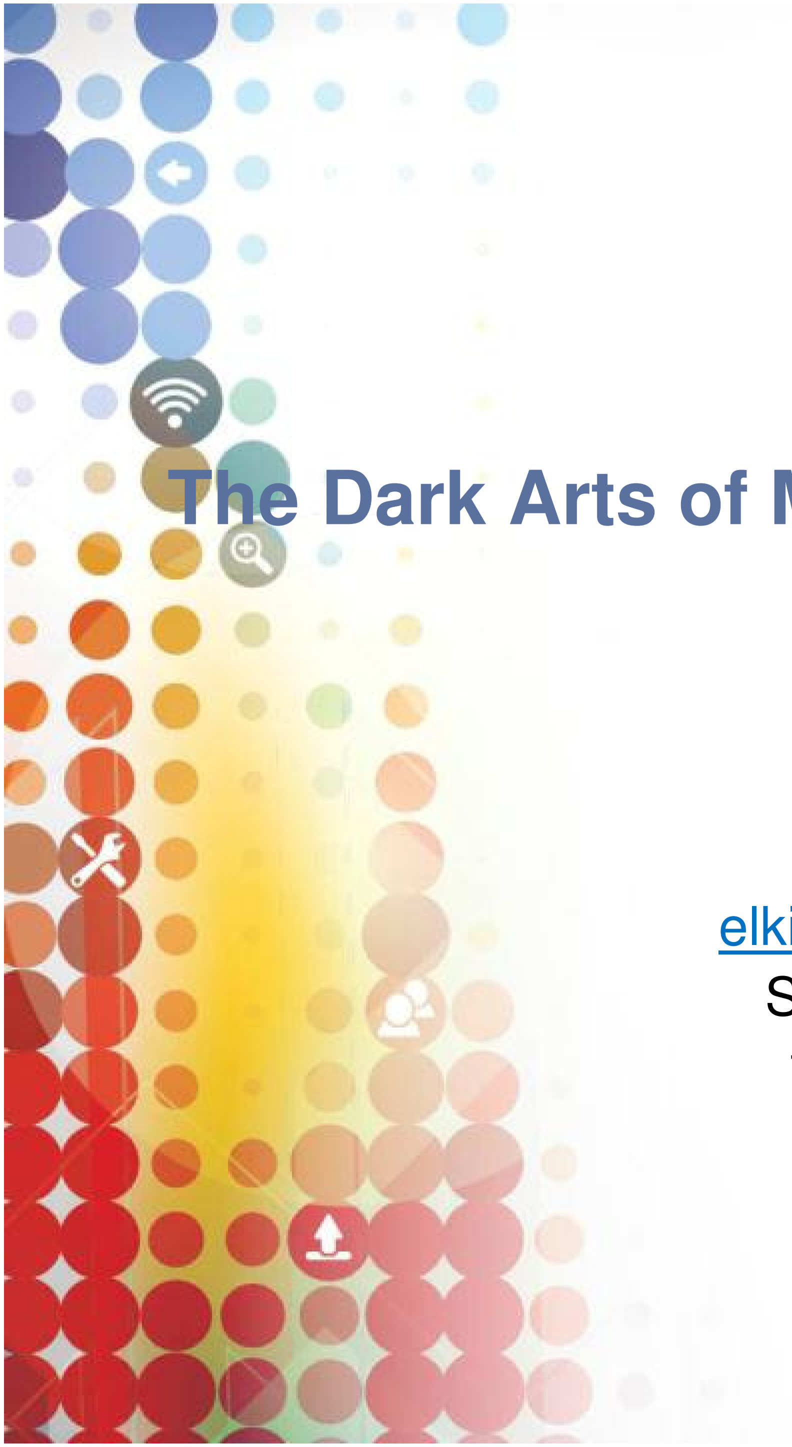
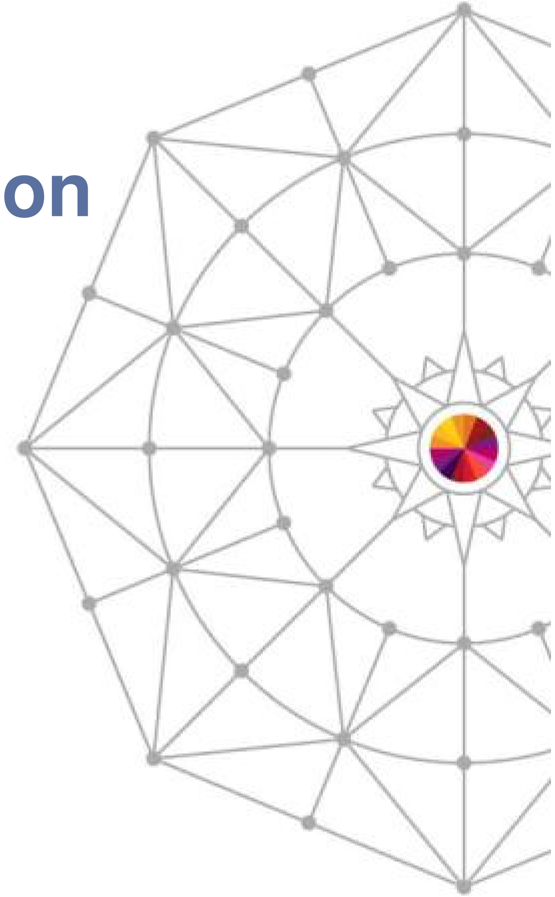


The Dark Arts of MQ SMF Evaluation

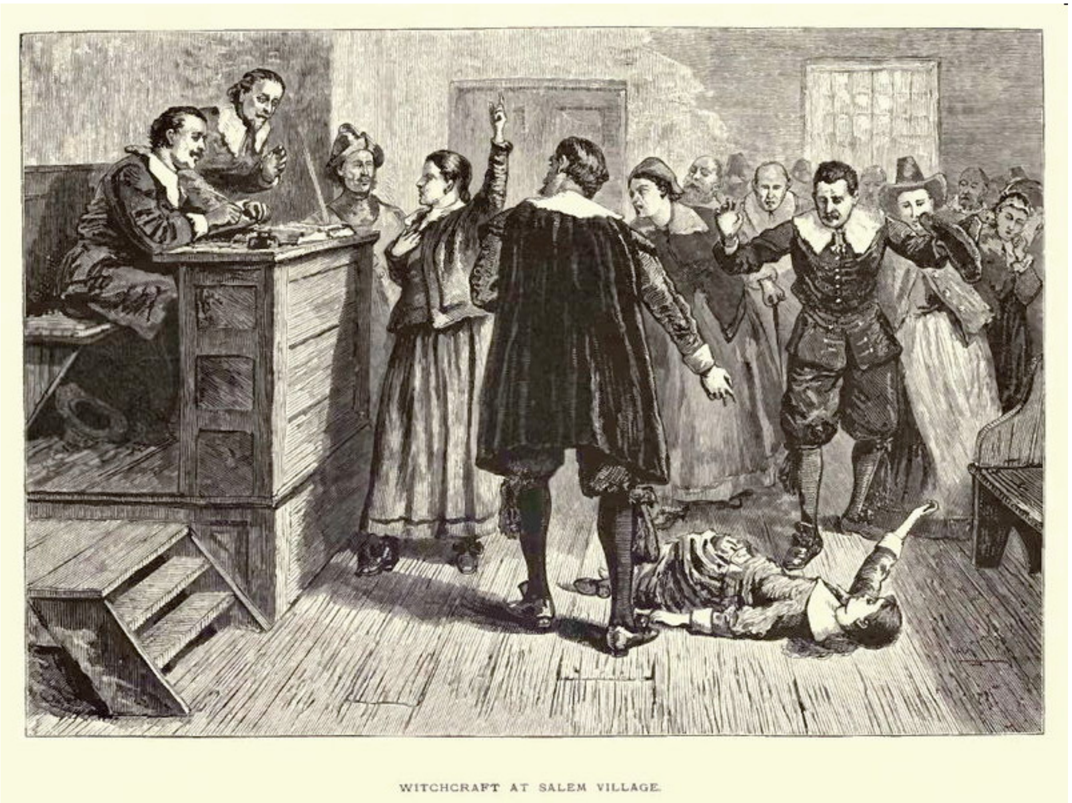


Lyn Elkins
elkinsc@us.ibm.com

Session # 15024
10 March 2014

Code!

The witch trial – MQ is broken!



WITCHCRAFT AT SALEM VILLAGE.

Why is this session first?

- The SFM data, especially the SMF115 data is more interesting after you have learned about the internals.
  - 1:30 on Wednesday
- It’s interesting after you have learned more about problem determination
  - 9:30 on Thursday
- So as to WHY this is the first MQ session in the agenda
  - It just works out that way!

Agenda

- Review of SMF 115 and SMF 116 class 3 data
- Hunting down the culprit
  - SMF115 Data
    - Bufferpool behaving badly
    - Volume growth
    - Log manager getting cranky
    - Other SMF115 data of interest
  - SMF116 Data
    - What queues are being used and how?
    - Can I find out which queues are the most active?
    - Pulling the data for one CICS transaction or batch job
    - Long running tasks
- Summary

Objectives

- This session is to delve a bit deeper into how the ATS team use the SMF data to find transaction and WMQ problems, based on situations we've tried to resolve.
- It will bore you to death.

Review of SMF115

- The SMF 115 data is the statistical information produced by a WMQ for z/OS queue manager.
  - Primarily used to track major trends and resolve performance problems with the queue manager
  - Very lightweight
  - Broken down into the major ‘managers’ within WMQ
  - The ‘old’ MP1B provides several views into the data:
    - MQ1150 – detailed SMF115 report
    - MQCSMF – extracts specific information from SMF115 and 116 in a column format
      - *Particularly useful for building spreadsheets*
  - The ‘new’ MP1B provides two views of the data
    - Report from for each manager
    - Comma separated values

Review of SMF116 – Class 3 data

- The SMF 116 data is the accounting information produced by a WMQ for z/OS queue manager.
  - Primarily used to determine what is going on within WMQ workload
  - Heavyweight
  - Broken down into the transactions within WMQ
  - The old MP1B provides several views into the data:
    - MQ1160 – prints the SMF116 class 1 report
    - MQ116S – prints the detailed SMF116 class 3 report, including the queue information
    - MQCSMF – extracts specific information from SMF115 and 116 in a column format
      - *Particularly useful for building spreadsheets*
  - The new MP1B provides:
    - The 'TASK' output
      - *Somewhat like the MQ116S report*
      - *I am currently writing a paper on the differences/similarities*
    - Other files, much like the 'old' MQCSMF output

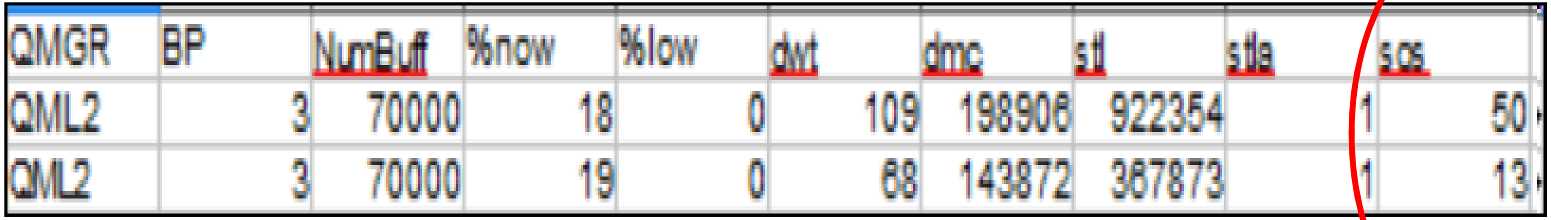
Finding the problem

SMF 115 data – Hunting down the culprit Red Flags for bufferpools

- In the next few slides, an analysis of a bufferpool under stress is shown
- First the raw SMF data for two weeks was processed thru the “old MP1B” MQCSMF and the MQ1150 format and print programs
- The Buffer Manager statistics were downloaded into a spreadsheet
- The spread sheet was sorted to find:
  - Non-Zero Short on storage counts
  - Non-Zero DMC counts
  - Percent of free pages
- This showed the areas that needed to be looked at in greater detail, and it became apparent that there were some processing patterns that need evaluation

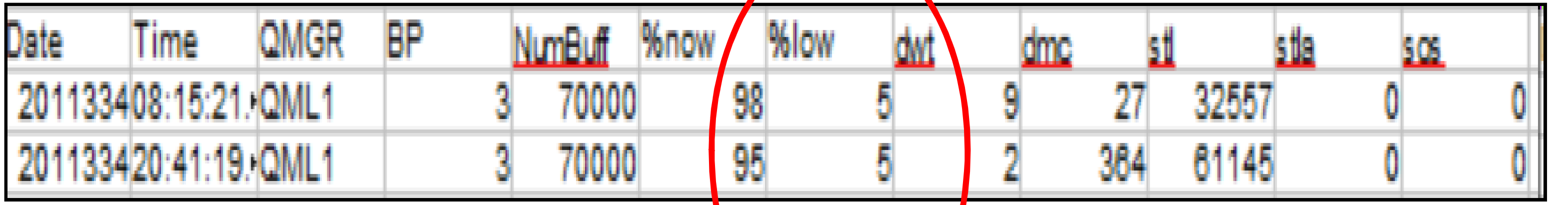
SMF 115 data – Hunting down the Culprit

- **Red Flags for Bufferpools**
  - SOS



QMGR	BP	NumBuff	%now	%low	dwt	dmc	stl	stla	sos
QML2	3	70000	18	0	109	198906	922354	1	50
QML2	3	70000	19	0	68	143872	367873	1	13

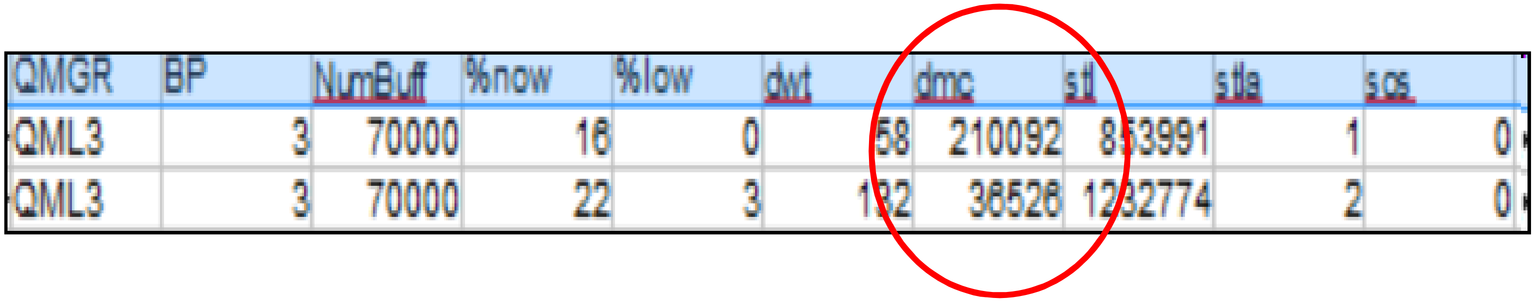
  - Freepages at 5% or less



Date	Time	QMGR	BP	NumBuff	%now	%low	dwt	dmc	stl	stla	sos
2011334	08:15:21.	QML1	3	70000	98	5	9	27	32557	0	0
2011334	20:41:19.	QML1	3	70000	95	5	2	384	61145	0	0

SMF 115 data

- **Red Flags for Bufferpools** - Continued
  - DMC – synchronous write process kicks off

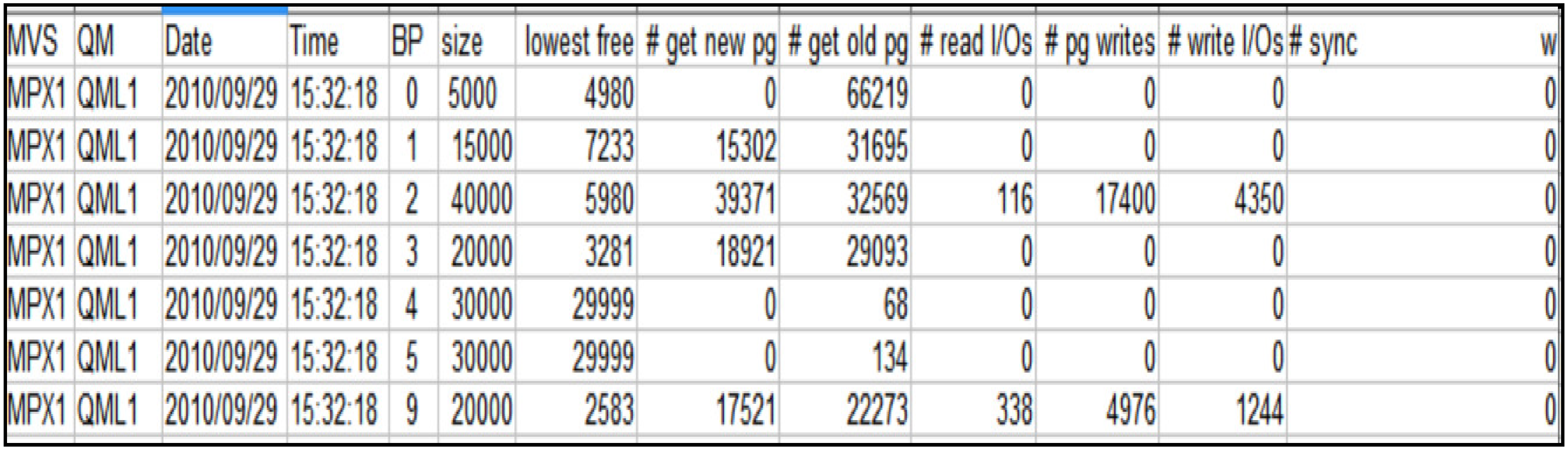


QMGR	BP	NumBuff	%now	%low	dwt	dmc	stl	stla	sos
QML3	3	70000	16	0	58	210092	853991	1	0
QML3	3	70000	22	3	132	36526	1232774	2	0

  - The DMC count should be used in conjunction with the IMW field from the SMF115 report to see how many synchronous writes were actually performed.

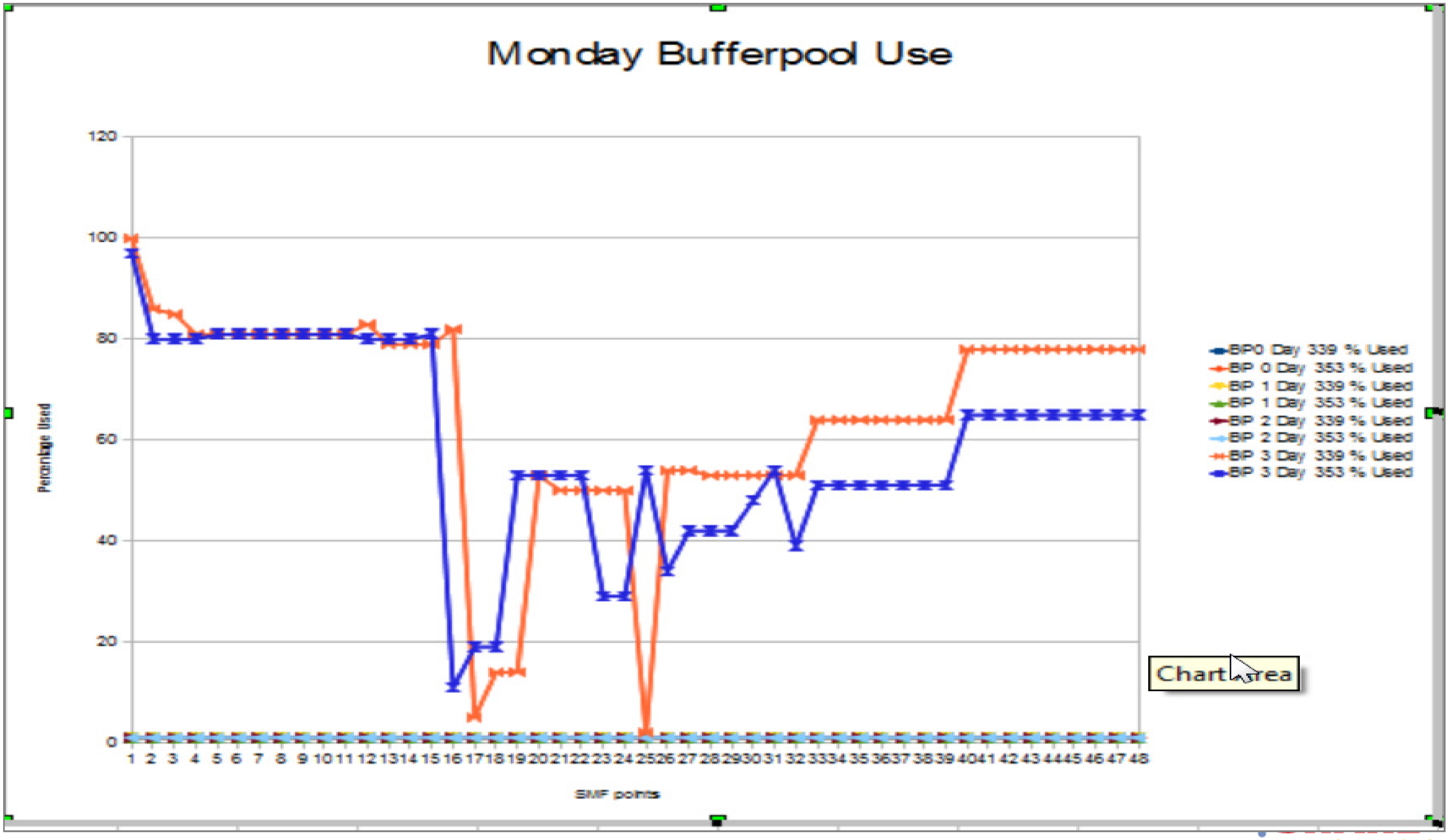
The NEW SMF print – BUFCSV file

- This spreadsheet image was ceated from WMQ V7.0.1 data thru the new MP16 print program
  - Note the data produced is different from the MQCSMF report from the old version. Important fields missing include the important SOS counts, deferred writes, and synchronous writes fields.



MVS	QM	Date	Time	BP	size	lowest free	# get new pg	# get old pg	# read I/Os	# pg writes	# write I/Os	# sync	w
MPX1	QML1	2010/09/29	15:32:18	0	5000	4980	0	66219	0	0	0		0
MPX1	QML1	2010/09/29	15:32:18	1	15000	7233	15302	31695	0	0	0		0
MPX1	QML1	2010/09/29	15:32:18	2	40000	5980	39371	32569	116	17400	4350		0
MPX1	QML1	2010/09/29	15:32:18	3	20000	3281	18921	29093	0	0	0		0
MPX1	QML1	2010/09/29	15:32:18	4	30000	29999	0	68	0	0	0		0
MPX1	QML1	2010/09/29	15:32:18	5	30000	29999	0	134	0	0	0		0
MPX1	QML1	2010/09/29	15:32:18	9	20000	2583	17521	22273	338	4976	1244		0

SMF115 – Bufferpool Trends and Analysis

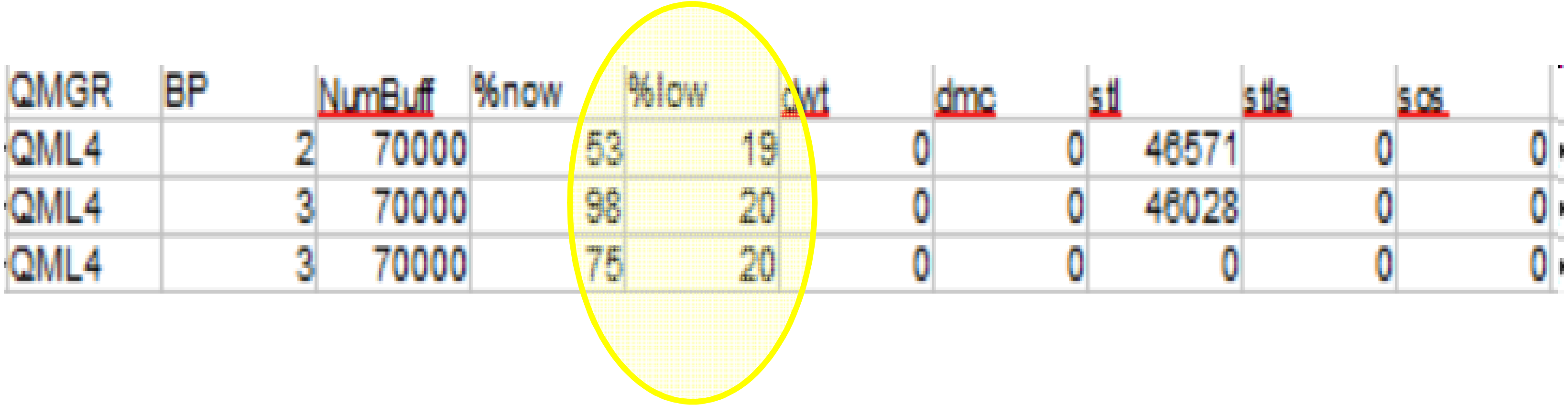


SMF115 – Bufferpool Trends and Analysis - Notes

- In the chart shown two high volume days were compared to see if there was a pattern to the BP use.
- BP 0, 1 an 2 showed almost no utilization.
- BP 3 was in very heavy use, some of the time.
- BP 3 is under some stress.
- Having multiple days worth of data is vital, had there just been one heavy day it may have been an anomaly. Data from longer periods of time, when compared like this can be very useful in tracking usage, etc.
- In this case there was a clear pattern of overuse of bufferpool 3, in further evaluation the SMF116 data showed that all the queues that were being used for this queue manager were defined on the same pageset/bufferpool. By moving some of the queues to another resource pool, the stress was reduced, work flowed faster and the CPU usage was reduced.
- In attempting to replicate the issues, the information on the previous slides was used to create the charts, but also to show that charting the pattern might be helpful in the evaluation.

SMF 115 data

- Yellow Flags for Bufferpools
  - Consistently Approaching/Achieving 20 % Free pages



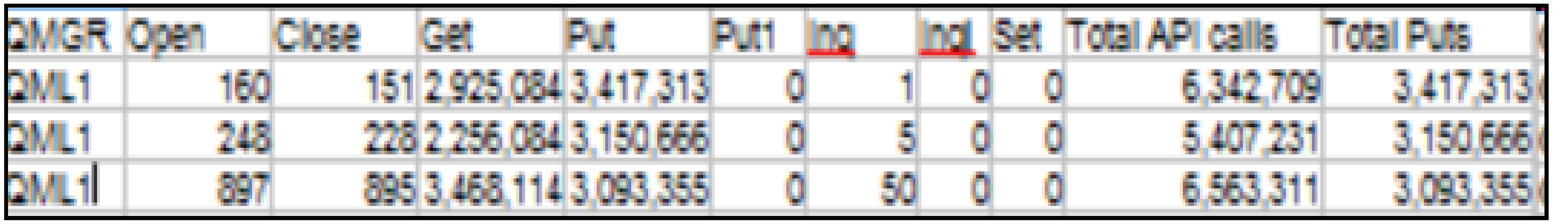
QMGR	BP	NumBuff	%now	%low	dwt	dmc	stl	stla	sos
QML4	2	70000	53	19	0	0	46571	0	0
QML4	3	70000	98	20	0	0	46028	0	0
QML4	3	70000	75	20	0	0	0	0	0

Bufferpool Use - Warnings

- In the data shown, two bufferpools were approaching the 20% freepage threshold.
- At 20% the async write task is initiated, which is not catastrophic, but if it can be avoided it should be.
- In this case, when several weeks worth of data were examined the 20% threshold was being broken on a regular basis. After evaluation fo the SMF116 class-3 data it was found this was batch oriented workload, and messages were expected to queue up for long periods of time this was not a problem. It is something to watch.

SMF 115 data – Hunting down the Culprit

- Message Manager Information
  - Good indication of queue manager usage
    - This is only a count of API calls, not one of successful calls
    - Volume trends can be approximated from the MQPUT and MQPUT1 calls, as these are generally successful
    - MQGETs may or may not have data returned

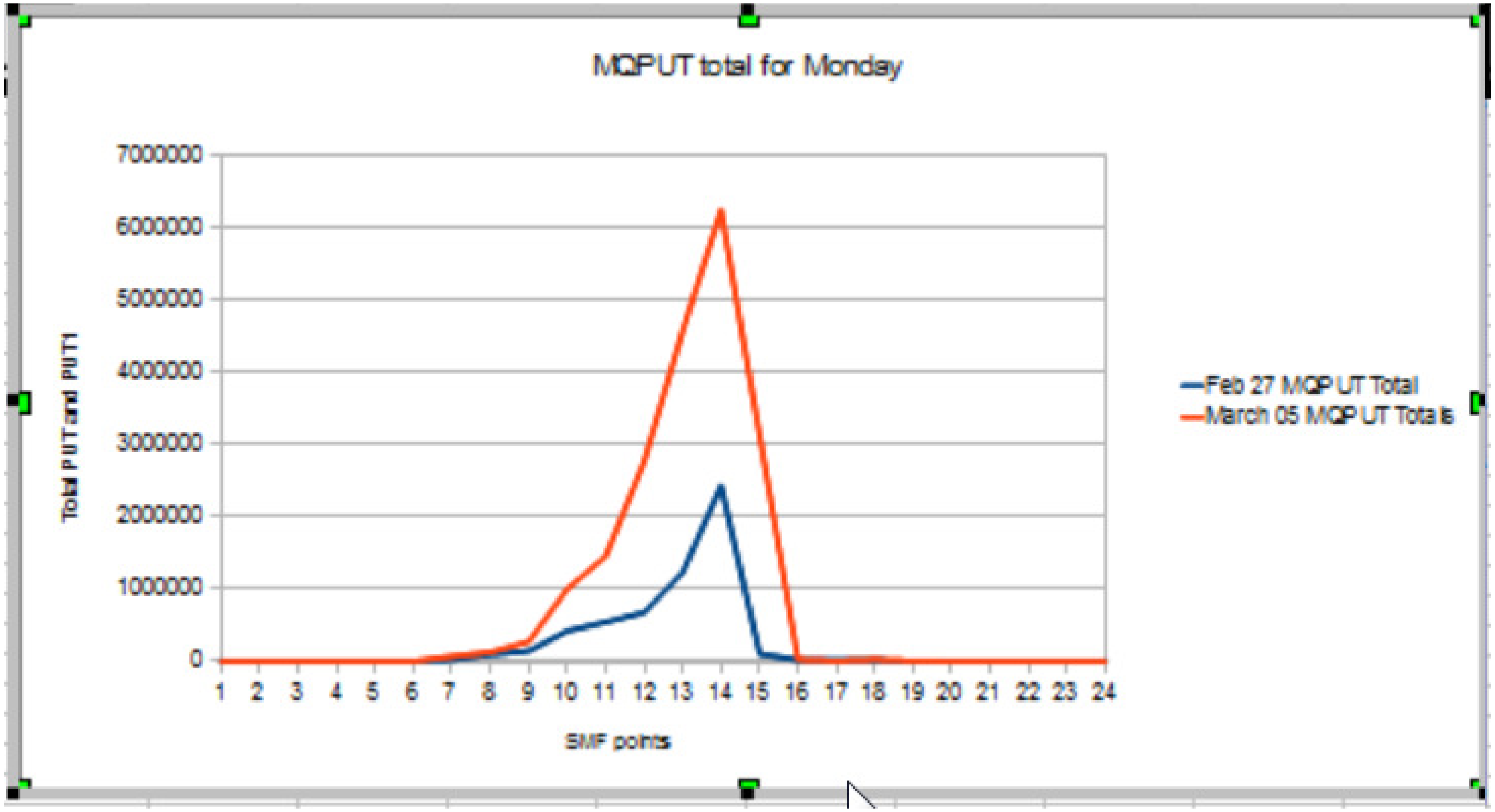


QMGR	Open	Close	Get	Put	Put1	Inq	Inql	Set	Total API calls	Total Puts
QML1	160	151	2,925,084	3,417,313	0	1	0	0	6,342,709	3,417,313
QML1	248	228	2,256,084	3,150,666	0	5	0	0	5,407,231	3,150,666
QML1	897	895	3,468,114	3,093,355	0	50	0	0	6,563,311	3,093,355

Message Manager Statistics

- This data was taken from the message manager output from the old MQCSMF format and print program.
- Two columns were added to calculate the Total API calls and Total Puts (sum of MQPUT and MQPUT1 calls)
- When charted over a few weeks an upward curve was noticed.

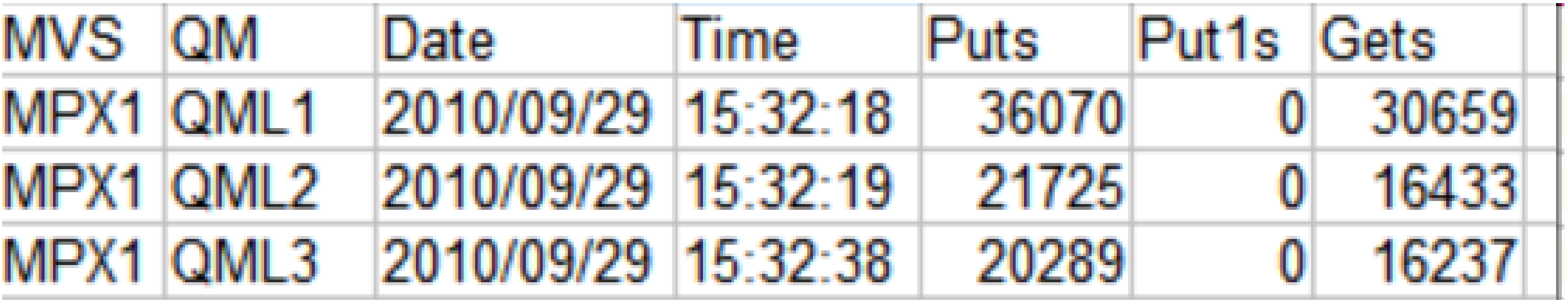
Message Manager - Trend Chart



Message Manager Trend Chart

- Two days data is not really a trend analysis, but it’s a start
- If more Mondays are charted, a real trend may emerge and show that volume is increasing allowing a good admin to plan for additional workload.
- This is an overall count for the queue manager, individual queue activity can be evaluated from the SMF116 class 3 data.

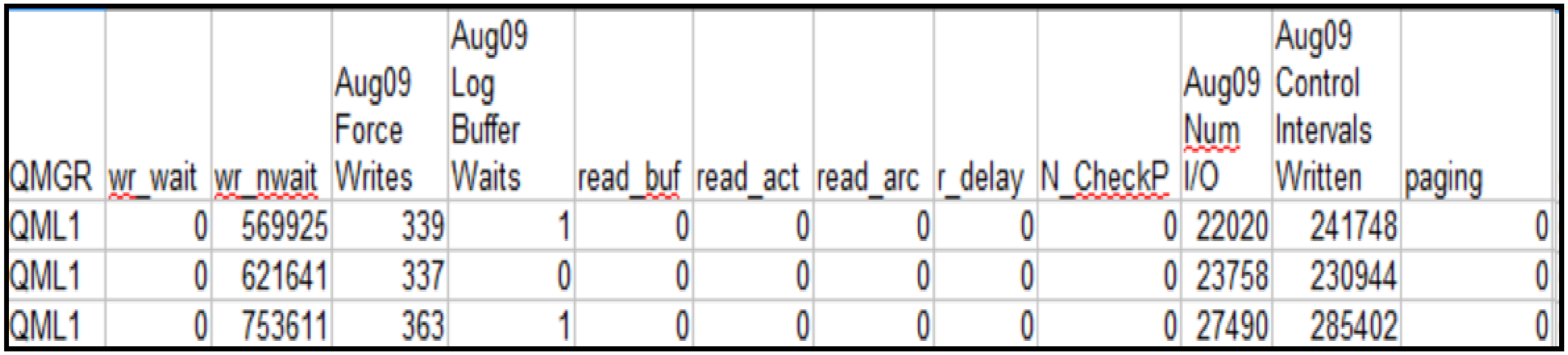
New MP1B Print program – Message Manager



MVS	QM	Date	Time	Puts	Put1s	Gets
MPX1	QML1	2010/09/29	15:32:18	36070	0	30659
MPX1	QML2	2010/09/29	15:32:19	21725	0	16433
MPX1	QML3	2010/09/29	15:32:38	20289	0	16237

SMF 115 data – Hunting down the Culprit

- Log Manager Information
  - Good indication of persistent messaging use
    - As has been mentioned before some of the counts are not complete, the checkpoints does not include those from queue manager switching



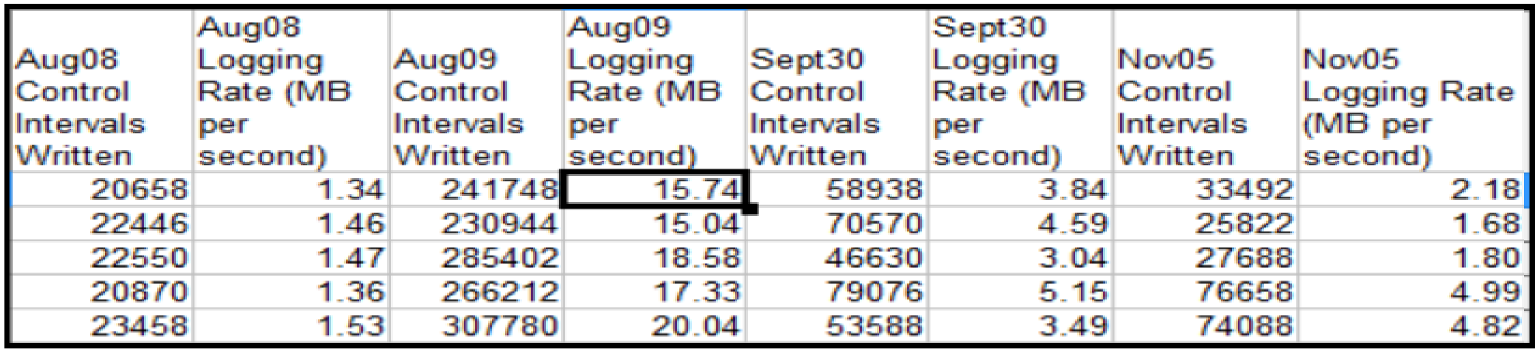
QMGR	wr_wait	wr_nwait	Aug09 Force Writes	Aug09 Log Buffer Waits	read_buf	read_act	read_arc	r_delay	N_CheckP	Aug09 Num I/O	Aug09 Control Intervals Written	paging
QML1	0	569925	339	1	0	0	0	0	0	22020	241748	0
QML1	0	621641	337	0	0	0	0	0	0	23758	230944	0
QML1	0	753611	363	1	0	0	0	0	0	27490	285402	0

Message Manager – New MP1B output

- This is from the Message Manager CSV file.
- Note that all it reports are the MQPUT, MQPUT1 and MQGET requests.
- If you are looking for any of the other requests, the TASK report (from the MP116 data) must be used.

SMF 115 data – Hunting down the Culprit

- Log manager – I/O rate
  - The I/O rate is calculated as
    - The number of CIs written * 4096 (CI size)
    - Divided by 1 M (1024*1024)
    - Divided by the number of seconds in the interval
  - The I/O rate is the throttle for many queue managers

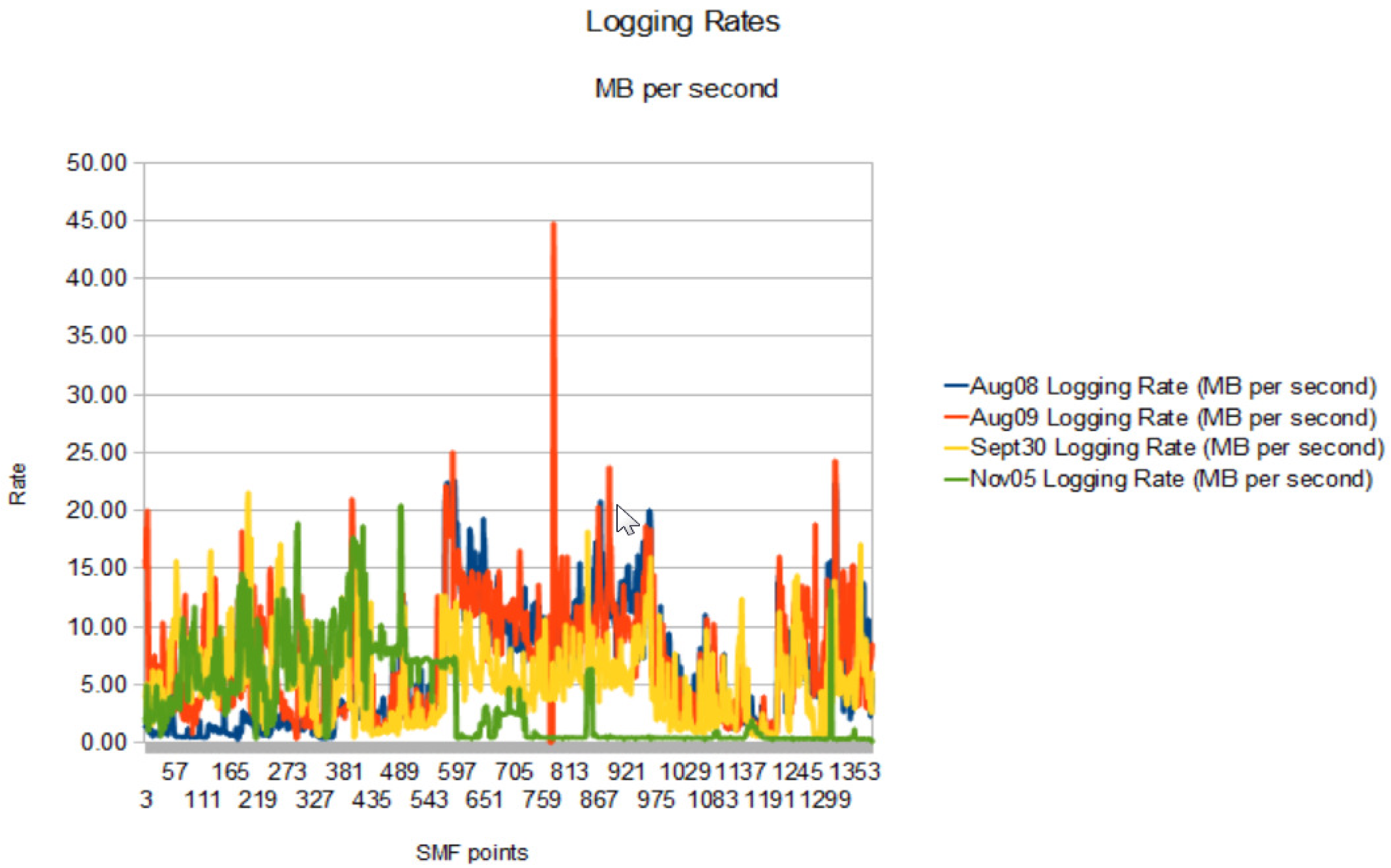


Aug08 Control Intervals Written	Aug08 Logging Rate (MB per second)	Aug09 Control Intervals Written	Aug09 Logging Rate (MB per second)	Sept30 Control Intervals Written	Sept30 Logging Rate (MB per second)	Nov05 Control Intervals Written	Nov05 Logging Rate (MB per second)
20658	1.34	241748	15.74	58938	3.84	33492	2.18
22446	1.46	230944	15.04	70570	4.59	25822	1.68
22550	1.47	285402	18.58	46630	3.04	27688	1.80
20870	1.36	266212	17.33	79076	5.15	76658	4.99
23458	1.53	307780	20.04	53588	3.49	74088	4.82

Log Manager Statistics

- This data was taken from the log manager output from the old MQCSMF format and print program.
- The log buffer waits indicates the number of times during the interval there were not free log buffers. This is somewhat tunable, but most production environments have it set to the recommended 40,000. If this count goes very high and the maximum number of buffers are allocated, then the queue manager may be saturated.
- Another critical factor is the I/O rate that can be achieved

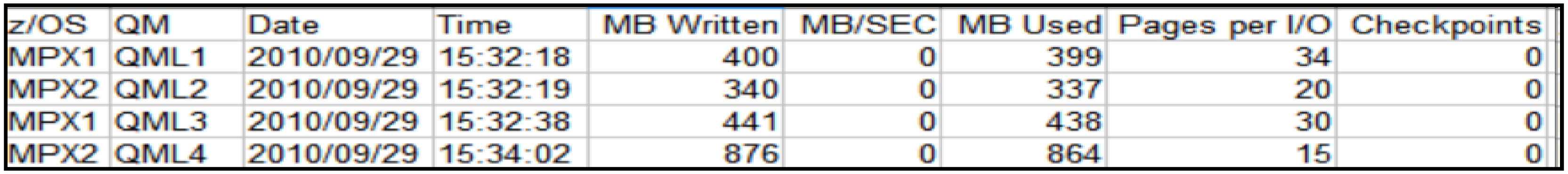
Logging Rates - Charted



Logging Rates Charted - Notes

- These rates were charted from days when there were reported slowdowns.
- Notice the huge spike in the Aug 09 data, that is an anomaly caused by a change to the SMF interval in the middle of the day. It is left in to illustrate that spikes do happen and should be investigated. They may not indicate a sudden growth rate, but can indicate a problem with the data itself.
- In looking at the data, the logging rate is frequently at the 20/25 MB per second rate. For the environment, this was quite high. It was discussed with the capacity planning team. This is continually being monitored, there may be workload that has to shift to another queue manager in the near future.

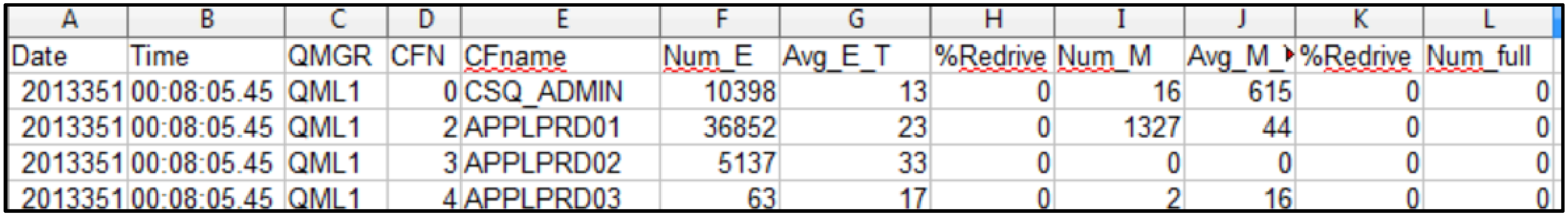
Log Manager CSV file from new MP1B



z/OS	QM	Date	Time	MB Written	MB/SEC	MB Used	Pages per I/O	Checkpoints
MPX1	QML1	2010/09/29	15:32:18	400	0	399	34	0
MPX2	QML2	2010/09/29	15:32:19	340	0	337	20	0
MPX1	QML3	2010/09/29	15:32:38	441	0	438	30	0
MPX2	QML4	2010/09/29	15:34:02	876	0	864	15	0

- The new MQCSMF print program will calculate the MB/Second written
  - A caution, it uses the number of seconds per SMF interval defined for the run. If you allow this to default, your results are likely to be incorrect.

SMF 115 data – CF Statistics



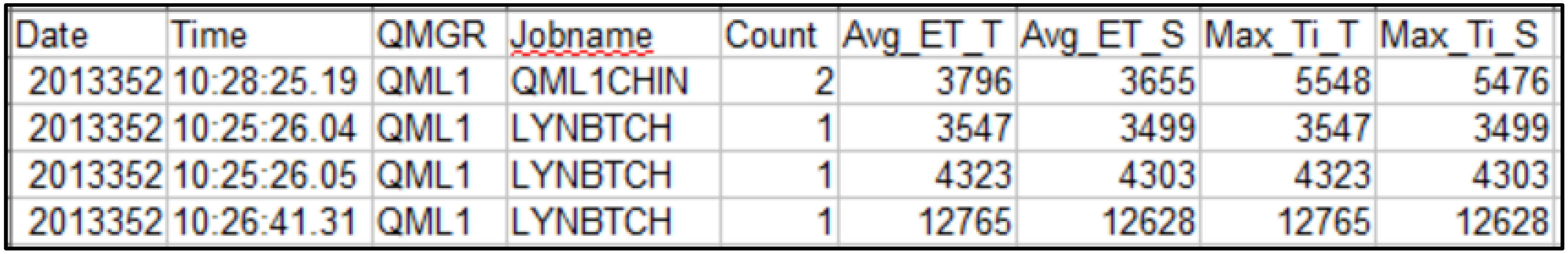
A	B	C	D	E	F	G	H	I	J	K	L
Date	Time	QMGR	CFN	CFname	Num_E	Avg_E_T	%Redrive	Num_M	Avg_M_	%Redrive	Num_full
2013351	00:08:05.45	QML1	0	CSQ_ADMIN	10398	13	0	16	615	0	0
2013351	00:08:05.45	QML1	2	APPLPRD01	36852	23	0	1327	44	0	0
2013351	00:08:05.45	QML1	3	APPLPRD02	5137	33	0	0	0	0	0
2013351	00:08:05.45	QML1	4	APPLPRD03	63	17	0	2	16	0	0

- The CF data from MQ should be used in conjunction with the Coupling Facility Activity Reports (CFRM).

- The average elapsed time is reported in microseconds, and in this example is low. As it happens the CF in use is ‘local’ – in the same CEC, so they should be low.

- Recommendation is to chart the values over time, like the other statistics looking for anomalies and use patterns.

SMF 115 data – DB2 Statistics

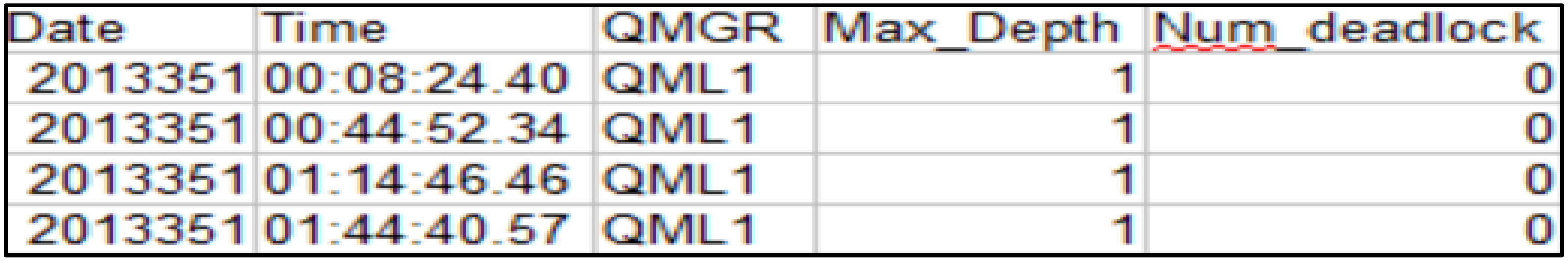


Date	Time	QMGR	Jobname	Count	Avg_ET_T	Avg_ET_S	Max_Ti_T	Max_Ti_S
2013352	10:28:25.19	QML1	QML1CHIN	2	3796	3655	5548	5476
2013352	10:25:26.04	QML1	LYNBTCH	1	3547	3499	3547	3499
2013352	10:25:26.05	QML1	LYNBTCH	1	4323	4303	4323	4303
2013352	10:26:41.31	QML1	LYNBTCH	1	12765	12628	12765	12628

- This data includes the average am maximum times spent on DB2 requests
  - Average elapsed time on the DB2 task
  - Average elapsed time on the DB2 server
  - Maximum elapsed time on the DB2 task
  - Maximum elapsed time on the DB2 server

- Recommendation is to chart the values over time, like the other statistics looking for anomalies and use patterns.

SMF 115 data – SDB2 Statistics



Date	Time	QMGR	Max_Depth	Num_deadlock
2013351	00:08:24.40	QML1	1	0
2013351	00:44:52.34	QML1	1	0
2013351	01:14:46.46	QML1	1	0
2013351	01:44:40.57	QML1	1	0

- The additional DB2 information shows the
  - Maximum depth of queues requests into DB2
  - Whether there were deadlocks
  - As with the other statistics, these should be charted to show usage patterns and detect anomalies

SMF116 Class 3 data

- Reviewing this copious data can feel like searching for the spell to turn lead into gold. It's more like panning for gold
- As a WMQ admin, you have more information at your fingertips   about your environment than we at IBM reviewing this data will have. There are a number of things that we do to look for patterns or particular problems that are discussed.

SMF116 - Hunting the culprit

- The scenario is simple:
  - ‘We are missing our SLAs on some of our transactions’
  - The SMF 115 may or may not show bottlenecks
  - You have over 3M SMF116 class 3 records from one SMF interval to see if you can find the problem
  - And, of course, ‘MQ is the problem’

What queues are being used and how?

- SMF116 class 3 data shows the use of queues
- Helpful because even as a WMQ admin, it may be a challenge to find out where the queues are
- Some specific problems:
  - Non-indexed queues
  - High volume request/reply queues in same resource pool
  - Overuse of Temporary dynamic queues

What queues are being used and how?

- We have seen some specific problems/issues at a number of customers that were addressed with an evaluation of the SMF116 data.
- In this section we are going to show some of the more common ones, and how the SMF data lead to the improvment

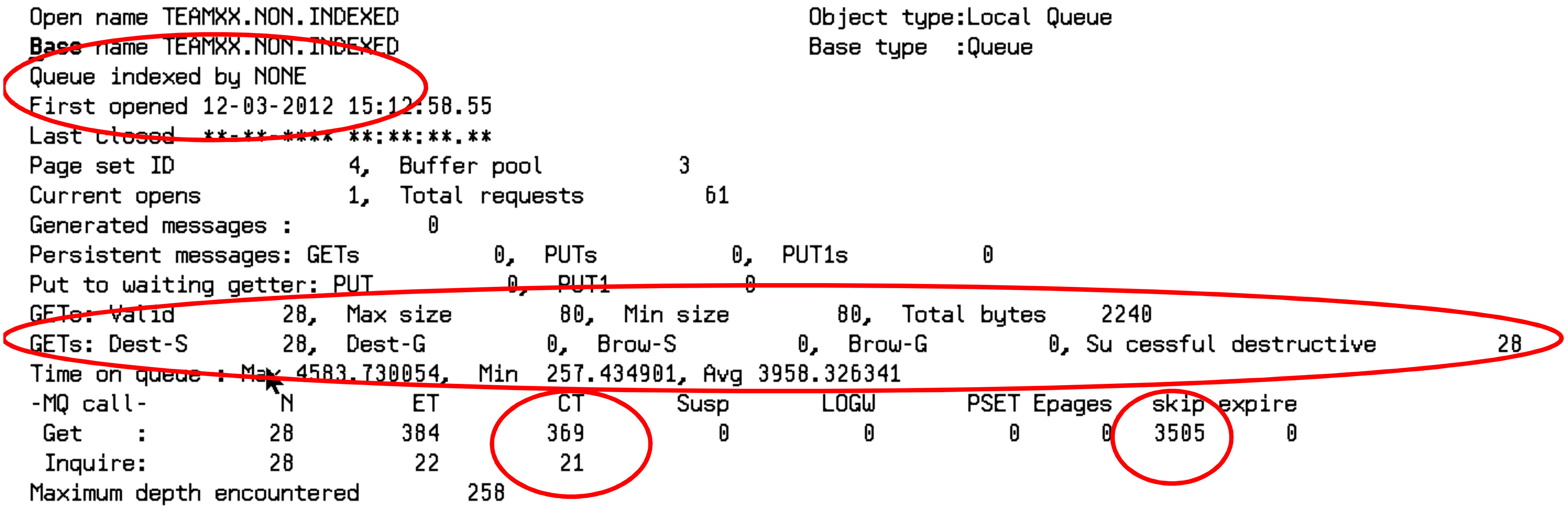
What queues are being used and how?

- Queue Indexing
  - Messages that are retrieved using an index-able field benefit from being indexed even when the depth is not high.
    - Message ID
    - Correlation ID
    - Token
    - Group ID
  - The greater the depth of the queue the greater the benefit.
  - The SMF116 queue records show when messages are retrieved using a 'known' field

Queue Indexing - Notes

- Queue indexing is unique to WMQ on z/OS
- The use of a proper index can substantially improve performance an CPU consumption, as will be shown
- Anecdotally, we’ve heard of it making a difference when queue depths were as low as 5 on a busy system
- Often the first report of a problem is when there has been a slowdown elsewhere and queue depths have grown unexpectedly

Non-Indexed Queue retrieval

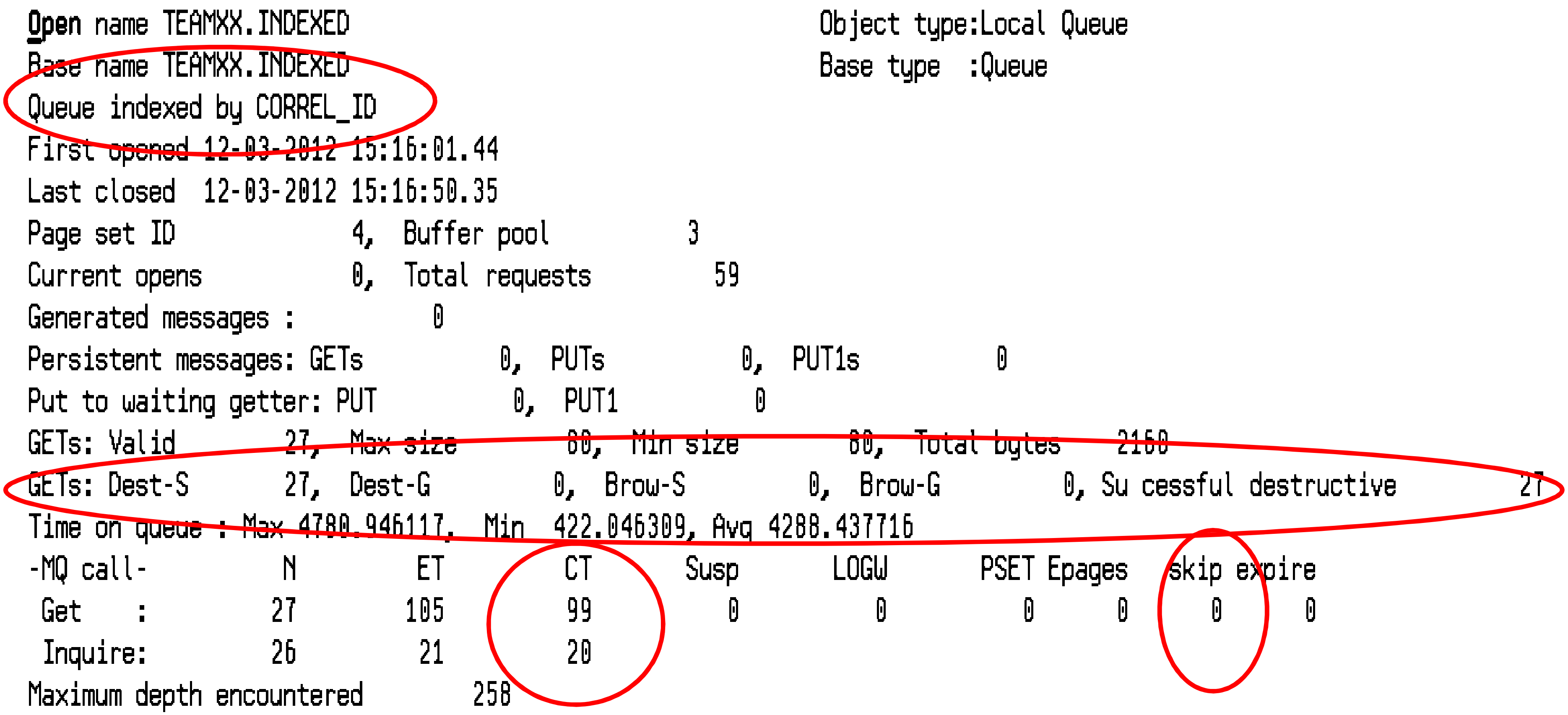


```
Open name TEAMXX.NON.INDEXED                               Object type:Local Queue
Base name TEAMXX.NON.INDEXED                               Base type  :Queue
Queue indexed by NONE
First opened 12-03-2012 15:12:58.55
Last closed  **-**-**** **:**:**.**
Page set ID             4,  Buffer pool          3
Current opens           1,  Total requests         61
Generated messages :          0
Persistent messages: GETs          0,  PUTs          0,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
GETs: Valid        28,  Max size        80,  Min size        80,  Total bytes    2240
GETs: Dest-S       28,  Dest-G         0,  Brow-S         0,  Brow-G         0, Su cessful destructive         28
Time on queue : Max 4583.730054,  Min  257.434901, Avg 3958.326341
-MQ call-          N         ET        CT        Susp       LOGW       PSET Epages   skip expire
 Get     :        28        384        369          0          0         0       0  3505       0
 Inquire:         28         22         21
Maximum depth encountered        258
```

Non-Indexed Queue retrieval - Notes

- In the SMF record, the fields of interest are:
  - The Queue Indexing
  - The Type of GET request being made, those with a '-S' are for specific messages (Get by correlid, get by message id, etc.)
  - The total CPU expenditure for the successful gets – the 'CT' column highlighted
  - The number of pages skipped while finding matching messages

Indexed Queue Retrieval



```
Open name TEAMXX.INDEXED                              Object type:Local Queue
Base name TEAMXX.INDEXED                              Base type  :Queue
Queue indexed by CORREL_ID
First opened 12-03-2012 15:16:01.44
Last closed  12-03-2012 15:16:50.35
Page set ID            4,  Buffer pool         3
Current opens          0,  Total requests        59
Generated messages :         0
Persistent messages: GETs          0,  PUTs          0,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
GETs: Valid        27,  Max size          80,  Min size          80,  Total bytes     2160
GETs: Dest-S       27,  Dest-G          0,  Brow-S          0,  Brow-G          0, Su cessful destructive          27
Time on queue : Max 4780.946117,  Min  422.046309, Avg 4288.437716
-MQ call-          N        ET         CT        Susp        LOGW        PSET Epages  skip expire
 Get    :         27       105         99          0           0           0      0     0      0
 Inquire:         26        21         20
Maximum depth encountered        258
```

Indexed Queue retrieval - Notes

- In the SMF record, the fields of interest are:
  - The Queue Indexing
  - The Type of GET request being made, those with a ‘-S’ are for specific messages (Get by correlid, get by message id, etc.)
  - The total CPU expenditure for the successful gets – the ‘CT’ column highlighted
  - The number of pages skipped while finding matching messages

Indexed vs Non - comparison

- Comparing the CPU time, both queues with the same max message depth:
  - Indexed 27 messages at 99 CPU microseconds
    - 3.667 ms per message retrieved
  - Non-indexed 28 messages at 369 CPU microseconds
    - 13.18 ms per message

- Comparing the number of pages that had to be skipped
  - Indexed = 0
  - Non-indexed = 3585

What queues are being used and how?

- High volume request and reply queue in the same resource pool
  - This is a case of ‘define like’ run amok
  - The request queue and reply queue for a high volume application were defined in the same storage class (same bufferpool and pageset)
  - By moving the reply queue to a different storage class, the resource usage was better distributed

High volume request and reply queue in the same resource pool

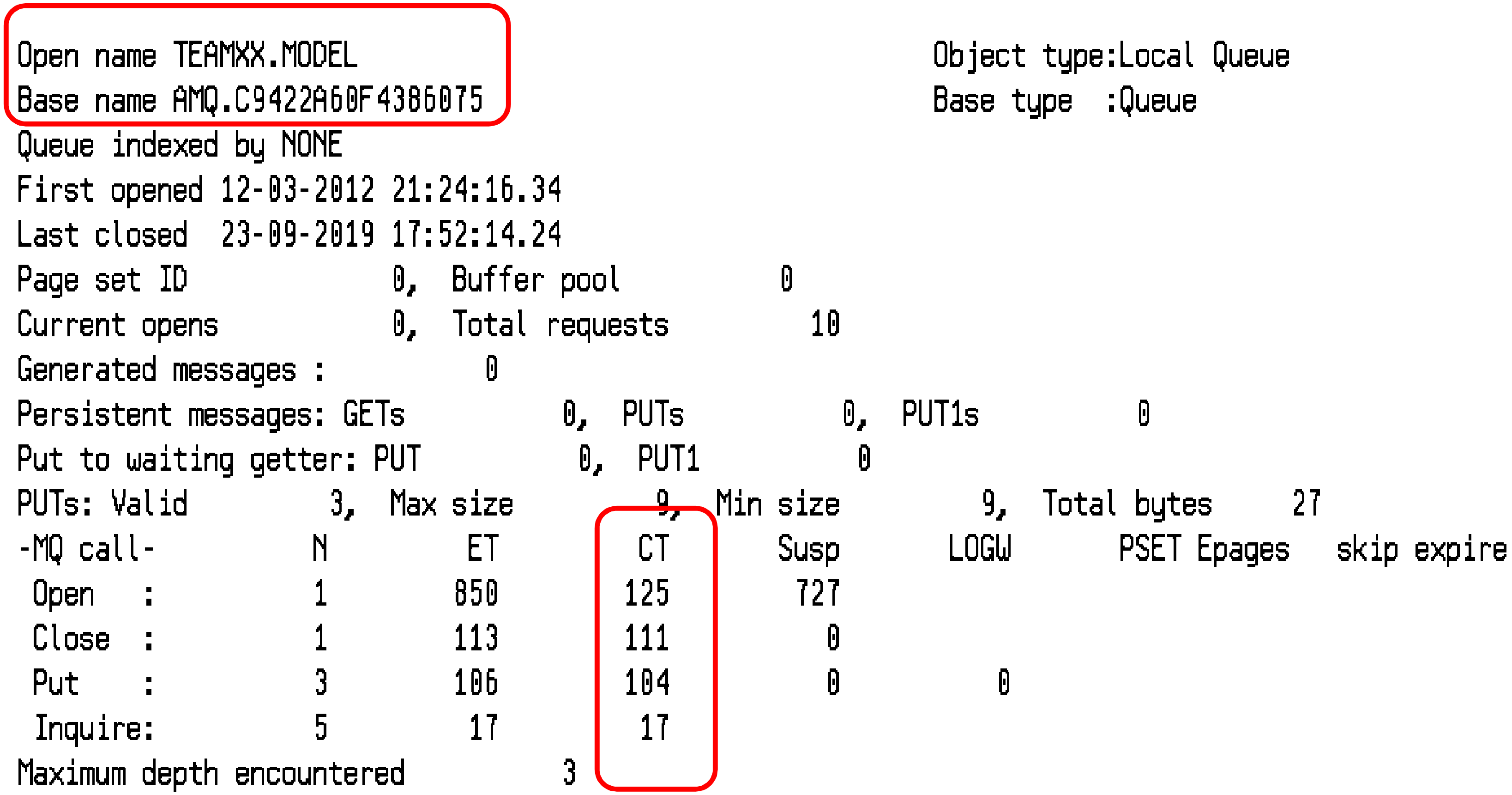
- Note this is often not seen until there is stress in one or more bufferpools due to volume.

What queues are being used and how?

- Overuse of Temporary dynamic queues
  - Often used for responses on both RYO and traditional monitoring tools
  - All queues created will be in the same resource pool
  - Quite expensive in CPU
- Temp dynamic queues are identifiable by their name
  - For example for the MQExplorer uses temporary dynamic queues. The name looks like this

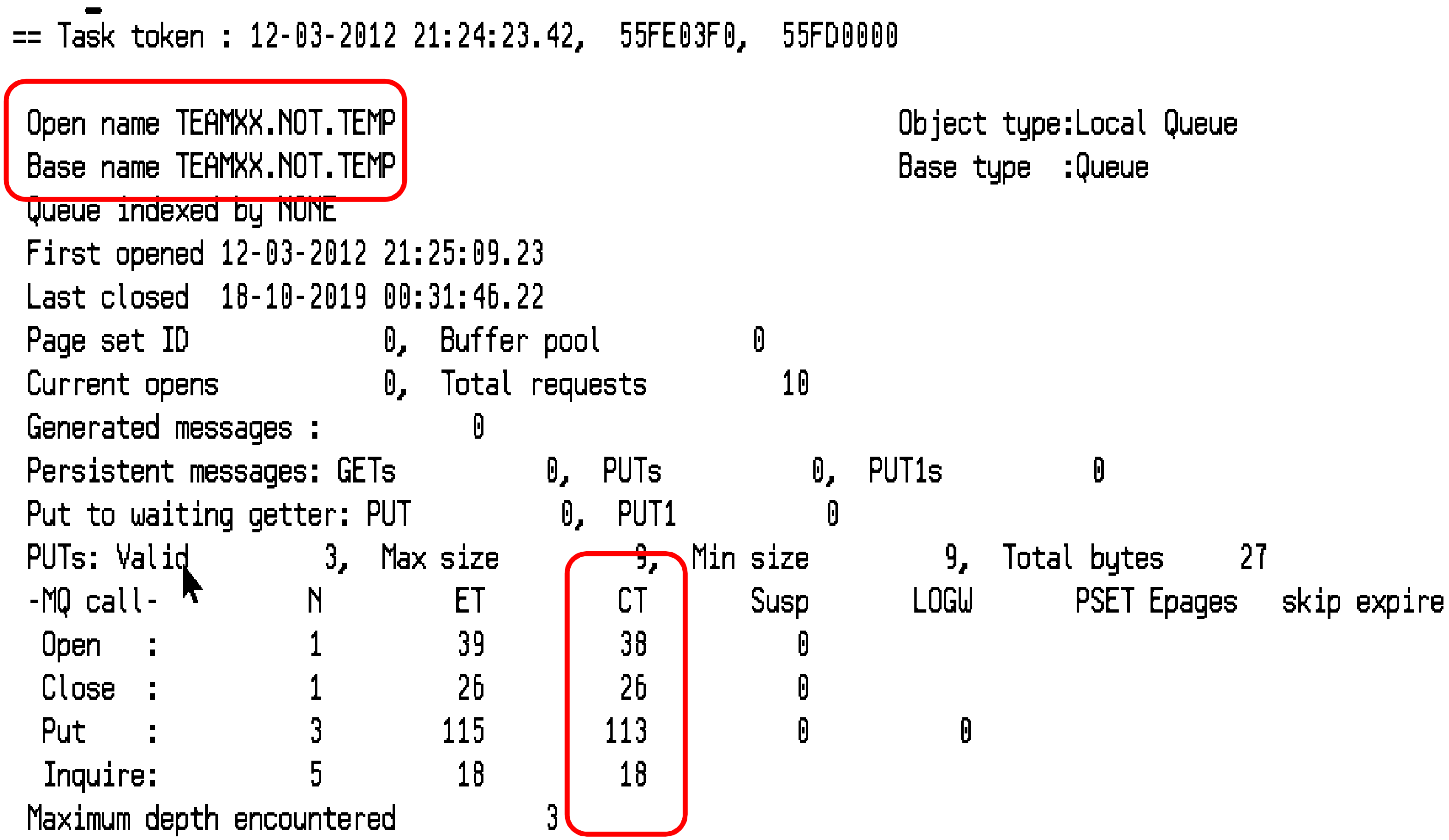
    AMQ.MQEXPLORER.1363497285

Temporary Dynamic Queues



```
Open name TEAMXX.MODEL                                     Object type:Local Queue
Base name AMQ.C9422A60F4386075                             Base type  :Queue
Queue indexed by NONE
First opened 12-03-2012 21:24:16.34
Last closed  23-09-2019 17:52:14.24
Page set ID            0,  Buffer pool         0
Current opens          0,  Total requests        10
Generated messages :         0
Persistent messages: GETs          0,  PUTs          0,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
PUTs: Valid         3,  Max size          9,  Min size          9,  Total bytes     27
-MQ call-           N          ET          CT          Susp          LOGW          PSET Epages    skip expire
 Open    :          1          850         125         727
 Close   :          1          113         111           0
 Put     :          3          106         104           0             0
 Inquire:           5           17          17
Maximum depth encountered          3
```

Permanent Queues



```
== Task token : 12-03-2012 21:24:23.42,  55FE03F0,  55FD0000

 Open name TEAMXX.NOT.TEMP                           Object type:Local Queue
 Base name TEAMXX.NOT.TEMP                           Base type  :Queue
 Queue indexed by NONE
 First opened 12-03-2012 21:25:09.23
 Last closed  18-10-2019 00:31:46.22
 Page set ID           0,  Buffer pool        0
 Current opens         0,  Total requests      10
 Generated messages :        0
 Persistent messages: GETs        0,  PUTs          0,  PUT1s         0
 Put to waiting getter: PUT        0,  PUT1         0
 PUTs: Valid        3,  Max size        9,  Min size        9,  Total bytes     27
 -MQ call-          N          ET          CT       Susp        LOGW       PSET Epages   skip expire
  Open   :          1          39          38          0
  Close  :          1          26          26          0
  Put    :          3         115         113          0          0
  Inquire:          5          18          18
 Maximum depth encountered          3
```

Temp vs. Permanent

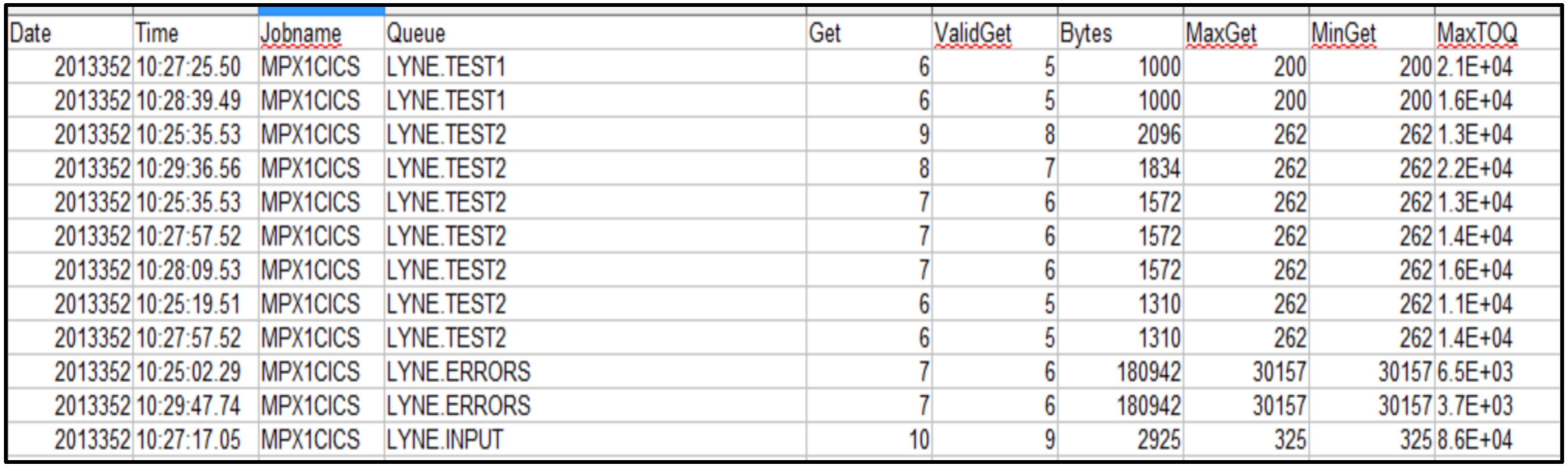
- The CPU cost comparison

Verb	TDQ	Permanent
Open	125	38
Close	111	26
Put	104	113
Inquire	17	18

- The Elapsed Time comparison

Verb	TDQ	Permanent
Open	850	39
Close	113	26
Put	106	115
Inquire	17	18

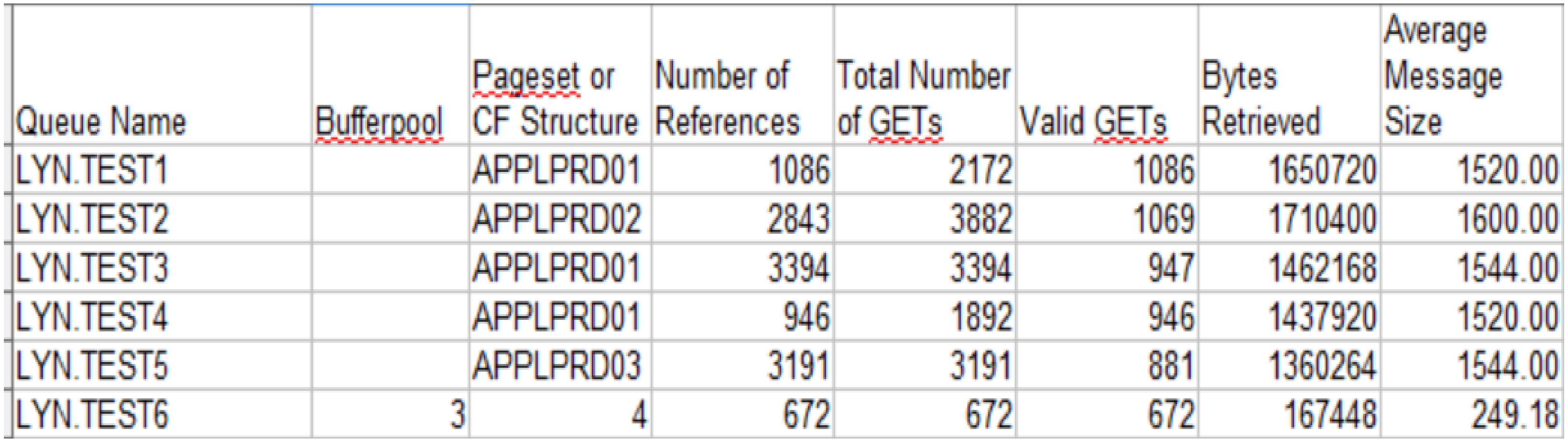
What queues are actually in use?



Date	Time	Jobname	Queue	Get	ValidGet	Bytes	MaxGet	MinGet	MaxTOQ
2013352	10:27:25.50	MPX1CICS	LYNE.TEST1	6	5	1000	200	200	2.1E+04
2013352	10:28:39.49	MPX1CICS	LYNE.TEST1	6	5	1000	200	200	1.6E+04
2013352	10:25:35.53	MPX1CICS	LYNE.TEST2	9	8	2096	262	262	1.3E+04
2013352	10:29:36.56	MPX1CICS	LYNE.TEST2	8	7	1834	262	262	2.2E+04
2013352	10:25:35.53	MPX1CICS	LYNE.TEST2	7	6	1572	262	262	1.3E+04
2013352	10:27:57.52	MPX1CICS	LYNE.TEST2	7	6	1572	262	262	1.4E+04
2013352	10:28:09.53	MPX1CICS	LYNE.TEST2	7	6	1572	262	262	1.6E+04
2013352	10:25:19.51	MPX1CICS	LYNE.TEST2	6	5	1310	262	262	1.1E+04
2013352	10:27:57.52	MPX1CICS	LYNE.TEST2	6	5	1310	262	262	1.4E+04
2013352	10:25:02.29	MPX1CICS	LYNE.ERRORS	7	6	180942	30157	30157	6.5E+03
2013352	10:29:47.74	MPX1CICS	LYNE.ERRORS	7	6	180942	30157	30157	3.7E+03
2013352	10:27:17.05	MPX1CICS	LYNE.INPUT	10	9	2925	325	325	8.6E+04

- The SMF116 class 3 data hold the information into the actual queue use
  - This information can be critical in tracking down a performance problem or for capacity planning
  - The MQCSMF GET and PUT files can be used to track the use
  - This file can contain millions of records

What is the volume on the different queues?



Queue Name	Bufferpool	Pageset or CF Structure	Number of References	Total Number of GETs	Valid GETs	Bytes Retrieved	Average Message Size
LYN.TEST1		APPLPRD01	1086	2172	1086	1650720	1520.00
LYN.TEST2		APPLPRD02	2843	3882	1069	1710400	1600.00
LYN.TEST3		APPLPRD01	3394	3394	947	1462168	1544.00
LYN.TEST4		APPLPRD01	946	1892	946	1437920	1520.00
LYN.TEST5		APPLPRD03	3191	3191	881	1360264	1544.00
LYN.TEST6	3	4	672	672	672	167448	249.18

- A spreadsheet can be used to consolidate the queues that are actually used
- In the example shown above the GET file was used to illustrate actual queue use.
    - Should be combined with the PUT information
- Note the size of the files make this better suited for a database

What queues are actually in use? Continued - Notes

- The spread sheet used a number of formula to get the totals, an example of the formula to calculate the number of GET references is: =COUNTIF(AllGets.D03:D22219;A3)
- The formula to calculate the number of Valid GETs is: =SUMIF(AllGets.$D03:$D22219;A3;AllGets.$E03:$E22219)
- The bufferpool, pageset, and CF structure information was manually drawn from the SMF 116 print program

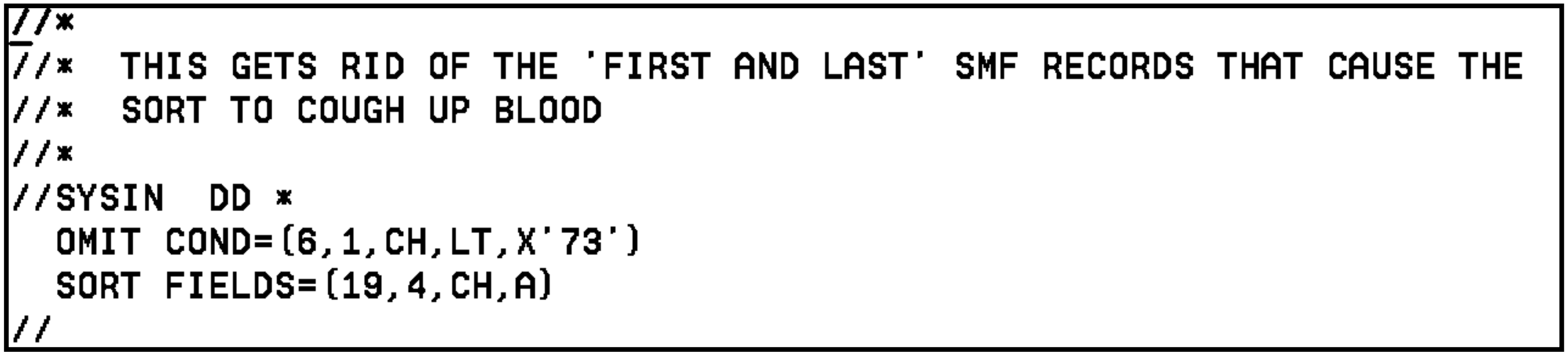
Hunting down the culprit – finding a transaction in the SMF116

- Many times you want to look at the information from a CICS transaction or batch job
  - No way to turn SMF116 class 3 on for just one TX or job
  - Use SORT
    - Remember you will have 2 passes!
      - *First pass to sort out the 'short' records that the SMFDUMP program applies*
      - *Second pass to pull out the records for the transaction/batch job you want*

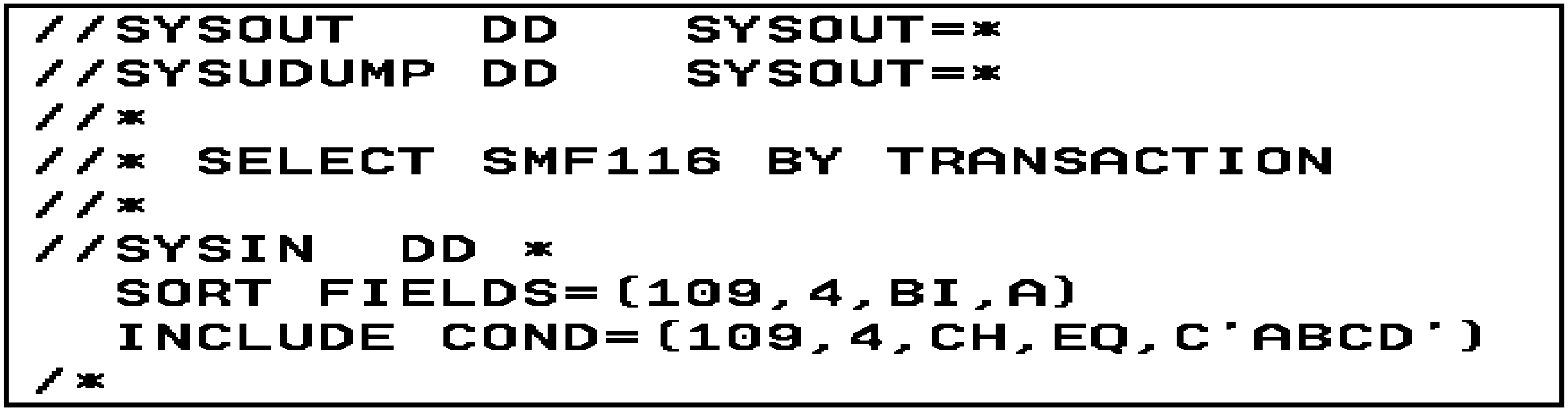
Finding a specific transaction or batch job

- In a group of millions of records, pulling the information for a specific transaction to ‘map’ it’s behavior can be critical in both problem resolution and performance issues
- The SMFDUMP program has few options for getting subsets of the data
- Using a simple sort is a quick solution to dividing up this massive volume into manageable groups

Finding a transaction

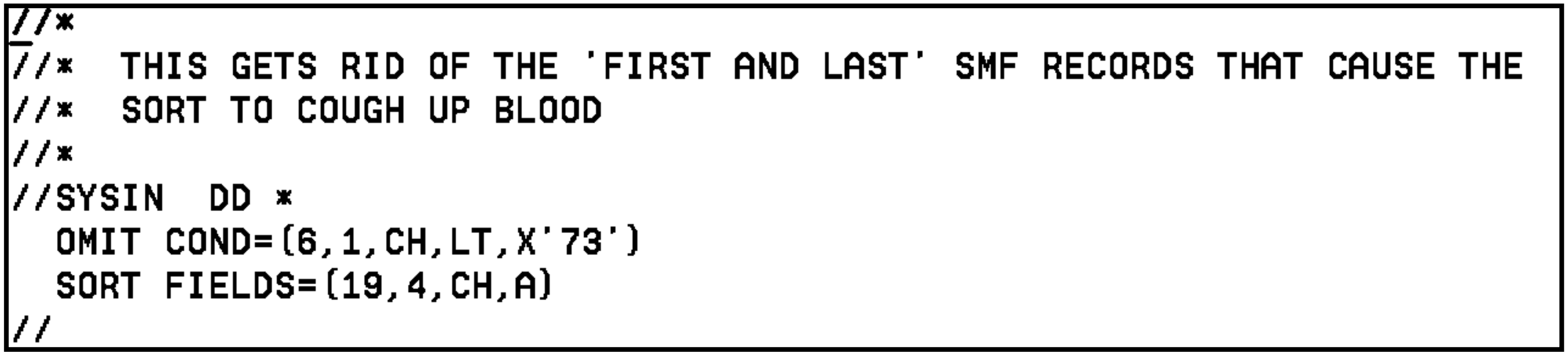


```
//*
//*  THIS GETS RID OF THE 'FIRST AND LAST' SMF RECORDS THAT CAUSE THE
//*  SORT TO COUGH UP BLOOD
//*
//SYSIN  DD *
  OMIT COND=(6,1,CH,LT,X'73')
  SORT FIELDS=(19,4,CH,A)
//
```

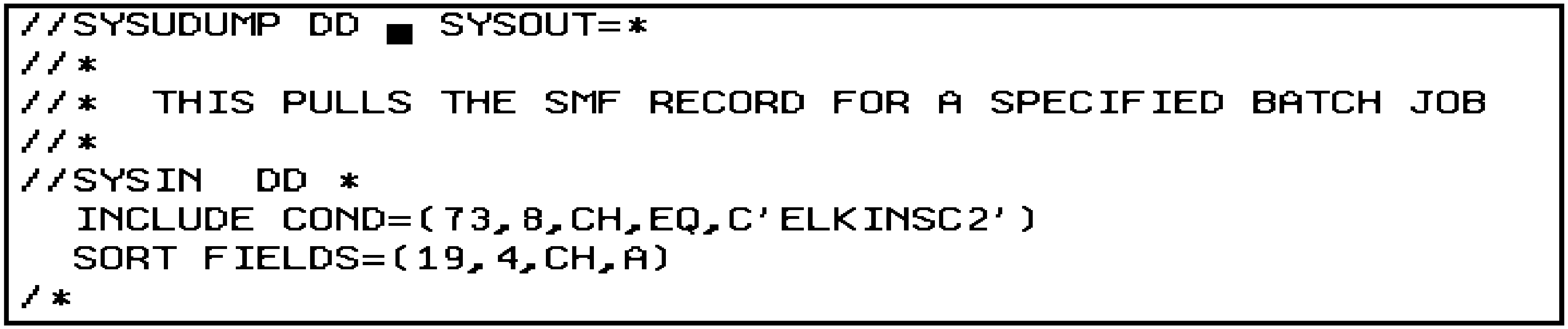


```
//SYSOUT   DD   SYSOUT=*
//SYSUDUMP DD   SYSOUT=*
//*
//* SELECT SMF116 BY TRANSACTION
//*
//SYSIN  DD *
  SORT FIELDS=(109,4,BI,A)
  INCLUDE COND=(109,4,CH,EQ,C'ABCD')
/*
```

Finding a Batch job



```
//*
//*  THIS GETS RID OF THE 'FIRST AND LAST' SMF RECORDS THAT CAUSE THE
//*  SORT TO COUGH UP BLOOD
//*
//SYSIN  DD *
  OMIT COND=(6,1,CH,LT,X'73')
  SORT FIELDS=(19,4,CH,A)
//
```



```
//SYSUDUMP DD ■ SYSOUT=*
//*
//*  THIS PULLS THE SMF RECORD FOR A SPECIFIED BATCH JOB
//*
//SYSIN  DD *
  INCLUDE COND=(73,8,CH,EQ,C'ELKINSC2')
  SORT FIELDS=(19,4,CH,A)
/*
```

SMF116 and Long running tasks

- IF the long running task is started after the Class 3 trace
  - SMF 116 records will be cut at each SMF interval and at task end
- If the task is started before the trace is
  - No records are cut
  - APAR PM58798 has been taken on this

Summary

- The SMF data can be used in many ways to find patterns of use, problems with the queue managers, and programming problems.

- There are many other things within the data that are helpful, and more to come with the 7.1 interpretations and print programs.

- Thank you

This was session 15024 - The rest of the week ……

	Monday	Tuesday	Wednesday	Thursday	Friday
08:00			What's Available in MQ and Broker for High Availability and Disaster Recovery?	Best Practices in Enhancing our Security with WebSphere MQ	MQ & CICS Workload Balancing in a 'Plexed' World
09:30				What's Wrong with MQ?	
11:00	The Dark Side of Monitoring MQ - SMF 115 and 116 Record Reading and Interpretation			IIIB - Internals of IBM Integration Bus	
12:15				Hands-on Labs for MQ - Take Your Pick!	
01:30		What's New in the MQ Family	MQ on z/OS – Vivisection	MQ Clustering - The Basics, Advances and What's New	
03:00	Introduction to MQ		WebSphere MQ CHINIT Internals	Using IBM WebSphere Application Server and IBM WebSphere MQ Together	
04:30	First Steps with IBM Integration Bus: Application Integration in the new world	What's New in IBM Integration Bus & WebSphere Message Broker	MQ & DB2 – MQ Verbs in DB2 & InfoSphere Data Replication (Q Replication) Performance	MQ Parallel Sysplex Exploitation, Getting the Best Availability From MQ on z/OS by Using Shared Queues	

You are HERE!

Shameless Promotion

IBM WebSphere MQ V7.1 and V7.5 Features and Enhancements

Maximize your investment in WebSphere MQ

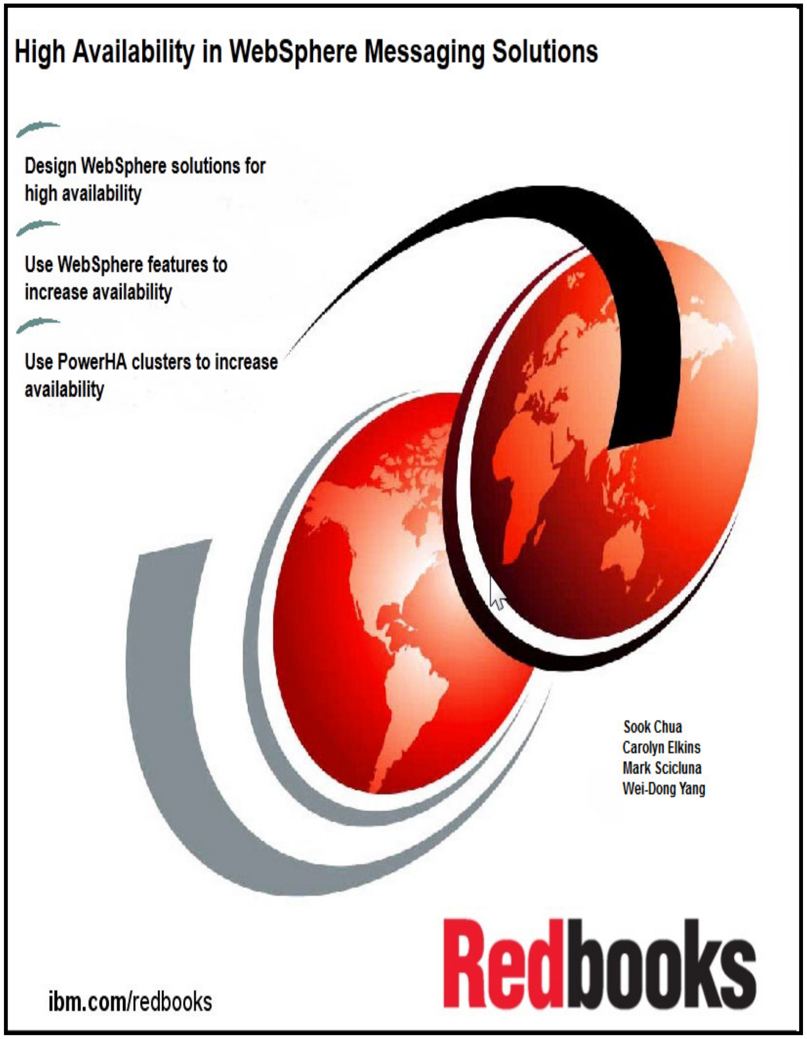
Discover new features that bring value to your business

Learn from scenarios with sample configurations

Cezar Aranha
Craig Both
Barry Dearfield
Carolyn (Lyn) Elkins
Alexander Ross
Jamie Squibb
Mark Taylor

ibm.com/redbooks

Redbooks



High Availability in WebSphere Messaging Solutions

Design WebSphere solutions for high availability

Use WebSphere features to increase availability

Use PowerHA clusters to increase availability

Sook Chua
Carolyn Elkins
Mark Scicluna
Wei-Dong Yang

ibm.com/redbooks

Redbooks

Complete your session evaluations online at www.SHARE.org/AnaheimEval

Please fill in evaluations at share.org/SFEval #15024