Everything about subsystems LAB 2

Table of Contents

Everything about subsystems Lab 2

Create a subsystem to handle batch jobs

If you did Lab 1 you may skip this step

In this task you will create a 'normal' subsystem used to handle batch work

Task 1 – Create the subsystem description

You should create the subsystem description in your library (SBSCLASSxx) and the subsystem should have one memory pool assigned (*BASE), put a relevant description on the object

Task 2 – Create a class to hold your run attributes

The class should be created in your library and named CLASS50, define the job to run at priority 50 and get 2000 MS per time slice, the job should wait 30 seconds for any locks it needs, use a relevant description.

Task 3 – Create a job queue for your batch job to use

The job queue should be created in your library and named BATCHJOBS. It should be able to be controlled by the system operator (and anyone else with job control authority).

Task 4 – Attach the job queue to your subsystem

Attach the job queue to your newly created subsystem at sequence 50, it should allow up to 5 jobs to run

Task 5 – Start your subsystem and submit a batch job

Start your subsystem. Do a DSPSBSJOBS to ensure it started correctly.

What memory pool is your subsystem running in? Why is this different than what you stated when you created the subsystem.

Then submit a job that delays 90 seconds (to give you time to see it), named sbsxxTest1 and repeat the DSPSBSJOBS

What do you see? If the job isn't running, look for its joblog. What do you see?

Task 6 - Add a routing entry to your subsystem

Add a routing entry to your subsystem at sequence 9999, it should catch any routing data the user uses, and call the normal command processor (QSYS/QCMD), use the class you created in Task 2. You will allow as many jobs to use this entry as wish to.

Resubmit your batch job (task 5) what happens? If the job is running, look at it and check the run attributes, what is the run priority, time slice, and wait time?

Task 7 – End your subsystem

Issue the command to end your subsystem

Adding a memory pool to your subsystem

In this task you will add a shared memory pool to isolate work within your subsytem

Task 23 – Look at the predefined memory pool

Use the WRKSHRPOOL command and page down to *SHRPOOL40

Record the following:

Defined size: ____ Max Active: ____ Paging option: ____

Priority: ____ Min Size: ____ Max Size: ____

Description: _____

Task 24 – Change your subsystem to use this pool

Use the CHGSBSD command to change your subsystem to have 2 pools, the first to use *BASE and the second to use *SHRPOOL40

Task 25 – Start your subsystem and confirm memory pools

Use the DSPSBSD command to display your subsystem, display the pools to confirm *SHRPOOL40 is there

Use the WRKSYSSTS command to see the pool is now active

Task 26 – End your subsystem

Issue the command to end your subsystem

Task 27 - Add a routing entry to use the new pool

Add a routing entry to your subsystem at sequence 100, it should look for routing data of 'POOL2', use your new memory pool, and call the normal command processor (QSYS/QCMD), use the class you created in Task 2. You will allow as many jobs to use this entry as wish to.

Task 28 – Test your routing entry and memory pool

Start your subsystem and submit a job use your jobq (BATCHJOBS), the command should be DLYJOB DLY(120), and the routing data should be 'POOL2', and name the job POOL2TEST. Use DSPSBSJOBS to validate the memory pool being used.

Task 29 – End your subsystem

Issue the command to end your subsystem

Using Prestart jobs

In this task you will add entries for ODBC prestart jobs we will be using the QZDASOINIT (ODBC) and QZDASSINIT (SSL ODBC)

Task 30- – Create a class for jobs running at priority 20

The class should be created in your library and named CLASS20, define the job to run at priority 20 and get 2000 MS per time slice, the job should wait 30 seconds for any locks it needs, use a relevant description.

Task 31- – Add a prestart job for ODBC

The program used to handle ODBC connections is QSYS/QZDASOINIT, you will set up a prestart job for this program in your subsystem using the ADDPJE command, you should use the following attributes: Job name is ODBC, the program is QSYS/QZDASOINIT, you should initially start 3 jobs, the memory pool should be *BASE, and the class should be the one you created in Task 30

Task 32- – Test that the jobs are started

Start your subsystem and then use the DSPSBSJOBS command to see what is running, an attribute of prestart jobs is only active (attached) jobs show in WRKACTJOB. Press F14 to show all jobs.

Task 33 – End your subsystem

Issue the command to end your subsystem

Task 34- – Create a class for jobs running at priority 30

We will run our SSL ODBC jobs at priority 30. The class should be created in your library and named CLASS30, define the job to run at priority 30 and get 2000 MS per time slice, the job should wait 30 seconds for any locks it needs, use a relevant description.

Task 35- – Add a prestart job for SSL ODBC

The program used to handle ODBC connections is QSYS/QZDASOINIT, you will set up a prestart job for this program in your subsystem using the ADDPJE command, you should use the following attributes: Job name is ODBC, the program is QSYS/QZDASOINIT, you should initially start 3 jobs, the memory pool should be *BASE, and the class should be the one you created in Task 30

Task 36- – Test that the jobs are started

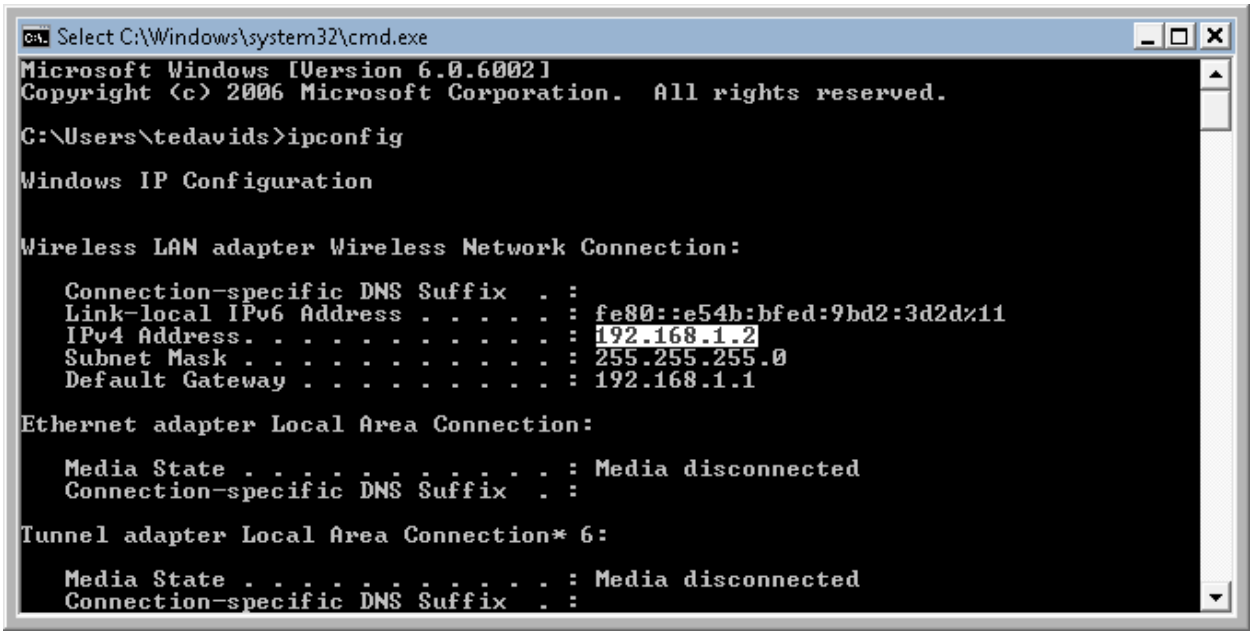
Start your subsystem and then use the DSPSBSJOBS command to see what is running, an attribute of prestart jobs is only active (attached) jobs show in WRKACTJOB. Press F14 to show all jobs.

Task 37- – Change client access to use your subsystem

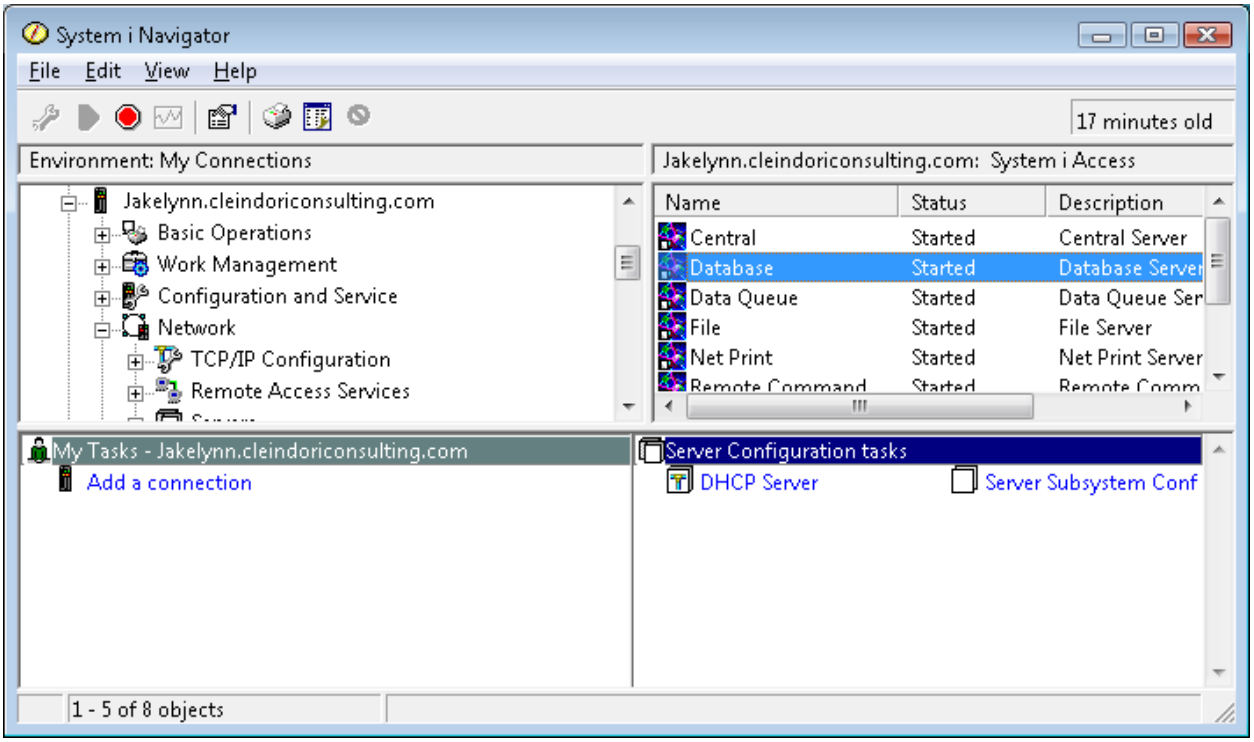
Prestart job requests are routed to the appropriate subsystems based on IP address, you will need to tell the system via System i Navigator to send your requests to your subsystem.

Note: This is the process to set this up, however these entries have already been created for you.

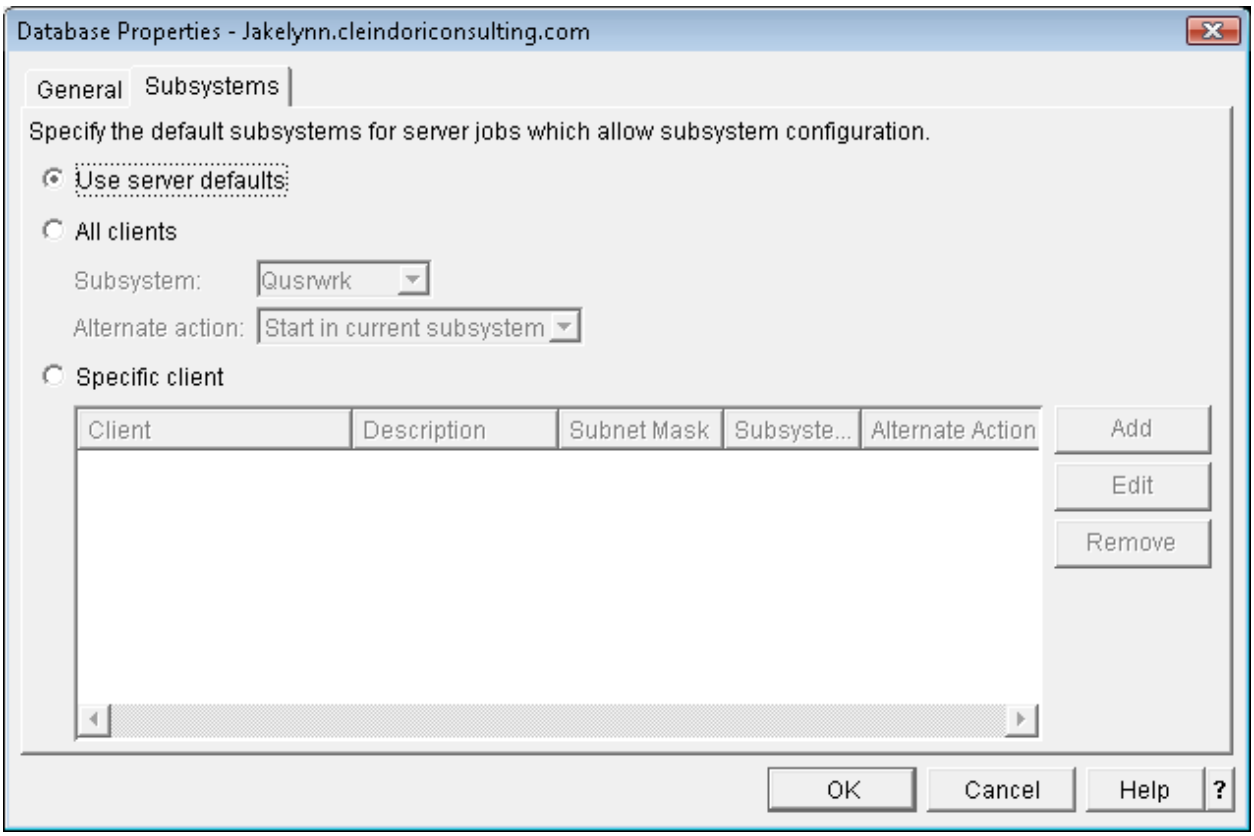
First you need to get your IP address, I do this via the ipconfig command from windows, use Start/Run then enter CMD to get a DOS window:



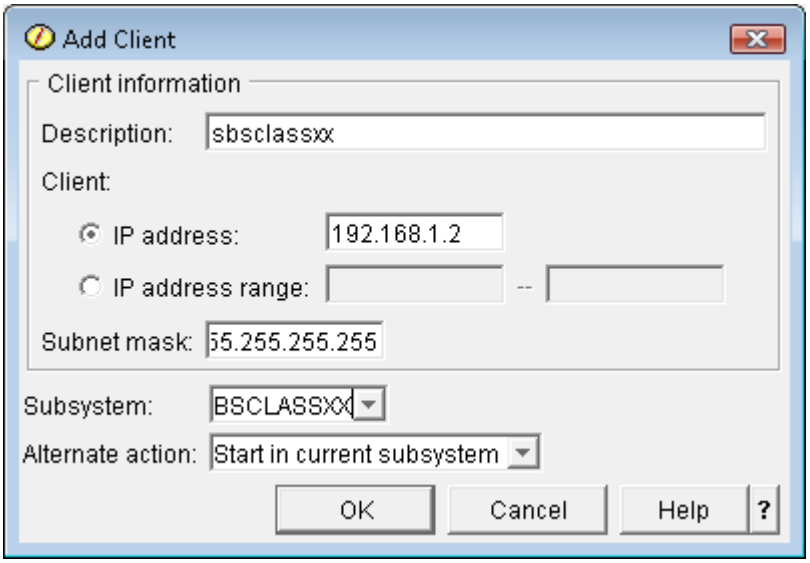
Get your IP address, then open System i Navigator. Go to system/Network/Servers/System i Access then right click on Database and choose Properties (Note in real life you will need *SERVICE authority)



You will receive this screen:

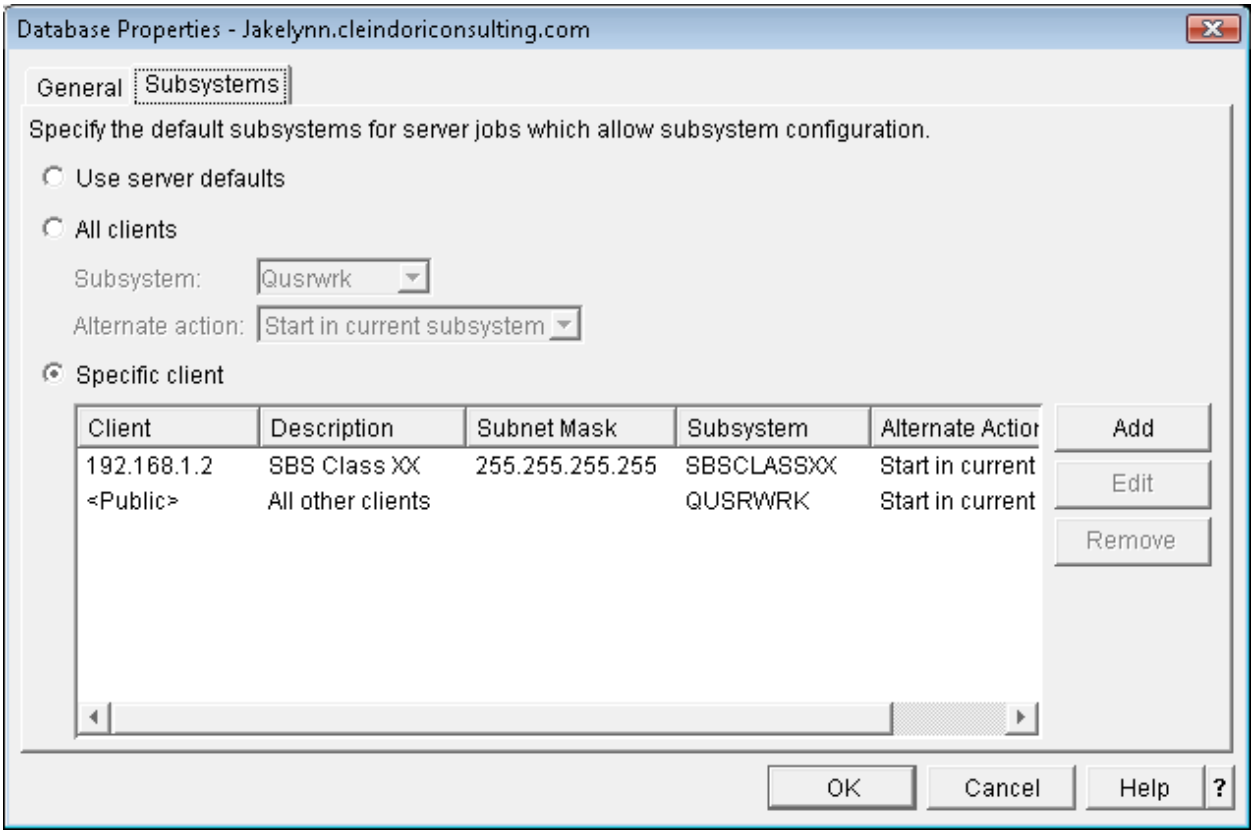


Click on Specific Client then Add



Enter your IP address, subnet mask of 255.255.255.255 (IMPORTANT, do not mess this up), and your subsystem (SBSCLASSXX) in the screens and press OK

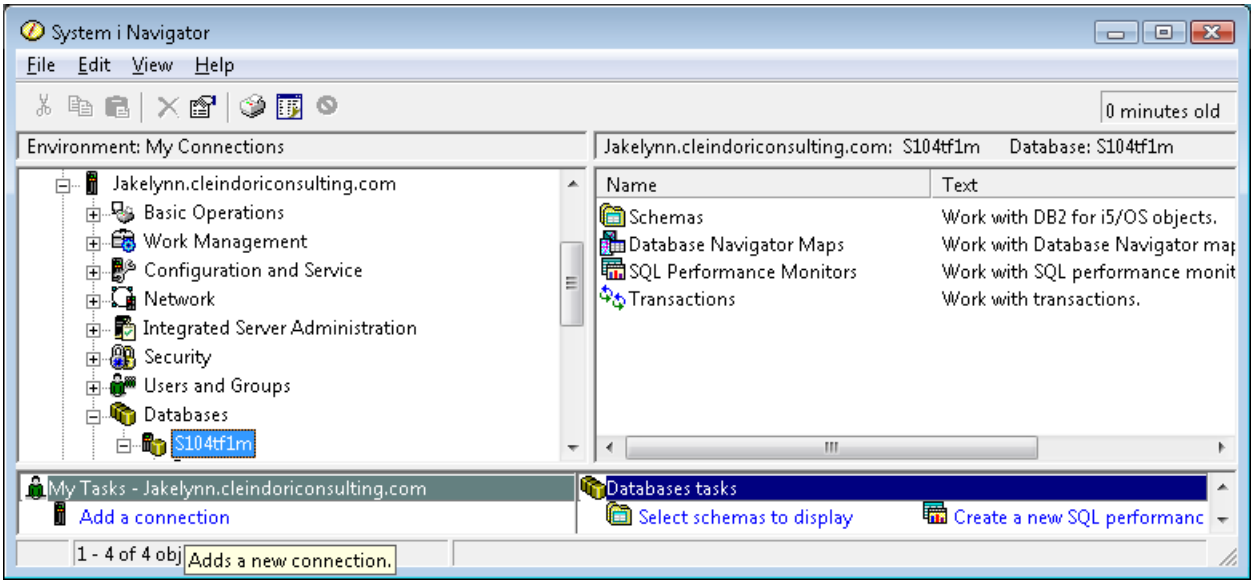
You should get a screen similar to this:



Task 38- – Test your subsystem is being used

Run the DSPSBSJOBS command you should show your subsystem active, if not start it

From iSeries Nav start a SQL session (system/Databases/serial#)



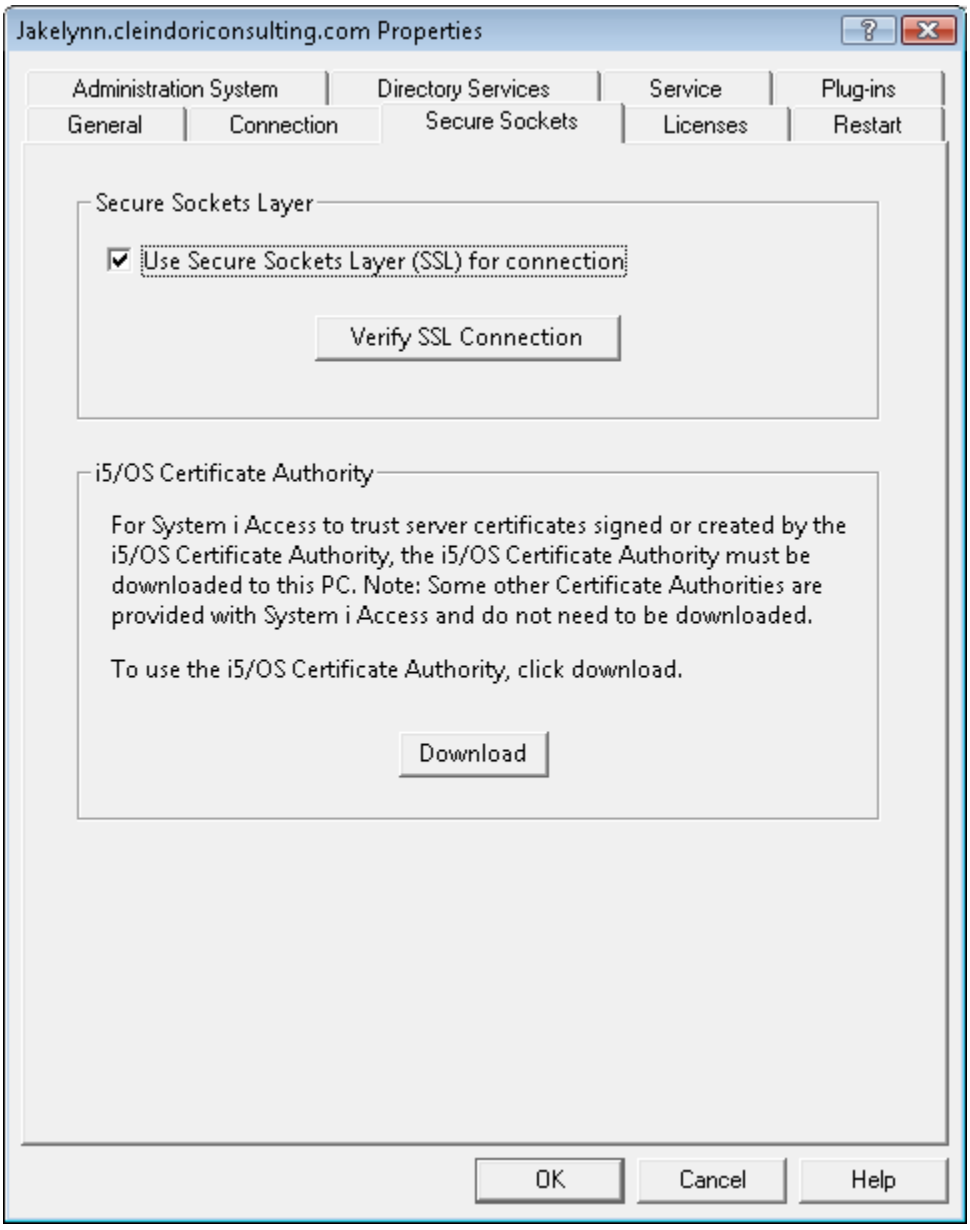
Right clickand select ‘Run SQL scripts…’

After the script window opens check your subsystem, you should show your jobs as ODBC. Also display the job and check the run priority and memory pool assigned.

Close the SQL window.

Right click on the system name, select properties:

Select the Secure sockets tab



Select Secure sockets, and click OK, then close down iSeries nav.

Reopen iSeries Nav, and return to run SQL scripts.

Check your subsystem and you should see the SSLODBC jobs active.

Using a different initial program

In this task you will change the program that is being called when a job starts in your subsystem

Task 39 – Create your new initial program

Copy the source from SBSCLASS/QCLSRC(MYINTIPGM) to your library

Change the message to something unique to you and compile it into your library.

Task 40 – Add a routing entry to use your new initial program

Add a routing entry to your subsystem at sequence 200, it should look for routing data of 'NEWPGM', use *BASE as your memory pool, and call your new command processor (SBSCLASSXX/MYINTIPGM), use any class you wish. You will allow as many jobs to use this entry as wish to.

Task 41 – Test your routing entry and program

Start your subsystem and submit a job use your jobq (BATCHJOBS), the command should be DLYJOB DLY(120), and the routing data should be 'NEWPGM', and name the job NEWPGMTEST. Use DSPSBSJOBS and check the joblog of your job. You should see the message from your initial program in the job log, you should also see the program in your invocation stack

Task 42 – End your subsystem

Issue the command to end your subsystem

Solutions

Create a subsystem to handle batch jobs

Task 1 – Create the subsystem description

CRTSBSD SBSD(SBSCLASSXX/SBSCLASSXX) POOLS((1 *BASE)) TEXT('SBSCLASSxx subsystem')

Task 2 – Create a class to hold your run attributes

CRTCLS CLS(SBSCLASSXX/CLASS50) RUNPTY(50) TIMESLICE(2000) DFTWAIT(30)

Task 3 – Create a job queue for your batch job to use

CRTJOBQ JOBQ(SBSCLASSXX/BATCHJOBS) OPRCTL(*YES) AUTCHK(*OWNER)

Task 4 – Attach the job queue to your subsystem

ADDJOBQE SBSD(SBSCLASSXX/SBSCLASSXX) JOBQ(SBSCLASSXX/BATCHJOBS) MAXACT(5) SEQNBR(50)

Task 5 – Start your subsystem and submit a batch job

strsbs sbsclassxx/sbsclassxx

The WRKACTJOB (and other commands) use the actually memory pool #, when you create your subsystem the memory pool # is just a sequence #, the actual pool number as shown by wrkshrpool or wrksyssts should match the *BASE pool ID (always 2)

SBMJOB CMD(DLYJOB DLY(90)) JOB(SBSXXTEST1) JOBQ(SBSCLASSXX/BATCHJOBS)

Job isn't running. Job log shows message CPC1117, no routing entry

Task 6 - Add a routing entry to your subsystem

ADDRTGE SBSD(SBSCLASSXX/SBSCLASSXX) SEQNBR(9999) CMPVAL(*ANY) PGM(QSYSL/QCMD)

CLS(SBSCLASSXX/CLASS50) MAXACT(*NOMAX)

You can see your batch job running. Attributes should match your class

Task 7 – End your subsystem

ENDSBS SBS(SBSCLASSXX)

Adding a memory pool to your subsystem

Task 23 – Look at the predefined memory pool

Defined size: varies Max Active: _30_ Paging option: *CALC

Priority: __2_ Min Size: _1%_ Max Size: _5%_

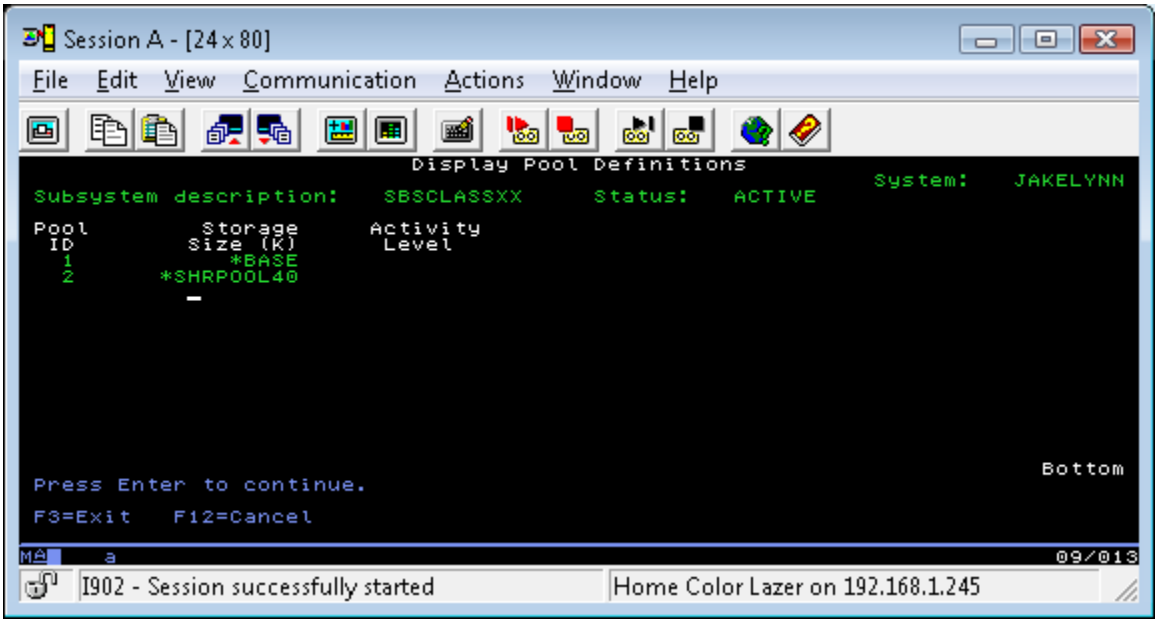
Description: Shared pool for SBS Class_____

Task 24 – Change your subsystem to use this pool

CHGSBSD SBSD(SBSCLASSXX) POOLS((1 *BASE) (2 *SHRPOOL40))

Task 25 – Start your subsystem and confirm memory pools

DSPSBSD SBSD(SBSCLASSXX)



WRKSYSSTS – Then F11 to show pools, you may have to scroll down

Task 27 - Add a routing entry to use the new pool

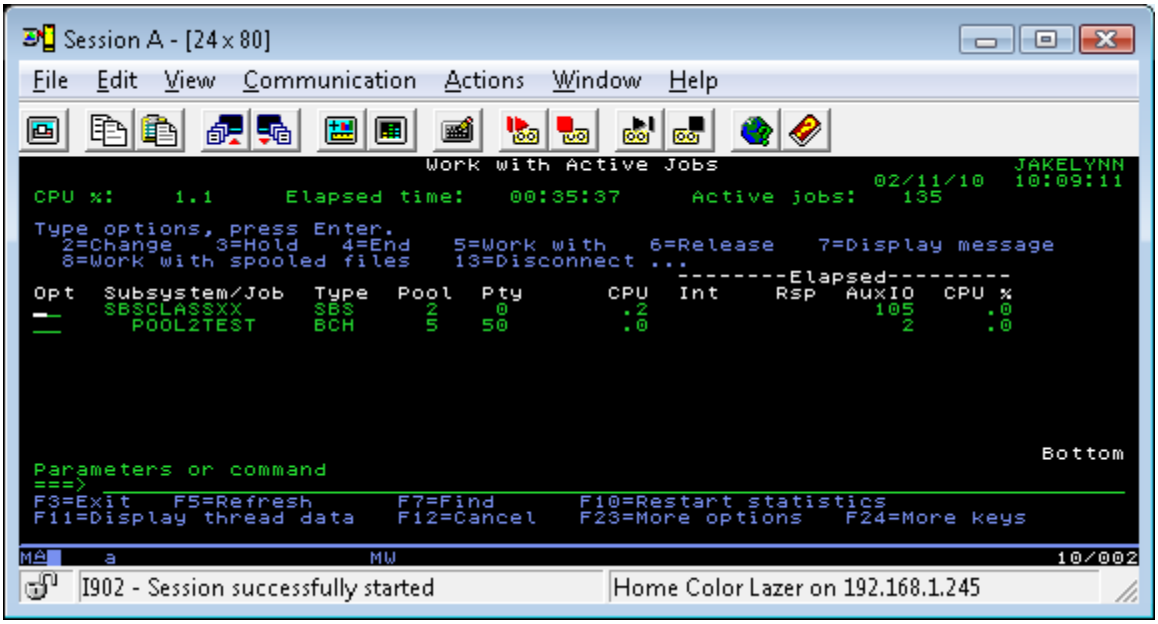
ADDRTGE SBSD(SBSCLASSXX/SBSCLASSXX) SEQNBR(100) CMPVAL('POOL2') PGM(QSYSL/QCMD)

CLS(SBSCLASSXX/CLASS50) MAXACT(*NOMAX) POOLID(2)

Task 28 – Test your routing entry and memory pool

SBMJOB CMD(DLYJOB DLY(120)) JOB(POOL2TEST) JOBQ(BATCHJOBS) RTGDTA(POOL2)

DSPSBSJOBS – Press F11 and you should see a pool other than 2 being used by the batch job



Task 30- – Create a class for jobs running at priority 20

CRTCLS CLS(SBSCLASSXX/CLASS20) RUNPTY(20) TIMESLICE(2000) DFTWAIT(30)

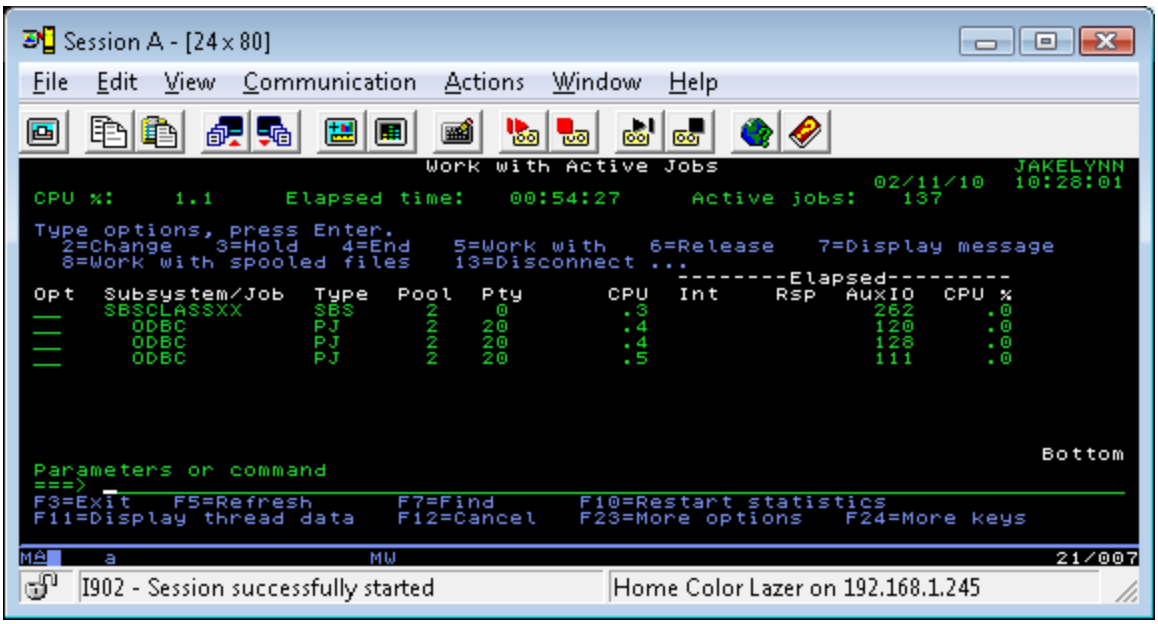
Task 31- – Add a prestart job for ODBC

ADDPJE SBSD(SBSCLASSXX/SBSCLASSXX) PGM(QSYS/QZDASOINIT) INLJOBS(3) JOB(ODBC)

POOLID(1) CLS(SBSCLASSXX/CLASS20)

Task 32- – Test that the jobs are started

DSPSBSJOBS, then press F14



Task 34- – Create a class for jobs running at priority 30

CRTCLS CLS(SBSCLASSXX/CLASS30) RUNPTY(30) TIMESLICE(2000) DFTWAIT(30)

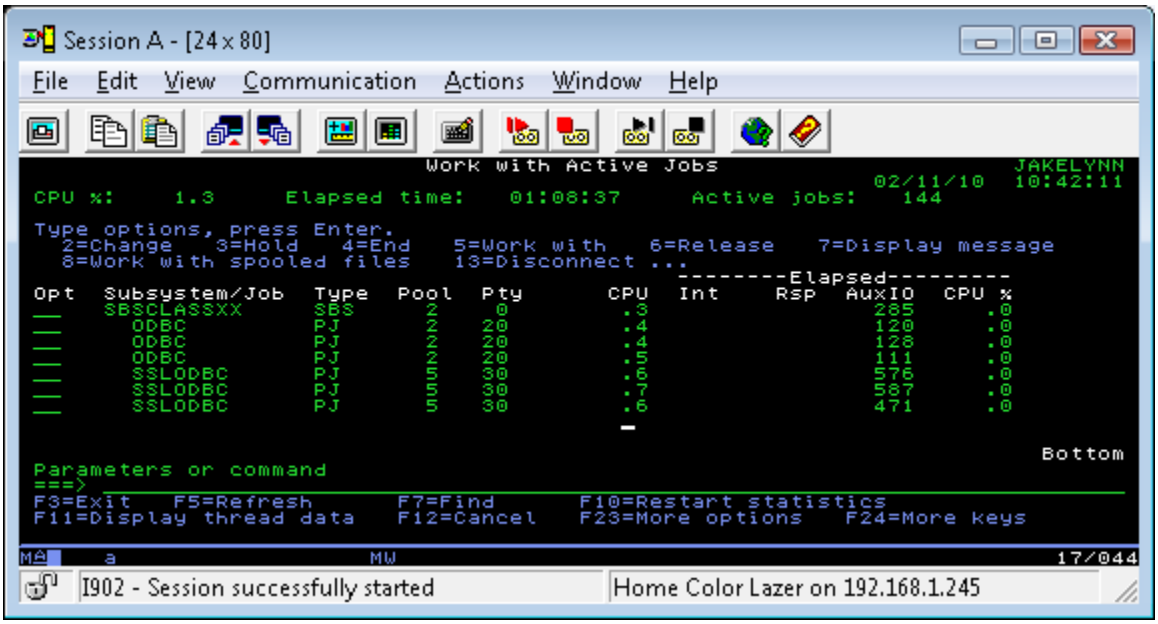
Task 35- – Add a prestart job for SSL ODBC

ADDPJE SBSD(SBSCLASSXX/SBSCLASSXX) PGM(QSYS/QZDASSINIT) INLJOBS(3) JOB(SSLODBC)

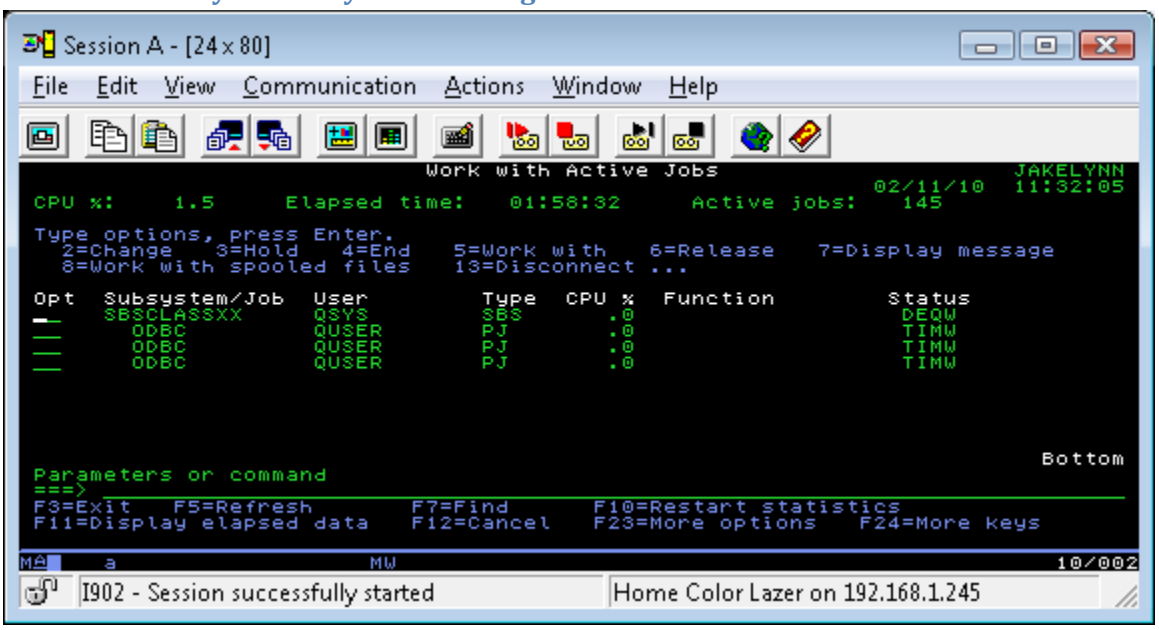
POOLID(2) CLS(SBSCLASSXX/CLASS30)

Task 36- – Test that the jobs are started

DSPSBSJOBS, then press F14



Task 38- – Test your subsystem is being used



Task 40 – Add a routing entry to use your new initial program

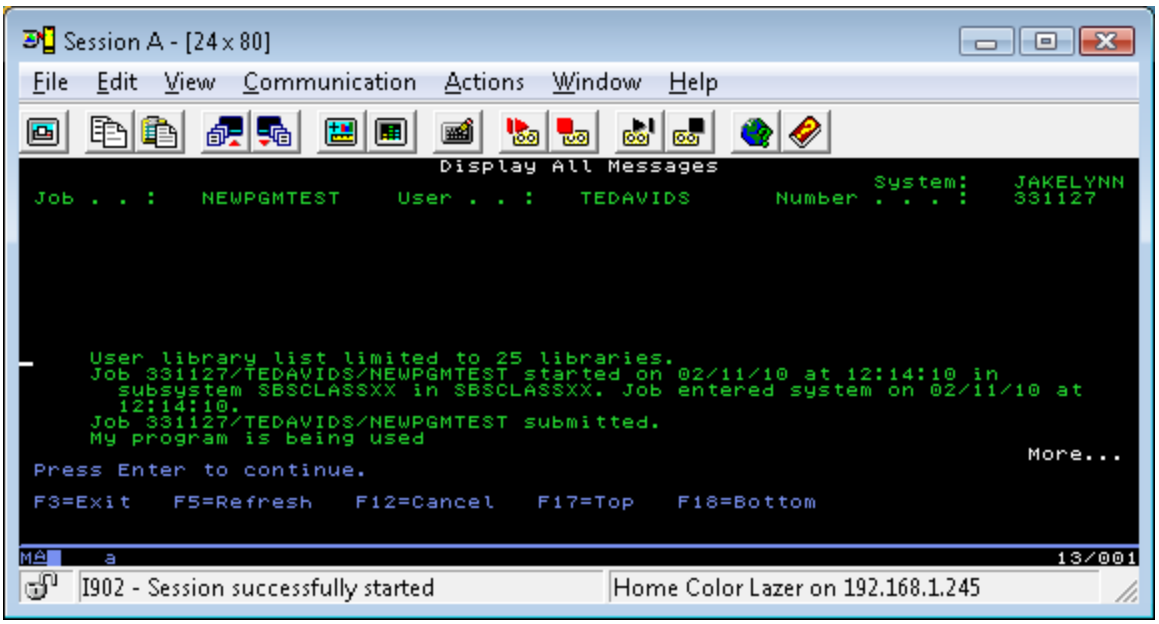
ADDRTGE SBSD(SBSCLASSXX/SBSCLASSXX) SEQNBR(200) CMPVAL('NEWPGM')

PGM(SBSCLASSXX/MYINTLPGM) CLS(SBSCLASSXX/CLASS50) MAXACT(*NOMAX) POOLID(1)

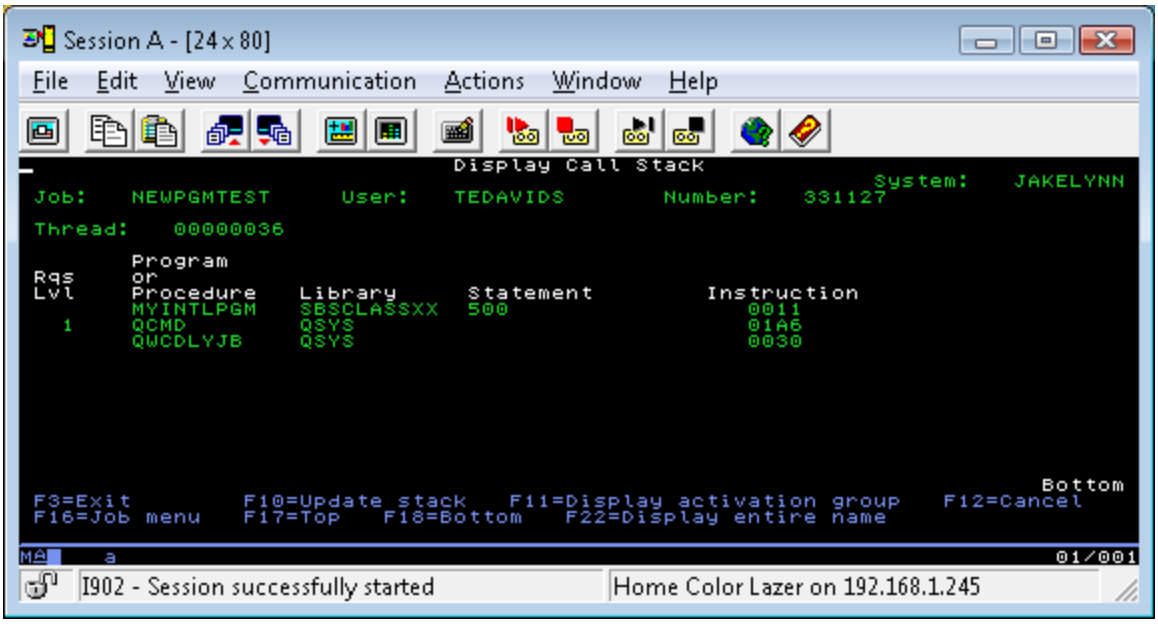
Task 41 – Test your routing entry and program

SBMJOB CMD(DLYJOB DLY(120)) JOB(NEWPGMTEST) JOBQ(BATCHJOBS) RTGDTA(NEWPGM)

Here is the joblog



And invocation stack



Instructor notes: set up ODBC routing before class begins