Comparing SMDS and DB2 Blobs for queue-sharing

Audience level: knowledge of MQ or z/OS

Skillset: z/OS Systems Programming, MQ Administration

Background:

Shared message data sets (SMDS) are the preferred method for offloading large messages in queue-sharing groups. SMDS's are designed to handle large messages efficiently, so in this exercise, we will test two CF structures, one with SMDS and the other with BLOBs to examine the differences between the two offloading mechanisms.

Overview of exercise:

I. Run OEMPUT program against SMDS-enabled CF structure (TEST1)
II. Run OEMPUT program against BLOB-enabled CF structure (TEST2)
III. Compare the output from both

Steps of exercise:

I. *Run OEMPUT program against SMDS-enabled CF structure (TEST1)*

1. Using MQ Explorer, verify that the below configuration is in place. You should see connections to ZQS1, ZQS2, and you should see a QSGA queue-sharing group visible.
2. In MQ Explorer, navigate to the queue-sharing group QSGA's Coupling Facility Structures by clicking '>' next to the QSGA label then pressing Coupling Facility Structures to display more information.
3. Your structures should look like the following:

Coupling Facility Structures

Filter: Standard for Coupling Facility Structures

Coupling facility ...	Status	Level	Description	Recovery	Loss of CF connectivity	Automatic recovery	Alteration time	Alteration date	Offload	Offlo
CSQSYSAPPL	Active	3	Recoverable System CF structure	Yes	Terminate	No	9:14:40 AM	Sep 11, 2024	None	0
NRMLMSGS	None	3		No	Terminate	No	12:33:02 PM	Apr 19, 2024	None	0
TEST1	Active	5	Recoverable System CF structure	Yes	Tolerate	Yes	10:21:30 AM	Sep 12, 2024	SMDS	50
TEST2	Active	5	Recoverable System CF structure	Yes	Tolerate	Yes	10:20:58 AM	Sep 11, 2024	DB2	50

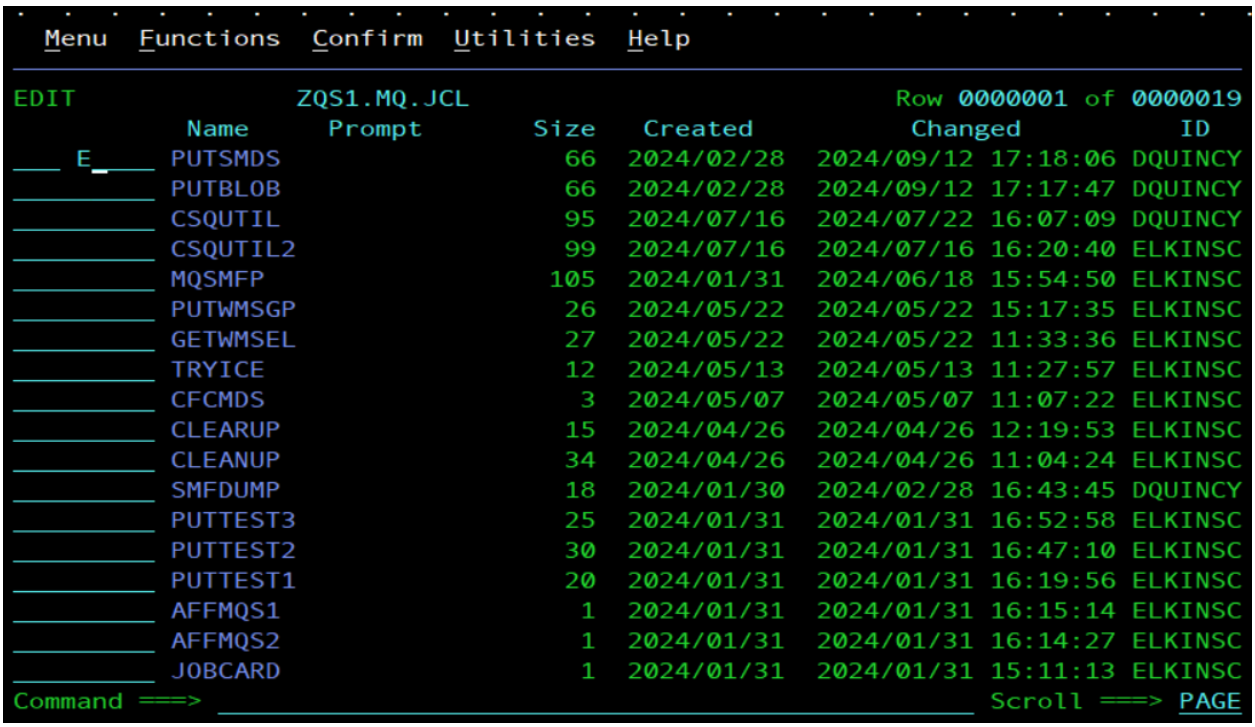
4. Scroll to the right, making sure that all offload rules are the same for TEST1 and TEST2 except for the 'Offload' and 'Group data set name' fields.

Coupling Facility Structures

Filter: Standard for Coupling Facility Structures

rule 1 size	Offload rule 2 threshold (%)	Offload rule 2 size	Offload rule 3 threshold (%)	Offload rule 3 size	Generic data set name	Logical block size	Number of buffers	Expand data s
	0	0K	0	0K		Default	Default	Default
	0	0K	0	0K		Default	Default	Default
	60	2K	70	0K	QSGA.*.TEST1.SMDS	256K	100	Yes
	60	2K	70	0K		256K	100	Yes

5. Now, navigate to the MQS1 z/OS image.
6. Use option 3.4 to navigate to the ZQS1.MQ.JCL data set. Navigate to PUTSMDS and type an ‘e’ to edit the member.



7. In PUTSMDS, you will see an execution of OEMPUT. This JCL puts a large amount of large messages on our SMDS.QUEUE, defined to TEST1.

Which parameters are we using with OEMPUT here?

Parameter	Description
- mZQS1	Specify target queue manager
-qSMDS.QUEUE	Specify target queue
-fileDD:MSGIN	Specify messages to be used
-ts500	Specify how long message stream should last (500 seconds)

-s650000	Specify the size of the message, note we are using a message larger than 63KB here, as to necessitate the use of offloading, since large messages can't be held in the CF list structures.
-l10	Loop MQPUT and MQGET 10 times during execution
-cgcpc	Mimic client application program by procressing a commit after both MQPUTs and MQGETs
-crlf	Each line in the input message file is used in sequence as message data
-rSMDS.QUEUE	Reply-to-queue from which replies will be retrieved (MQGET). If the -r option is omitted, MQGETs will not be issued.

****Note: If we specified persistent messages here, the contrast between SMDS and BLOBs would be less noticeable because transactions on both sides would have to wait on logging.**

8. Type 'submit' in the command line and press your enter key. You should see a reason code (RC) of 0000. This execution will take a few minutes to complete.

II. *Run OEMPUT program against BLOB-enabled CF structure (TEST2)*

1. Now, we will repeat the steps to submit another execution of OEMPUT, this time for our queue tied to BLOB storage.
2. Use F3 to back out to the ZQS1.MQ.JCL data set. Place an 'E' next to PUTBLOB and press enter to edit the member.
3. As you look through PUTBLOB, navigating up and down the screen using the F7 and F8 keys, you will notice that the only difference between PUTSMDS and PUTBLOB is the queue name. We are keep all other variables constant, especially message size.
4. Type 'submit' in the command line and press your enter key. You should see a reason code (RC) of 0000. This execution will take a few minutes to complete.

III. *Compare the output from both*

1. From the ISPF main menu, type 'sdsf' or 'd' on the command line and press enter to access the SDSF menu.
2. Here, type 'ST' on the command line and press enter to access the status of recent jobs

3. In our JCL, our OEMPUT jobs were named OEMPSMDS and OEMBLOB, respectively. We can search for all jobs beginning with OEM, but typing in the command line ‘pre OEM*’ and pressing enter.
4. Both jobs should appear in a list. Let’s look at OEMPSMDS first. Place a question mark to the left of the job name and press enter.
5. A list should appear with output on how successful the job was and any output from the job. We are interested in the SYSPRINT output. Place a ‘s’ to the left of SYSPRINT and press enter.
6. Scroll down on the SYSPRINT output until you see the following output. Make not of the Total Transaction value, the Transaction Rate value, and the Avg App CPU per msg value. This gives us information about how many transactions were completed in the allotted time with SMDS storage specified, the efficiency of those transactions, and the CPU consumption required.

****If you are unable to see the SYSPRINT screen for any reason, we have prepared sample examples at the end of this lab for reference.****

7. Now, let’s check out the same information for BLOB storage. Use F3 to back out twice until you reach the list containing OEMPSMDS and OEMPBLOB.
8. Place a ‘?’ next to the OEMPBLOB job and press enter.
9. Place a ‘s’ next to the SYSPRINT output and press enter.
10. Navigate until you see the Total Transaction value, the Transaction Rate value, and the Avg App CPU per msg value.

****If you are unable to see the SYSPRINT screen for any reason, we have prepared sample examples at the end of this lab for reference.****

11. You have now compared the performance and storage consumption of SMDS and BLOB offloading in our test environment! Hopefully, this helps you see the advantages of using SMDS in terms of throughput. While CPU consumption is higher for SMDS in this test environment,

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY OEMPSMDS JOB01038  DSID   102 LINE 36      COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
-------------------------------------------------------
Total Transactions  : 155080
Elapsed Time        :  500.024 seconds
Application CPU Time:   26.843 seconds  (5.4%)
Transaction Rate    :  310.145 trans/sec
-------------------------------------------------------
Round trip per msg  :     3224 microseconds
Avg App CPU per msg :      173 microseconds
-------------------------------------------------------
 Jobname.ASID  TCB(uS)   SRB(uS) SSRB(uS)   Tot(uS) (%)
                /tran     /tran   /tran      /tran
------------- -------- -------- -------- -------- ----
ZQS1MSTR.004C 00000059 00000015 00000000  00000075 2.3
ZQS1CHIN.004D 00000000 00000000 00000000  00000000 0.0
ZQS1BRK*      00000000 00000000 4494815659  4494815659 139404585.5
Total       CPUmicrosecs/tran                    75
Grand Total CPUmicrosecs/msg                    248
-------------------------------------------------------
Ending loop at 2024-09-12 21:26:24.341198
```

Figure 1. SMDS performance

```
 Display  Filter  View  Print  Options  Search  Help
-------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY OEMPBLOB JOB01037  DSID   102 LINE 37      COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
-------------------------------------------------------
Total Transactions  : 23220
Elapsed Time        :  500.083 seconds
Application CPU Time:    2.662 seconds  (0.5%)
Transaction Rate    :   46.432 trans/sec
-------------------------------------------------------
Round trip per msg  :    21536 microseconds
Avg App CPU per msg :      114 microseconds
-------------------------------------------------------
 Jobname.ASID  TCB(uS)   SRB(uS) SSRB(uS)   Tot(uS) (%)
                /tran     /tran   /tran      /tran
------------- -------- -------- -------- -------- ----
ZQS1MSTR.004C 00000447 00000098 00000000  00000546 2.5
ZQS1CHIN.004D 00000001 00000000 00000000  00000002 0.0
ZQS1BRK*      00000000 00000000 30019638778  30019638778 139388175.6
Total       CPUmicrosecs/tran                   548
Grand Total CPUmicrosecs/msg                    663
-------------------------------------------------------
Ending loop at 2024-09-12 21:26:02.561851
```

Figure 2 BLOB performance