Load testing PeopleSoft: Using JMeter, PTF, OEM, and shell scripts to CONNECT the dots

Geoff Putney
July 18, 2019
Session 101630

PeopleSoft RECONNECT

#PSRECONNECT

UNIVERSITY OF Nebraska
LINCOLN | OMAHA | KEARNEY | MEDICAL CENTER

Nebraska State College System
CHADRON | PERU | WAYNE

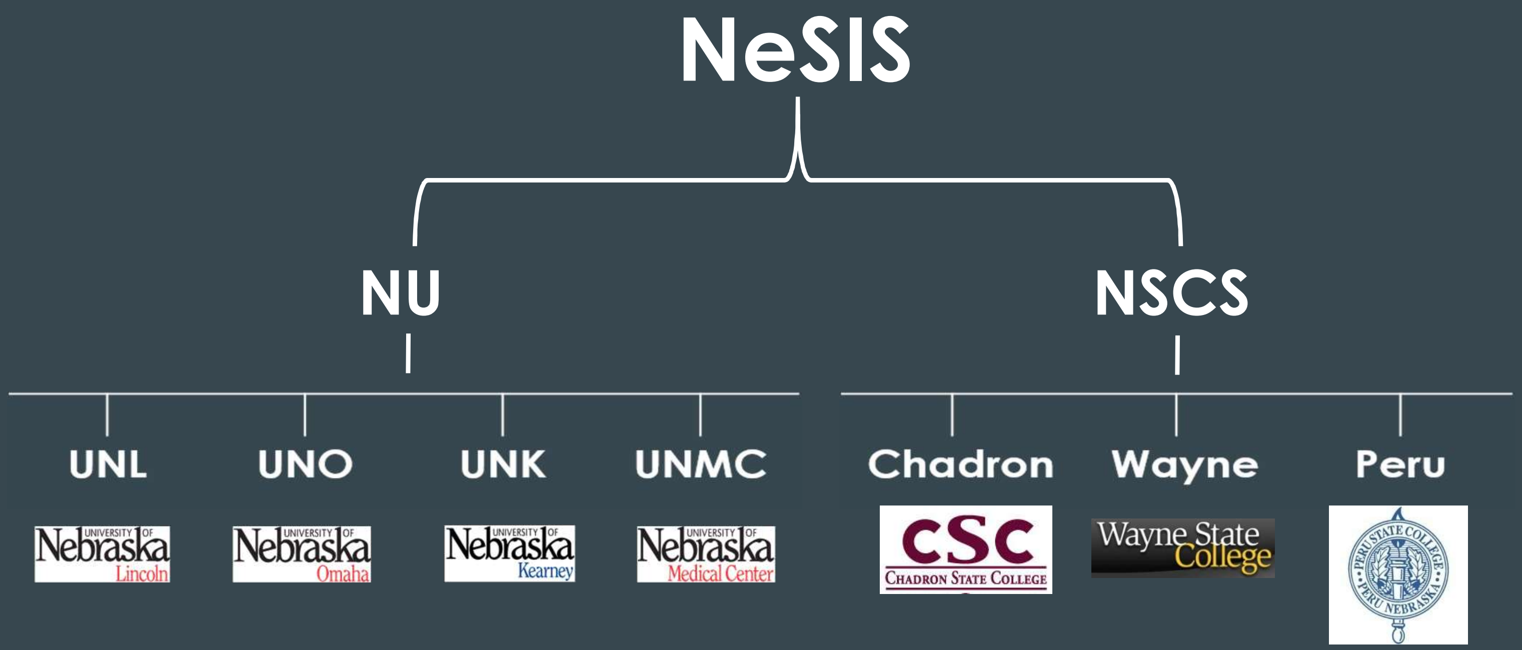
Geoff Putney

System Administrator
University of Nebraska
gputney@nebraska.edu

Agenda

- Nebraska Student Information System (NeSIS)
- System architecture overview
- Load testing methodology
  - Scenarios
  - Software – Jmeter, PTF
  - Monitoriing – Shell scripts, OEM
- Results
- Lessons Learned
- Questions

Nebraska Student Information System (NeSIS)



NeSIS

NU

NSCS

UNL

UNO

UNK

UNMC

Chadron

Wayne

Peru

University of Nebraska Lincoln
University of Nebraska Omaha
University of Nebraska Kearney
University of Nebraska Medical Center
CSC
Chadron State College
Wayne State College
Peru State College
Peru Nebraska

Quick facts

- 7 institutions
- University of Nebraska and Nebraska State College Systems
- 60,000+ students
- Campus Solutions 9.2 PUM 13
- PeopleTools 8.56.17

System Architecture

On-Premise

- Production
  - 5 Dell R720
    - 2x18 core CPU
    - 512 GB memory
      - 5 application hosts
        - 128GB memory
        - 36 CPU threads (18 cores hyperthreaded)
      - 5 web hosts
        - 40GB memory
        - 4 CPU threads
      - Database
        - Oracle C@C Exadata

- QA
  - 2 Dell R720
    - 2x18 core CPU
    - 512 GB memory
      - 2 app/web hosts
        - 128GB memory
        - 36 CPU threads (18 cores hyperthreaded)
      - Database
        - Oracle C@C Exadata

We estimate QA to be ~40% of production capacity

Databases – Oracle Cloud at Customer Exadata

- Production in Lincoln, DR and non-production in Omaha
  - 22 CPUs
  - 1440GB of Memory
  - 76TB Flash Storages
  - 106TB of Spinning Disk

Chadron
Valentine
RESERVATION
Yankton
Vermillion
Le Mars
Sioux City
Wayne
Norfolk
29
Alliance
Scottsbluff
NEBRASKA
Broken Bow
Columbus
Omaha
Papillion
80
Sidney
Ogallala
North Platte
80
Gothenburg
Cozad
Lexington
Grand Island
York
80
Lincoln
Curtis
Kearney
Peru
76
Sterling
Holdrege
Beatrice
McCook

PIAs and App Servers During Testing

- 4 PIA sites
  - 1 integration gateway, 3 campus sites
- 3 app servers
  - 1 for integration gateway
  - 1 for 2 campus sites
  - 1 for 1 campus site

Other architecture

- On premise F5 load balancer
- On premise load testing servers
  - 2 servers, 4 cpus, 16GB memory
- On premise authentication

Load Testing Methodology

Load Testing

- Why do we load test?

Scenarios

- We load test at least every quarter
    - PUM
    - PeopleTools Patch
    - PeopleTools Upgrade
    - and also with any major modifications to the system...
        - Federation (Appsian formerly Grey Heller)
        - Exadata
        - Oracle Classic Compute trial
        - Etc...

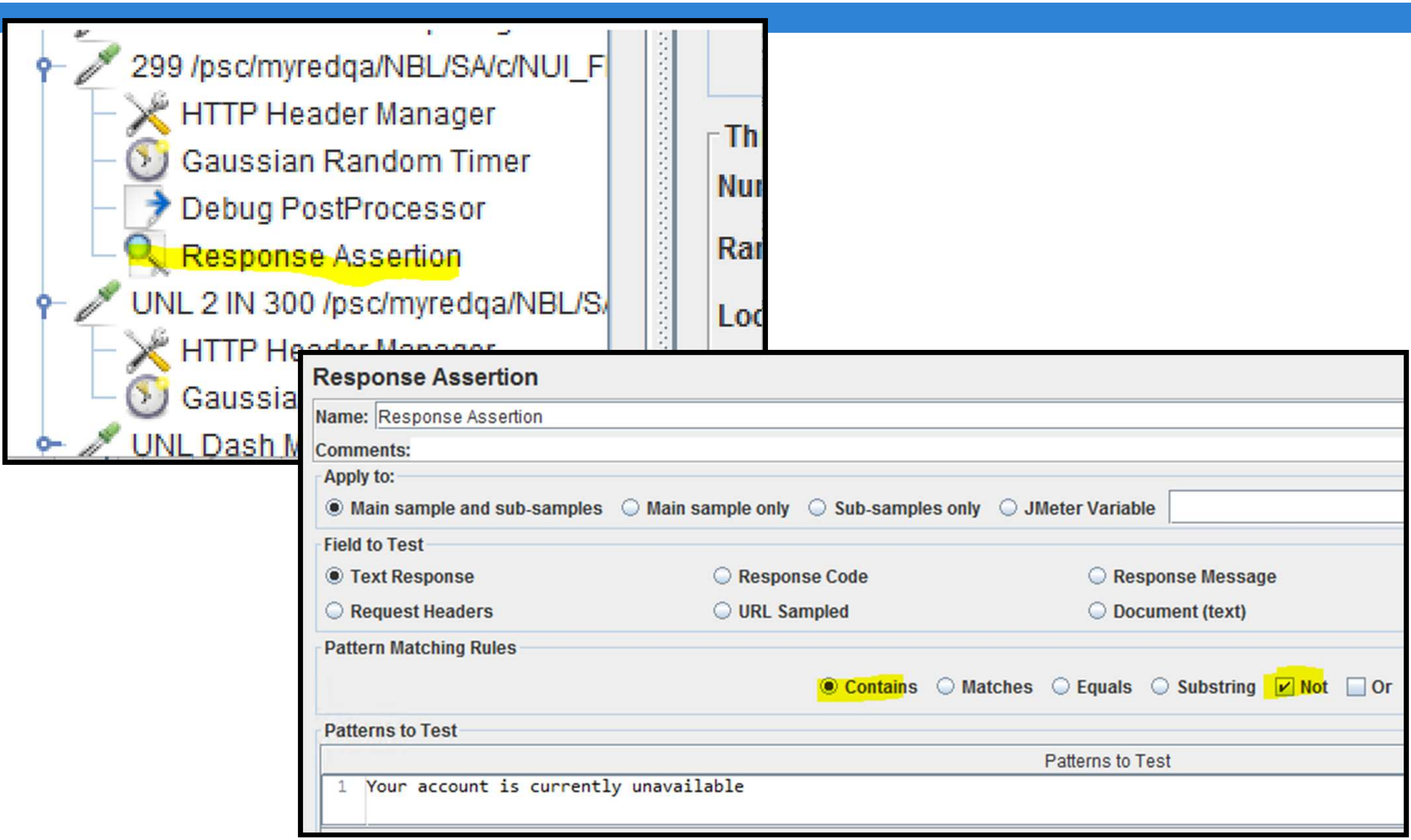
Scenarios

- Enrollment (drop/add) and first day of classes
  - For any PT patch, PUM, or upgrade
- Evaluate what we're trying to test for one off items
  - For example, with federation, we mainly focused on logging in to the system
  - Design a test to match that requirement
- Almost always from the student perspective
  - We have custom bolt-on student/faculty/advisor dashboards

Load Testing Software

- Apache JMeter
  - Open source
  - Command line and GUI options
  - Highly configurable
    - Random timers to mimic user wait times
    - Variety of regex validation and variable assignment
      - Capture necessary session variables
  - Takes some investment to get the tests working as expected
  - Increase the heap size

*Attached a generic template JMeter script on HEUG



- 299 /psc/myredqa/NBL/SA/c/NUI_F
  - HTTP Header Manager
  - Gaussian Random Timer
  - Debug PostProcessor
  - Response Assertion
- UNL 2 IN 300 /psc/myredqa/NBL/S
  - HTTP Header Manager
  - Gaussia
- UNL Dash M

Response Assertion

Name: Response Assertion

Comments:

Apply to:

● Main sample and sub-samples ○ Main sample only ○ Sub-samples only ○ JMeter Variable

Field to Test

● Text Response ○ Response Code ○ Response Message

○ Request Headers ○ URL Sampled ○ Document (text)

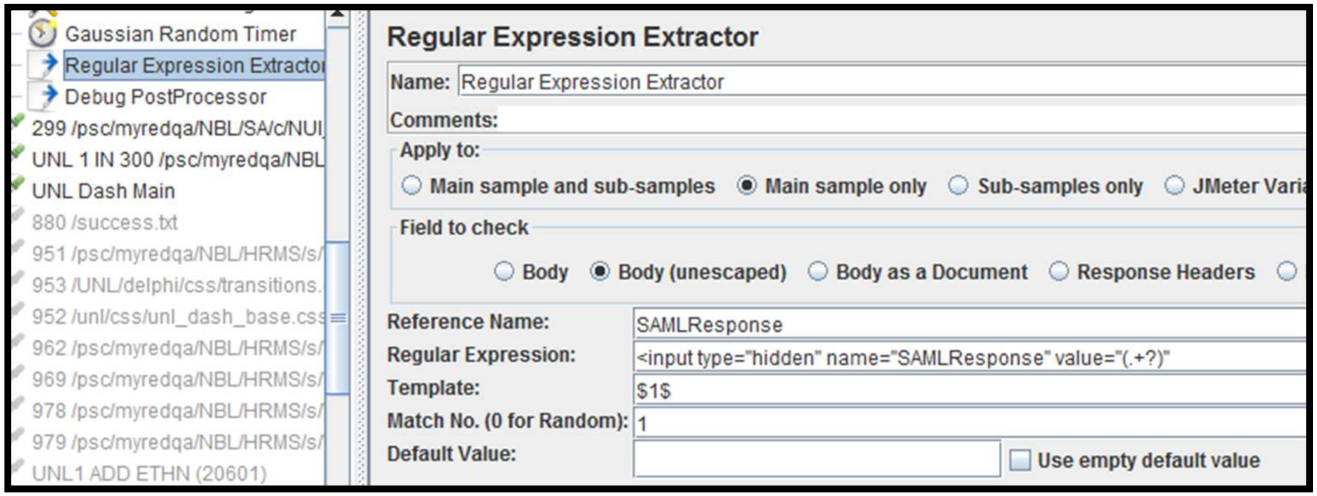
Pattern Matching Rules

● Contains ○ Matches ○ Equals ○ Substring ☑ Not □ Or

Patterns to Test

Patterns to Test

```
Your account is currently unavailable
```



- Gaussian Random Timer
- Regular Expression Extractor
- Debug PostProcessor
- 299 /psc/myredqa/NBL/SA/c/NUI
- UNL 1 IN 300 /psc/myredqa/NBL
- UNL Dash Main
- 880 /success.txt
- 951 /psc/myredqa/NBL/HRMS/s/
- 953 /UNL/delphi/css/transitions.
- 952 /unl/css/unl_dash_base.css
- 962 /psc/myredqa/NBL/HRMS/s/
- 969 /psc/myredqa/NBL/HRMS/s/
- 978 /psc/myredqa/NBL/HRMS/s/
- 979 /psc/myredqa/NBL/HRMS/s/
- UNL1 ADD ETHN (20601)

Regular Expression Extractor

Name: Regular Expression Extractor

Comments:

Apply to:
○ Main sample and sub-samples ● Main sample only ○ Sub-samples only ○ JMeter Varia

Field to check
○ Body ● Body (unescaped) ○ Body as a Document ○ Response Headers ○

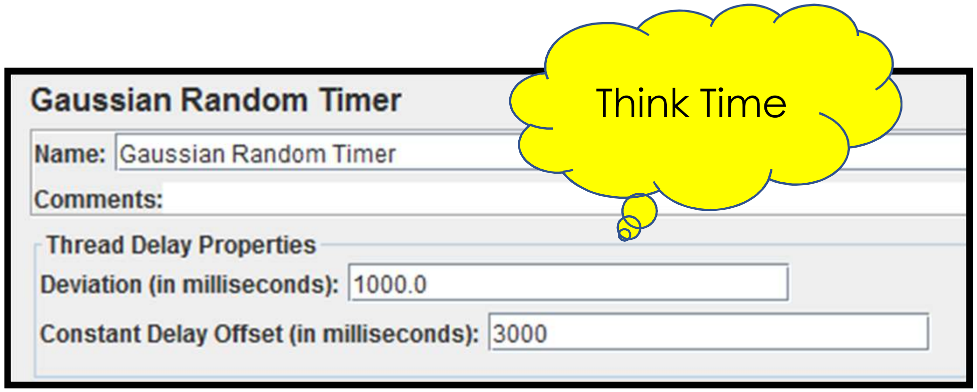
Reference Name: SAMLResponse

Regular Expression: <input type="hidden" name="SAMLResponse" value="(.+?)"

Template: 1

Match No. (0 for Random): 1

Default Value: □ Use empty default value



Gaussian Random Timer

Name: Gaussian Random Timer

Comments:

Thread Delay Properties

Deviation (in milliseconds): 1000.0

Constant Delay Offset (in milliseconds): 3000

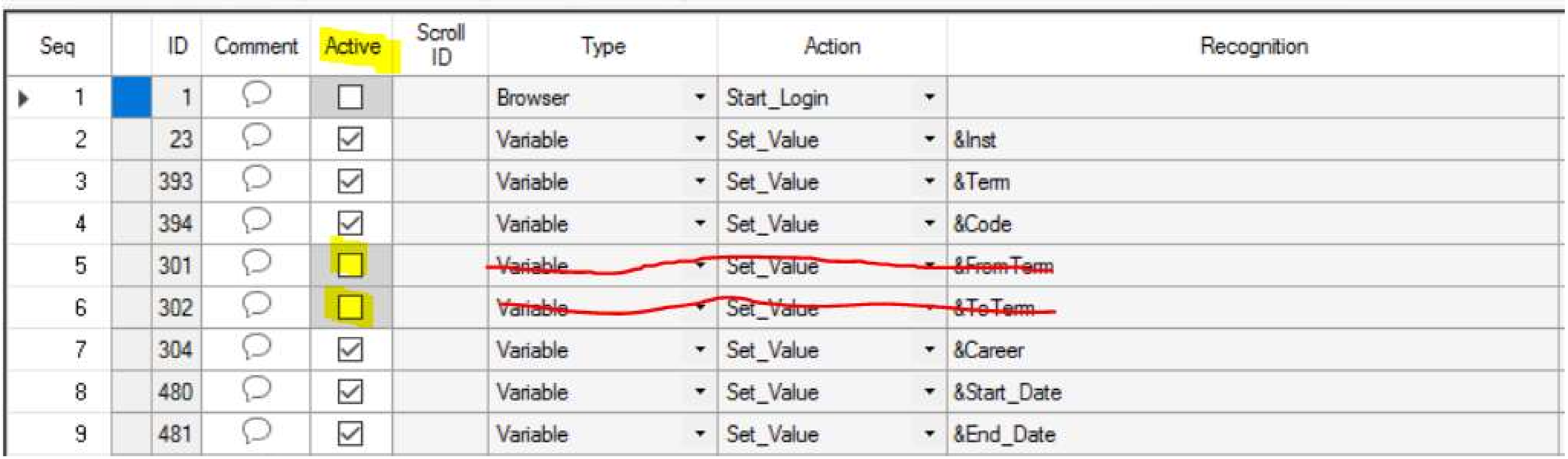
Think Time

Load Testing Software

- PeopleSoft Test Framework
  - Created scripts to:
    - Opens specific entry level courses/labs for enrollment
      - Avoid prereqs
    - Allow for taking a class more than once
      - Many students have already taken the 100 level courses
    - Increase the number of students that can be in the class to 9999

PTF – “Driver” script

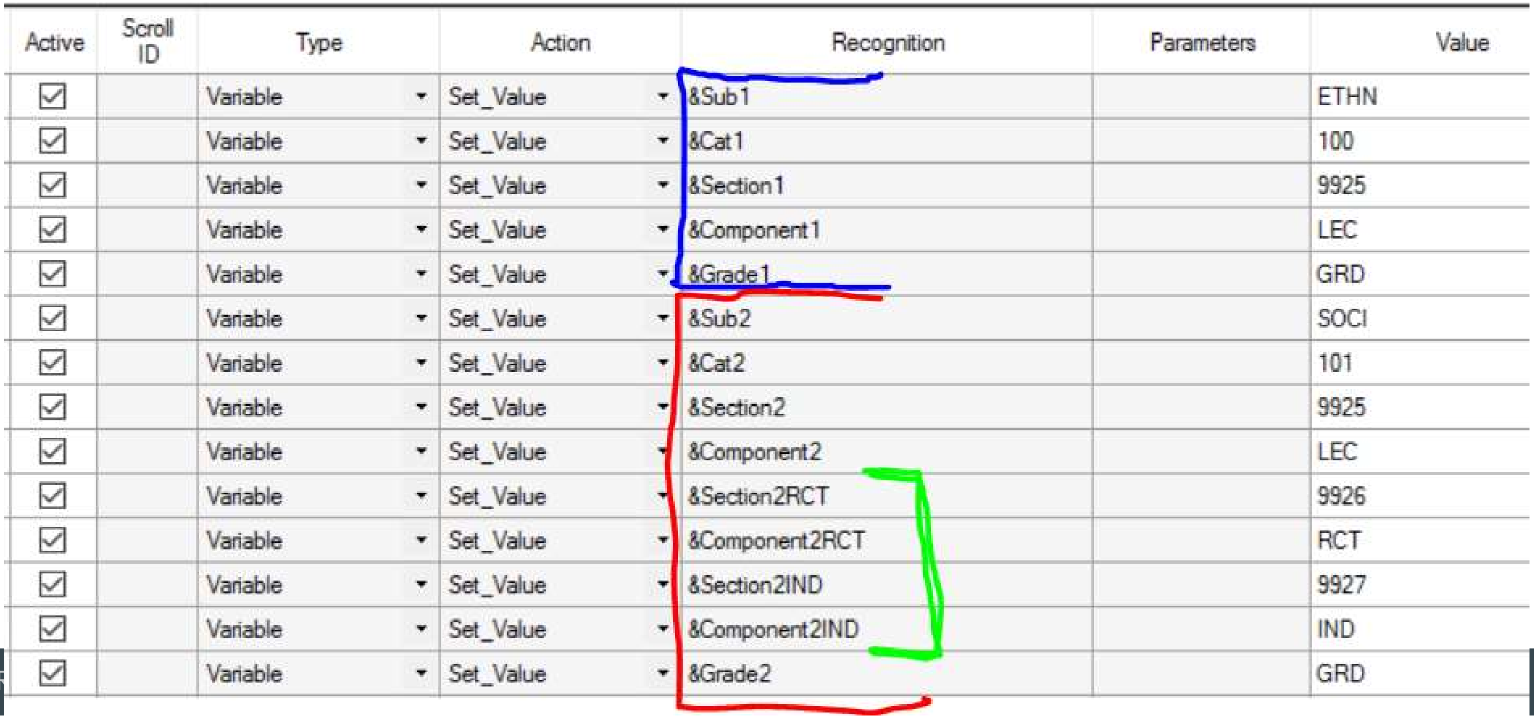
- Set variables for global use
- Call sub tests that utilize the variables for other setup
  - Institution, Term, Code, Career, Start Date, End Date



Seq		ID	Comment	Active	Scroll ID	Type	Action	Recognition
▸ 1		1	💬	☐		Browser	Start_Login	
2		23	💬	☑		Variable	Set_Value	&Inst
3		393	💬	☑		Variable	Set_Value	&Term
4		394	💬	☑		Variable	Set_Value	&Code
5		301	💬	☐		~~Variable~~	~~Set_Value~~	~~&FromTerm~~
6		302	💬	☐		~~Variable~~	~~Set_Value~~	~~&ToTerm~~
7		304	💬	☑		Variable	Set_Value	&Career
8		480	💬	☑		Variable	Set_Value	&Start_Date
9		481	💬	☑		Variable	Set_Value	&End_Date

PTF – “Driver” script

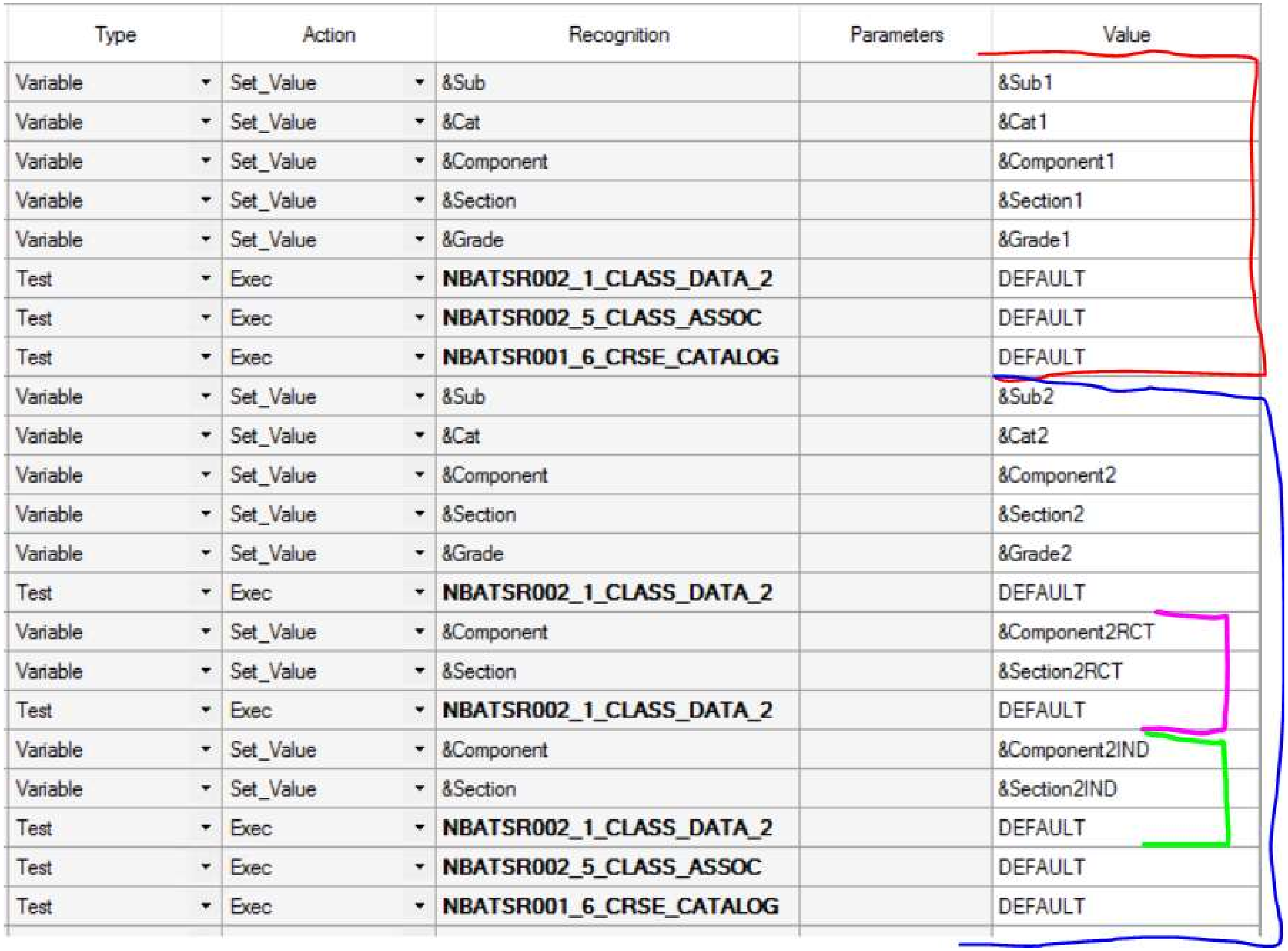
- Set variables for each class to use in sub tests
    - A little painful at first, but much easier to maintain. We have 10 classes worth of variables configured.



Active	Scroll ID	Type	Action	Recognition	Parameters	Value
☑		Variable	Set_Value	&Sub1		ETHN
☑		Variable	Set_Value	&Cat1		100
☑		Variable	Set_Value	&Section1		9925
☑		Variable	Set_Value	&Component1		LEC
☑		Variable	Set_Value	&Grade1		GRD
☑		Variable	Set_Value	&Sub2		SOCI
☑		Variable	Set_Value	&Cat2		101
☑		Variable	Set_Value	&Section2		9925
☑		Variable	Set_Value	&Component2		LEC
☑		Variable	Set_Value	&Section2RCT		9926
☑		Variable	Set_Value	&Component2RCT		RCT
☑		Variable	Set_Value	&Section2IND		9927
☑		Variable	Set_Value	&Component2IND		IND
☑		Variable	Set_Value	&Grade2		GRD

PTF – “Driver” Script

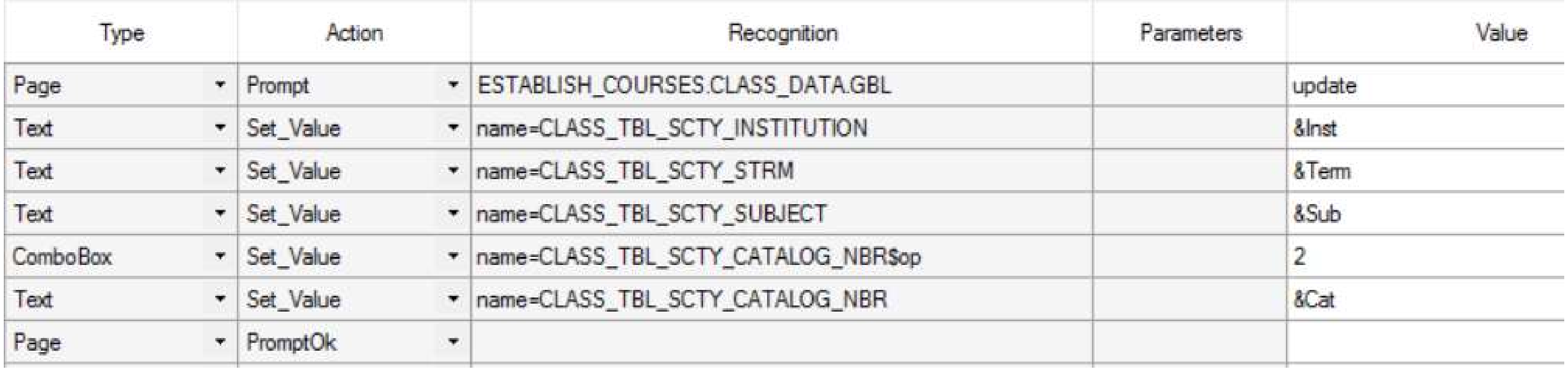
- Running the sub tests with the global variables
- One gotcha – the tests expect certain variable names, so make sure they are assigned correctly ahead of the test



Type	Action	Recognition	Parameters	Value
Variable	Set_Value	&Sub		&Sub1
Variable	Set_Value	&Cat		&Cat1
Variable	Set_Value	&Component		&Component1
Variable	Set_Value	&Section		&Section1
Variable	Set_Value	&Grade		&Grade1
Test	Exec	**NBATSR002_1_CLASS_DATA_2**		DEFAULT
Test	Exec	**NBATSR002_5_CLASS_ASSOC**		DEFAULT
Test	Exec	**NBATSR001_6_CRSE_CATALOG**		DEFAULT
Variable	Set_Value	&Sub		&Sub2
Variable	Set_Value	&Cat		&Cat2
Variable	Set_Value	&Component		&Component2
Variable	Set_Value	&Section		&Section2
Variable	Set_Value	&Grade		&Grade2
Test	Exec	**NBATSR002_1_CLASS_DATA_2**		DEFAULT
Variable	Set_Value	&Component		&Component2RCT
Variable	Set_Value	&Section		&Section2RCT
Test	Exec	**NBATSR002_1_CLASS_DATA_2**		DEFAULT
Variable	Set_Value	&Component		&Component2IND
Variable	Set_Value	&Section		&Section2IND
Test	Exec	**NBATSR002_1_CLASS_DATA_2**		DEFAULT
Test	Exec	**NBATSR002_5_CLASS_ASSOC**		DEFAULT
Test	Exec	**NBATSR001_6_CRSE_CATALOG**		DEFAULT

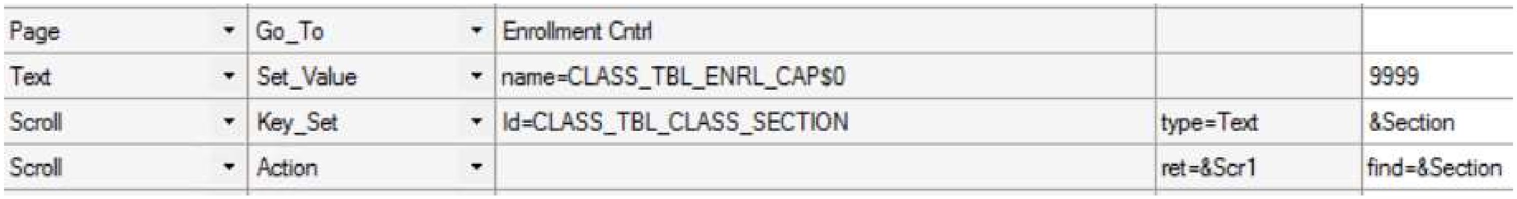
PTF – Class setup sub test

- Create course based on variables passed
- Set capacity to 9999



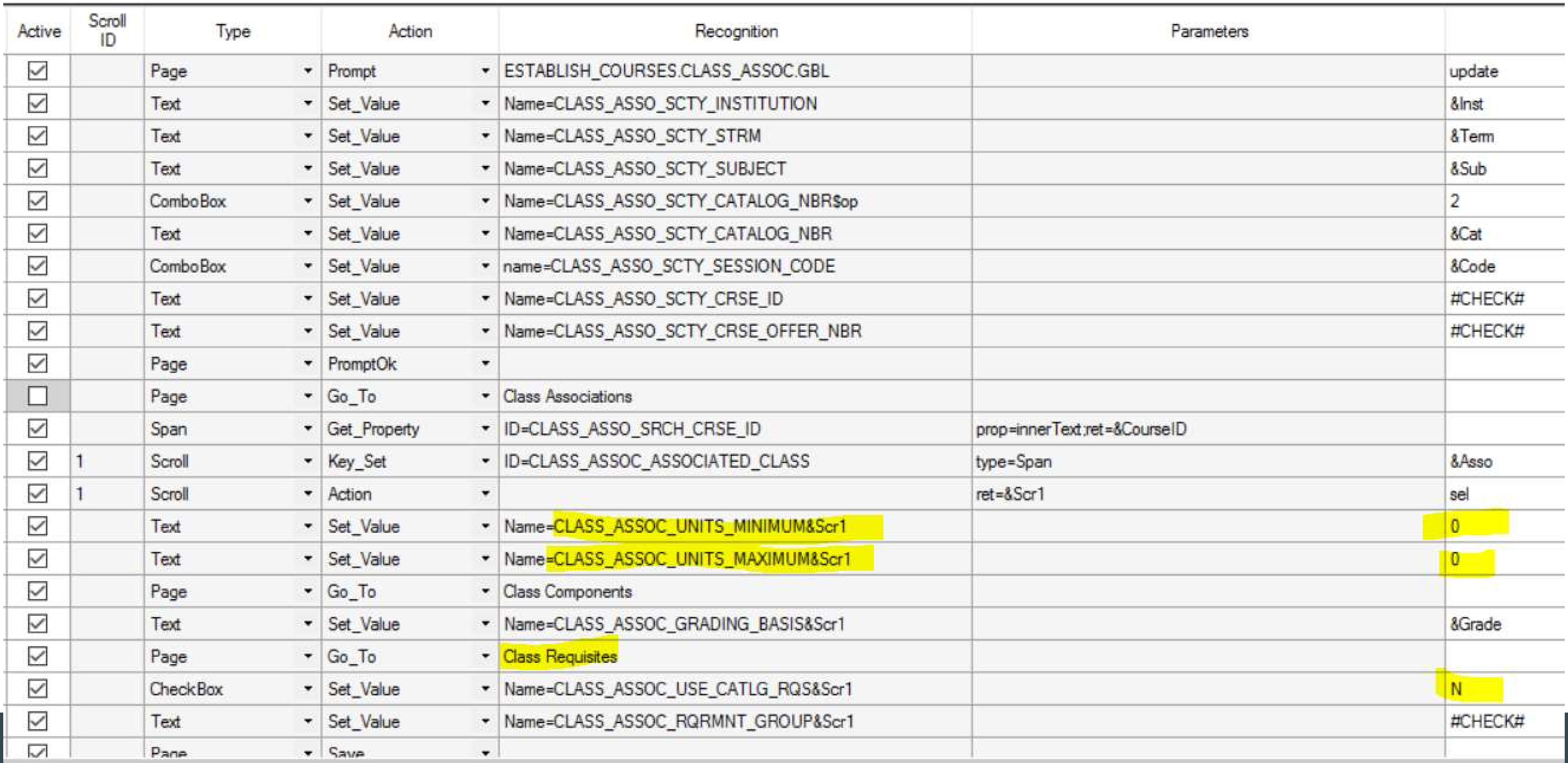
Type	Action	Recognition	Parameters	Value
Page	Prompt	ESTABLISH_COURSES.CLASS_DATA.GBL		update
Text	Set_Value	name=CLASS_TBL_SCTY_INSTITUTION		&Inst
Text	Set_Value	name=CLASS_TBL_SCTY_STRM		&Term
Text	Set_Value	name=CLASS_TBL_SCTY_SUBJECT		&Sub
ComboBox	Set_Value	name=CLASS_TBL_SCTY_CATALOG_NBR$op		2
Text	Set_Value	name=CLASS_TBL_SCTY_CATALOG_NBR		&Cat
Page	PromptOk			

• • •



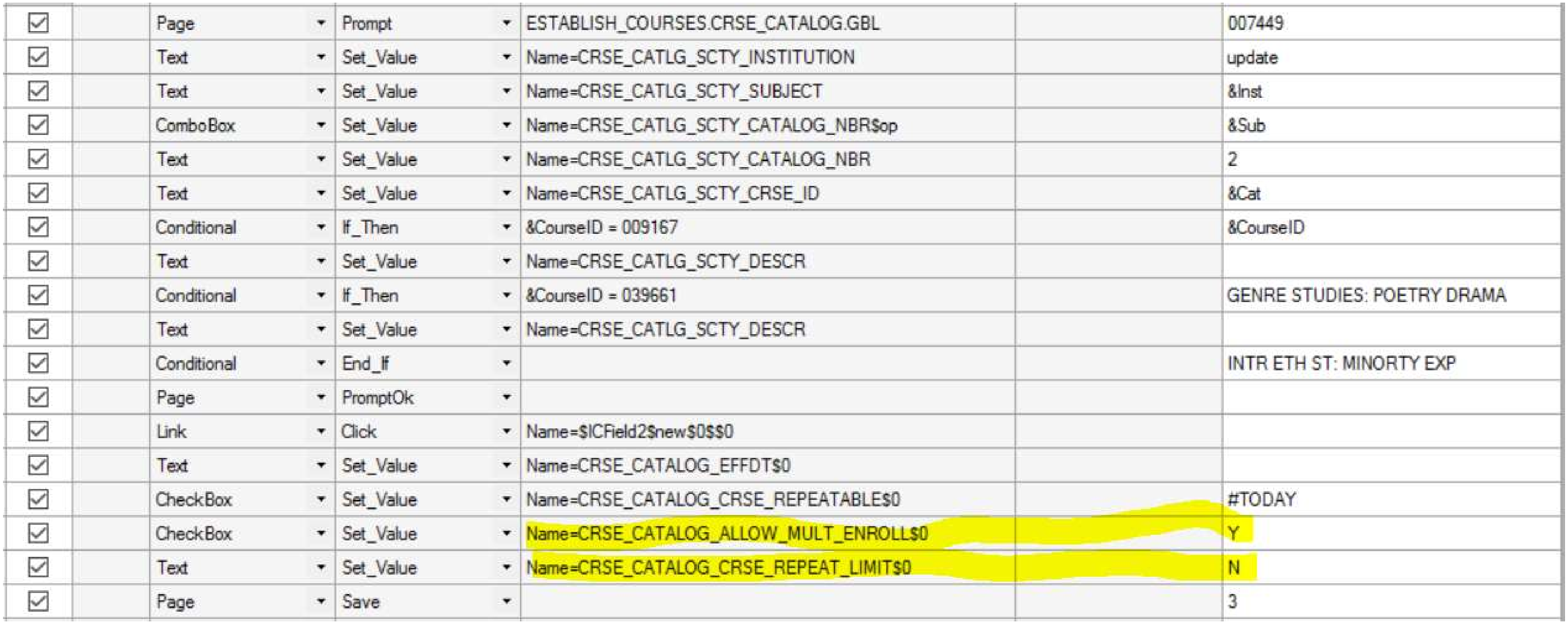
Page	Go_To	Enrollment Cntrl		
Text	Set_Value	name=CLASS_TBL_ENRL_CAP$0		9999
Scroll	Key_Set	Id=CLASS_TBL_CLASS_SECTION	type=Text	&Section
Scroll	Action		ret=&Scr1	find=&Section

PTF – Class association sub test



Active	Scroll ID	Type	Action	Recognition	Parameters	
☑		Page	Prompt	ESTABLISH_COURSES.CLASS_ASSOC.GBL		update
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_INSTITUTION		&Inst
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_STRM		&Term
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_SUBJECT		&Sub
☑		ComboBox	Set_Value	Name=CLASS_ASSO_SCTY_CATALOG_NBR$op		2
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_CATALOG_NBR		&Cat
☑		ComboBox	Set_Value	name=CLASS_ASSO_SCTY_SESSION_CODE		&Code
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_CRSE_ID		#CHECK#
☑		Text	Set_Value	Name=CLASS_ASSO_SCTY_CRSE_OFFER_NBR		#CHECK#
☑		Page	PromptOk			
□		Page	Go_To	Class Associations		
☑		Span	Get_Property	ID=CLASS_ASSO_SRCH_CRSE_ID	prop=innerText;ret=&CourseID	
☑	1	Scroll	Key_Set	ID=CLASS_ASSOC_ASSOCIATED_CLASS	type=Span	&Asso
☑	1	Scroll	Action		ret=&Scr1	sel
☑		Text	Set_Value	Name=CLASS_ASSOC_UNITS_MINIMUM&Scr1		0
☑		Text	Set_Value	Name=CLASS_ASSOC_UNITS_MAXIMUM&Scr1		0
☑		Page	Go_To	Class Components		
☑		Text	Set_Value	Name=CLASS_ASSOC_GRADING_BASIS&Scr1		&Grade
☑		Page	Go_To	Class Requisites		
☑		CheckBox	Set_Value	Name=CLASS_ASSOC_USE_CATLG_RQS&Scr1		N
☑		Text	Set_Value	Name=CLASS_ASSOC_RQRMNT_GROUP&Scr1		#CHECK#
☑		Page	Save			

PTF – Course catalog sub test



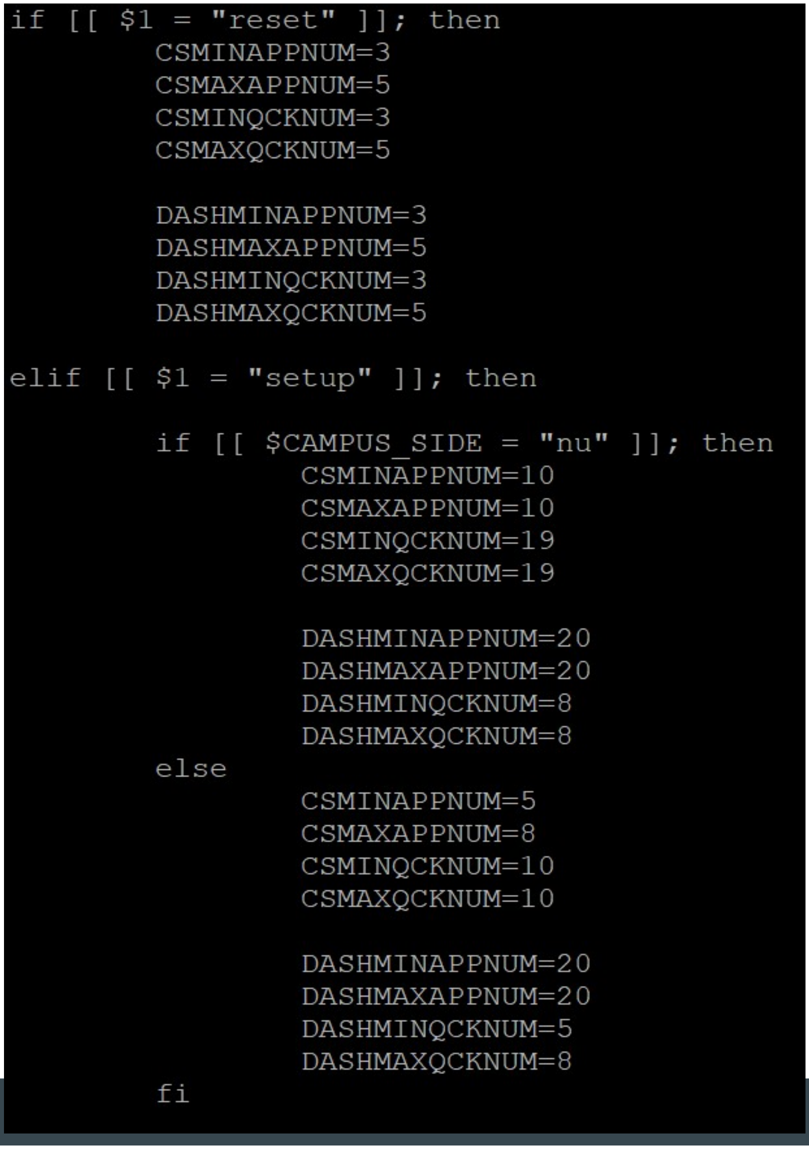
☑		Page	Prompt	ESTABLISH_COURSES.CRSE_CATALOG.GBL		007449
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_INSTITUTION		update
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_SUBJECT		&Inst
☑		ComboBox	Set_Value	Name=CRSE_CATLG_SCTY_CATALOG_NBR$op		&Sub
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_CATALOG_NBR		2
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_CRSE_ID		&Cat
☑		Conditional	If_Then	&CourseID = 009167		&CourseID
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_DESCR		
☑		Conditional	If_Then	&CourseID = 039661		GENRE STUDIES: POETRY DRAMA
☑		Text	Set_Value	Name=CRSE_CATLG_SCTY_DESCR		
☑		Conditional	End_If			INTR ETH ST: MINORTY EXP
☑		Page	PromptOk			
☑		Link	Click	Name=$ICField2$new$0$$0		
☑		Text	Set_Value	Name=CRSE_CATALOG_EFFDT$0		
☑		CheckBox	Set_Value	Name=CRSE_CATALOG_CRSE_REPEATABLE$0		#TODAY
☑		CheckBox	Set_Value	Name=CRSE_CATALOG_ALLOW_MULT_ENROLL$0		Y
☑		Text	Set_Value	Name=CRSE_CATALOG_CRSE_REPEAT_LIMIT$0		N
☑		Page	Save			3

PTF

- Phew….I'm glad someone else set all that up!
- However, it does have a lot of benefits
  - Saves us a tremendous amount time for manual set up each time we load test.
  - It also helps reduce human error related to setting up all those courses manually.
  - Load tests are much cleaner and have fewer errors.
  - More confidence in our load testing results because we know we have consistency.

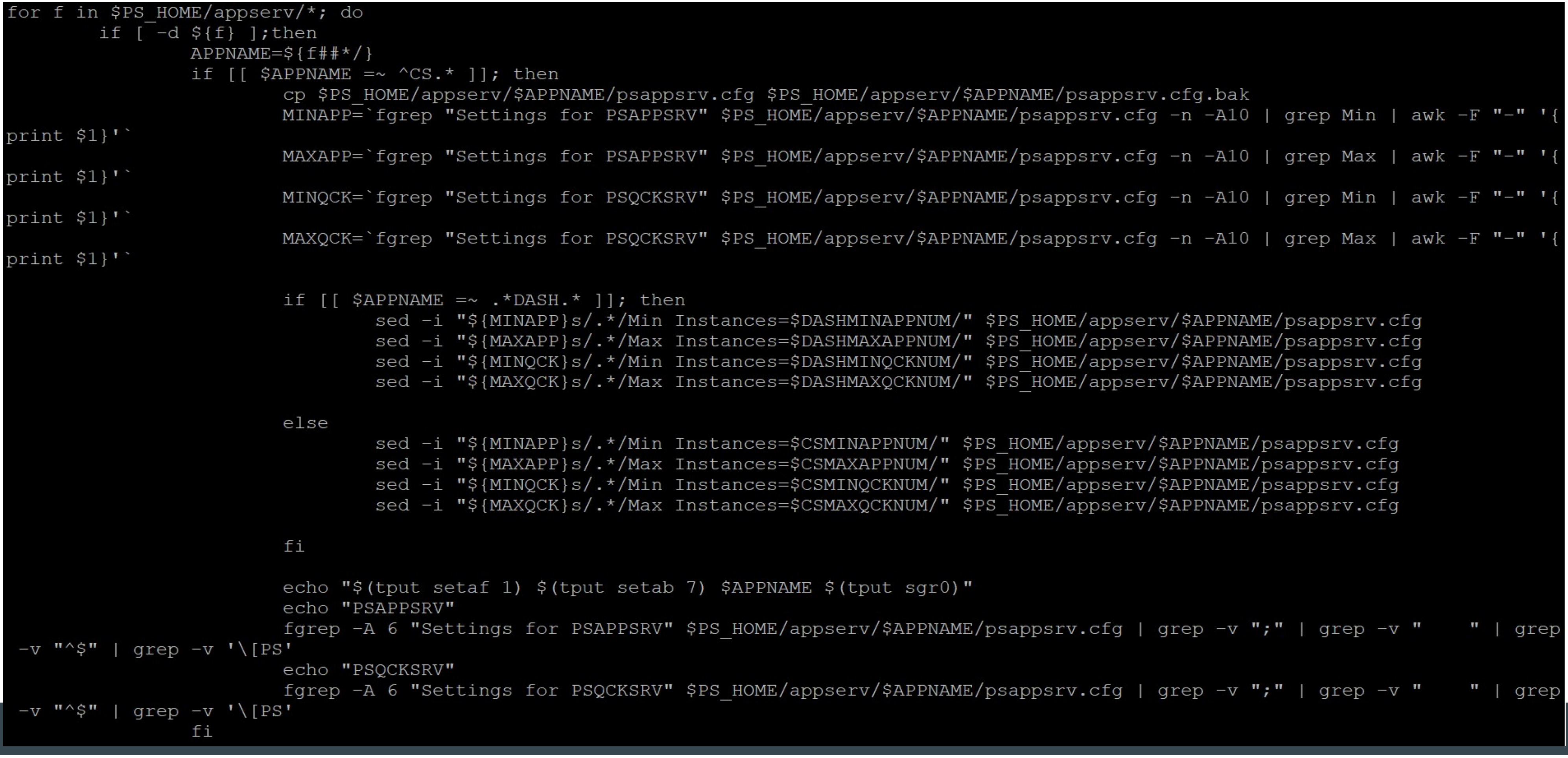
Load Test Server Setup

- Users don’t like it when we leave our app servers configured for production load, so we made a script to setup and reset our app server min/max.



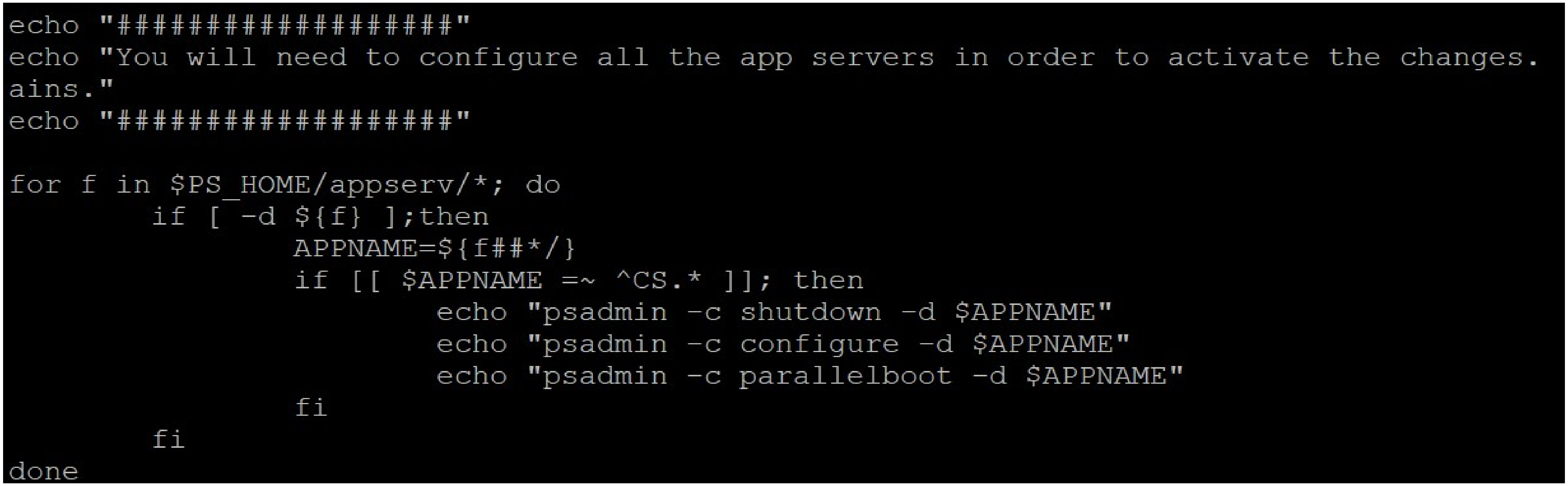
```
if [[ $1 = "reset" ]]; then
        CSMINAPPNUM=3
        CSMAXAPPNUM=5
        CSMINQCKNUM=3
        CSMAXQCKNUM=5

        DASHMINAPPNUM=3
        DASHMAXAPPNUM=5
        DASHMINQCKNUM=3
        DASHMAXQCKNUM=5

elif [[ $1 = "setup" ]]; then

        if [[ $CAMPUS_SIDE = "nu" ]]; then
                CSMINAPPNUM=10
                CSMAXAPPNUM=10
                CSMINQCKNUM=19
                CSMAXQCKNUM=19

                DASHMINAPPNUM=20
                DASHMAXAPPNUM=20
                DASHMINQCKNUM=8
                DASHMAXQCKNUM=8
        else
                CSMINAPPNUM=5
                CSMAXAPPNUM=8
                CSMINQCKNUM=10
                CSMAXQCKNUM=10

                DASHMINAPPNUM=20
                DASHMAXAPPNUM=20
                DASHMINQCKNUM=5
                DASHMAXQCKNUM=8
        fi
```

Load Test Server Setup



```
for f in $PS_HOME/appserv/*; do
        if [ -d ${f} ];then
                APPNAME=${f##*/}
                if [[ $APPNAME =~ ^CS.* ]]; then
                        cp $PS_HOME/appserv/$APPNAME/psappsrv.cfg $PS_HOME/appserv/$APPNAME/psappsrv.cfg.bak
                        MINAPP=`fgrep "Settings for PSAPPSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg -n -A10 | grep Min | awk -F "-" '{
print $1}'`
                        MAXAPP=`fgrep "Settings for PSAPPSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg -n -A10 | grep Max | awk -F "-" '{
print $1}'`
                        MINQCK=`fgrep "Settings for PSQCKSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg -n -A10 | grep Min | awk -F "-" '{
print $1}'`
                        MAXQCK=`fgrep "Settings for PSQCKSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg -n -A10 | grep Max | awk -F "-" '{
print $1}'`


                        if [[ $APPNAME =~ .*DASH.* ]]; then
                                sed -i "${MINAPP}s/.*/Min Instances=$DASHMINAPPNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MAXAPP}s/.*/Max Instances=$DASHMAXAPPNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MINQCK}s/.*/Min Instances=$DASHMINQCKNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MAXQCK}s/.*/Max Instances=$DASHMAXQCKNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg

                        else
                                sed -i "${MINAPP}s/.*/Min Instances=$CSMINAPPNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MAXAPP}s/.*/Max Instances=$CSMAXAPPNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MINQCK}s/.*/Min Instances=$CSMINQCKNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg
                                sed -i "${MAXQCK}s/.*/Max Instances=$CSMAXQCKNUM/" $PS_HOME/appserv/$APPNAME/psappsrv.cfg

                        fi

                        echo "$(tput setaf 1) $(tput setab 7) $APPNAME $(tput sgr0)"
                        echo "PSAPPSRV"
                        fgrep -A 6 "Settings for PSAPPSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg | grep -v ";" | grep -v "    " | grep
 -v "^$" | grep -v '\[PS'
                        echo "PSQCKSRV"
                        fgrep -A 6 "Settings for PSQCKSRV" $PS_HOME/appserv/$APPNAME/psappsrv.cfg | grep -v ";" | grep -v "    " | grep
 -v "^$" | grep -v '\[PS'
                fi
```

Load Test Server Setup



```
echo "###################"
echo "You will need to configure all the app servers in order to activate the changes.
ains."
echo "###################"

for f in $PS_HOME/appserv/*; do
        if [ -d ${f} ];then
                APPNAME=${f##*/}
                if [[ $APPNAME =~ ^CS.* ]]; then
                        echo "psadmin -c shutdown -d $APPNAME"
                        echo "psadmin -c configure -d $APPNAME"
                        echo "psadmin -c parallelboot -d $APPNAME"
                fi
        fi
done
```

Load Testing Population

- Undergraduate Students
- Eligible to enroll
- Don't have staff roles or elevated access
- Don't have service indicators preventing them from enrolling
- Security team gathers population and exports it as an excel file
  - We convert to csv so jMeter can pull from it easily

*Our query is at the end of the slides

Load Tests

Our enrollment tests are for 10 minutes

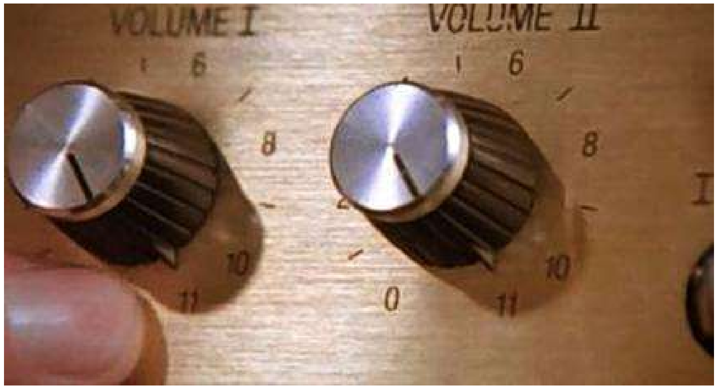
- Warm up
  - 100 users/12 second ramp up
  - Builds cache on the app servers
- Intermediate
  - 200 users/24 second ramp up
- Pretty close to max
  - 300 users/36 second ramp up
- Let’s see what we can break
  - 400 users+/48+ second ramp up

Load Tests Continued

- Jmeter hits each link individually, so we disable images, css, js, which should be cached by most browsers
- We use random timers on each “click”, 1 to 3 seconds
- We use validation methods on some of the steps
  - Login
  - Ensure term activated for the correct term
  - Enrollment success

Standard Targets to Hit

- Logins
  - Queries to determine high average logins per second/minute/hour
  - 8 users/sec, 400 users/min
- Class Add/Drop
  - Query to find the max enrollment add/drops in a minute
  - 900/min for us

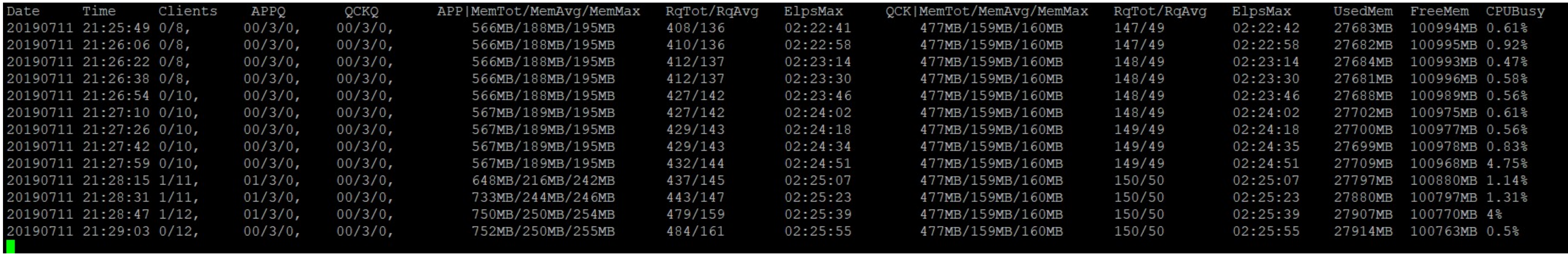


What We Monitor

- OEM to monitor SQL activity in the database
  - DBAs can then tune, verify performance vs production
- We built a custom shell script to run tmadmin commands to check app server queues, CPU utilization
  - https://docs.oracle.com/cd/E13203_01/tuxedo/tux64/sect1/tmadmin.htm
- Look at average, min, and max values for key steps in jMeter
  - Compare them to previous years/quarters

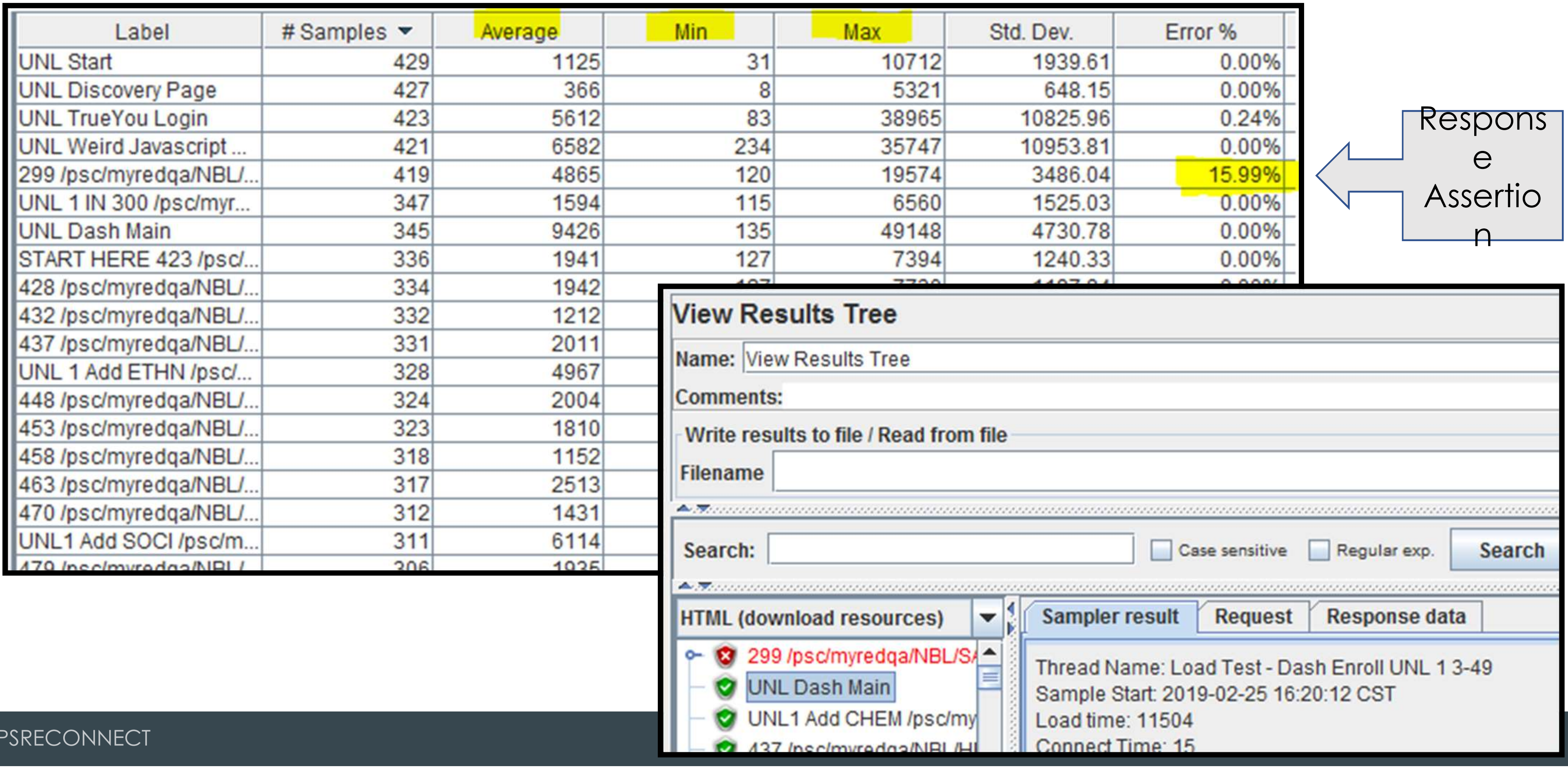
What We Monitor

- We built a custom shell script to run tmadmin commands to check app server queues, CPU utilization
  - https://docs.oracle.com/cd/E13203_01/tuxedo/tux64/sect1/tmadmin.htm



```
Date     Time     Clients    APPQ       QCKQ       APP|MemTot/MemAvg/MemMax    RqTot/RqAvg   ElpsMax     QCK|MemTot/MemAvg/MemMax    RqTot/RqAvg   ElpsMax     UsedMem  FreeMem  CPUBusy
20190711 21:25:49 0/8,      00/3/0,    00/3/0,         566MB/188MB/195MB      408/136       02:22:41        477MB/159MB/160MB      147/49        02:22:42    27683MB  100994MB 0.61%
20190711 21:26:06 0/8,      00/3/0,    00/3/0,         566MB/188MB/195MB      410/136       02:22:58        477MB/159MB/160MB      147/49        02:22:58    27682MB  100995MB 0.92%
20190711 21:26:22 0/8,      00/3/0,    00/3/0,         566MB/188MB/195MB      412/137       02:23:14        477MB/159MB/160MB      148/49        02:23:14    27684MB  100993MB 0.47%
20190711 21:26:38 0/8,      00/3/0,    00/3/0,         566MB/188MB/195MB      412/137       02:23:30        477MB/159MB/160MB      148/49        02:23:30    27681MB  100996MB 0.58%
20190711 21:26:54 0/10,     00/3/0,    00/3/0,         566MB/188MB/195MB      427/142       02:23:46        477MB/159MB/160MB      148/49        02:23:46    27688MB  100989MB 0.56%
20190711 21:27:10 0/10,     00/3/0,    00/3/0,         567MB/189MB/195MB      427/142       02:24:02        477MB/159MB/160MB      148/49        02:24:02    27702MB  100975MB 0.61%
20190711 21:27:26 0/10,     00/3/0,    00/3/0,         567MB/189MB/195MB      429/143       02:24:18        477MB/159MB/160MB      149/49        02:24:18    27700MB  100977MB 0.56%
20190711 21:27:42 0/10,     00/3/0,    00/3/0,         567MB/189MB/195MB      429/143       02:24:34        477MB/159MB/160MB      149/49        02:24:35    27699MB  100978MB 0.83%
20190711 21:27:59 0/10,     00/3/0,    00/3/0,         567MB/189MB/195MB      432/144       02:24:51        477MB/159MB/160MB      149/49        02:24:51    27709MB  100968MB 4.75%
20190711 21:28:15 1/11,     01/3/0,    00/3/0,         648MB/216MB/242MB      437/145       02:25:07        477MB/159MB/160MB      150/50        02:25:07    27797MB  100880MB 1.14%
20190711 21:28:31 1/11,     01/3/0,    00/3/0,         733MB/244MB/246MB      443/147       02:25:23        477MB/159MB/160MB      150/50        02:25:23    27880MB  100797MB 1.31%
20190711 21:28:47 1/12,     01/3/0,    00/3/0,         750MB/250MB/254MB      479/159       02:25:39        477MB/159MB/160MB      150/50        02:25:39    27907MB  100770MB 4%
20190711 21:29:03 0/12,     00/3/0,    00/3/0,         752MB/250MB/255MB      484/161       02:25:55        477MB/159MB/160MB      150/50        02:25:55    27914MB  100763MB 0.5%
```

Results



Label	# Samples	Average	Min	Max	Std. Dev.	Error %
UNL Start	429	1125	31	10712	1939.61	0.00%
UNL Discovery Page	427	366	8	5321	648.15	0.00%
UNL TrueYou Login	423	5612	83	38965	10825.96	0.24%
UNL Weird Javascript ...	421	6582	234	35747	10953.81	0.00%
299 /psc/myredqa/NBL/...	419	4865	120	19574	3486.04	15.99%
UNL 1 IN 300 /psc/myr...	347	1594	115	6560	1525.03	0.00%
UNL Dash Main	345	9426	135	49148	4730.78	0.00%
START HERE 423 /psc/...	336	1941	127	7394	1240.33	0.00%
428 /psc/myredqa/NBL/...	334	1942				
432 /psc/myredqa/NBL/...	332	1212				
437 /psc/myredqa/NBL/...	331	2011				
UNL 1 Add ETHN /psc/...	328	4967				
448 /psc/myredqa/NBL/...	324	2004				
453 /psc/myredqa/NBL/...	323	1810				
458 /psc/myredqa/NBL/...	318	1152				
463 /psc/myredqa/NBL/...	317	2513				
470 /psc/myredqa/NBL/...	312	1431				
UNL1 Add SOCI /psc/m...	311	6114				
479 /psc/myredqa/NBL/...	306	1935				

Response Assertion

View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename

Search: □ Case sensitive □ Regular exp.

- 299 /psc/myredqa/NBL/S
- UNL Dash Main
- UNL1 Add CHEM /psc/my
- 437 /psc/myredqa/NBL/HI

Thread Name: Load Test - Dash Enroll UNL 1 3-49
Sample Start: 2019-02-25 16:20:12 CST
Load time: 11504
Connect Time: 15

Results

- Oracle Compute Classic
  - We created a mimic of our QA environment
  - Database load seemed to be similar, but overall performance was slower
- PUM 11
  - Experienced a lot of locking in the database
    - Varied the courses being enrolled in to avoid
- Federation
  - System performance didn't really change much

Results

- Oracle Exadata Cloud @ Customer
    - Initial performance was terrible, it was taking 15+ seconds to load the navigator. Functional users were complaining about slowness in all of our non-prods. Load testing confirmed the slowness.
    - Found that the network infrastructure wasn't working as we'd anticipated, so our non-production app servers were routing from Omaha through Lincoln and then back to Omaha.
    - Found that we were missing some parameters that we'd had on our old database. Adding them decreased CPU utilization (due to the way PeopleSoft uses bind variables).

Lessons Learned

- If you're updating data, always validate that it's actually getting updated
- If you're changing hardware or moving to the cloud, verify well in advance that your infrastructure will not be geographically separated
- If you're using tmadmin, make sure you use it in read-only mode
  - tmadmin -r
- Work closely with functional staff to make sure you're testing functionality the same way it's being used
- Keep other users out while you're load testing

Lessons Learned

- Use multiple sets of classes to avoid locking (PUM 11)
- If you have DoS, ASM, or other security filters in front of your environment, make sure they don't impact the load tests
- Consider comprehensive testing for everything
  - Innocent sounding modifications or bolt-ons may come back to bite you
    - Type ahead class search
    - External fonts
    - DTD validation

Questions?

Or send to gputney@nebraska.edu

Appendix

Querying for Logins

Grep in App Server Logs for successful logins and analyze from there

We have a small mod in our signon peoplecode that writes to a table when users log in

```
/* SIGONS PER MINUTE */
select count(*), eo_pe_dt, EO_PE_YEAR, EO_PE_MONTH, EO_PE_DATE, EO_PE_HOUR, EO_PE_MINUTE
from  SYSADM.ps_eo_pe_stats
where EO_PE_YEAR = '2016'
and EO_PE_MONTH >= '01'
AND EO_PE_DATE = '12'
group by eo_pe_dt, EO_PE_YEAR, EO_PE_MONTH, EO_PE_DATE, EO_PE_HOUR, EO_PE_MINUTE
order by count(*) desc;

/* SIGONS PER SECOND*/
select count(*), eo_pe_dt, EO_PE_YEAR, EO_PE_MONTH, EO_PE_DATE, EO_PE_HOUR, EO_PE_MINUTE,EO_PE_SECONDS
from  SYSADM.ps_eo_pe_stats
where EO_PE_YEAR = '2016'
and EO_PE_MONTH >= '01'
AND EO_PE_DATE = '12'
group by eo_pe_dt, EO_PE_YEAR, EO_PE_MONTH, EO_PE_DATE, EO_PE_HOUR, EO_PE_MINUTE, EO_PE_SECONDS
order by count(*) desc;
```

Querying for Add/Drop

```
/* enrollment total during time period */
select  COUNT(*), to_char(DTTM_STAMP_SEC, 'YYYYMMDD') AS RUNDATE from
ps_enrl_req_dt_vw2
where to_char(DTTM_STAMP_SEC, 'YYYYMMDDHH24MISS') >= '20180223133400'
and to_char(DTTM_STAMP_SEC, 'YYYYMMDDHH24MISS') <= '20180223134530'
group by to_char(DTTM_STAMP_SEC, 'YYYYMMDD');

/* enrollment per minute during time period */
select  COUNT(*), to_char(DTTM_STAMP_SEC, 'YYYY/MM/DD  HH24:MI') AS RUNDATE from
ps_enrl_req_dt_vw2
where to_char(DTTM_STAMP_SEC, 'YYYYMMDDHH24MISS') >= '20180110000000'
and to_char(DTTM_STAMP_SEC, 'YYYYMMDDHH24MISS') <= '20180110235959'
group by to_char(DTTM_STAMP_SEC, 'YYYY/MM/DD  HH24:MI')
order by count(*) desc;
```

Querying for Students

```
SELECT A.EMPLID, A.ACAD_CAREER, A.ELIG_TO_ENROLL, A.STRM, E.ACCTLOCK
  FROM PS_STDNT_CAR_TERM A, PSOPRDEFN E
  WHERE ( A.INSTITUTION = :1
     AND A.ACAD_CAREER = :2
     AND ( A.STRM = :3
     AND A.ELIG_TO_ENROLL = 'Y')
     AND A.EMPLID NOT IN (SELECT C.ROLEUSER
  FROM PSROLEUSER C
  WHERE C.ROLEUSER = A.EMPLID
     AND C.ROLENAME IN (‘STAFF_ROLE1',‘STAFF_ROLE2',‘STAFF_ROLE3'))
     AND A.EMPLID = E.OPRID
     AND A.EMPLID NOT IN (SELECT B.EMPLID
  FROM PS_SRVC_IND_DATA B
  WHERE B.POS_SRVC_INDICATOR = 'N'))
```

Please complete a session evaluation

Session ID: 101630

Contact Info:

Geoff Putney

gputney@nebraska.edu

PeopleSoft RECONNECT

Who is the Quest Community?

A 55,000+ member user community for Oracle Cloud, JD Edwards and PeopleSoft customers.

What the Quest PeopleSoft Community offers:

- Customized digital content
- Official PeopleSoft newsletter
- Customer success stories
- Virtual and face-to-face events
- PeopleSoft networking groups

Visit **www.QuestOracleCommunity.org** for more information!