Task Records information

A question we have already started receiving is whether we shall continue to need the Accounting class 3 records once customers have fully implemented the new queue statistics that became available in MQ V9.3.3. The answer is simply yes, and we have a good example of why that detailed level of information may be needed in this post. We have now found that there are occasions when we must look at the details the requests made to Coupling Facility when using shared queues. This post is the beginning of how we examined the queue manager conversion from an MQ API request to the CF requests. That information is only found in the WQ records, part of the Task Accounting data.

1) At the WSC we use MQSMFCSV to convert the MQ SMF records (all types and sub-types) to CSV files and to generate the DDL for Db2. We use the DDL to build the tables within a Db2 database, then load the CSV files into the appropriate tables. The names of the fields used in this document are those set by MQSMFCSV. If you are using a different SMF record parser or a database other than Db2, this information will have to be translated into the formats used by the different tools.
2) A task to MQ is created for a connection to the QM from any type of workload (CICS, IMS, Db2, RRS, TSO, Batch, CHIN, or client connections via the channel initiator). A task will always have a task ID associated with it, it is a 33 byte character field that is used to correlate the WTID, WTAS and WQ records associated with the task. The field has different names based on the record and is called WTAS_CORRELATOR in the WTID, or task identification record, CORREL in the WTAS, or Task Statistics record, and CORRELATION in the WQ or queue information record(s).
3) The task records are always created at task end and may also be created at the SMF intervals for tasks that span more than one SMF interval. These are known as Long Running Tasks (LRT). The accounting SMF interval is controlled by the ACCTIME in the ZPRM member and defaults to -1 or produce Accounting SMF at the same intervals as the Statistics.
4) The WTID record contains information that remains consistent for the duration of the task, with the exception of the 'Date and Time' fields. It also contains some potentially very useful information like the connection name and application information.
5) The WTAS record (tasks statistics) contains the date and time of the interval, the start date and time for the task, and task specific data that we often use for performance issues – including latching information and CF calls that are at the task level.
6) The WQ records (task queue records) contain the interval data and time, and very detailed information about the queue's disposition and use by this task. It includes specifics about the MQAPI calls not found elsewhere and calls to the CF to satisfy the API request. The queue statistics added with IBM MQ for z/OS 9.3.3 does not include this detailed breakdown of the API requests.
7) LRTs require special treatment, for example after the first set of records for the LRT, counts are set to zero at SMF intervals.

   a) If you do not have the 'first instance' of the task in the data being examined, some information may be hard to discern. For example, the use of selectors on MQGET processing. The Selector count and length are set at MQOPEN time, and when an open is only done at the start of a task subsequent records do not clearly indicate selector use. However, if all MQGET requests show that they are for specific records and the queue is both shared and non-indexed

it is usually safe to assume that selectors are in use.
b) Note that if an MQGET is done against a shared queue , has a 'typical' match option, and the queue is not indexed properly; that MQGET will fail with a 2207 (CORREL_ID_ERROR).
c) If the information is needed from start to finish for a particular task, the Accounting Class(3) data should be started at queue manager start-up, or before the task is started and continue until the task has ended, or until the next restart of the queue manager.

8) To get the correct records to align for LRTs, several fields must be matched between the three tables. They are listed here with the name of the table included as a qualifier:
WTID.WTAS_CORRELATOR, WTAS.CORREL, WQ.CORRELATION
WTID.DATE, WTAS.DATE, WQ.DATE
WTID.TIME, WTAS.TIME, WQ.TIME
WTID.LPAR, WTAS.LPAR, WQ.LPAR
WTID.QMGR, WTAS.QMGR, WQ.QMGR

9) The WHERE clause in queries to associate the correct WTID, WTAS and WQ records looks something like this:
WHERE (WTID.WTAS_CORRELATOR = WTAS.CORREL AND
WTAS.CORREL = WQ.CORRELATION AND
WTID.DATE = WTAS.DATE AND
WTAS.DATE = WQ.DATE AND
WTID.TIME = WTAS.TIME AND
WTAS.TIME = WQ.TIME AND
WTID.LPAR = WTAS.LPAR AND
WTAS.LPAR = WQ.LPAR AND
WTID.QMGR = WTAS.QMGR AND
WTAS.QMGR = WQ.QMGR)

10) The WHERE clause may be extended to return rows for just about anything, including a specific date, queues that are on a specific structure (WQ.CF_STRUCTURE), a specific queue, or as in this investigation looking for all queues hosted on a designated structure and where the get count (WQ.GET_COUNT) is greater than zero.

11) If your investigation includes data from more than one queue sharing group, defining views for each QSG is necessary when the same structure names are used by more than one QSG. Please see the previous article for directions on setting up those views.

12) All MQ API requests are broken down into one or more calls to the coupling facility, depending on the attributes of the request. This activity is reported in the WQ records, for queue-based activity, and the WTAS record for task related requests (commit, backout) and each record type includes the range of CF requests that may be used to fulfill the MQ API request. We had to focus on queue activity for this investigation.

13) Don't feel daunted by the volume and complexity of the task related data, it can be overwhelming when you first look at it. Thought is required to determine just how to consolidate the data into useful information that can be used to make decisions about both application and infrastructure patterns. It took several rounds of SQL attempts to get data in a usable form with the information we were looking for, and we work with this kind of data regularly.