#SHAREorg

The Dark Side of Monitoring MQ SMF 115 and 116 Record Reading and Interpretation

Neil Johnston (neilj@uk.ibm.com)
IBM

7th February, 2013

Session Number 12610

SHARE in San Francisco 2013

Session Agenda

- Introduction to SMF 115 records
- SMF 115 records in detail
- A look at SMF 116 accounting records

SHARE in San Francisco 2013

Introduction to SMF115

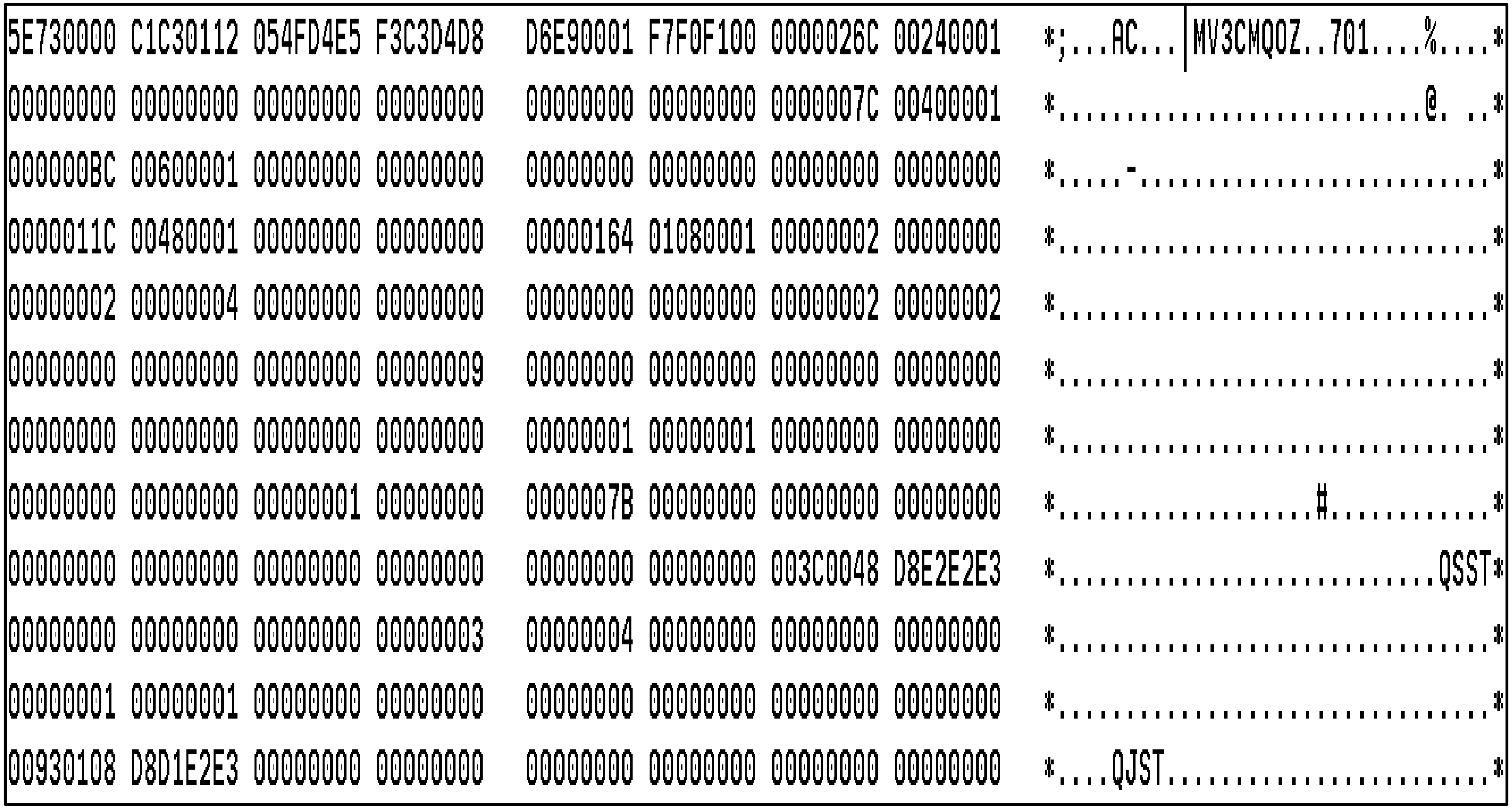
- Statistics records for the Queue Manager
- Enabled via:
  - **CSQ6SYSP** macro
    - SMFSTAT=YES
  - **START TRACE** command
    - START TRACE(STAT) DEST(SMF) CLASS(1)
- Statistics gathering interval controlled via:
  - **CSQ6SYSP** macro
    - STATIME=0¦integer (in mins) - default 30 mins
  - **SET SYSTEM** command
    - SET SYSTEM STATIME(0¦integer) (in mins)

Introduction to SMF115 cont'd

- Two records cut per SMF interval per queue manager
  - SMF 115 subtype 1
    - Storage Manager and Log Manager
  - SMF 115 subtype 2
    - Buffer Manager, Message Manager, Data Manager, CF Manager, DB2 Manager, Topic Manager, Lock Manager
- Lightweight (subtype 1 < 1K, subtype 2 < 7K)
- Negligible CPU cost
- Recommendations:
  - Always gather and examine this data
  - Useful to store for trend analysis

SHARE in San Francisco 2013

SMF 115 subtype 1 – in the raw



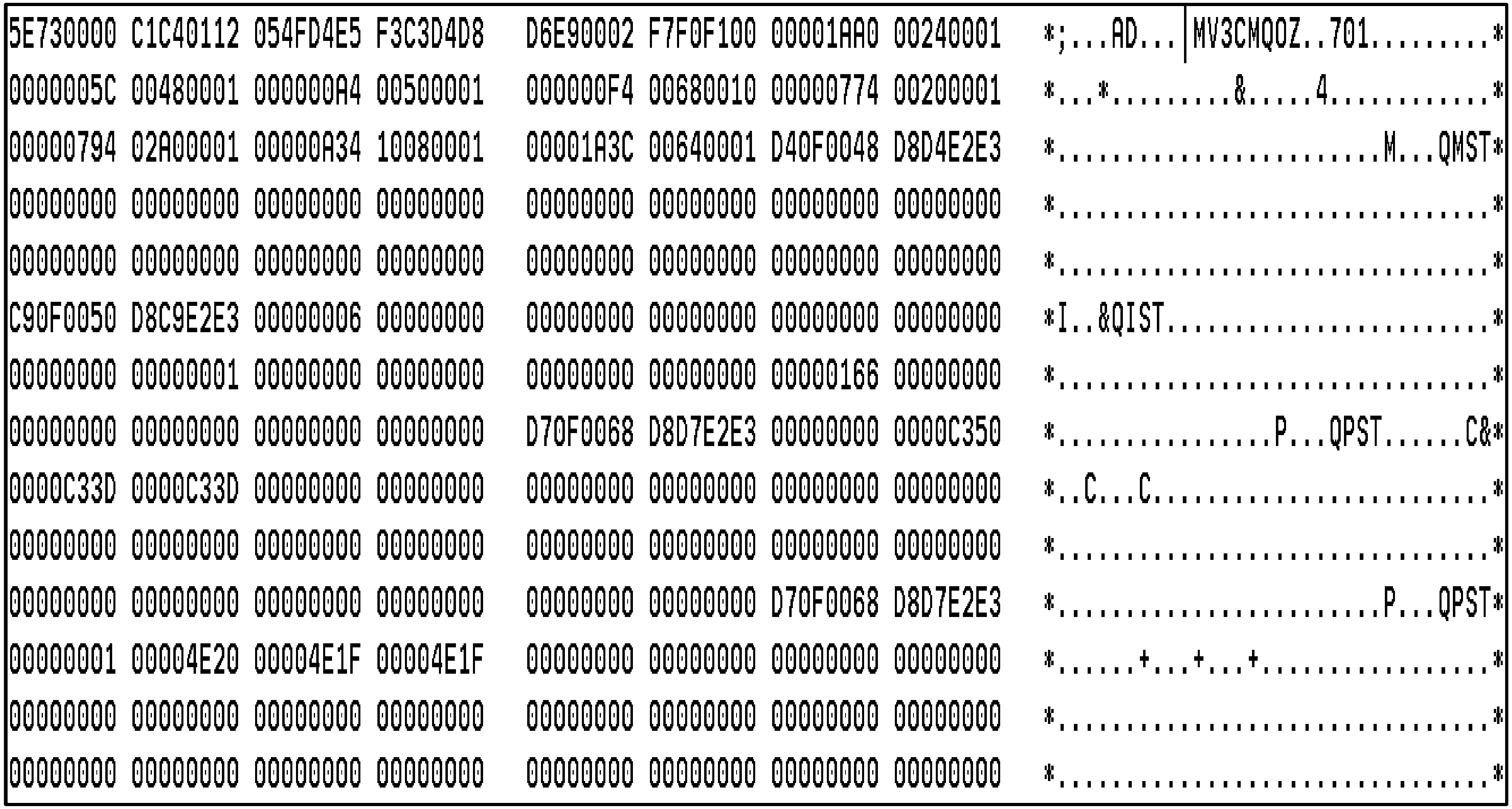
```
5E730000 C1C30112 054FD4E5 F3C3D4D8  D6E90001 F7F0F100 0000026C 00240001  *;...AC...|MV3CMQOZ..701....%....*
00000000 00000000 00000000 00000000  00000000 00000000 0000007C 00400001  *...........................@. ..*
000000BC 00600001 00000000 00000000  00000000 00000000 00000000 00000000  *.....-..........................*
0000011C 00480001 00000000 00000000  00000164 01080001 00000002 00000000  *................................*
00000002 00000004 00000000 00000000  00000000 00000000 00000002 00000002  *................................*
00000000 00000000 00000000 00000009  00000000 00000000 00000000 00000000  *................................*
00000000 00000000 00000000 00000000  00000001 00000001 00000000 00000000  *................................*
00000000 00000000 00000001 00000000  0000007B 00000000 00000000 00000000  *...................#............*
00000000 00000000 00000000 00000000  00000000 00000000 003C0048 D8E2E2E3  *............................QSST*
00000000 00000000 00000000 00000003  00000004 00000000 00000000 00000000  *................................*
00000001 00000001 00000000 00000000  00000000 00000000 00000000 00000000  *................................*
00930108 D8D1E2E3 00000000 00000000  00000000 00000000 00000000 00000000  *....QJST........................*
```

Notes:

1) IDCAMs print job used to produce this.

2) SMF data is typically written to SYS1.*.MANA and SYS1.*.MANB SMF datasets.

SMF 115 subtype 2 – in the raw



```
5E730000 C1C40112 054FD4E5 F3C3D4D8   D6E90002 F7F0F100 00001AA0 00240001   *;...AD...MV3CMQOZ..701........*
0000005C 00480001 000000A4 00500001   000000F4 00680010 00000774 00200001   *...*.........&.....4...........*
00000794 02A00001 00000A34 10080001   00001A3C 00640001 D40F0048 D8D4E2E3   *......................M...QMST*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
C90F0050 D8C9E2E3 00000006 00000000   00000000 00000000 00000000 00000000   *I..&QIST........................*
00000000 00000001 00000000 00000000   00000000 00000000 00000166 00000000   *................................*
00000000 00000000 00000000 00000000   D70F0068 D8D7E2E3 00000000 0000C350   *................P...QPST......C&*
0000C33D 0000C33D 00000000 00000000   00000000 00000000 00000000 00000000   *..C...C.........................*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
00000000 00000000 00000000 00000000   00000000 00000000 D70F0068 D8D7E2E3   *........................P...QPST*
00000001 00004E20 00004E1F 00004E1F   00000000 00000000 00000000 00000000   *......+...+...+.................*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
00000000 00000000 00000000 00000000   00000000 00000000 00000000 00000000   *................................*
```

Notes:

1) IDCAMs print job used to produce this.

2) SMF data is typically written to SYS1.*.MANA and SYS1.*.MANB SMF datasets.

Formatting and Understanding SMF 115 records

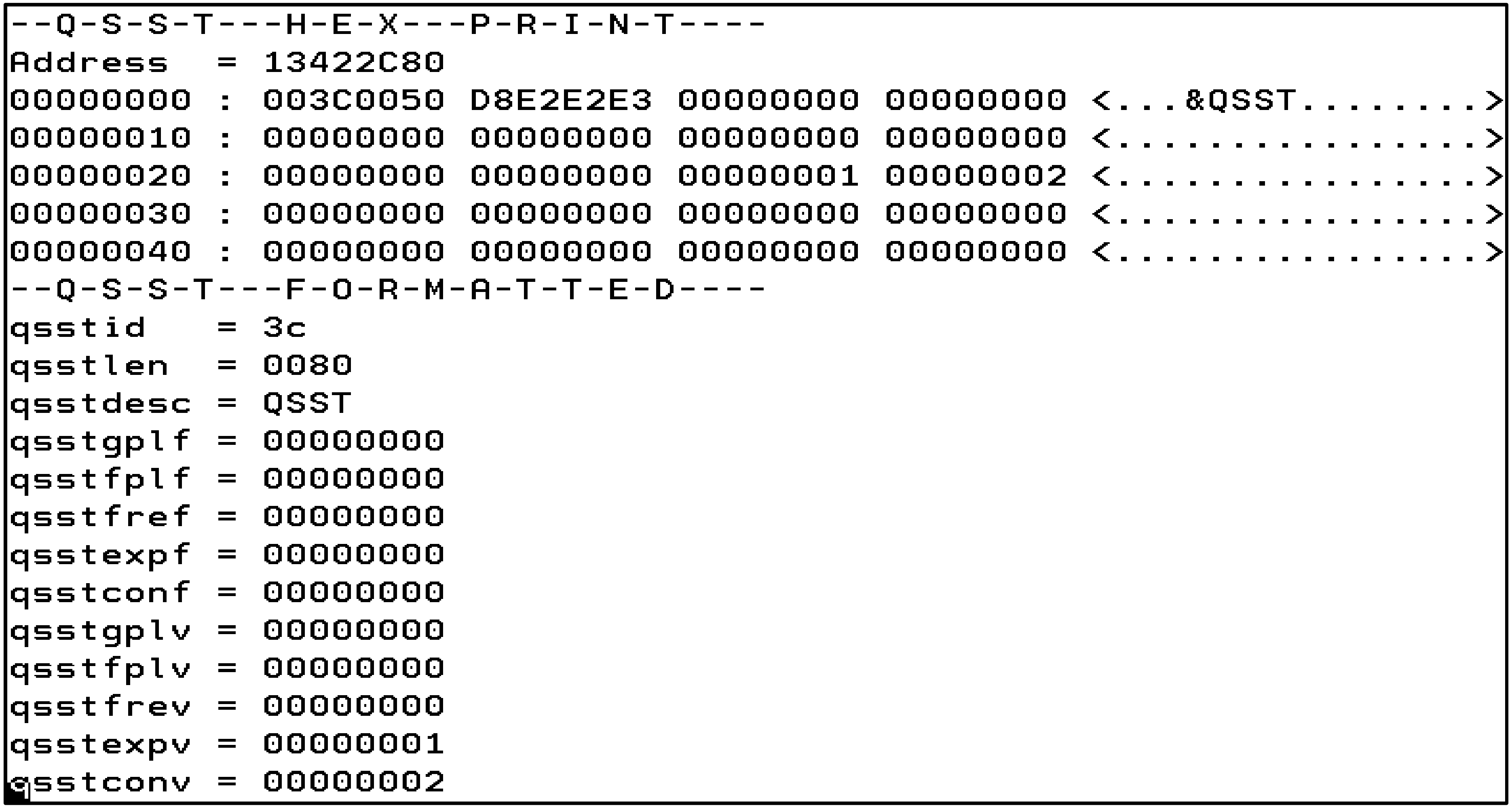
- Format of SMF 115 blocks
  - Assembler macros SCSQMACS
    - CSQDQSST – Storage Manager Statistics
    - CSQDQMST – Message Manager Statistics, etc.
  - C header file SCSQC370(CSQDSMFC)
    - Contains declarations for most statistics blocks
- CSQ4SMFD
  - Sample C program shipped with base product to print SMF 115 and 116 records 'dump style' and extract major fields

Macros for Statistics Record Formats

Resource Manager/Function	Resource Manager ID	Macro	Eyecarcher
DB2	5	CSQDQ5ST	Q5ST
Coupling Facility	E	CSQDQEST	QEST
Coupling Facility (SMDS)	E	CSQDQESD	QESD
Data Manager	I	CSQDQIST	QIST
Log Manager	J	CSQDQJST	QJST
Lock Manager	L	CSQDQLST	QLST
Message Manager	M	CSQDQMST	QMST
Buffer Manager	P	CSQDQPST	QPST
Storage Manager (getmain)	S	CSQDQSGM	QSGM
Storage Manager (pool header)	S	CSQDQSPH	QSPH
Storage Manager (region summary)	S	CSQDQSRS	QSRS
Storage Manager	S	CSQDQSST	QSST
Topic Manager	T	CSQDQTST	QTST

SHARE in San Francisco 2013

CSQ4SMFD example



```
--Q-S-S-T---H-E-X---P-R-I-N-T----
Address  = 13422C80
00000000 : 003C0050 D8E2E2E3 00000000 00000000 <...&QSST........>
00000010 : 00000000 00000000 00000000 00000000 <................>
00000020 : 00000000 00000000 00000001 00000002 <................>
00000030 : 00000000 00000000 00000000 00000000 <................>
00000040 : 00000000 00000000 00000000 00000000 <................>
--Q-S-S-T---F-O-R-M-A-T-T-E-D----
qsstid   = 3c
qsstlen  = 0080
qsstdesc = QSST
qsstgplf = 00000000
qsstfplf = 00000000
qsstfref = 00000000
qsstexpf = 00000000
qsstconf = 00000000
qsstgplv = 00000000
qsstfplv = 00000000
qsstfrev = 00000000
qsstexpv = 00000001
qsstconv = 00000002
```

Formatting and Understanding SMF 115 records cont'd

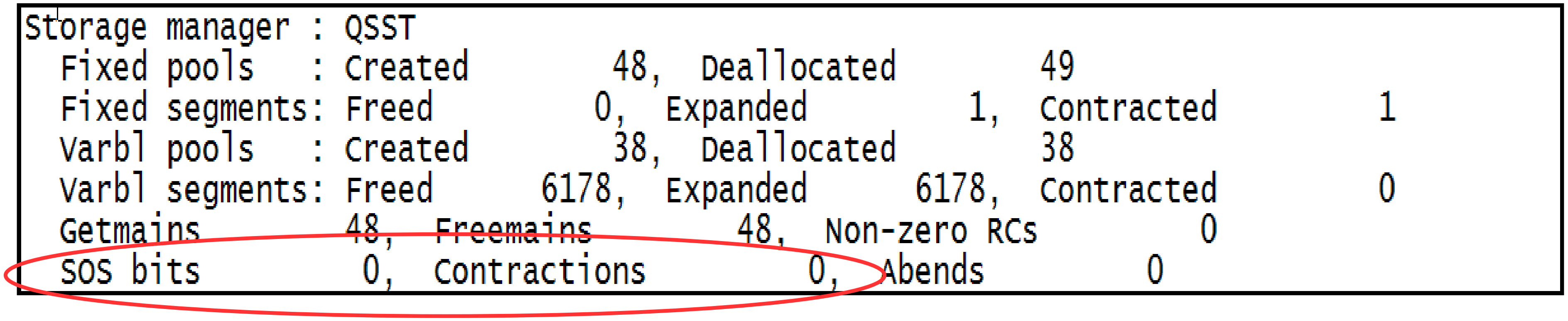
- **SupportPac MP1B** – WebSphere MQ for z/OS Interpreting Accounting and Statistics Data
  - Sample C programs to print SMF 115 and 116 records in a more easily understandable manner
  - Documentation on how to use and interpret the information
- **SupportPac MP16** – WebSphere MQ for z/OS Capacity Planning & Tuning
  - “The WMQ for z/OS handbook”

SMF 115 details

- The examples that follow show the output from program MQ1150, supplied with SupportPac MP1B

SHARE in San Francisco 2013

Storage Manager – QSST



```
Storage manager : QSST
  Fixed pools    : Created         48,  Deallocated         49
  Fixed segments: Freed           0,  Expanded            1,  Contracted           1
  Varbl pools    : Created         38,  Deallocated         38
  Varbl segments: Freed        6178,  Expanded         6178,  Contracted           0
  Getmains            48,  Freemains           48,  Non-zero RCs          0
  SOS bits             0,  Contractions         0,  Abends           0
```

- 'SOS bits' (QSSTCRIT) – count of critical short on storage conditions
- 'Contractions' (QSSTCONT) - short on storage was detected and storage contractions had to be done (below-the-bar storage)
- New for V710 – QSSTCN64 and QSSTCR64 – contractions and short on storage for above-the-bar storage (not formatted yet)
- Information not available:
  - High and low watermark storage use, both below and above the bar
  - Storage use by type (security caching, index, etc.)
    BUT new for V710 - START TRACE(STAT) CLASS(2¦3)
  - Storage used in the CHIN by clients and channels

Storage Manager

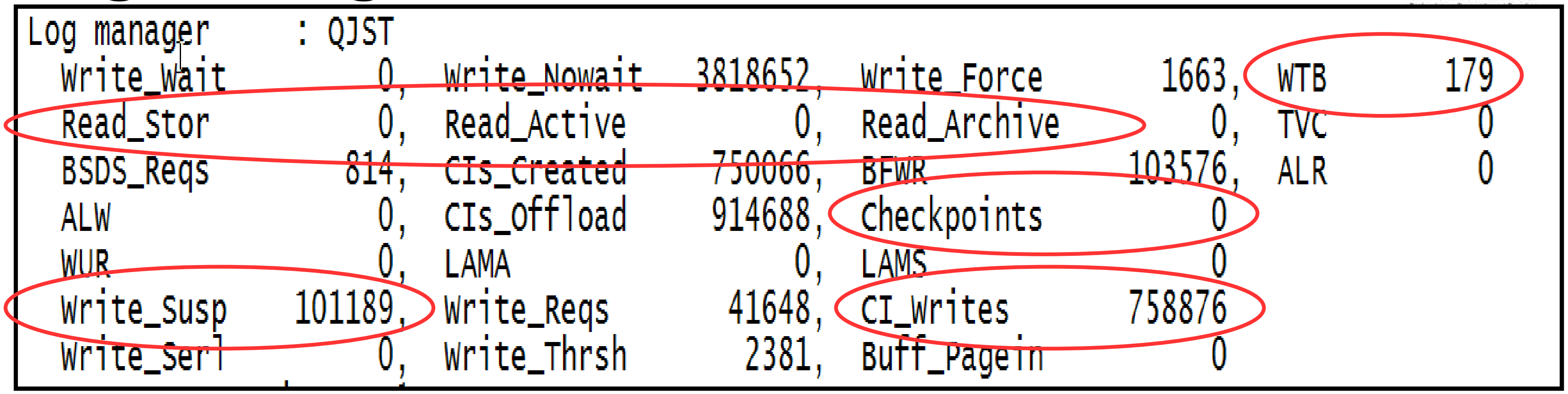
- In addition to the storage manager statistics, review the JES log for the storage use messages

```
CSQY220I QML1 Queue manger is using 627 MB of local
Storage, 1105 MB are free
```

  - If storage use keeps increasing and the free storage goes to less than 100 MB:
    - Queue manager may need to be stopped and restarted to avoid an abend soon.
    - Investigation should take place to determine why storage is not being freed.
- Information about CF structure storage use can be found in the CF activity reports

SHARE in San Francisco 2013

Log Manager – QJST



```
Log manager      : QJST
  Write_Wait          0,  Write_Nowait   3818652,  Write_Force       1663,  WTB     179
  Read_Stor           0,  Read_Active          0,  Read_Archive         0,  TVC       0
  BSDS_Reqs         814,  CIs_Created     750066,  BFWR            103576,  ALR       0
  ALW                 0,  CIs_Offload     914688,  Checkpoints          0
  WUR                 0,  LAMA                 0,  LAMS                 0
  Write_Susp     101189,  Write_Reqs       41648,  CI_Writes       758876
  Write_Serl          0,  Write_Thrsh       2381,  Buff_Pagein          0
```

- 'Read_*' fields indicate that work is being backed out
- 'Checkpoints' counts LOGLOAD checkpoints, NOT log switch checkpoints
- 'WTB' is the wait count for unavailable buffers
  - If WTB > 0, need to increase value of OUTBUFF

    (but in this stress run, OUTBUFF was already set to maximum)
- 'Write-Susp' is number of suspends as a result of log buffers being forced to the log datasets on disk due to commit processing
- 'CI_Writes' is number of 4K pages written to the logs

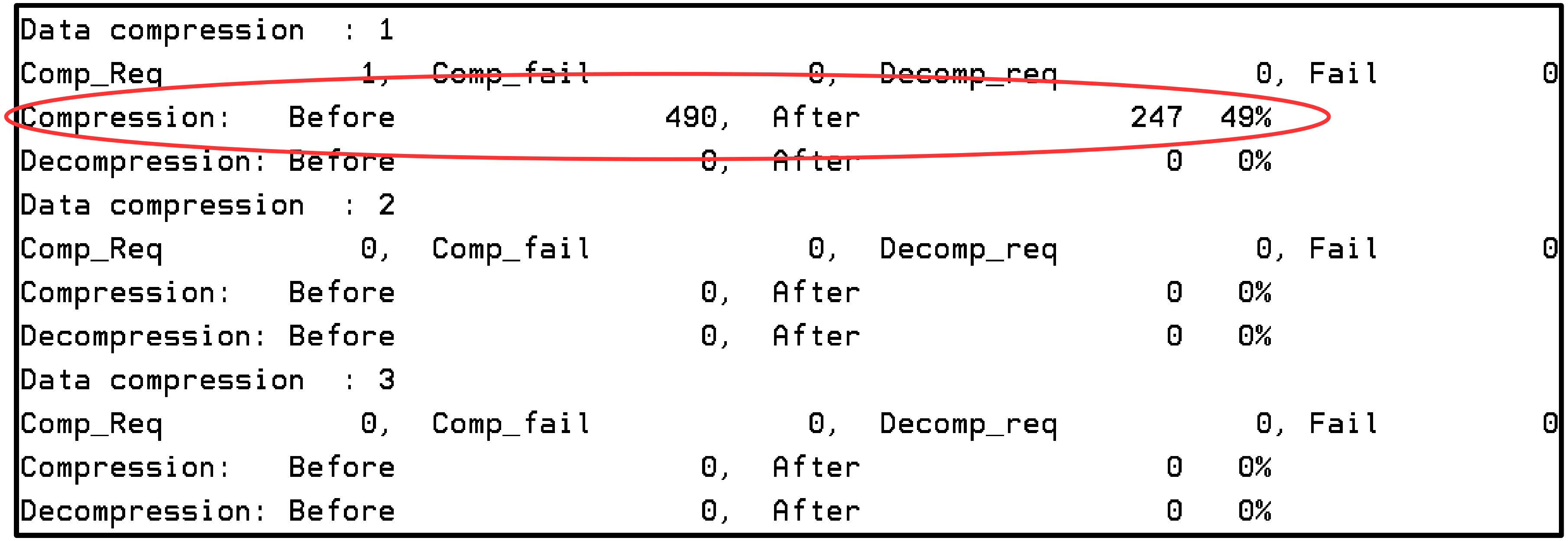
Log Manager – QJST

- Important for customers using a lot of persistent messaging, and for those who don't think they are !
- Some of the interesting fields include:
  - Checkpoint
    - Important - only includes when the LOGLOAD has been hit, not when log switching has occurred. May indicate LOGLOAD is too small.
  - Any of the Read_ fields – indicating work is being backed out
  - Wait for buffers, WTB. Increase OUTBUFF.
  - Buff_Pagein. Increase real storage or decrease OUTBUFF.
  - Write_Susp – tasks are suspended until the write completes (commit or out-of-sync (implicit syncpoint); for 2 phase commit processing, Write_Susp is incremented twice for each commit request)).
  - New for V701 – log compression performance
  - CI_Writes – number of 4K Control Intervals written (with dual logging, includes both logs)
  - Formula for calculating logging rate:

    (CI_Writes/ 256) / SMF_interval = MB/min
    (758876 / 256) / 30 = 99 MB/min
- Information not available:
  - Number of log switches / shunts / long-running UOWs

SHARE in San Francisco 2013

Log Manager – QJST cont'd



```
Data compression   : 1
Comp_Req             1,   Comp_fail           0,   Decomp_req          0, Fail          0
Compression:   Before                490,   After                 247   49%
Decompression: Before                  0,   After                   0    0%
Data compression   : 2
Comp_Req             0,   Comp_fail           0,   Decomp_req          0, Fail          0
Compression:   Before                  0,   After                   0    0%
Decompression: Before                  0,   After                   0    0%
Data compression   : 3
Comp_Req             0,   Comp_fail           0,   Decomp_req          0, Fail          0
Compression:   Before                  0,   After                   0    0%
Decompression: Before                  0,   After                   0    0%
```

- Log compression statistics (COMPLOG=RLE – run length encoding)
  - Message data compressed when writing to log (MQPUT)
  - Three separate compression sections, but only first one used at present
  - Gave > 50% saving in the amount of space used in the logs

Log Manager – QJST cont'd

- To enable LOG compression (although we call it log compression, it is actually the compression of messages before they are written to the logs), issue the following command:

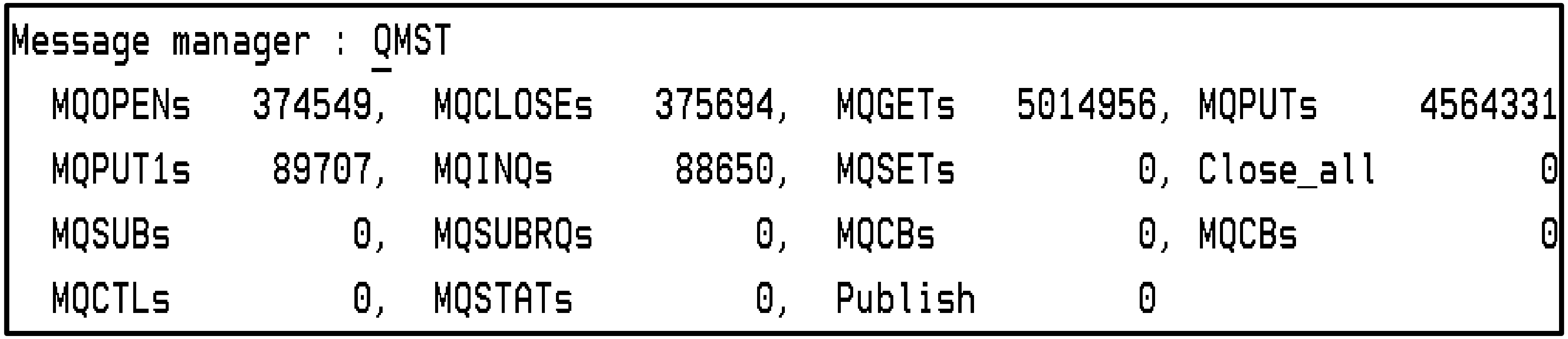
  SET LOG COMPLOG(RLE)   where RLE = Run Length Encoding

  (e.g. 0000000000 is compressed to short form representation of 10 0's)

- The MQMD has potential for compression:
  - If short queue names are in use, but field length is 48 chars
  - Queue Manager name is 4 chars but field length is 48 chars
  - Correlation id field may not be in use
  - Many other MQMD fields may have defaulted to 0 or blanks, etc.
- The actual CPU costs for log compression vary with the nature of the message data and are not readily predictable. Though note that if message compression is used, and it is necessary to recover messages from logs, the overhead of message decompression will add to the overall time it takes to recover.
  - As Log data is written in 4K chunks, 4K chunks are compressed. So, if you have a 100K message, it will be split into 25 chunks with each chunk being compressed separately.
  - In a CPU constrained system, the transaction rate may be degraded if log compression is used.
- If the message data does not contain repeating chars, it may be pointless to use log compression.
- The performance report provided in supportpac MP1G covers log compression.
- zHPF (High Performance FICON) offers a more efficient way of accessing disks and is likely to be better than to use log compression.

SHARE in San Francisco 2013

Message Manager – QMST



```
Message manager : QMST
  MQOPENs    374549,  MQCLOSEs    375694,  MQGETs    5014956, MQPUTs      4564331
  MQPUT1s     89707,  MQINQs       88650,  MQSETs          0, Close_all         0
  MQSUBs          0,  MQSUBRQs         0,  MQCBs           0, MQCBs             0
  MQCTLs          0,  MQSTATs          0,  Publish         0
```

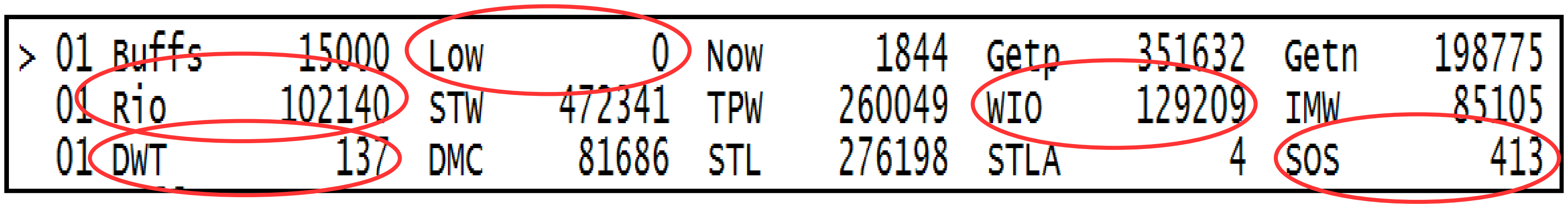
- Reports number of MQ API requests that have been made
  - NOT the number of successful requests
- Useful for tracking the overall volume of MQ API calls
  - Good initial indication of workload/workload change

SHARE in San Francisco 2013

Buffer Manager – QPST

- Often biggest bang for the buck on performance tuning
- For each bufferpool it reports:
  - The number of pages allocated
  - The ‘low’ point (number of pages available to steal in the buffer pool)
  - How the pool is used
  - Short on Storage
- What it doesn’t tell you:
  - How many pagesets use a certain buffer pool
  - Number of pages written to/read from each pageset
  - Number of pageset expansions
- There is no value in increasing the size of bufferpools for shared queues (SMDS uses a different buffer pool)

Buffer Manager – QPST cont'd



```
> 01 Buffs      15000 Low          0 Now       1844 Getp      351632 Getn     198775
  01 Rio       102140 STW     472341 TPW     260049 WIO       129209 IMW       85105
  01 DWT          137 DMC      81686 STL     276198 STLA           4 SOS         413
```

Bufferpool usage during a stress test:

- ‘low’ value of 0 (means, there were NO pages available to steal in the buffer pool during this interval)
- Asynchronous write processor started (since no. of 'dirty' pages >= 85% total pages) 137 times
- Bufferpool went short on storage (SOS) 413 times in a 5 minute interval
- 81,686 synchronous writes (since no. of 'dirty' pages >= 95% total pages)
- 102,140 reads and 129,209 writes from/to the pagesets
- JES log also had repetitions of the following messages

```
CSQP020E QML1 CSQP1RSW Buffer pool 1 is too small
CSQP020E QML1 CSQP3GET Buffer pool 1 is too small
```

SHARE in San Francisco 2013

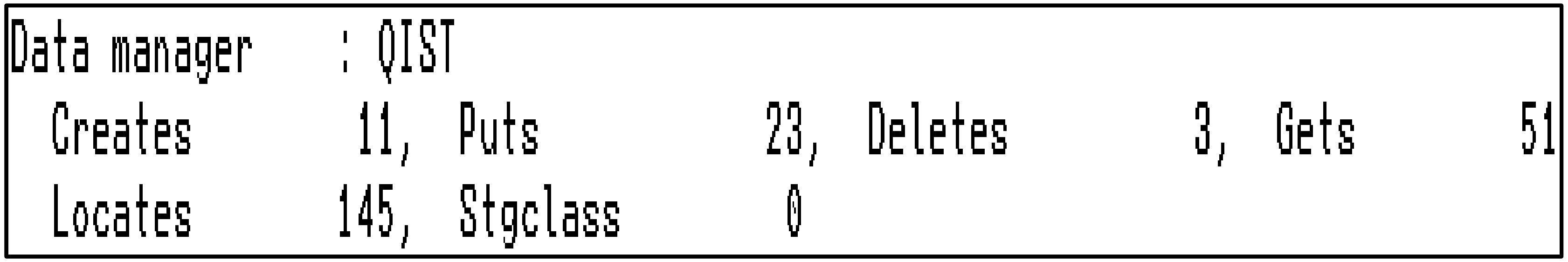
Buffer Manager – QPST cont'd

- The information in interpretation is taken from MP1B
- While this example is from a stress test, we have seen similar situations in production environments
- If the bufferpool becomes completely exhausted and nothing can be freed, the queue manager will abend with a '**00D70120**' reason code
- There is no indication of pageset expansions, that information can be obtained from the JES log, or DISPLAY USAGE command

```
CSQP017I QML1 CSQPEXT1 EXPANSION STARTED FOR PAGE SET 1
CSQP013I QML1 CSQPEXT1 NEW EXTENT CREATED FOR PAGE SET 1.
NEW EXTENT WILL NOW BE FORMATTED
```

- Dirty pages are pages in memory that have been written to (modified) and which now differ from the pages on the pagesets. Within a 3 checkpoint interval, these pages are put on a list to be written out to the pageset(s). The asynchronous write is responsible for processing the list and for ensuring that the modified pages are properly written to the pageset(s).

Data Manager – QIST



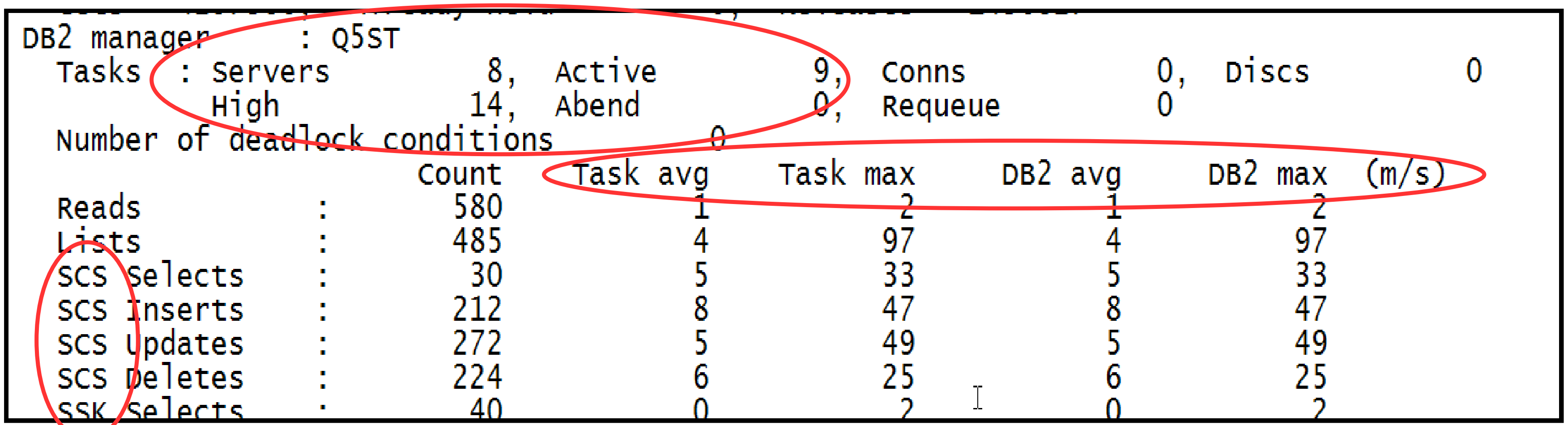
```
Data manager    : QIST
  Creates        11,  Puts             23,  Deletes          3,  Gets        51
  Locates       145,  Stgclass          0
```

- 'Creates' gives the number of objects defined
- 'Puts' gives the number of objects changed (ALTER/MQSET)
- Also provides information about the number of read ahead and gets that required real I/O. However these fields are not included in the MP1B sample SMF reports

Lock manager – QLST

- Gives counts of lock gets/releases
- The lock manager statistics are only of interest to IBM

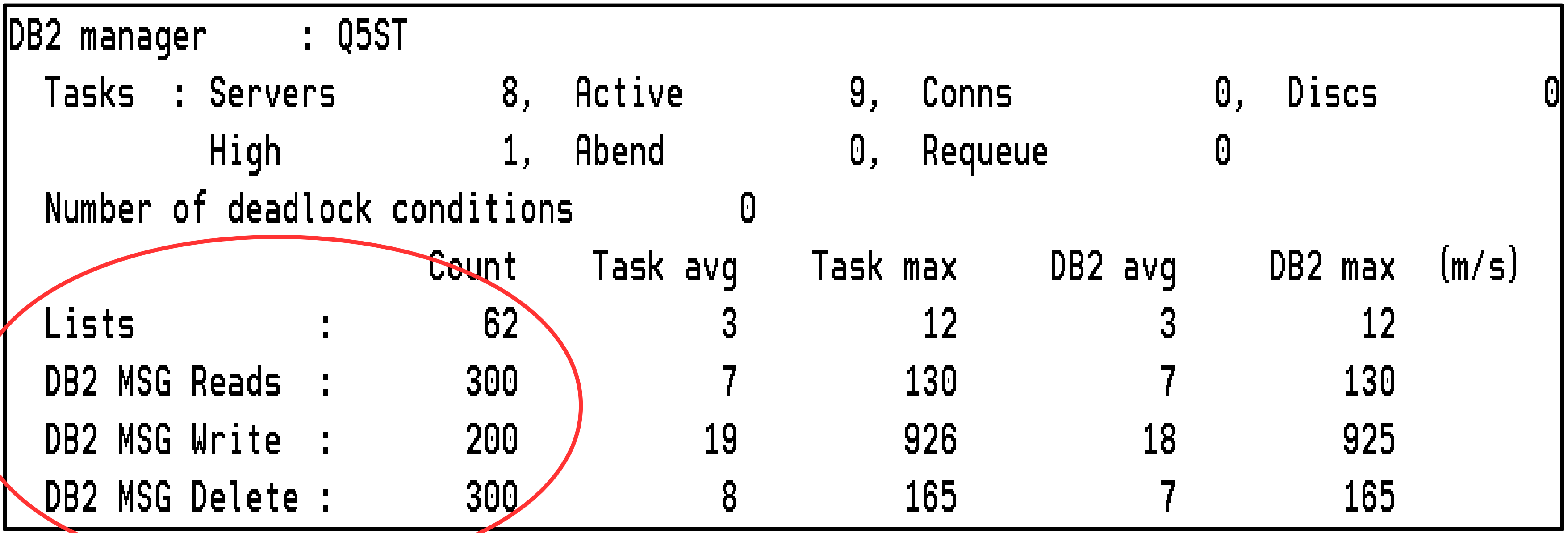
DB2 Manager – Q5ST



```
DB2 manager       : Q5ST
  Tasks  : Servers          8,  Active          9,  Conns          0,  Discs          0
           High            14,  Abend           0,  Requeue        0
  Number of deadlock conditions             0
                          Count   Task avg    Task max     DB2 avg     DB2 max   (m/s)
  Reads            :       580          1           2           1           2
  Lists            :       485          4          97           4          97
  SCS Selects      :        30          5          33           5          33
  SCS Inserts      :       212          8          47           8          47
  SCS Updates      :       272          5          49           5          49
  SCS Deletes      :       224          6          25           6          25
  SSK Selects      :        40          0           2           0           2
```

- 8 server tasks + 1 monitor task hence 9 active tasks
- ‘High’ represents the high water mark across all requests to the servers.
- 'Task avg' and 'Task max' are the average/maximum elapsed times for each request in millisecs. This includes queuing.
- 'DB2 avg' and 'DB2 max' are the average/maximum elapsed times for SQL requests. This does not include queuing.
- SCS / SSK are for shared channels (status table and keyfile)

DB2 Manager – Q5ST cont'd



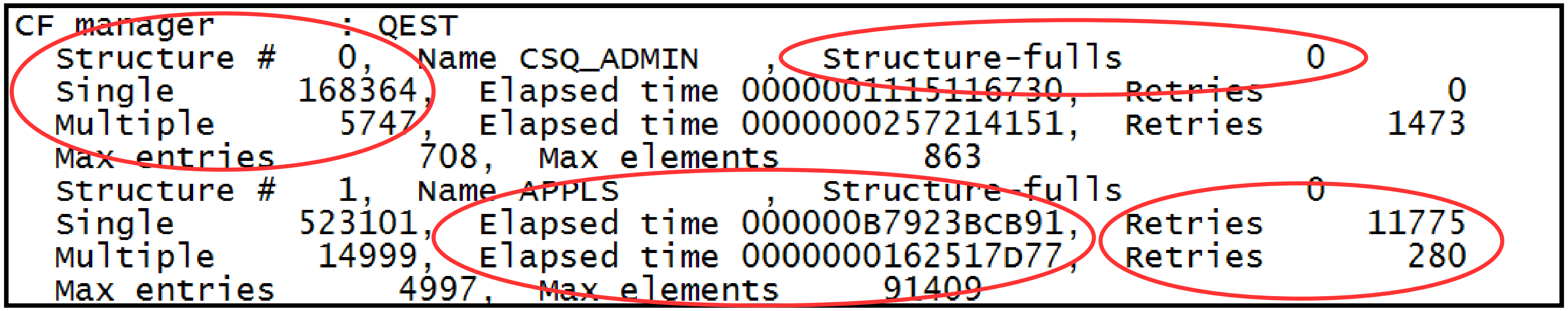
```
DB2 manager      : Q5ST
  Tasks   : Servers          8,  Active          9,  Conns            0,  Discs            0
            High             1,  Abend           0,  Requeue          0
  Number of deadlock conditions          0
                       Count    Task avg     Task max     DB2 avg      DB2 max   (m/s)
  Lists          :        62           3           12           3           12
  DB2 MSG Reads  :       300           7          130           7          130
  DB2 MSG Write  :       200          19          926          18          925
  DB2 MSG Delete :       300           8          165           7          165
```

- The above example shows large messages are being put to shared queues and off-loaded to DB2
  - DB2 MSG Write is for MQPUTs
  - DB2 MSG Read/Delete are for MQGETs

DB2 Manager – Q5ST

- Only used when in a queue-sharing group
- Is used to report on the queue manager interaction with DB2
- DB2 response time can impact the WMQ response times (MQOPEN/MQPUT/MQGET) and should be monitored
- Use in conjunction with DB2 performance reports
- High number of Lists – could be due to DISPLAY QLOCAL commands (from monitoring tool perhaps)

CF Manager – QEST



```
CF manager          : QEST
  Structure #    0,   Name CSQ_ADMIN    ,  Structure-fulls        0
  Single       168364,   Elapsed time 000000111511673O,  Retries           0
  Multiple       5747,   Elapsed time 000000025721415l,  Retries        1473
  Max entries           708,   Max elements          863
  Structure #    1,   Name APPLS        ,  Structure-fulls        0
  Single       523101,   Elapsed time 000000B7923BCB91,  Retries       11775
  Multiple      14999,   Elapsed time 000000162517D77,  Retries         280
  Max entries          4997,   Max elements        91409
```

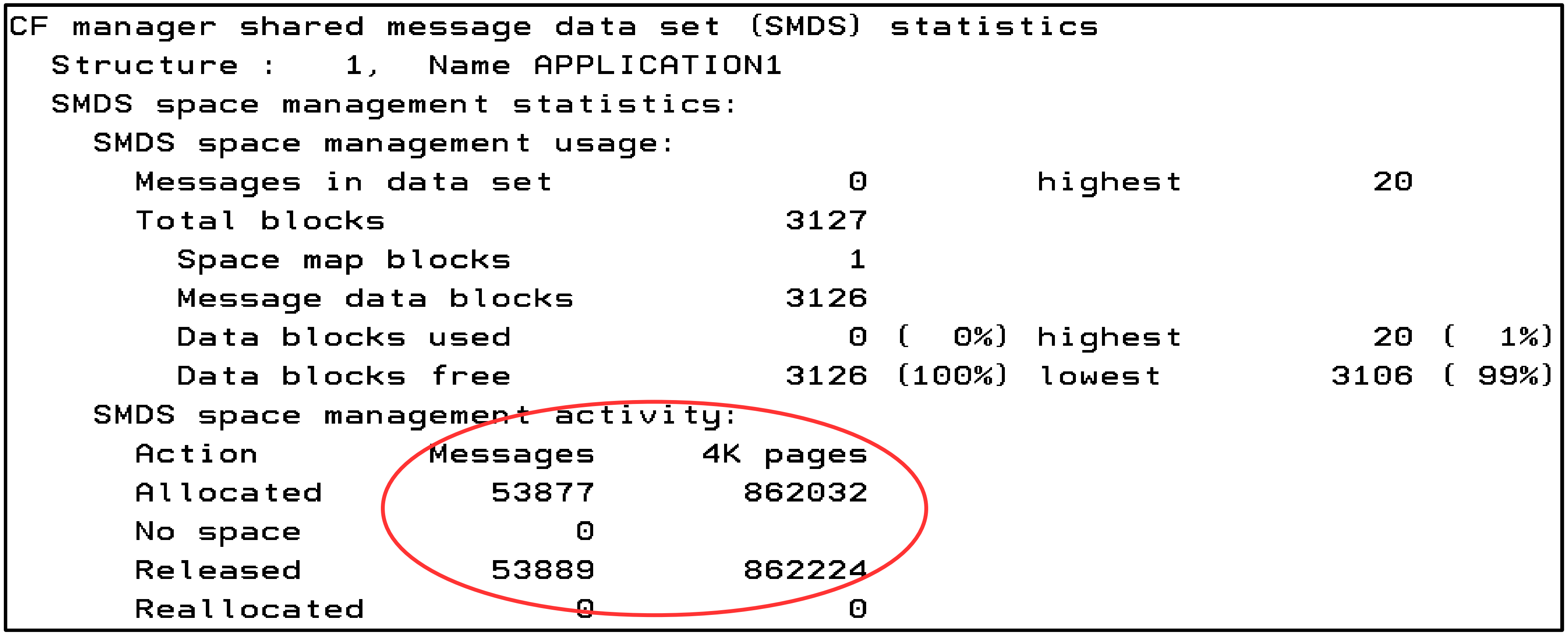
- In the sample above there were no Structure full conditions
- Requests to the CF can be to update a single entry or multiple entries, based on the type of request. They are reported separately in the statistics.
- ‘Retries’ indicates the number of times a 4K buffer was not sufficient to retrieve the data from the CF and the request had to be retried with a larger (64K) buffer or CF timed-out a request
- 'Elapsed time' is total, in hex (STCK units so divide by decimal 4096 to convert to microseconds; i.e. ignore last 3 hex digits)

CF Manager – QEST

- The CF Manager data
  - Only used when in a queue-sharing group
  - Is used to report on the interaction with the CF structures
  - Should be used in conjunction with the CF Activity Report

CF Manager – QESD (SMDS)

V7.1



```
CF manager shared message data set (SMDS) statistics
  Structure :    1,  Name APPLICATION1
  SMDS space management statistics:
    SMDS space management usage:
      Messages in data set              0        highest          20
      Total blocks                   3127
        Space map blocks                1
        Message data blocks          3126
        Data blocks used                0 (  0%) highest          20 (  1%)
        Data blocks free             3126 (100%) lowest         3106 ( 99%)
    SMDS space management activity:
      Action          Messages      4K pages
      Allocated          53877        862032
      No space               0
      Released           53889        862224
      Reallocated            0             0
```

- One QESD per CFSTRUCT
- Above details show usage of local SMDS dataset
- 862032 / 52877 = 16*4K pages = 64KB of storage per message
- Above is sneak preview as MP1B not yet updated for QESD

SHARE in San Francisco 2013

CF Manager – QESD (SMDS)

V7.1

- When off-loading to SMDS, the data written to the CF structure includes:
  - The number of the queue manager in the QSG (1-32)
  - A list of blocks of where the message is in the SMDS
  - MQMD
- The actual message payload is written to SMDS of course.
- The number of messages released is 12 more than the number allocated because the 12 messages are from the previous statistics interval. That is they were put and accounted for during the previous statistics interval but they were got during this statistics interval.

CF Manager – QESD (SMDS) cont'd

V7.1

```
SMDS buffer pool statistics:
  SMDS buffer pool usage:
    Buffer size (DSBLOCK)           64K
    Total buffers                   12
    Buffers in use                   0 (  0%)  highest     10 ( 83%)
    Buffers free                    12 (100%)  lowest       0 (  0%)
      Saved buffers                  0
      Empty buffers                 12
    Waiting request queues
      For free buffer                0         highest      1
      For busy buffer                0         highest      0
```

- SMDS buffer pool is in 64-bit storage
- There were times when there were no free buffers
  - Need to increase DSBUFS
- Above is sneak preview as MP1B not yet updated for QESD

SHARE in San Francisco 2013

CF Manager – QESD (SMDS) cont'd

V7.1

```
SMDS buffer pool activity:
  Acquired buffers                   107742
    Got valid buffer                        26910 ( 24%)
    Got matching, empty buffer                  0 (  0%)
    Got free, empty buffer                  33270 ( 30%)
    Stole a saved buffer                    47562 ( 44%)
  No buffer available                     0
  Waited for free buffer                  0 (  0%) avg time 0.000000s
  Waited for busy buffer                  0 (  0%) avg time 0.000000s
  Buffer read issued                  53869
    Data already valid                26910 ( 49%)
    Data partly valid                     0 (  0%)
    Data read from disk               26959 ( 50%)
  Freed valid buffer                 107750
  Marked buffer deleted               33280
  Buffer write issued                 53881
```

- Half the messages were retrieved from buffer pool
- Above is sneak preview as MP1B not yet updated for QESD

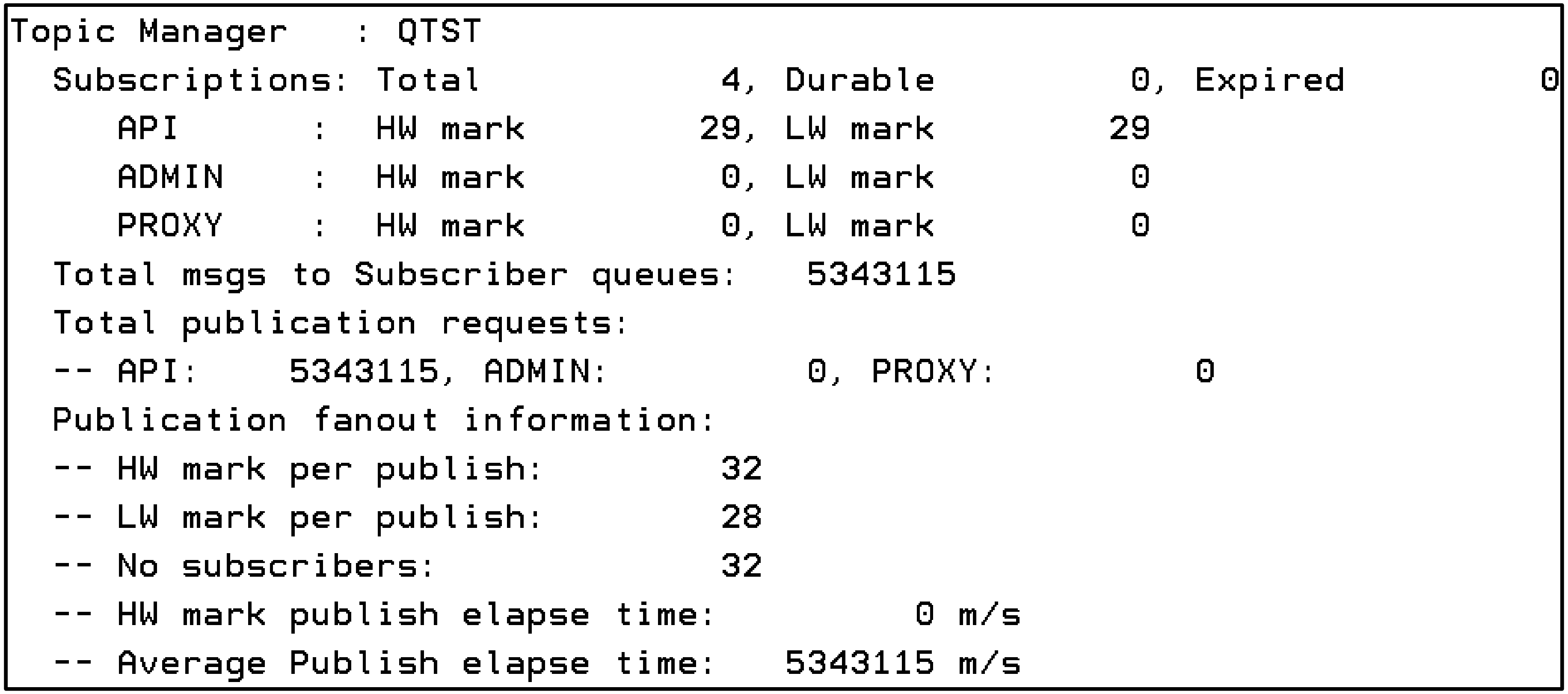
CF Manager – QESD (SMDS) cont'd

V7.1

```
SMDS I/O statistics:
  SMDS data set usage:
    High allocated CI              50040
    High formatted CI              50040
    Control interval size           4096
    Control area size             737280
  SMDS I/O activity:
    Type              Requests    4K pages      avg I/O time      avg wait time
    Format                   0           0         0.000000s          0.000000s
    Write                53881      862096         0.000753s          0.000748s
    Read (local)         26959      431344         0.000619s          0.000615s
    Read (Other)             0           0         0.000000s          0.000000s
```

- No 'Read (Other)' as all messages put and got on the same queue-manager
- Above is a sneak preview as MP1B is not yet updated for QESD

Topic Manager – QTST



```
Topic Manager    : QTST
  Subscriptions: Total           4, Durable         0, Expired         0
     API       :  HW mark       29, LW mark        29
     ADMIN     :  HW mark        0, LW mark         0
     PROXY     :  HW mark        0, LW mark         0
  Total msgs to Subscriber queues:   5343115
  Total publication requests:
  -- API:    5343115, ADMIN:         0, PROXY:         0
  Publication fanout information:
  -- HW mark per publish:        32
  -- LW mark per publish:        28
  -- No subscribers:             32
  -- HW mark publish elapse time:         0 m/s
  -- Average Publish elapse time:   5343115 m/s
```

- Details on pub/sub usage
- 'HW mark publish elapse time' and 'Average Publish elapse time' are incorrect – bug in MQ1150 reported!

SHARE in San Francisco 2013

Topic Manager – QTST

- Subscriptions:
  - API MQSUB
  - ADMIN DEFINE SUB
  - PROXY internal (routing publications through a queue manager)

SHARE in San Francisco 2013

SMF 115 subtype 7

Storage Manager Region Summary – QSRS

V7.1

- START TRACE(STAT) DEST(SMF) CLASS(1,2)

• QSRSLOAL	-	< 16M USER Region alloc value
• QSRSOLOAL	-	previous value of QSRSLOAL
• QSRSELOAL	-	> 16M USER Region alloc value
• QSRSOELOAL	-	previous value of QSRSELOAL
• QSRSGBYTES	-	high water mark for number of usable bytes of above-bar storage
• QSRSAVAL	-	amount of free 31-bit storage
• QSRSAVAL64	-	amount of free above-bar storage

SMF 115 subtype 5, 6 and 7
Storage Manager Pool Header Statistics – QSPH
Storage Manager Getmain Statistics – QSGM
Storage Manger Region Summary – QSRS

V7.1

- START TRACE(STAT) DEST(SMF) CLASS(1,3)
- Also gives QSRS (subtype 7)

- SMF 115 subtype 5
  - One QSPH per storage pool
  - Shows current and previous size of pool

- SMF 115 subtype 6
  - One QSGM per module/offset
  - Shows internal getmains by module/offset

- Of most interest to IBM Support

SMF 115 recap

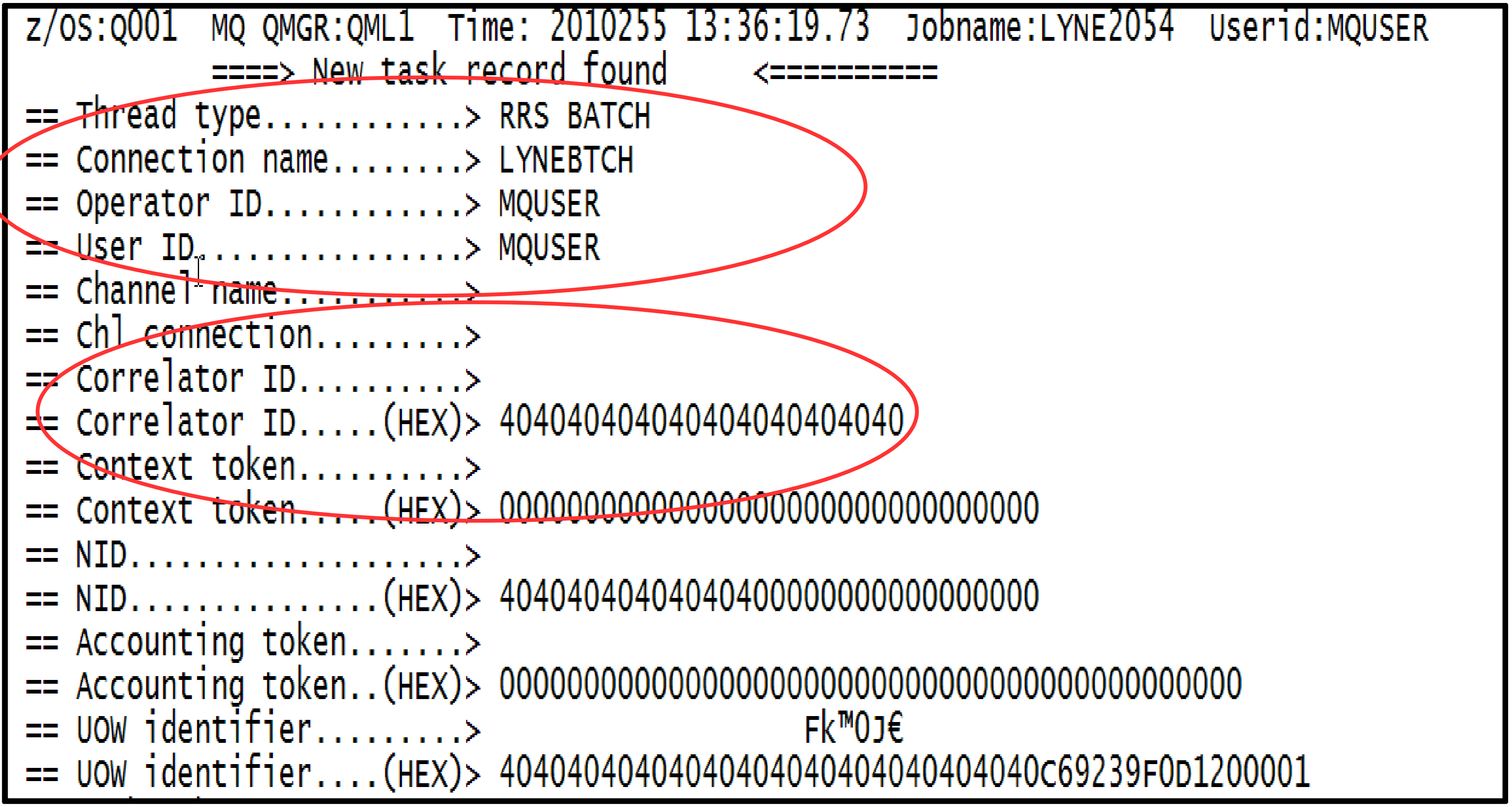
- Number of checkpoints (QJSTLLCP)
- Reads from active/archive logs (QJSTRACT/QJSTRARH) - indicative of backouts
- Amount of data being logged (QJSTCIWR)
- Is log buffer big enough (QJSTWTB)
- How effective is log compression (QJSTCmpUncmp/QJSTCmpComp)
- Number of MQ verbs being issued
    - e.g. Number of MQGETs (QMSTGET)
- Buffer pool usage
    - Is buffer pool being stressed (QPSTDWT)
- Queuing on DB2 server TCBs (Q5ST.DHIGMAX)

SHARE in San Francisco 2013

Intro to SMF116 – Class 3, subtypes 1, 2

- START TRACE(ACCTG) DEST(SMF) CLASS(3)
- Control at queue-level with ACCTQ
- Costs 5-10% CPU overhead
- Heavyweight – multiple records may be cut for each transaction, and at SMF intervals for long running UoWs
  - Turning this on has been known to swamp an SMF environment
  - But you get marvelous information about what is actually happening
  - Often used in tracking down an application problem, and in performance tuning
- Task/thread and queue statistics
- Time values are generally in microsecs or millisecs (where stated)
- Recommendation - Even though they are prolific:
  - At least once a month turn on class 3 accounting for one SMF interval
  - Become familiar with the data and with the patterns of WMQ usage

SMF116 – Task/Thread Identification – WTID



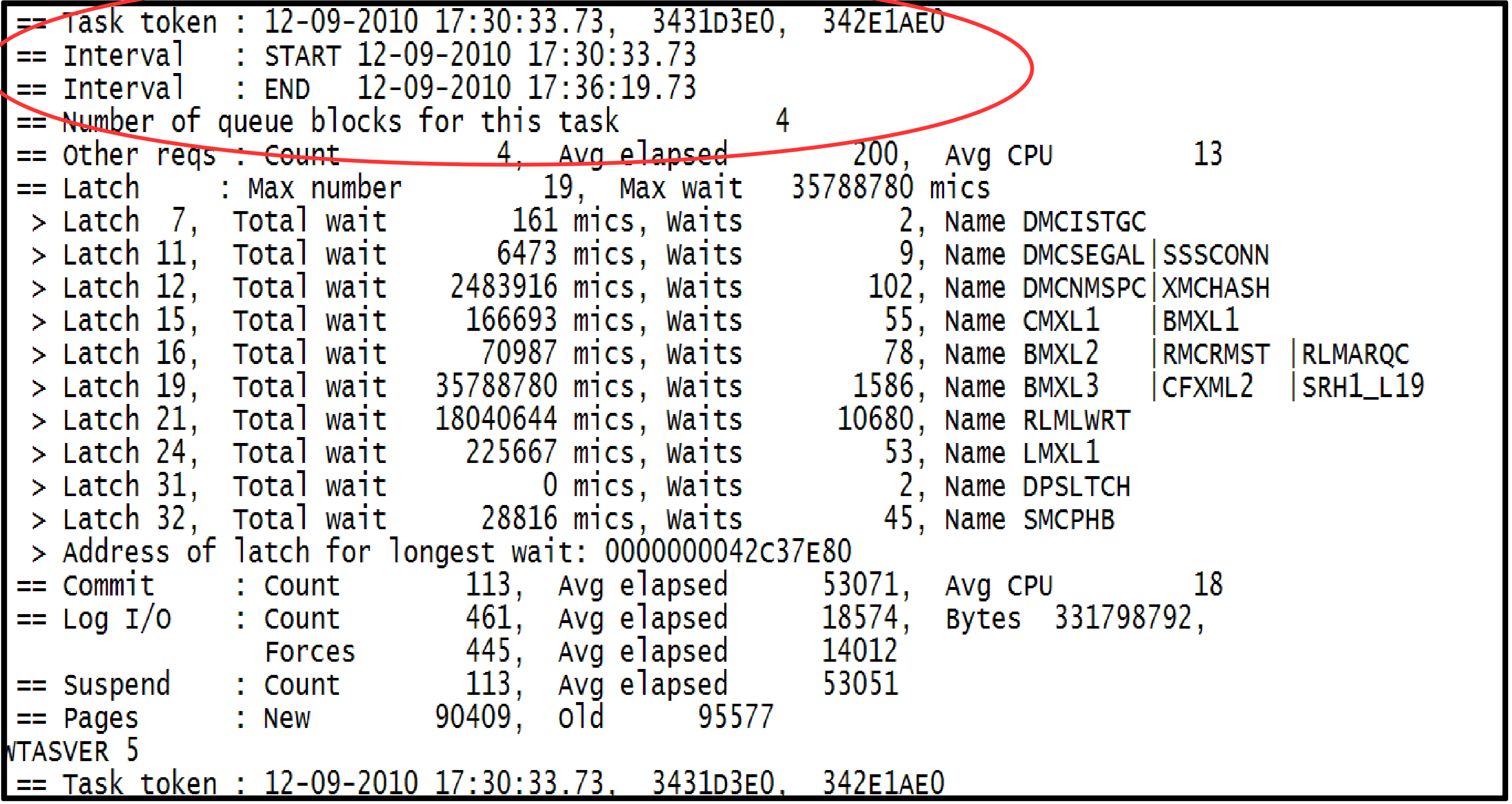
```
z/OS:Q001  MQ QMGR:QML1  Time: 2010255 13:36:19.73  Jobname:LYNE2054  Userid:MQUSER
           ====> New task record found      <==========
== Thread type...........> RRS BATCH
== Connection name.......> LYNEBTCH
== Operator ID...........> MQUSER
== User ID...............> MQUSER
== Channel name..........>
== Chl connection........>
== Correlator ID.........>
== Correlator ID.....(HEX)> 404040404040404040404040
== Context token.........>
== Context token.....(HEX)> 00000000000000000000000000000000
== NID...................>
== NID...............(HEX)> 40404040404040400000000000000000
== Accounting token......>
== Accounting token..(HEX)> 0000000000000000000000000000000000000000000000
== UOW identifier........>                  Fk™0J€
== UOW identifier....(HEX)> 404040404040404040404040404040C69239F0D1200001
```

SMF116 – Task/Thread Identification – WTID

- The Thread type gives you information about the task, in this case it's a batch process. It may also be CHINIT (for channels), CICS and IMS
- Connection name is the jobname
- The channel name will be present when this is a CHINIT thread
- The correlator ID is NOT the MQMD correlation ID
  - If the SMF data is for a CICS transaction, it will contain the transaction ID.

    e.g. The transaction ID for this record is QPUB and the taskid is 43219:
    - == Correlator ID..........>   .®.ÇQPUB.
    - == Correlator ID.....(HEX)> 20AF4B68D8D7E4C20043219C

SMF116 – Task related statistics - WTAS



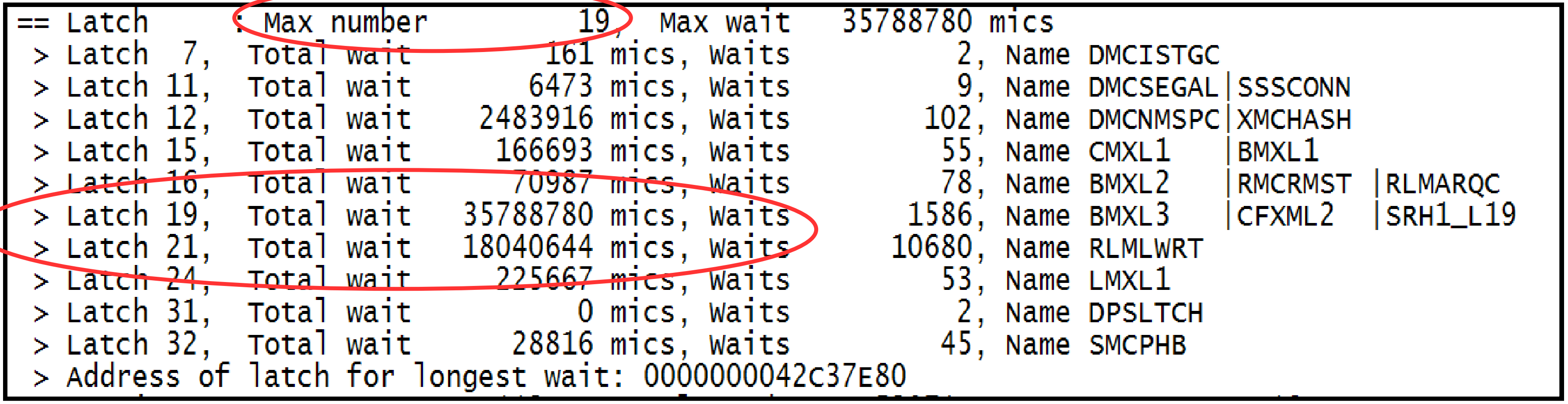
```
== Task token : 12-09-2010 17:30:33.73,  3431D3E0,  342E1AE0
== Interval   : START 12-09-2010 17:30:33.73
== Interval   : END   12-09-2010 17:36:19.73
== Number of queue blocks for this task          4
== Other reqs : Count            4,  Avg elapsed        200,  Avg CPU         13
== Latch      : Max number         19,  Max wait   35788780 mics
 > Latch  7,  Total wait       161 mics, Waits            2, Name DMCISTGC
 > Latch 11,  Total wait      6473 mics, Waits            9, Name DMCSEGAL|SSSCONN
 > Latch 12,  Total wait   2483916 mics, Waits          102, Name DMCNMSPC|XMCHASH
 > Latch 15,  Total wait    166693 mics, Waits           55, Name CMXL1    |BMXL1
 > Latch 16,  Total wait     70987 mics, Waits           78, Name BMXL2    |RMCRMST |RLMARQC
 > Latch 19,  Total wait  35788780 mics, Waits         1586, Name BMXL3    |CFXML2  |SRH1_L19
 > Latch 21,  Total wait  18040644 mics, Waits        10680, Name RLMLWRT
 > Latch 24,  Total wait    225667 mics, Waits           53, Name LMXL1
 > Latch 31,  Total wait         0 mics, Waits            2, Name DPSLTCH
 > Latch 32,  Total wait     28816 mics, Waits           45, Name SMCPHB
 > Address of latch for longest wait: 0000000042C37E80
== Commit     : Count          113,  Avg elapsed      53071,  Avg CPU         18
== Log I/O    : Count          461,  Avg elapsed      18574,  Bytes  331798792,
                Forces         445,  Avg elapsed      14012
== Suspend    : Count          113,  Avg elapsed      53051
== Pages      : New          90409,  Old        95577
WTASVER 5
== Task token : 12-09-2010 17:30:33.73,  3431D3E0,  342E1AE0
```

- Task token identifies the task
- Task was long running as it ran for just under 6 mins
- Number of queue blocks indicates number of queues accessed

SMF116 – Task related statistics - WTAS

- Task token is the task identifying information
- Since this is a long running task, the interval start and end information may be of interest
- The queue blocks gives you the number of queues that have been accessed
- The Commit Count indicates the number of Commit requests
- Then there’s the latches………

SMF116 – Task related statistics – Latching – WTAS



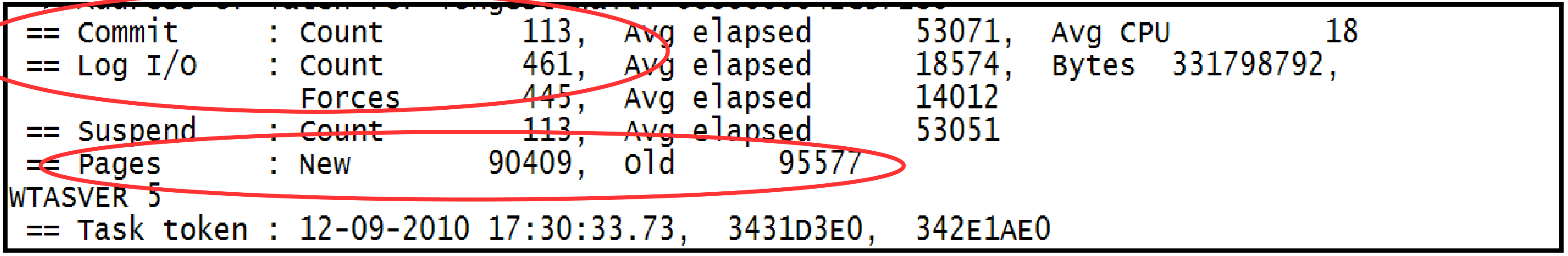
```
== Latch      : Max number          19   Max wait   35788780 mics
 > Latch  7,  Total wait        161 mics, Waits           2, Name DMCISTGC
 > Latch 11,  Total wait       6473 mics, Waits           9, Name DMCSEGAL|SSSCONN
 > Latch 12,  Total wait    2483916 mics, Waits         102, Name DMCNMSPC|XMCHASH
 > Latch 15,  Total wait     166693 mics, Waits          55, Name CMXL1   |BMXL1
 > Latch 16,  Total wait      70987 mics, Waits          78, Name BMXL2   |RMCRMST |RLMARQC
 > Latch 19,  Total wait   35788780 mics, Waits        1586, Name BMXL3   |CFXML2  |SRH1_L19
 > Latch 21,  Total wait   18040644 mics, Waits       10680, Name RLMLWRT
 > Latch 24,  Total wait     225667 mics, Waits          53, Name LMXL1
 > Latch 31,  Total wait          0 mics, Waits           2, Name DPSLTCH
 > Latch 32,  Total wait      28816 mics, Waits          45, Name SMCPHB
 > Address of latch for longest wait: 0000000042c37E80
```

- Latching is performed to serialize requests within the queue manager
- There is always latching going on
  - But there are times when it gets a bit excessive, and needs to be investigated
  - This is one of those times

SMF116 – Task related statistics – Latching – WTAS

- The 'Max number' is really the latch type that showed the longest wait, in this case latch type 19
- Latch types may be used for multiple purposes
- MP1B has a list of some of the more typical entries, latch 19 is used for serialization to bufferpools
- Latch 21, the second largest wait count, is used when updating log buffers.
- Using these numbers, and looking at the JES message log for the queue manager indicates that during this interval there were numerous log switches and one of the bufferpools expanded
- Further investigation uncovered I/O subsystem issues – the logs and the pagesets were on the same devices for this environment, leading to significant contention

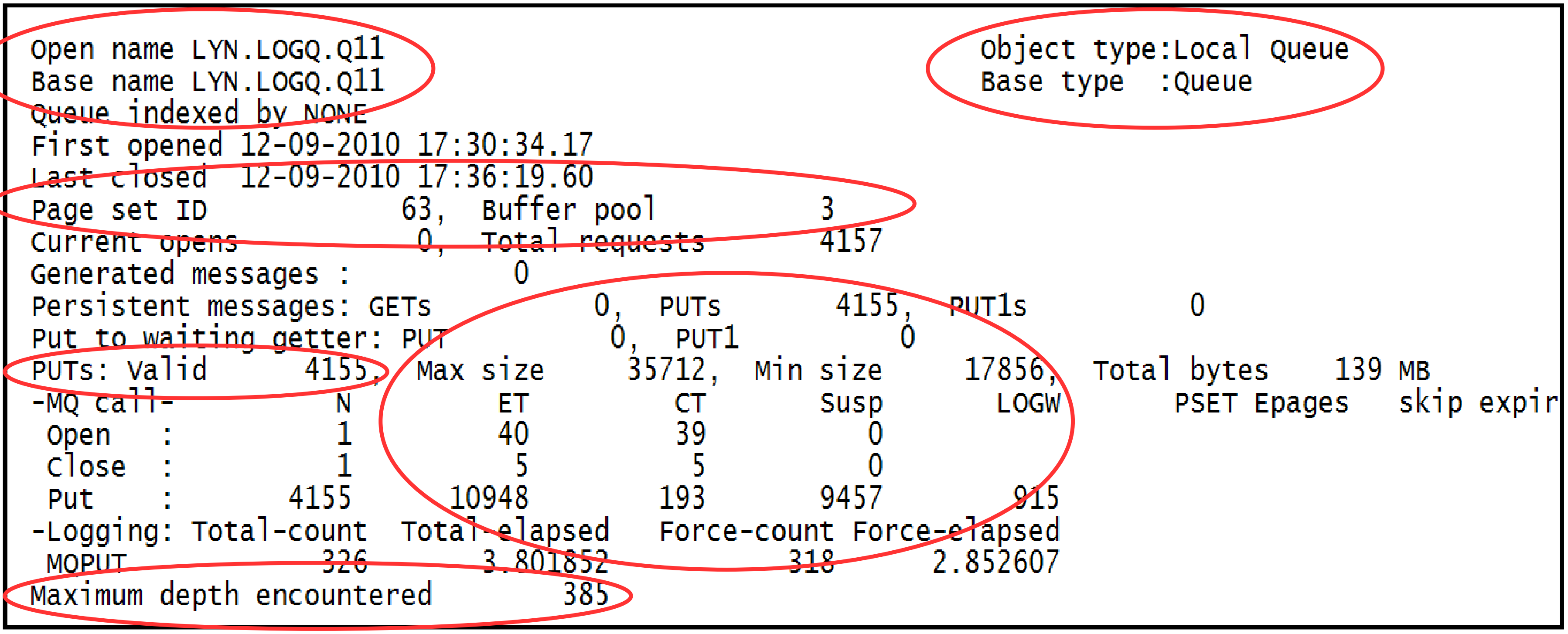
SMF116 – Task related statistics cont'd - WTAS



```
 == Commit      : Count          113,  Avg elapsed      53071,  Avg CPU          18
 == Log I/O     : Count          461,  Avg elapsed      18574,  Bytes   331798792,
                  Forces         445,  Avg elapsed      14012
 == Suspend     : Count          113,  Avg elapsed      53051
 == Pages       : New          90409,  Old       95577
WTASVER 5
 == Task token : 12-09-2010 17:30:33.73,  3431D3E0,   342E1AE0
```

- The commit count is useful, especially when working with long running tasks
- The ‘Pages’ values show how many new and old buffer pages have been used during this interval by this task

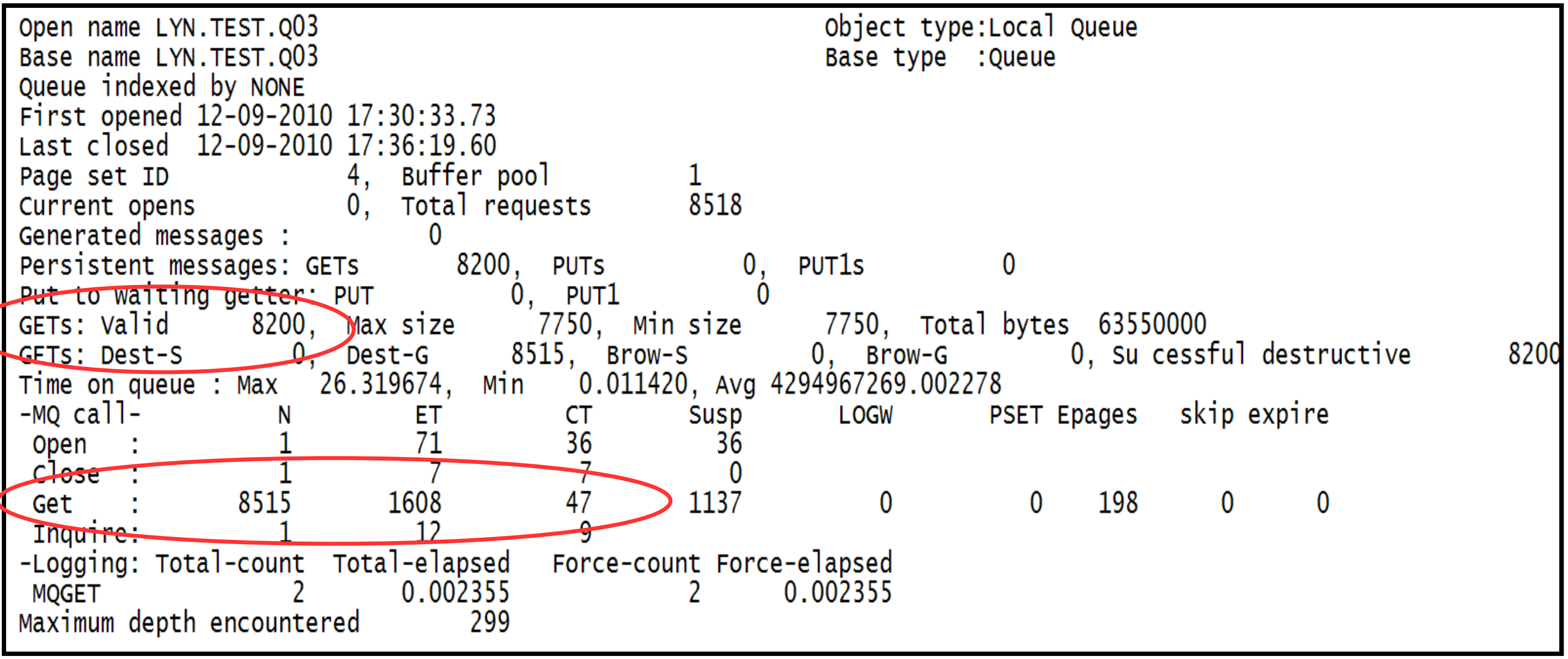
SMF116 – Queue Accounting statistics – WQST



```
Open name LYN.LOGQ.Q11                                       Object type:Local Queue
Base name LYN.LOGQ.Q11                                       Base type  :Queue
Queue indexed by NONE
First opened 12-09-2010 17:30:34.17
Last closed  12-09-2010 17:36:19.60
Page set ID              63,  Buffer pool        3
Current opens             0,  Total requests     4157
Generated messages :          0
Persistent messages: GETs           0,  PUTs        4155,  PUT1s          0
Put to waiting getter: PUT           0,  PUT1          0
PUTs: Valid        4155,  Max size      35712,  Min size      17856,  Total bytes     139 MB
-MQ call-            N          ET          CT          Susp        LOGW       PSET Epages    skip expir
 Open   :            1          40          39            0
 Close  :            1           5           5            0
 Put    :         4155       10948         193         9457          915
-Logging: Total-count  Total-elapsed   Force-count Force-elapsed
 MQPUT           326       3.801852          318       2.852607
Maximum depth encountered           385
```

- Detailed information about the queue’s used by a task, including:
  - Pageset and bufferpool
  - Number of valid put requests
  - Record size range, you can calculate the average size
  - Total elapsed time and cpu time for the requests
    - N = number or calls, ET = Elapsed time, CT = CPU time
  - Maximum depth

SMF116 – Queue Accounting statistics – WQST



```
Open name LYN.TEST.Q03                                       Object type:Local Queue
Base name LYN.TEST.Q03                                       Base type  :Queue
Queue indexed by NONE
First opened 12-09-2010 17:30:33.73
Last closed  12-09-2010 17:36:19.60
Page set ID            4,  Buffer pool        1
Current opens          0,  Total requests     8518
Generated messages :         0
Persistent messages: GETs        8200,  PUTs           0,  PUT1s          0
Put to waiting getter: PUT          0,  PUT1          0
GETs: Valid       8200,  Max size       7750,  Min size      7750,  Total bytes  63550000
GETs: Dest-S         0,  Dest-G       8515,  Brow-S          0,  Brow-G          0, Su cessful destructive       8200
Time on queue : Max   26.319674,  Min    0.011420, Avg 4294967269.002278
-MQ call-            N          ET         CT        Susp        LOGW       PSET Epages   skip expire
 Open   :            1          71         36          36
 Close  :            1           7          7           0
 Get    :         8515        1608         47        1137           0          0    198       0      0
 Inquire:            1          12          9
-Logging: Total-count  Total-elapsed   Force-count Force-elapsed
 MQGET               2       0.002355            2      0.002355
Maximum depth encountered         299
```

- Detailed information about the queue's used by a task, including:
  - Number of valid get requests V's total number of get requests issued (valid get requests = number of get requests that returned data and completed with a reason code of none or truncated message accepted)
  - The amount of time (max, min, average) messages have been on the queue

SMF116 – Queue Accounting statistics – WQST

- In addition to the information common to all queues, the following should be noted on the GET queues
  - Number of valid gets as compared to the total gets issued
    - The difference means that a number of gets returned no message, often due to a get wait expiring
  - Time on queue – average sometimes overflows
  - PSET is the average I/O time for a read from a pageset
  - Epages is the number of empty pages there were skipped during a get
  - Skip is the number of pages with messages that were skipped
  - Expire is the number of expired messages that were skipped

SMF116 Uses

- Channel usage
- Bufferpool/pageset balancing
  - In a high volume request reply scenario if the two queues are on the same pageset, separating them may improve performance
  - When queues have become concentrated in one resource pool
- Preparation for migration to shared queues
  - Min/Max/Average message size and duration on queue
- Application Performance tuning
  - Proper Indexing
  - Elimination of ‘hot spots’ – reducing contention
- Problem determination

SMF116 – What it does not tell you

- Often a consolidated view is needed
  - How many tasks are concurrently using this set of queues?
  - What tasks are related?
    - Can be determined via the queues accessed, but not easily
- Were security calls made during this task?
- No accounting for the IMS Bridge
- Consolidation of z/OS and distributed information to provide a complete view

SMF116 – A few gotcha's

- Size of WQST (WQSTAT) records increased by around 2K between V6 and V7.0.1 – so a lot more SMF 116 data written
- Starting SMF 116 Queue Level accounting via START TRACE, used to only start accounting for subsequent MQOPENs (so, accounting data was not collected for long running tasks that had already issued MQOPEN calls)
  - **APAR PM58798, PTFs UK80090 (V7.0.1)** and **UK80091 (V7.1)** fixes this
- Long-running tasks have SMF 116 records at STATIME. However, STAT tracing must be started. If not, records only written when task ends.
  - **APAR PM46937, PTF UK72801(V7.0.1)** fixes this
  - Fix was in the base for V7.1

SMF 116 accounting recap

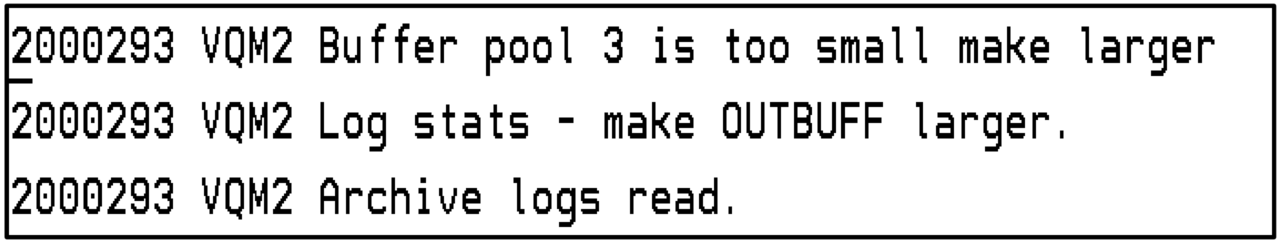
- Queues used by applications
  - MQAPI calls issued
  - Sizes of messages
- CPU cost per MQAPI call per queue
- Is “Put to a waiting getter” being used
- Number of messages read from disk
- Persistent V's Non Persistent message counts
- Number of MQGETs by msgid/correlid
- Number of messages skipped to find the right message
- Expired messages

SHARE in San Francisco 2013

MQCSMF

- Another sample program in MP1B
- Analyzes SMF 115 and 116 records and gives notification of any major problems found.

  e.g.



```
2000293 VQM2 Buffer pool 3 is too small make larger
2000293 VQM2 Log stats - make OUTBUFF larger.
2000293 VQM2 Archive logs read.
```

Summary

- We looked at:
  - SMF 115 records in some details
  - SMF 116 Accounting records

SHARE in San Francisco 2013

Thank-you

Any questions?

This was session 12610 - The rest of the week ……

	Monday	Tuesday	Wednesday	Thursday	Friday
08:00					Are you running too many queue managers or brokers?
09:30		What's New in WebSphere Message Broker			Diagnosing Problems for MQ / CICS and WMQ - The Resurrection of Useful
11:00		Extending IBM WebSphere MQ and WebSphere Message Broker to the Cloud	WMQ - Introduction to Dump Reading and SMF Analysis - Hands-on Lab	BIG Data Sharing with the cloud - WebSphere eXtreme Scale and WebSphere Message Broker integration	Getting the best availability from MQ on z/OS by using Shared Queues
12:15					
01:30	Introduction to MQ	MQ on z/OS – Vivisection	Migration and maintenance, the necessary evil	The Dark Side of Monitoring MQ - SMF 115 and 116 Record Reading and Interpretation	
03:00	First Steps With WebSphere Message Broker: Application Integration for the Messy	BIG Connectivity with WebSphere MQ and WebSphere Message Broker	WebSphere MQ CHINIT Internals	Using IBM WebSphere Application Server and IBM WebSphere MQ Together	
04:30	WebSphere MQ application design, the good, the bad and the ugly	What's New in the WebSphere MQ Product Family	MQ & DB2 – MQ Verbs in DB2 & Q-Replication	WebSphere MQ Channel Authentication Records	
06:00			Clustering - The Easier Way to Connect Your Queue Managers		

SHARE in San Francisco 2013