Over the past year we have seen more customers with multiple queue sharing groups (QSG) in production. Managing the SMF data is more challenging when this is the case, especially when there are multiple QSGs that have the same structure names (we see this a lot) and the same shared queue names. This is because most of the MQ SMF data is completely queue manager centric. Except for the QCCT (channel initiator statistics) and the new QQST (queue statistics records) this important information does not appear in the others. So, when I am looking at the MQ reported Coupling Facility statistics for a structure named APP1, if that structure is in multiple QSGs, the simple query we have commonly used those results will be misleading. Misleading data leads to errors when trying to track down workload skewing and performance problems. And while those of us using this hope that the QSG name gets added to more records in the future, what do we at the WSC do to address this in the meantime?

Up until 2022, I hand coded SQL. Then Dorothy coded up Python scripts to generate the 'usual suspects' queries for our health check work. To use those in situations where there are multiple queue sharing groups, we have then had to replicate and tailor those scripts to pull the data for the queue managers in individual QSGs. And we have been in situations where the customer's documentation about QSG members has not been kept up to date - new queue managers have been added, older queue managers have been retired, or (even worse!) been moved to a different QSG.

Looking for simpler, more efficient, less subject to error methods, and because these multi-QSGs have become more common, we talked about reconfiguring the python scripts, etc. But as I was working with data over this past week, and Dorothy was fighting fires on another front, I decided to try using Db2 Views to separate the queue managers into the QSGs. I am not an SQL expert, and as we all know the IBM documentation can be quite effective when you know what you are looking for and painful to follow when you are searching for the answers to ‘this is how you do this.’ So, in case others are in this same position, I felt like it was time to write this up.

The first thing to do was figure out which queue managers are in which queue sharing group. After loading the data processed by MQSMFCSV into Db2 we run this very simple SQL:

```
SELECT DISTINCT LPAR, QMGR, QSG_NAME
FROM MQSMF.QCCT;
```

The results from this query look something like this:

LPAR	QMGR	QSG_NAME
MQZ1	ZMQA	SMQ1
MQZ2	ZMQB	SMQ1
MQZ1	ZMQC	SMQ1
MQZ2	ZMQD	SMQ1
MQZ1	ZMQE	SMQ1
MQZ2	ZMQF	SMQ1
MQZ1	ZMI2	
MQZ1	ZMQ1	SMQ2
MQZ2	ZMQ2	SMQ2
MQZ1	ZMQ3	SMQ2
MQZ2	ZMQ4	SMQ2
MQZ1	ZMQ5	SMQ2
MQZ2	ZMQ6	SMQ2
MQZ1	ZMI3	
MQZ1	ZMI4	
MQZ1	ZMI5	
MQZ2	ZMI6	
MQZ2	ZMI7	
MQZ2	ZMI8	

What this shows is that there are two queue sharing groups represented each with 6 queue managers, and 7 standalone queue managers. Using this information, I created views over all the tables generated by MQSMFCSV. The pattern looks like this:

```
DROP VIEW QSGSMQ1_QEST;
CREATE VIEW QSGSMQ1_QEST
AS
SELECT * from MQSMF.QEST as QEST
where
 (QMGR = 'ZMQA' OR QMGR = 'ZMQB' OR QMGR = 'ZMQC' OR
QMGR = 'ZMQD' OR QMGR = 'ZMQE' OR QMGR = 'ZMQF')
;

DROP VIEW QSGSMQ2_QEST;
CREATE VIEW QSGSMQ2_QEST
AS
SELECT * from MQSMF.QEST as QEST
where
 (QMGR = 'ZMQ1' OR QMGR = 'ZMQ2' OR QMGR = 'ZMQ3' OR
QMGR = 'ZMQ4' OR QMGR = 'ZMQ5' OR QMGR = 'ZMQ6')
;
```

A view for the tables for the standalone queue managers was also created.

What this allowed us to do was use the views to isolate the activity reported on the individual QSGs, making the queries simpler, more readable, and giving us more accurate data about the use of the CF, Db2 and SMDS. It also allowed us to review the queue and task activity by QSG, giving better insight into their use as well.

To illustrate the simplification, the SQL WHERE clause we had to compose to retrieve the QEST data (the queue managers calls to a structure data) before creating the views looked like this:

```
FROM MQSMF.QEST
WHERE (CFSTRUCT <>  '      ' AND
(QMGR = 'ZMQA' OR QMGR = 'ZMQB' OR QMGR = 'ZMQC' OR
QMGR = 'ZMQD' OR QMGR = 'ZMQE' OR QMGR = 'ZMQF') )
```

After creating the views, the same Where clause looks like this:

```
FROM QSGSMQ1.QEST
WHERE (Structure_Name <>'    ')
```

This does not seem like much, but using views has once again saved much needed time and prevented typos. And it means that we are looking at data from a single queue sharing group, that we do not have the possibility of comingling data from multiple QSGs and can evaluate more complex situations more accurately.

Please note that all samples considered open source, feel free to use them but they are provided AS-IS. There is no guarantee of support.