

MQ SMF Formatting – How to Use and Analyse (or Analyze) the Data

Mark Taylor
marke_taylor@uk.ibm.com
IBM Hursley

Lyn Elkins
elkinsc@us.ibm.com
IBM Washington Systems
Center

speaker.setName(Lyn)

What is SMF

- Not the Sacramento International Airport
- SMF is the z/OS System Management Facility
 - A common utility for all z/OS subsystems to report activity
 - What they report and when they report it is up to them
- Each subsystem is assigned one or more SMF Types
 - MQ uses:
 - SMF 115 – or MQ Statistics records
 - SMF 116 – or MQ Accounting records
- SMF data is (1) collected, (2) dumped to a data set, then (3) **formatted and analysed**

How do you collect data (1)

- Collecting MQ SMF is controlled two ways:
- SYSP Macro
 - SMFSTAT attribute owns the Statistical (SMF 115) record production
 - This should be on all the time, set to (01,04) for all queue managers
 - SMFACCT attribute owns the Accounting (SMF 116) record production
 - Typically not on all the time, controlled by the START TRACE command

How do you collect data (2)

- Collecting MQ SMF is controlled two ways:
- START TRACE command
 - +cpf START TRACE(A) CLASS(?)
- Starting and stopping the accounting trace is typically dynamic
 - CLASS(3) controls production of the Task Accounting (including queue) data
 - CLASS(4) controls production of the Channel Accounting data
 - CLASS(1) is no longer used
 - The classes are not inclusive, so if you want both Task and Channel accounting you need to turn on both 3 and 4

Dumping records to SMF output data set

- Once SMF data has been collected, it must be sent to output data sets
 - These can be considered an intermediate state
 - IFASMFDL copies data from SMF data sets to a sequential data set
 - IFASMFDL copies data from SMF logstreams to a sequential data set
- Options filter which records are copied to the output data set
 - Once the output data set is created, then it can be formatted

SMF Streaming

- New capability with z/OS 2.x via PTF OA49263
- “Live” access to SMF buffers without needing to dump for offline processing
 - Can then process data for real-time analytics
- Tool described here does not exploit that. But the SQL examples could be used

Working with the data

- Various tools have been around for a while
- CSQ4SMFD is a sample program provided with MQ
 - Dumps records from the data sets created by IFASMFDP/L jobs in a readable but unconsumable format
- SupportPac MP1B – free tool to create reports from records
- Other commercially available tools for interpretation
 - Sometimes do not keep up with changes
 - Do not capture/use some critical data
 - This column means what?

CSQ4SMFD Output – Message Manager

message manager statistics data

--Q-M-S-T---H-E-X---P-R-I-N-T----

Address = 13B2AC08

00000000 : D40F0048 D8D4E2E3 00000001 00000001 <M...QMST.....>

00000010 : 00000013 00000003 00000000 00000002 <.....>

00000020 : 00000000 00000000 00000000 00000000 <.....>

00000030 : 00000000 00000000 00000001 00000008 <.....>

00000040 : 00000000 00000000 <.....>

--Q-M-S-T---F-O-R-M-A-T-T-E-D----

qmstid = d40f

qmstll = 0072

qmsteyec = QMST

qmstopen = 00000001

qmstclos = 00000001

qmstget = 00000019

qmstput = 00000003

qmstput1 = 00000000

qmstinq = 00000002

Challenges

- Tools sometimes broke with different MQ levels
- Calculations were not always clear, or correct
- Difficult to validate they were doing the right thing
- Filled up JES spool with reports



JES Spool Example

SMF Record Type	Number of records
2	1
3	1
115	66
116	2,684,149

\$HASP375 ELKINSE1
ESTIMATE EXCEEDED BY
167,100,000

Output file name	Number of Lines
BUFF	316
BUFFCSV	57
LOG	326
TASK	163M

speaker.setName(Mark)

Challenges

- Would get calls asking how formatters actually worked
 - As I could see source code
- Not always able to understand it
 - But could see inconsistencies



Solution

- I decided I had to learn how to process SMF
- Investigation ...
- Found various tools and toolkits but none suitable
 - Java code that only runs on z/OS because of I/O
 - Parser using DFDL for IIB records

As a Distributed person

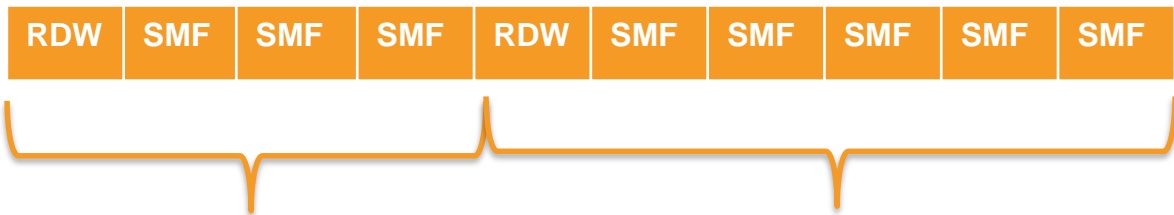
- I know how to develop code that runs on Unix and Windows
 - Editors, compilers, debuggers etc
- Lyn wanted to import to spreadsheets which run on those platforms
 - And different programs were better able to handle large data
 - So formatting SMF on these platforms made sense

Project Goal

- Develop a tool that did not get in the way of analyses
- Format all the data and nothing but the data
- Syntax. Not semantics.

Some issues

- Formatting RDW
 - z/OS data sets are structured (embedded record lengths)
 - Files on Unix/Windows are mostly byte-streams
 - Need to be able to deal with the Record Descriptor Words
 - ftp options to keep RDW bytes when transferring bytes
 - > QUOTE SITE RDW
 - > BINARY



Yet more issues

- C headers and Assembler macros did not always match
- Incompatible changes made across some versions
 - Fields inserted in middle of structures
- Data formats not always cross-platform C-friendly
 - Assumptions about data type sizes
 - Assumptions about bit fields
 - Assumptions about endian-ness
 - Assumptions about padding
 - Structures not always complete/overlap

SMF not as self-describing as advertised

- Despite claims, SMF is not really self-describing
 - Unlike MQ's PCF
 - Model is header followed by “triplets” which say where each real element is, how long it is, and how many there are
- MQ SMF has some undocumented triplets, or skipped fields
 - Can't tell without reading docs and looking at the sample source code (and sometimes verifying in product source code)
 - Not everything has an eyecatcher (newest CHIN records)
- A whole class of subtypes seems undocumented

Starting on the formatter

- Started with RDW record reader, hex and EBCDIC dumper
 - Similar to the raw output from CSQ4SMFD
- To ensure I was processing one complete record at a time
 - One SMF record may be split across multiple dataset records
- Program evolved ...
 - Simple structure for formatting MQ structures such as QPST
 - Adding the V9 pageset statistics took minutes
 - Might choose a different approach (Java?) if restarting
- Adding other record types (AMS is 180) is feasible

Post Processing Challenges

- Formatting the output data also had “opportunities”
- Spreadsheets try to be clever when importing CSVs
 - Date, time formats
 - Treating strings as numbers
 - And sometimes get it wrong
- So this formatter went through several iterations testing with Excel and LibreOffice to ensure data could be imported
 - Compromises needed on timestamp formats

Unexpected popularity

- After first version running, mentioned it at Interconnect 2016
- “How many people interested”
 - Expected only the co-presenter to raise hand
 - Rather more than that did
- So quickly got a version on github



Github project

<http://github.com/ibm-messaging/mq-smf-csv>

Downloading data

```
$ ftp winmvs41
Connected to 9.20.1.1
User (winmvs41:(none)): met
331 Send password please.
Password:
230 MET is logged on. Working directory is "MET.".
ftp> BINARY
200 Representation type is Image
ftp> QUOTE SITE RDW
200 SITE command was accepted
ftp> GET 'MET.SMF.DATA' c:\smf\data\test.bin
200 Port request OK
125 Sending data set MET.SMF.DATA
250 Transfer completed successfully
ftp: 792532 bytes received in 0.30 Seconds 2641.77Kbytes/sec.
ftp> quit
```


Running the program

```
C:\smf>mqsmfcsv -i c:\smf\data\test.bin -o c:\smf\out -m 200 -s
```

```
MQ SMF CSV - Build Jul 17 2016 11:45:19
```

```
Swapping bytes in input records
```

```
Processed 146 records total
```

Ignored		record count:	2
Formatted 115 subtype	1	record count:	48
Formatted 115 subtype	2	record count:	48
Formatted 115 subtype	215	record count:	48

A raw formatted CSV file

```
Date,Time,LPAR,QMgr,MQ_Version,Interval_Start (DATE),Interval_Start (TIME),Interval_Duration,BufferPool,Buffer_Count,Lowest_Stealable,Current_Stealable,Getp_Old_Requests,Getp_New_Requests,DASD_Read,Set_Write_Pages,Pages_Written,DASD_Write,Sync_Writes,Defer_Write_THold_Reached,Sync_Write_THold_Reached,Buffer_Steals,Buffer_Steals_Hash_Changes,Suspend_No_Buffers,LOC,LOC_FIX,
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,0,50000,49981,49984,966636,0,0,966532,84,31,17,0,0,0,0,0,"Above","Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,1,100000,42308,99487,1771124,328513,0,1736279,111053,6993,41,0,0,0,0,0,"Above","Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,2,50000,49999,49999,221,0,0,13,13,13,13,0,0,0,0,0,"Above","Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,3,100000,98961,99385,284845,56959,0,280496,4117,276,13,0,0,0,0,0,"Above",
"Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,4,50000,49999,49999,741,0,0,13,13,13,13,0,0,0,0,0,"Above","Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,5,100000,99393,99393,516284,103803,0,508323,828,71,13,0,0,0,0,0,"Above",
"Paged",
"2015/11/23","21:10:04,930000","H019","MQPC",="800","2015/11/23","20:40:12,045359",1792,6,100000,99992,99999,2229,167,0,764,36,19,13,0,0,0,0,0,"Above","Paged",
@
"SMF-QPST.csv" 553 lines, 87171 characters
```

Imported to a spreadsheet

SMF-QPST.csv - LibreOffice Calc

File Edit View Insert Format Tools Data Window Help

Liberation Sans 10

A1

	A	B	C	D	E	F	G	H	I	
1	Date	Time	LPAR	QMgr	MQ_Version	Interval_Start (DATE)	Interval_Start (TIME)	Interval_Duration	BufferPool	Buff
2	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	0	
3	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	1	
4	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	2	
5	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	3	
6	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	4	
7	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	5	
8	2015/11/23	21:10:04,930000	H019	MQPC	800	2015/11/23	20:40:12,045359	1792	6	
9	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	0	
10	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	1	
11	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	2	
12	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	3	
13	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	4	
14	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	5	
15	2015/11/23	21:39:58,000000	H019	MQPC	800	2015/11/23	21:10:04,929903	1793	6	

Import to SQL tables

- After working with just CSV, Lyn tried importing data to DB2
 - For very large data volumes that challenge spreadsheets
- But DB2 cannot simply import CSV files
 - Needs tables to be created with columns and datatypes
 - Unlike MS Access, which does it automatically
- Tried creating tables by hand
 - Was easier to do it in code to cover all tables
 - Get simple DDL to define columns with appropriate types

Some DDL

```
DROP TABLE MQSMF.QPST;  
CREATE TABLE MQSMF.QPST (  
  Date DATE  
  , Time CHAR(16)  
  , LPAR CHAR(4)  
  , QMgr CHAR(4)  
  , MQ_Version CHAR(3)  
  , Interval_Start_Date DATE  
  , Interval_Start_Time CHAR(19)  
  , Interval_Duration INTEGER  
  , BufferPool INTEGER  
  , Buffer_Count INTEGER  
  , Lowest_Stealable INTEGER  
  , Current_Stealable INTEGER  
  , Getp_Old_Requests INTEGER  
  , Getp_New_Requests INTEGER  
  , DASD_Read INTEGER  
  , Set_Write_Pages INTEGER  
  , Pages_Written INTEGER  
  , DASD_Write INTEGER  
  , Sync_Writes INTEGER  
  , Defer_Write_Thold_Reached INTEGER  
  , Sync_Write_Thold_Reached INTEGER  
  , Buffer_Steals INTEGER  
  , Buffer_Steals_Hash_Changes INTEGER  
  , Suspend_No_Buffers INTEGER  
  , LOC CHAR(6)  
  , FIX CHAR(6)  
);
```



Date = Date

Time = String

**And now some
examples of using the
data**

speaker.setName(Lyn)

How do I use this?

- Use MP1B and mqsmfcsv together for fuller picture
- MP1B
 - Looking at messages
 - Examine complete task record
 - What queues used
- MQSMFCSV
 - Looking for specifics

Some Common Analysis

- Bufferpool issues:

QMGr	MQ_Versio	Interval	Interval_D	BufferPool	Buffer_Co	Lowest_St	Current_St	Getp_Old_Req	Getp_New	DASD_Rea	Set_Write	Pages_Wri	DASD_Wri	Sync_Writ	D
QML1	800	#####	09:0	2352	0	10000	9974	9975	5559431	0	0	5559392	27	8	4
QML1	800	#####	09:0	2352	1	15000	14692	14694	221325	44412	0	217085	226	18	3
QML1	800	#####	09:0	2352	2	15000	2180	3236	12546086	2893875	613685	12578779	615763	38488	3
QML1	800	#####	09:0	2352	3	15000	11298	11902	49210	7789	3145	47718	3703	237	3
QML1	800	#####	09:0	2352	4	15000	12827	13874	5710808	1140222	0	5674805	1611	105	4
QML1	800	#####	10:0	1798	0	10000	9974	9975	4118894	0	0	4118864	5	4	2
QML1	800	#####	10:0	1798	1	15000	14691	14692	174429	35017	0	171216	156	13	2
QML1	800	#####	10:0	1798	2	15000	2205	3579	9290884	2150664	478006	9302808	479682	29982	2
QML1	800	#####	10:0	1798	3	15000	11061	11863	37386	6145	2525	36203	2376	151	2
QML1	800	#####	10:0	1798	4	15000	13873	14073	4186589	835675	0	4159757	569	40	3
QML1	800	#####	11:0	28928	0	10000	9972	9975	57745022	0	0	57744539	294	92	48
QML1	800	#####	11:0	28928	1	15000	14687	14694	2688933	539818	0	2638670	2285	189	31
QML1	800	#####	11:0	28928	2	15000	2200	3506	126619240	29014948	8186300	1.27E+08	8011007	500717	31
QML1	800	#####	11:0	28928	3	15000	2240	3506	560717	123511	41705	564050	60376	3820	33
QML1	800	#####	11:0	28928	4	15000	13031	14475	54915056	10960000	0	54557571	6975	487	38
QML1	800	#####	19:0	1584	0	10000	9975	9975	2142890	0	0	2142863	3	2	1
QML1	800	#####	19:0	1584	1	15000	14693	14694	164143	32961	0	161439	55	5	1
QML1	800	#####	19:0	1584	2	15000	3505	3506	3834660	1002494	0	3835563	1	1	1
QML1	800	#####	19:0	1584	3	15000	9002	9674	20267	3600	1470	19101	1983	125	1
QML1	800	#####	19:0	1584	4	15000	14474	14595	1857590	370519	0	1844758	127	10	2

Some Common Analysis

- Who is using the bufferpool?

BASE_NAME	PAGESET_	BUFFERPOOL_ID	
XMITQ1	2	2	
SYSTEM.ADMIN.CHANNEL.EVENT	2	2	
REPLY_Q_1	2	2	

Some Common Analysis

- Long Latching:

A	B	C	D	E	F
QMGR	LONGEST_LATCH	MAX_LATCH_WAIT_	MAX_LATCH_WAIT_I	START_TIME	TIME
QML1	0000000045702C28	938725	24	13:19:20,816071	
QML1	0000000045702C28	929327	24	13:19:26,184211	
QML1	0000000045702C28	928027	24	13:19:27,066160	
QML1	000000007E7684A8	855952	24	13:19:20,327533	
QML1	0000000045702C28	837519	24	13:19:22,095385	
QML1	0000004806A00920	835626	24	13:19:24,936686	
QML1	0000000045702C28	767101	24	13:19:39,856699	
QML1	0000000045702C28	684788	24	13:19:21,996386	
QML1	0000000045702C28	596139	24	13:19:35,333268	
QML1	0000000045702C28	496932	24	13:19:23,312622	
QML1	0000000045702C28	481161	24	13:19:46,452283	
QML1	0000000045702C28	471263	24	13:19:21,597983	
QML1	0000000045702C28	396968	24	13:19:30,589287	

Queries against the data

- Reading a million-plus row report for potential issues is impossible
 - With V7.0.1 we developed a series of searches that worked well against the task report
 - Quit working with V7.1 because the format changed dramatically
- Using queries to find things which might be problems

Some queries I have found useful (to date)

- Looking for skipped or expired messages?
 - `SELECT LPAR, QMgr, Correlation, Base_Name from MQSMF.WQ WHERE Get_Messages_Skipped_Count >0;`
 - `SELECT LPAR, QMgr, Correlation, Base_Name from MQSMF.WQ WHERE Get_Messages_Expired_Count >0;`
- Put to waiting getter active on a queue?
 - `SELECT * from MQSMF.WQ WHERE LPAR = 'MPX1' AND "Base_Name" = 'LYNS.TEST.QUEUE' AND "Put_Waiting_Getter_Count" > 0 ;`

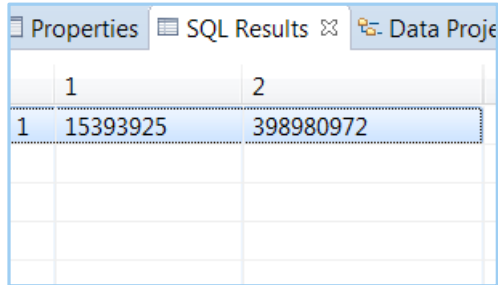
Some queries I have found useful (to date)

- How many transactions had unfulfilled MQGETs?
 - SELECT QMGR, Base_Name, Get_Valid, Get_Count, Get_Invalid from MQSMF.WQ where (GET_Valid < Get_Count and Base_Name= 'LYN.TEST.Q2');

QMGR	BASE_NAME	GET_VALID	GET_COUNT	GET_INVALID	
QML2	LYN.TEST.Q2	897	21529	0	
QML2	LYN.TEST.Q2	929	21328	0	
QML2	LYN.TEST.Q2	920	21419	0	
QML2	LYN.TEST.Q2	1012	23133	0	
QML2	LYN.TEST.Q2	329	13718	0	
QML2	LYN.TEST.Q2	1099	23601	0	
QML2	LYN.TEST.Q2	1070	23942	0	
QML2	LYN.TEST.Q2	1043	23624	0	

Some queries I have found useful (to date)

- How many valid MQGETs were from a queue?
 - `SELECT SUM(Get_Valid), SUM(Get_Count) from MQSMF.WQ where (GET_Valid < Get_Count and Base_Name='LYNE.QUEUE.2');`
 - Results - Column 1 the number of valid gets, Column 2 is total get requests:

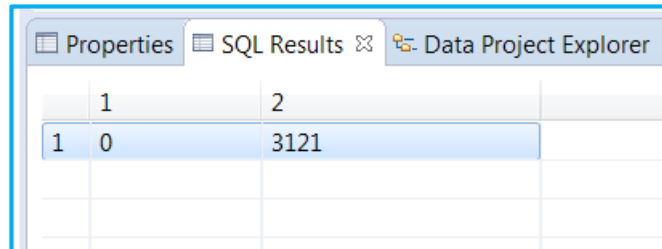


The screenshot shows a database query results window with three tabs: 'Properties', 'SQL Results', and 'Data Project'. The 'SQL Results' tab is active, displaying a table with two columns. The first column is labeled '1' and the second column is labeled '2'. The first row of data shows the values '15393925' and '398980972' respectively. The table has a light blue header and a light gray body.

1	2
15393925	398980972

Some queries I have found useful (to date)

- What was my largest message size retrieved for this queue?
 - `SELECT MAX(Get_Max_Msg_Size) from MQSMF.WQ where (Base_Name= 'LYNS.TEST.QUEUE');`
 - Result was 11,189 (application people insisted it was 3,800)
- How many MQPUTs and MQPUT1s were completed?
 - `SELECT SUM (Put_Count), SUM (Put1_Count) from MQSMF.WQ where (Base_Name = 'LYNS.TEST.QUEUE');`
 - Results:



The screenshot shows a software interface with three tabs: 'Properties', 'SQL Results', and 'Data Project Explorer'. The 'SQL Results' tab is active, displaying a table with two columns labeled '1' and '2'. The first row of data shows the values '0' and '3121' respectively. The table is highlighted with a blue border.

1	2
0	3121

Useful Queries - How much are my puts and gets costing?

- Query to get total costs for MQGETs and MQPUTs
 - SELECT SUM (Get_Count), SUM (Get_CT_us), SUM (Total_Valid_Gets),
SUM (Total_Bytes_Get),
SUM (Put_Count), SUM (Put_CT_us), SUM (Put1_Count),
SUM (Put1_CT_us), SUM (Total_Valid_Puts), SUM (Total_Bytes_Put)
FROM MQSMF.WQ
WHERE (Base_Name = 'ELKINSC.SHARED.QUEUE' AND
QMGR = 'QML1');

	1	2	3	4	5	6	7	8	9	10
1	204808121	1181322809	9173709	15415705192	0	0	0	0	0	0

Useful Queries - How much are my puts and gets costing?

- The raw sums are not all that useful by themselves
- But when averaged and used for comparisons, they can be

Queue Name	Queue Manager	Queue Type	Average CPU for Valid MQGET	Average CPU for Valid MQPUT
ELKINSC.SHARED.QUEUE	QML1	SHARED	128.77	0
ELKINSC.SHARED.QUEUE	QML2	SHARED	245.67	0
ELKINSC.SHARED.QUEUE	QML3	SHARED	0	18.42
ELKINSC.SHARED.QUEUE	QML4	SHARED	0	30.83
Sum FOR SHARED QUEUE			176.01	24.6
SYSTEM.CLUSTER.TRANSMIT.QUEUE	QML1	PRIVATE	21.15	51.76
SYSTEM.CLUSTER.TRANSMIT.QUEUE	QML2	PRIVATE	13.13	45.85
SYSTEM.CLUSTER.TRANSMIT.QUEUE	QML3	PRIVATE		
SYSTEM.CLUSTER.TRANSMIT.QUEUE	QML4	PRIVATE		
SUM FOR CLUSTER TRANSMIT QUEUE			17.91	49.37

And can surprise you!

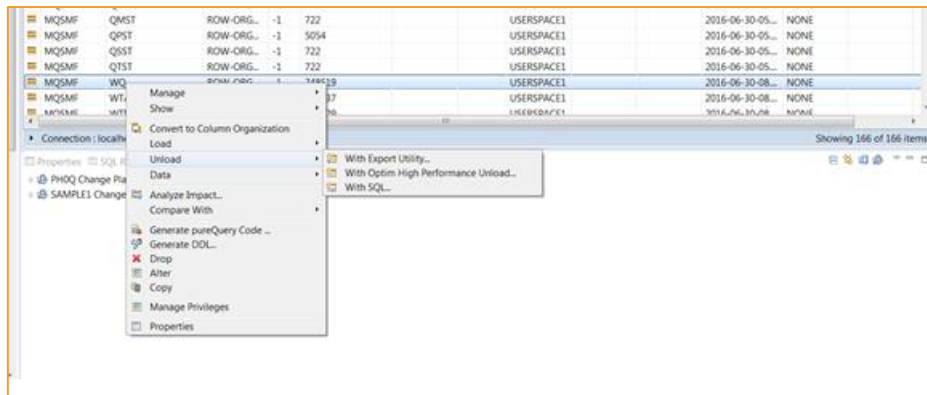
- SELECT QMgr, Interval_Start_Date, Interval_Start_Time,
Interval_Duration, Checkpoints, Log_CI
FROM MQSMF.QJST;

	QMGR	INTERVAL_START_DATE	INTERVAL_START_TIME	INTERVAL_DURATION	CHECKPOINTS	LOG_CI
1	QML1	2016-09-12	11:29:55,926548	59	1	7709
2	QML1	2016-09-12	11:30:55,695439	59	52	243762
3	QML1	2016-09-12	11:31:55,464300	59	53	242837
4	QML1	2016-09-12	11:32:54,742262	60	52	247384
5	QML1	2016-09-12	11:33:55,002061	59	53	242765
6	QML1	2016-09-12	11:34:54,417852	60	51	241894
7	QML1	2016-09-12	11:35:54,539833	59	0	0
8	QML1	2016-09-12	11:36:54,308729	59	0	0

Queries – warnings and lessons learned

- Using Data Studio
 - Makes things easy for those of us who are not very SQL literate
 - Single quotes are typically used for literals
 - A query defaults to 500 rows
 - If you need to see more, use an EXPORT

- Using an EXPORT



Using an Export - continued

1. Target

2. Options

3. Source

Specify the file name and format that you want to use to export data

Export operations require at least one output file. You can use the Options tab to specify additional, optional file specifications for each file type. Use the LOB and XML fields to specify where to store these types of data.

Select the file format type for the file.

☒ Delimited (DEL)

☐ Integrated exchange format (IXF)

Select or create an export file

☒ Create a new file

☐ Use an existing file



Path:

File name:

Using an Export - continued

1. Target	<p>When you select an output file format of delimited, you can specify additional options and change default values. No additional options exist for the other file formats.</p>
2. Options	
3. Source	
	<p>Code page: <input type="text"/></p> <p><input type="checkbox"/> Prefix positive decimal values with a blank</p> <p><input type="checkbox"/> Use ISO date format</p> <p><input type="checkbox"/> Suppress the recognition of double character delimiters</p> <p><input type="checkbox"/> Remove leading zeros from all decimal columns</p> <p>Custom timestamp format: <input type="text"/></p> <p>Delimiters</p> <p>The values of the column delimiter, character string delimiter, and decimal point character must all be different. The default values for these delimiters are a comma, a double quotation mark, and a period, respectively.</p> <p>Column delimiter: <input type="text"/></p> <p>Character string delimiter: <input type="text"/></p> <p>Decimal point character: <input type="text"/></p>

Using an Export - continued

Settings
Specify any additional settings to use. Click Run when you are done.
 [Preview Command](#) Run method: 

1. Target

2. Options

3. Source

Specify the columns to export to the output file

You can export all the columns in the table, or you can edit the SELECT statement to choose to export selected columns.

Select Statement

```
SELECT Base_Name, Get_Count, Put_Count, Put1_Count  
FROM MQSMF.WQ  
where Base_Name = "SYSTEM.CLUSTER.TRANSMIT.QUEUE";
```

2
single
quotes

2
single
quotes

The Export Command generated

```
CALL SYSPROC.ADMIN_CMD(  
  'EXPORT TO "C:\Users\IBM_ADMIN\Documents\Projects\SHARE\2016\Atlanta\How_Many.csv"  
  OF DEL MODIFIED BY COLDEL, CHARDEL'''' DECPT.  
  MESSAGES ON SERVER  
  SELECT Base_Name, Get_Count, Put_Count, Put1_Count  
    FROM MQSMF.WQ  
    where Base_Name = ''SYSTEM.CLUSTER.TRANSMIT.QUEUE'';' );
```

Results of the export (end of CSV file)

52432	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	1263	0
52433	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	6047	0	0
52434	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	6184	0	0
52435	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	6154	0	0
52436	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	6062	0	0
52437	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	432	0
52438	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	709	0
52439	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	1081	0
52440	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	468	0
52441	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	1174	0
52442	'SYSTEM.CLUSTER.TRANSMIT.QUEUE	'	0	784	0

New - Queries used to generate 'report like' sheets

- To generate a bufferpool report spreadsheet:

```
Select Date,  
Time, LPAR, QMgr, MQ_Version, Interval_Start_Date, Interval_Start_Time, Interval_Duration,  
BufferPool, Buffer_Count, Lowest_Stealable,  
  
/* The next statement calculates the greatest number of pages used during the interval. */  
INT(Buffer_Count - Lowest_Stealable) AS Highest_Used,  
  
/* The next statement calculates the highest used pages as a percentage of the total allocated for the SMF interval */  
DEC( (Buffer_Count - Lowest_Stealable) * 100.0 /DEC(Buffer_Count,8,2),8,2) AS Highest_Used_Percent,  
  
Current_Stealable, Getp_Old_Requests, Getp_New_Requests, DASD_Read, Set_Write_Pages,  
Pages_Written, DASD_Write, Sync_Writes,  
Defer_Write_THold_Reached, Sync_Write_THold_Reached,  
Buffer_Steals, Buffer_Steals_Hash_Changes,  
Suspend_No_Buffers,  
LOC,  
FIX  
FROM MQSMF.QPST  
WHERE (QMGR = 'QML1' and Buffer_Count>0)|
```

Buffer Manager spreadsheet – some of the data

Interval Duration (seconds)	Bufferpool	Buffer Count	Lowest Stealable	Highest Used	Highest used (Percent age)	Current Stealable	Getpage Old requests	Getpage New requests	DASD Reads	Set Write Pages	Pages Written	DASD Writes	Sync Writes	Deferred Write Threshold reached
1800	1	90000	13359	76641	85.15	14786	4528574	3097442	1175012	4942446	1597943	99908	35	902
1787	1	90000	13370	76630	85.14	18304	3755002	3063077	549347	4102442	1049826	66061	466	635
1769	1	90000	13380	76620	85.13	87546	3550094	2257593	950213	3077643	772478	48757	504	410
1788	1	90000	13381	76619	85.13	20338	4060327	3150003	1128696	4132427	1317630	83030	714	932
1788	1	90000	13382	76618	85.13	14804	3953180	3073483	863273	4265688	1277742	80317	483	935
1802	1	90000	13384	76616	85.12	18698	4225953	2587521	1406145	4008608	1254734	78460	37	793
1800	1	90000	13393	76607	85.11	14790	3681075	2733307	617118	4240813	1252508	78716	459	725
1811	1	90000	13394	76606	85.11	20949	4020174	2449481	1387535	3733815	1181575	73877	29	734
1786	1	90000	13399	76601	85.11	18320	3753811	2499659	1187206	3694460	1202872	75232	51	722
1783	1	90000	13399	76601	85.11	15555	4379280	2523709	1196764	4348455	1226417	76688	33	940

New - Queries used to generate 'report like' sheets

- To generate a log manager report spreadsheet:

```
SELECT
LPAR, QMgr, MQ_Version, Interval_Start_Date, Interval_Start_Time,
Interval_Duration,
Unavailable_Buffer_Count, Log_Read_Output_Buffer, Log_Read_Active_Log, Log_Read_Archive_Log,

/* The next statement calculates the total number of log reads completed during the interval */
INT(Log_Read_Output_Buffer + Log_Read_Active_Log + Log_Read_Archive_Log) AS TOTAL_LOG_READS,

Tape_Contention_Delays, Checkpoints, Log_CI,

/* The next statement calculates the MB per second written during the interval */
DEC(ROUND((((Log_CI*4)/(1024.00))/(Interval_Duration)),2),6,2),

IO_Total_Time_1_1_us, IO_Total_Suspend_Time_1_1_us, IO_Max_Duration_1_1_us,
IO_Max_Log_ID_1_1, IO_Max_Suspend_Dur_1_1_us, IO_Max_Suspend_Time_1_1_Date,
IO_Max_Suspend_Time_1_1_Time, IO_Max_Suspend_Log_ID_1_1,
IO_Total_Time_1_2_us, IO_Total_Suspend_Time_1_2_us,
IO_Max_Duration_1_2_us, IO_Max_Log_ID_1_2,
IO_Max_Suspend_Dur_1_2_us, IO_Max_Suspend_Time_1_2_Date,
IO_Max_Suspend_Time_1_2_Time, IO_Max_Suspend_Log_ID_2_1
FROM MQSMF.QJST
WHERE (QMGR = "QML1")
```

Log Manager spreadsheet – some of the data

INTERVAL_DURATION	UNAVAILABLE_BUFFER_COUNT	LOG_READ_OUTPUT_BUFFER	LOG_READ_ACTIVE_LOG	LOG_READ_ARCHIVE_LOG	TOTAL_LOG_READS	TAPE_CONTENTION_DELAYS	CHECKPOINTS	LOG_CI	MB_PER_SECOND
1795	0	623	4461	0	5084	0	10	2821634	6.14
1789	0	417	3337	0	3754	0	9	2825604	6.17
1796	0	540	2638	0	3178	0	12	3453542	7.51
1792	0	511	2307	0	2818	0	10	2972254	6.48
1789	0	449	2082	0	2531	0	10	2818718	6.15
1773	0	392	1952	0	2344	0	12	3445866	7.59
1798	0	424	1835	0	2259	0	10	3061346	6.65
1787	0	518	1725	0	2243	0	8	2460906	5.38
1797	0	381	1824	0	2205	0	14	4037442	8.78
1797	0	581	1597	0	2178	0	9	2778470	6.04
1791	0	306	1841	0	2147	0	11	3259292	7.11
1791	0	424	1689	0	2113	0	8	2451898	5.35

New - Queries used to generate 'report like' sheets

- To generate a message manager report spreadsheet:

```
Select Date,  
Time, LPAR, QMgr, MQ_Version, Interval_Start_Date, Interval_Start_Time, Interval_Duration,  
Open , Close , Get , Put , Put1 ,  
  
/* The next statement calculates the total number of put-type requests during the itnerval. */  
INT(Put + Put1) AS Total_Puts,  
  
Inq, Inql, Set, Endw, Close_Handles , Sub , SubReq , CB , CTL , Status , Pubs ,  
  
/* The next statement calculates the total number of API requests during the itnerval. */  
  
INT(Open + Close + Get + Put + Put1 + Inq + Inql + Set + Endw + Close_Handles + Sub +  
    SubReq + CB + CTL + Status + Pubs) AS Total_APIs  
FROM MQSMF.QMST  
WHERE (QMGR = 'QML1')
```

Message Manager spreadsheet – some of the data

Interval Start Time	Interval Duration (seconds)	Opens	Closes	Gets	Puts	Put1s	Total Put and Put1	Inq	Inq1	Sets	Endws	Close Handles	Subs	Subrqs	Callbacks	CTLs	Status	Pubs	Total API Requests
'17:26:25	1788	133418	130388	531761	436343	0	436343	801	0	0	0	0	0	0	17702	51084	0	0	1301497
'19:25:55	1800	122757	119906	516266	466991	0	466991	1035	0	0	0	0	0	0	17039	49662	0	0	1293656
'17:50:40	1791	135043	131332	515275	420723	0	420723	936	0	0	0	0	0	0	20885	59920	0	0	1284114
'17:56:13	1796	123061	120178	538814	439458	0	439458	929	0	0	0	0	0	0	14517	41819	0	0	1278776
'17:15:23	1786	134208	131756	513848	423742	0	423742	1058	0	0	0	0	0	0	15296	44041	0	0	1263949
'21:55:19	1790	119423	116761	552011	407681	0	407681	1131	0	0	0	0	0	0	15278	45281	0	0	1257566
'21:25:25	1794	123045	120359	526620	413822	0	413822	978	0	0	0	0	0	0	17198	49550	0	0	1251572
'17:20:52	1787	126553	123691	515540	425578	0	425578	943	0	0	0	0	0	0	14095	40276	0	0	1246676
'17:45:10	1790	127985	124277	502269	409181	0	409181	1229	0	0	0	0	0	0	20140	58516	0	0	1243597
'16:56:25	1800	121192	118505	523428	418266	0	418266	931	0	0	0	0	0	0	12798	37570	0	0	1232690

Other discoveries – Or why didn't I know this?

- I routinely ignored the 'seconds' fields on a lot of queries because for the vast majority of the time the time was not creeping into seconds...but....
 - When I added seconds on latches I found extraordinary things

CORREL	LONGEST_LATCH_H (Addr)	MAX_LATCH_WAIT_TIME_S	MAX_LATCH_WAIT_TIME_US	MAX_LATCH_WAIT_ID
D1EDA3C	0000004807D690F	5	731717	19
D1EDA3C	00000048069118D	3	394963	19
D1EDA3C	0000004807B9C0	2	769386	19
D1E9EAB	000000480690BD	2	701325	19
D1E9EAB	000000480690BD	2	701325	19
D1E9EAB	000000480690BD	2	701325	19
D1ECF994	000000480690BD	1	701842	19
D1ECF994	000000480690BD	1	701842	19
D1ECF994	000000480690BD	1	701842	19
D1ECF994	000000480690BD	1	701842	19

Summary

- MQ's SMF provides much insight for tuning and planning
- Experience has been needed to analyse data
- The discussion of tooling and queries here should enable better self-service



Any questions?