1. I recommend authority \*EXCLUDE because it lets you use the data collected to charge users for resource usage.

```
CRTJRN JRN (QSYS/QACGJRN) JRNRCV (USERLIB/ACGJRN0001) AUT (*EXCLUDE)
```

3. Change the journal accounting information (QACGLVL) system value. You can set the system value to journal job accounting information, printer information, or both. \*JOB produces job (JB) journal entries, while \*PRINT produces direct print (DP) or spooled print (SP) journal entries.

```
CHGSYSVAL SYSVAL (QACGLVL) VALUE ('*JOB *PRINT')
```

4. Set the accounting code parameter ACGCDE for each user profile. You can set the accounting code to any alphanumeric string up to 15 characters in length.

```
CHGUSRPRF USRPRF (USERID1) ACGCDE (USERID1)
```

You can also specify the accounting code for a group of users by using the Change Job Description (CHGJOB) or Create Job Description (CRTJOB) command.

The default accounting code for job descriptions is \*USRPRF, which means the job description uses the accounting code from the job's user profile. A value other than \*USRPRF in the job description takes precedence over the accounting code specified in the user profile.

After you activate job accounting of type \*JOB, a journal entry is deposited to the job accounting journal every time a job ends on the system. The job accounting journal entry contains information about the job's processing and resource consumption. This information includes the job entry, start and end timestamps, completion code, routing steps, interactive transactions, CPU processing time, auxiliary synchronous and asynchronous I/O, database I/O operations, and the associated accounting code. The job accounting information is stored in the journal entry's entry-specific data (JOESD) field. For the exact layout of the JOESD field, follow the link to *Job accounting journal entry field information* listed in "Find Out More."

Retrieving Journal Entries

The Analyze Job Accounting Journal command processing program (CPP) employs the QjoRetrieveJournalEntries API to extract the requested journal entries from the QACGJRN journal. (To learn how to create the ANZJOBACG command, see "How to Compile," below.) Figure 1 shows the parameter list of QjoRetrieveJournalEntries.

Figure 1: The QjoRetrieveJournalEntries API's parameter list

Retrieve Journal Entries (QjoRetrieveJournalEntries) API			
Required Parameter Group:			
1	Receiver variable	Output	Char (*)
2	Length of receiver variable	Input	Binary(4)
3	Qualified journal name	Input	Char(20)
4	Format name	Input	Char(8)
Omissible Parameter Group:			
5	Journal entries to retrieve	Input	Char (*)
6	Error code	I/O	Char (*)
Default Public Authority: *USE			

Here, the *Receiver variable* is where the API returns the journal entry data extracted from the journal specified as the third API parameter, *Qualified journal name*. The second API parameter, *Length of receiver variable*, indicates to the API the amount of space available to return the journal entry data. The *Format name* parameter defines the format of the returned journal entry data. At this point, two formats are supported: RJNE0100 and RJNE0200 (I use the latter in this article). For details and information about and the differences between the two formats, please refer to the *Journal and Commit APIs* manual listed in "Find Out More."

The fifth—and omissible as in Options(\*OMIT)—parameter is a complex data structure that lets you specify the criteria to use to position and select journal entries of interest. The parameters supported correspond to the parameters that the Display Journal (DSPJRN) command provides, but the data structure format is quite different compared with the command prompt. I've defined and included the first 16 parameter substructures in the CPP (on IBM i 7.1); however, the API supports a total of 23 criteria. Figure 2 shows an example of the parameter structure format (what the FROMTIME and TOTIME criteria look like in RPG IV).

Figure 2: FROMTIME and TOTIME criteria in RPG IV

**-- FROMTIME			
D	JrnVarR03	Ds	Qualified
D	RcdLen	10i 0	Inz( %Size( JrnVarR03 ))
D	Key	10i 0	Inz( 3 )
D	DtaLen	10i 0	Inz( %Size( JrnVarR03.Data ))
D	Data	26a	
**-- TOTIME			
D	JrnVarR05	Ds	Qualified

```

D   RcdLen          10i 0 Inz( %Size( JrnVarR05 ))
D   Key             10i 0 Inz( 5 )
D   DtaLen          10i 0 Inz( %Size( JrnVarR05.Data ))
D   Data            26a

```

After the CPP calculates the timestamps indicating the start and end of the journal entry retrieval, it sets the two parameters as follows before the QjoRetrieveJournalEntries API call:

```

JrnVarR03.Data = %Char( RtvBegDts );
JrnVarR05.Data = %Char( RtvEndDts );

```

The process of retrieving all requested journal entries potentially involves two steps:

1. Establish the starting point. The positioning parameters include the beginning journal receiver as well as either a date and time value or a journal entry sequence number marking the offset from which journal entries are requested.
2. Continue the journal entry retrieval. If more journal entries exist than the receiver variable can hold, the API will return the continuing journal receiver and journal entry sequence number. You then pass these values as the journal entry retrieval offset in the corresponding parameters on the next API call.

This method implies that you specify the starting date and time (FROMTIME) parameter (JrnVarR03) on the first QjoRetrieveJournalEntries API call. Doing so replaces this parameter with the Starting sequence number (FROMENT) parameter (JrnVarR02) on subsequent API calls, as described in step 2. Web Figure 1 displays how this approach translates into RPG IV code.

### Web Figure 1: RPG IV code retrieving selected journal entries

```

/Free

RtvJrnE( RJNE0200
      : RcvVarSiz
      : 'QACGJRN  QSYS '
      : 'RJNE0200'
      : JrnEntRtv +
      : JrnVarR01 +
      : JrnVarR03 +
      : JrnVarR05 +
      : JrnVarR06 +
      : JrnVarR07 +
      : JrnVarR08 +
      : JrnVarR09 +
      : JrnVarR11
      : ERRC0100
      );

If  ERRC0100.BytAvl > *Zero And  ERRC0100.MsgId <> 'CPF7062';
  ExSr  EscApiErr;
EndIf;

DoW  RJNE0200.NbrEntRtv > *Zero;

  pEntHdr = pRJNE0200 + RJNE0200.OfsHdrJrnE;

  For  Idx = 1 to  RJNE0200.NbrEntRtv;

    ExSr  PrcLstEnt;

    If  Idx < RJNE0200.NbrEntRtv;
      pEntHdr += EntHdr.OfsHdrJrnE;
    EndIf;
  EndFor;

  Select;
  When  RJNE0200.ConInd = LST_COMP;
    Leave;

  When  RJNE0200.ConInd = LST_CONT;
    JrnVarR01.RcvStr = RJNE0200.ConRcvStr;
    JrnVarR01.LibStr = RJNE0200.ConLibStr;

```

```

JrnVarR01.RcvEnd = '*CURRENT';
JrnVarR02.SeqNbr = RJNE0200.ConSeqNbr;
EndSl;

RtvJrnE( RJNE0200
: RcvVarSiz
: 'QACGJRN  QSYS '
: 'RJNE0200'
: JrnEntRtv  +
  JrnVarR01  +
  JrnVarR02  +
  JrnVarR05  +
  JrnVarR06  +
  JrnVarR07  +
  JrnVarR08  +
  JrnVarR09  +
  JrnVarR11
: ERRC0100
);

If  ERRC0100.BytAvl > *Zero;
  ExSr  EscApiErr;
EndIf;
EndDo;

/End-Free

```

## The ANZJOBACG Command

Figure 3 shows the ANZJOBACG command prompt. In addition to exposing some of the most significant parameters recognized from the DSPJRN command, pointing to the main scope of the journal entries to process, I've also added several selection criteria relating to the job accounting data in the entry -specific data section of the journal entry.

**Figure 3: Analyze Job Accounting Journal (ANZJOBACG) command prompt**

```

-----
                        Analyze Job Accounting Journal (ANZJOBACG)

Type choices, press Enter.

Range of journal receivers:
  Starting journal receiver . . . *CURRENT      Name, *CURRENT, *CURCHAIN
  Library . . . . .                Name, *LIBL, *CURLIB
  Ending journal receiver . . .      Name, *CURRENT
  Library . . . . .                Name, *LIBL, *CURLIB
Starting date and time:
  Starting date . . . . .          *FIRST       Date, *FIRST, *YESTERDAY...
  Starting time . . . . .          Time
Ending date and time:
  Ending date . . . . .            *LAST        Date, *LAST, *YESTERDAY
  Ending time . . . . .            Time
Number of journal entries . . .    *ALL         Number, *ALL
Job name . . . . .                *ALL         Name, *ALL
  User . . . . .                  Name
  Number . . . . .                000000-999999
Job type . . . . .                *ALL         *ALL, *ASJ, *BCH, *INT...
      + for more values
Accounting code . . . . .          *ALL
User profile . . . . .            *ALL         Name, *ALL
Job completion code . . . . .      *ALL         0-99, *ALL
CPU time . . . . .                *ANY
Number of transactions . . . . .   *ANY
Number of aux I/O operations . . . *ANY
Output . . . . .                  *            *, *PRINT
-----

```

These parameters let you narrow down the journal entry selection based on job type, accounting code, job

completion code, and job processing statistics in terms of CPU time consumed, interactive transactions, and auxiliary I/O operations performed.

The following example ANZJOBACG command locates the interactive jobs run by user profile CARSTEN that ended in the week from September 22, 2012, to September 29, 2012:

```
ANZJOBACG RCVRNG(*CURCHAIN)
          FROMTIME(220912)
          TOTIME(290912)
          JOBTYP(*INT)
          USRPRF(CARSTEN)
```

Note that you control the system accounting journal receiver retention period from menu CLEANUP, option—Change cleanup options. In the prompt in Figure 4, you specify the number of days you want to keep the accounting journals' (and other system journal's) receivers.

Figure 4: Specifying number of days to retain accounting journal's receivers

```
Number of days to keep:
...
System journals and system logs . . . . . 180          1-366, *KEEP
```

The help text explains all the journals affected by this setting. Depending on the importance of the information stored in the accounting journal (i.e., actual costs charged to business units, possible audit recovery), you might also consider including the system accounting journal receiver directory in your daily backup.

Figure 5 displays the Analyze Job Accounting Journal list panel that results from the ANZJOBACG command. The list options available from this panel include *Display entry details*, *Print entry details*, *Display Job*, *Work with User profile*, and *Display journal*.

Figure 5: Analyze Job Accounting Journal list panel

Analyze Job Accounting Journal

WYNDHAMW

29-09-12 11:32:10

Type options, press Enter.

5=Display 6=Print 7=Display job 8=User profile 9=Display journal

Opt	Name	User	Number	Date	Time	Type	Code	Comp Code
	QCTLPAS001	CARSTEN	604317	22-09-12	21:07:34	*INT	IT	30
	QCTLPAS001	CARSTEN	608239	23-09-12	15:00:47	*INT	IT	30
	QCTLPAS001	CARSTEN	608054	23-09-12	15:08:20	*INT	IT	30
	QCTLPAS001	CARSTEN	609012	23-09-12	20:01:00	*INT	IT	30
	QCTLPAS001	CARSTEN	609424	23-09-12	20:36:58	*INT	IT	0
	QCTLPAS001	CARSTEN	609010	23-09-12	20:37:01	*INT	IT	0
	QPADEV000B	CARSTEN	610608	24-09-12	08:00:37	*INT	IT	30
	QPADEV000F	CARSTEN	610607	24-09-12	09:28:40	*INT	IT	0
	QPADEV000B	CARSTEN	611558	24-09-12	16:17:42	*INT	IT	0
	QPADEV000B	CARSTEN	611709	24-09-12	16:26:50	*INT	IT	0
	QPADEV000B	CARSTEN	611763	24-09-12	16:45:29	*INT	IT	0
	QPADEV000B	CARSTEN	613085	24-09-12	21:07:57	*INT	IT	50
	QCTLPAS001	CARSTEN	616689	25-09-12	04:42:08	*INT	IT	30

More...

F3=Exit F4=Prompt F5=Refresh F8=Work with journal attributes

F11=View 2 F12=Cancel F22=Display entire field F24=More keys

The F8 key executes the *Work with Journal Attributes* option for the system accounting journal QACGJRN; F11 toggles between the three list views; and F22 displays the full accounting code value, in case it exceeds the list column width. Please refer to the online help text for more details. You can see an example of the Display Job Accounting Journal Entry panel in Figure 6.

Figure 6: Display Job Accounting Journal Entry panel

```

-----
                                Display Job Accounting Journal Entry                                WYNDHAMW
                                                                 29-09-12  11:32:10
Job   . . . . . : QCTLPAS001      Date . . . . . : 22-09-12
User  . . . . . : CARSTEN         Time . . . . . : 21:07:34
Number . . . . . : 604317        Job type . . . . . : INT

Date and time:
Entered . . . . . : 22-09-12  09:09:25
Started . . . . . : 22-09-12  09:09:25
Ended   . . . . . : 22-09-12  21:07:34  011:58:09

Activity:
CPU time used . . . . . : 14479
Time active . . . . . : 23426589
Time suspended . . . . . : 19661487
Total transaction time . . . : 73
Number of transactions . . . : 635

I/O operations:
Synchronous . . . . . : 8120
Asynchronous . . . . . : 595

                                                                 More...

F3=Exit   F6=Work with active user jobs  F12=Cancel
-----

```

Again, cursor-sensitive help text is available to explain all the sections and fields in the display panel.

## How to Compile

Below you'll find instructions about how to create the Analyze Job Accounting Journal command. The following sources are included with the code download associated with this article (to download the code bundle, go to [iprodeveloper.com/code](http://iprodeveloper.com/code)):

CBX258—RPGLE: Analyze Job Accounting Journal—CPP

CBX258E—RPGLE: Analyze Job Accounting Journal—UIM Exit Program

CBX258H—PNLGRP: Analyze Job Accounting Journal—Help

CBX258P—PNLGRP: Analyze Job Accounting Journal—Panel Group

CBX258V—RPGLE: Analyze Job Accounting Journal—VCP

CBX258X—CMD: Analyze Job Accounting Journal

CBX258M—CLP: Analyze Job Accounting Journal—Build command

To create all above command objects, compile and run the CBX258M CL program, following the instructions in the source header. You'll also find compilation instructions in the respective source headers of the individual sources.

## Find Out More

[Converting job accounting journal entries](#)

[Job accounting](#)

[Job Notification Exit Point](#)

[Managing job accounting](#)

[QHST History Log](#)

## **IBM I 7.1 Information Center documentation**

[Job accounting journal entry field information](#)

[Journal and Commit APIs](#)

[Retrieve Journal Entries \(QjoRetrieveJournalEntries\) API](#)

## **Articles at iProDeveloper.com**

["Analyze Your Audit Journal with RPG"](#)

["APIs by Example: Journal APIs Solving Spooled Files Mysteries"](#)

["APIs by Example: Realtime Job Monitor"](#)

["APIs by Example: Retrieve Journal Entries \(QjoRetrieveJournalEntries\)"](#)

["IFS Journal Monitor"](#)

["Printing Job Performance Information from the History Logs"](#)

["Retrieve Journal Entry Exit Program"](#)

**Source URL:** <http://iprodeveloper.com/rpg-programming/job-accounting-and-retrieve-journal-entries-api>