#!/bin/ksh

#---------------------------------------------------------------------------------------------------------------------------

# TITLE : aedc\_add\_device

# PURPOSE : HOW TO PROCESS NEW DATA IN A PARAGRAPH WHICH WE CAN GENERATE IT IN THE DATA BASE TABULES

#

# GROUP : SOFTWARE.SCC (S/W)

# AUTHOR : Alaa Nagy

# DATE :23 MARCH 2002

# MODIFICATION : 5 Apr 2007 , 15 Nov 2008 and 20 Nov 2011 and 4 Apr 2015 by Alaa Nagy

# > to can change connected node

# > to can change cable name

# > to can write date of modified cable

# > to can change load type

#----------------------------------------------------------------------------------------------------------------------------

path=/aedc/etc/work/aedc/SCC/NEEDED\_FILES/data\_base/temp\_add\_device/

path\_file=/aedc/data/dbgen

function cheak\_dev\_node

{

rm -f $path/outdevnode

echo "

if exists (select \* from T0201\_node

where C0201\_node\_name='${node\_nm}')

begin

if exists (select \* from T0203\_terminal where C0201\_node\_id\_A in

(select C0201\_node\_id from T0201\_node where C0201\_node\_name='${node\_nm}'))

begin

print ' =========================================='

set nocount on

select C0003\_group\_name, C0202\_dev\_name from T0202\_devices where

C0202\_dev\_id in (select C0202\_dev\_id from T0203\_terminal where

C0201\_node\_id\_A in (select C0201\_node\_id from T0201\_node where C0201\_node\_name='${node\_nm}'))

end

else

begin

print ' '

print ' Not Exists'

print ' '

end

end

else

begin

print ' '

print ' Not Exists'

print ' '

end

go

" | isql -U dbu -P dbudbu |tee -a $path/outdevnode

grep -w Not $path/outdevnode |read s

}

function replase\_data

{

if [[ ${replecs} = 1 ]]

then

sed "s/SWitch/$sw/" $path/$tempecs |sed "s/gname/$igname/" |sed "s/discr/$discrp\_1\/$discrp\_2/" |sed "s/d\_catg/$data\_catagr/" |sed "s/aso/$tlgd/" |sed "s/area/$aojd/" |sed "s/pointn/$PNo/" >> $path\_final/ecs-$rname

replecs=0

else

sed "s/SWitch/$sw/" $path/$tempecs |sed "s/gname/$igname/" |sed "s/discr/$dis/" |sed "s/d\_catg/$data\_catagr/" |sed "s/aso/$tlgd/" |sed "s/area/$aojd/" |sed "s/pointn/$PNo/" >> $path\_final/ecs-$rname

fi

if [[ ${repl} = 1 ]]

then

sed "s/device/$sw/" $path/$temppdb |sed "s/gname/$igname/" |sed "s/swtype/$sw\_part1\/$sw\_part2/" |sed "s/d\_catg/$data\_catagr/" |sed "s/rmark/$remark/" |sed "s/connode1/$co1/" |sed "s/connode2/$co2/" |sed "s/room/$rname/" >> $path\_final/pdb-$rname

repl=0

else

sed "s/device/$sw/" $path/$temppdb |sed "s/gname/$igname/" |sed "s/swtype/$sw\_type/" |sed "s/d\_catg/$data\_catagr/" |sed "s/rmark/$remark/" |sed "s/connode1/$co1/" |sed "s/connode2/$co2/" |sed "s/room/$rname/" >> $path\_final/pdb-$rname

fi

}

function node\_case

{

case $q2 in

1)read\_sw

;;

2)enter\_new\_node

;;

\*)

;;

esac

}

function read\_sw

{

function read\_dev

{

echo " "

read dev\_nl?" Enter Group\_name.device\_name <CR to exit> >>> "

echo $dev\_nl |awk 'FS=":"{print $2}'|read dev\_nm

}

function dsply

{

echo "

if exists (select C0202\_dev\_id from T0202\_devices

where (rtrim(C0003\_group\_name)+'.'+rtrim(C0202\_dev\_name))='${dev\_nm}' )

begin

print ' '

print 'The nodes connected to the device ${dev\_nm} and there sides are'

print ' '

set nocount on

select C0201\_node\_name , C0203\_side from T0201\_node , T0203\_terminal

where T0201\_node.C0201\_node\_id =T0203\_terminal.C0201\_node\_id\_A

and T0203\_terminal.C0202\_dev\_id = (

select C0202\_dev\_id from T0202\_devices

where (rtrim(C0003\_group\_name)+'.'+rtrim(C0202\_dev\_name))='${dev\_nm}' )

end

else

begin

print ' '

print ' >>>> THERE IS NO SUCH DEVICE <<<<'

end

go

" | isql -Udbu -Pdbudbu -S$sev

}

read\_dev

if [[ -z ${dev\_nm} ]]

then

break

else

dsply

read node?"

Enter Base Connected Node >> "

if [[ -z ${node} ]]

then

break

else

node\_nm=$node

fi

fi

}

function enter\_new\_node

{

while true

do

read node?"

Enter Base Connected Node >> "

if [[ -z ${node} ]]

then

break

else

node\_nm=$node

fi

cheak\_dev\_node

if [[ ${s} = 'Not Exists' ]]

then

break

else

echo "

Connected Node Is Already Exsits

Try Again

"

fi

done

}

function cheak\_frist\_node

{

clear

echo "

\*\*\* Waiting System Check The Base Connected Node ! \*\*\*

- Devices Of Connected Node << ${node\_nm} >> IS "

check\_dev\_node

if [[ ${s} = 'Not Exists' ]]

then

read q2?"

CONNECTED\_NODE1 ${node\_nm} Is Not Exists

----------------------------------------

1. Check Connected Node

2. Enter New Connected Node

Your choice ( <CR> New Room ) >> "

node\_case

else

read q2?"

CONNECTED\_NODE1 ${node\_nm} Is Already Exists

---------------------------------------

1. Check Connected Node

2. Enter New Connected Node

Your choies ( <CR> to continue ) >> "

node\_case

fi

echo "

The Base Connected Node For Room $Room Will Be << $node\_nm >>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

}

function cheak\_scnd\_node

{

loop\_snd=1

exit\_snd=0

while [ ${exit\_snd} -lt 8 ]

do

case $loop\_snd in

1)var2=A

;;

2)var2=B

;;

3)var2=C

;;

4)var2=D

;;

5)var2=E

;;

6)var2=F

;;

7)var2=G

;;

\*)

;;

esac

co2="$co1"$var2

node\_nm=$co2

echo " - Devices Of Connected Node << $co2 >> Is "

check\_dev\_node

if [[ ${s} = 'Not Exists' ]]

then

exit\_snd=10

else

loop\_snd=`expr $loop\_snd + 1`

fi

done

}

function display\_cable

{

while true

do

clear

read chng\_cab\_data?"

\*\*\*\*\*\*\*\*\*\* PDB File For ${ca} Will Be \*\*\*\*\*\*\*\*\*\*

CABLE\_SECT\_DEFINITION

DEVICE\_NAME = $ca

GROUP\_NAME = $center-CABLE

CONNECTED\_NODE1 = $co1b

CONNECTED\_NODE2 = $co2b

CABLE\_SECT\_TYPE = $dev\_typ

VOLT\_LIMIT\_SET\_NAME = V\_SET

PHASE = RST

FLOW\_ALARMING = N

LENGTH = $dev\_len

DEVICE\_REMARK = $ddd

THRESH\_MIN\_TRB\_CALL = 0

MODELLED\_C = Y

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. If You Want To Change Cable Length.

2. If You Want To Change Cable Type.

3. If You Want To Change Cable Name.

4. If There is modified date.

Select Your Choiec <CR> To Contnue !"

case $chng\_cab\_data in

1)

read dev\_len?"

Enter The New Cable Length ( IF NOT LENGTH=1001 ) >> "

if [[ -z ${dev\_len} ]]

then

dev\_len=1001

else

echo " The Cable Length Will Be $dev\_len "

fi

;;

2)read dev\_typ?"

Select The New Cable Type ( 3X240AL\_X , 3X240AL\_O , 3X1X400AL\_X , 3X150AL\_O , 3X240AL\_X\_20 , 3X240AL\_X\_22 ) >> "

if [[ -z ${dev\_typ} ]]

then

dev\_typ=3X240AL\_X

else

echo " The Cable Type Will Be $dev\_typ "

fi

;;

3)

read cN?"

Enter The New Cable Name ( IF NOT DEVICE\_NAME = $ca ) >> "

if [[ -z ${cN} ]]

then

ca=$ca

else

ca=$cN

echo " The Cable Length Will Be $ca "

fi

;;

4)

mktime `date +"%Y/%m/%d:%H:00:00"`|read daysec

var\_asctime $daysec "%Y/%m/%d"|read dayymd

read tim?"

Enter The New Modified Date ( For Example. MOD. $dayymd ) >> "

if [[ -z ${tim} ]]

then

ddd=$ddd

else

ddd=$tim

echo " The New Modified Date Will Be $ddd "

fi

;;

\*)break

;;

esac

done

}

function display\_load

{

while true

do

clear

read dis\_cab\_rat?"

\*\*\*\*\*\*\*\*\*\* PDB File For LD-$rname Will Be \*\*\*\*\*\*\*\*\*\*\*\*

LOAD\_DEFINITION

DEVICE\_NAME = LD-$rname

GROUP\_NAME = $center-LOAD

CONNECTED\_NODE1 = $co1a

LOAD\_TYPE = $load\_typ

FLOW\_ALARMING = N

VOLT\_LIMIT\_SET\_NAME = V\_SET

PHASE = RST

P\_WEEKLY\_PEAK = 40.000000

Q\_WEEKLY\_PEAK = 30.000000

LOAD\_RATING = $load\_rat

THRESH\_MIN\_TRB\_CALL = 0

MODELLED\_C = Y

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. If You Want To Change Load Rate.

2. If You Want To Change Load Type.

Select Your Choiec <CR> To Contuie !"

case $dis\_cab\_rat in

1)

read load\_rat?"

Enter The New Load Rate ( IF NOT Load Rating=505 ) >> "

if [[ -z ${load\_rat} ]]

then

load\_rat=505

else

echo " The Load Rate Will Be $load\_rat "

fi

;;

2)

read load\_typ?"

Select The New Load Type ( COMMERCIAL , RSIDNTIL\_20 , RSIDNTIL\_22 >> "

if [[ -z ${load\_typ} ]]

then

load\_typ=COMMERCIAL

else

echo " The Load Type Will Be $load\_typ "

fi

;;

\*)break

;;

esac

done

}

function display\_sw

{

while true

do

clear

read q?"

\*\*\*\*\*\*\*\*\*\*\*\* PDB File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*

SWITCH\_DEFINITION

DEVICE\_NAME = $sw

GROUP\_NAME = $igname

CONNECTED\_NODE1 = $co1

CONNECTED\_NODE2 = $co2

SWITCH\_TYPE = $sw\_type

PHASE = RST

SW\_POSITION\_POINT\_NAME = $sw

DATA\_CATEGORY = $data\_catagr

CLOSE\_STATE\_DESCRIPTION = CLOSE

OPEN\_STATE\_DESCRIPTION = OPEN

TIE\_SWITCH = N

OPTIMAL\_CONTROL = N

FLOW\_ALARMING = N

VOLT\_LIMIT\_SET\_NAME = V\_SET

PRIORITY = 0

THRESH\_MIN\_TRB\_CALL = 0

MODELLED\_C = Y

\*\*\*\*\*\*\*\*\*\*\*\* ECS File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*

DCC\_DI\_POINT\_DEFINITION

POINT\_NAME = $sw

GROUP\_NAME = $igname

DATA\_CATEGORY\_NAME = $data\_catagr

POINT\_DESCRIPTION = $dis

ASSOCIATED\_DISPLAY = $tlgd

ALARM\_PRIORITY = 1

AREA\_OF\_JURISDICTION = $aojd

MANUAL\_ENTRY\_INHIBITED = N

POINT\_TAG = Y

NORMAL\_STATE = CLOSE

POINT\_NO = $PNo



Would you like to change data (y,n) >> "

case $q in

y|Y)

change\_data

;;

\*)

break

;;

esac

done

}

function display\_sw\_tr\_bar

{

while true

do

clear

read q?"

\*\*\*\*\*\*\*\*\* PDB File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*\*\*\*

SWITCH\_DEFINITION

DEVICE\_NAME = $sw

GROUP\_NAME = $igname

CONNECTED\_NODE1 = $co1

CONNECTED\_NODE2 = $co2

SWITCH\_TYPE = $sw\_type

PHASE = RST

SW\_POSITION\_POINT\_NAME = $sw

DATA\_CATEGORY = $data\_catagr

CLOSE\_STATE\_DESCRIPTION = CLOSE

OPEN\_STATE\_DESCRIPTION = OPEN

TIE\_SWITCH = N

OPTIMAL\_CONTROL = N

FLOW\_ALARMING = N

VOLT\_LIMIT\_SET\_NAME = V\_SET

PRIORITY = 0

DEVICE\_REMARK = $remark

THRESH\_MIN\_TRB\_CALL = 0

MODELLED\_C = Y

\*\*\*\*\*\*\*\*\* ECS File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*\*\*

DCC\_DI\_POINT\_DEFINITION

POINT\_NAME = $sw

GROUP\_NAME = $igname

DATA\_CATEGORY\_NAME = $data\_catagr

POINT\_DESCRIPTION = $dis

ASSOCIATED\_DISPLAY = $tlgd

ALARM\_PRIORITY = 1

AREA\_OF\_JURISDICTION = $aojd

MANUAL\_ENTRY\_INHIBITED = N

POINT\_TAG = Y

NORMAL\_STATE = CLOSE

POINT\_NO = $PNo



Would you like to change data (y,n) >> "

case $q in

y|Y)

change\_data

;;

\*)

break

;;

esac

done

}

function display\_sw\_cba

{

while true

do

clear

read q?"

\*\*\*\*\*\*\*\*\*\* PDB File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*

SWITCH\_DEFINITION

DEVICE\_NAME = $sw

GROUP\_NAME = $igname

CONNECTED\_NODE1 = $co1

CONNECTED\_NODE2 = $co2

SWITCH\_TYPE = $sw\_type

PHASE = RST

SW\_POSITION\_POINT\_NAME = $sw

DATA\_CATEGORY = OPNCLS

CLOSE\_STATE\_DESCRIPTION = CLOSE

OPEN\_STATE\_DESCRIPTION = OPEN

TIE\_SWITCH = N

OPTIMAL\_CONTROL = Y

FLOW\_ALARMING = N

VOLT\_LIMIT\_SET\_NAME = V\_SET

PRIORITY = 0

DEVICE\_REMARK = CB $rname

FEEDER\_BREAKER\_COLOR = 6

THRESH\_MIN\_TRB\_CALL = 0

MODELLED\_C = Y

\*\*\*\*\*\*\*\*\*\* ECS File For ${sw} Will Be \*\*\*\*\*\*\*\*\*\*\*

DCC\_DI\_POINT\_DEFINITION

POINT\_NAME = $sw

GROUP\_NAME = $igname

DATA\_CATEGORY\_NAME = OPNCLS

POINT\_DESCRIPTION = $dis

ASSOCIATED\_DISPLAY = $tlgd

ALARM\_PRIORITY = 1

AREA\_OF\_JURISDICTION = $aojd

MANUAL\_ENTRY\_INHIBITED = N

POINT\_TAG = Y

NORMAL\_STATE = CLOSE

POINT\_NO = $PNo



Would you like to change data (y,n) >>"

case $q in

y|Y)

change\_data

;;

\*)

break

;;

esac

done

}

function change\_data

{

read choice?"

Change Data For SWitch $sw

=========================

1. Change POINT\_NAME

2. GROUP\_NAME

3. POINT\_DESCRIPTION

4. ASSOCIATED\_DISPLAY

5. AREA\_OF\_JURISDICTION

6. POINT\_NO

7. CONNECTED\_NODE1

8. CONNECTED\_NODE2

enter your choice <CR to quit> >> "

case $choice in

1)read sw?"enter new switch >>"

;;

2)

enter\_dcc

;;

3)read dis?" Enter new POINT\_DESCRIPTION >>"

;;

4)read tlgd?" Enter new ASSOCIATED\_DISPLAY >>"

;;

5)read aojd?" Enter new AREA\_OF\_JURISDICTION >>"

;;

6)read PNo?" Enter new POINT\_NO >>"

;;

7)read co1?" Enter new CONNECTED\_NODE1 >>"

;;

8)read co2?" Enter new CONNECTED\_NODE2 >>"

;;

\*)

;;

esac

}

function enter\_dcc

{

while true

do

read cent?"

Select DCC W , M , E , (<q> to quit) >> "

case $cent in

W|w) center=W

centdflt=WEST

center\_file=west

break

;;

M|m) center=M

centdflt=MIDDLE

center\_file=middle

break

;;

E|e) center=E

centdflt=EAST

center\_file=east

break

;;

q|Q) exit

;;

\*)echo "

You Must select DCC Try Agane "

;;

esac

done

}

function aoj

{

while true

do

case $center in

W) read aoj?"

Select Area Of Jurisdiction ( AMRIYA , DEKHELA , SHAMALI ) >> "

;;

E) read aoj?"

Select Area Of Jurisdiction ( SABABASH , SIDIBISH ) >> "

;;

M) read aoj?"

Select Area Of Jurisdiction ( IBRAHIM , WASAT ) >> "

;;

\*)

;;

esac

if [[ -z ${aoj} ]]

then

echo "

You Must Select Area Of Jurisdiction

Try Again "

else

break

fi

done

}

function enter\_data\_lbs

{

cont\_room=1

loop\_room=1

read rlbs\_type?"

Enter Type Of LBS ( KO , POS ) >> "

read rrout?"

Enter DP or SS Number >> "

# read rarrang?"

# Enter The rarrang In LBS >> "

co1a=$center-K$rname

node\_nm=$co1a

Room=$rname

cheak\_frist\_node

co1a=$node\_nm

co1=$node\_nm

echo "

\*\*\* Waiting System Check Second Connected Node ! \*\*\* "

cheak\_scnd\_node ${co1}

n=`expr $loop\_snd - 1`

co2a=$co2

echo "

Room $rname connected with $n Rooms We Will Start From $co2 "

sleep 5

}

function enter\_data

{

cont\_room=1

loop\_room=1

if [ ${barno} = ok ]

then

under=\_

co1a=$center-K$rname$under$bar

else

co1a=$center-K$rname

fi

node\_nm=$co1a

Room=$rname

cheak\_frist\_node ${node\_nm} ${Room}

co1a=$node\_nm

co1=$node\_nm

echo "

\*\*\* Waiting System Check Second Connected Node ! \*\*\* "

cheak\_scnd\_node ${co1}

n=`expr $loop\_snd - 1`

co2a=$co2

echo "

Room $rname connected with $n Rooms We Will Start From $co2 "

sleep 10

}

function conr\_data

{

read PNo?"

Enter GPID For $sw >>"

read dis?"

Enter POINT\_DESCRIPTION For $sw >>"

if [[ -z ${dis} ]]

then

dis=$discrp

else

replecs=0

fi

}

function conectr

{

loop\_room=$loop\_snd

while true

do

t=0

clear

read conr?"

Enter Name For Room Number $cont\_room ( Select DP , SS , LBS , TR ) >> "

case $loop\_room in

1)var=A

;;

2)var=B

;;

3)var=C

;;

4)var=D

;;

5)var=E

;;

6)var=F

;;

7)var=G

;;

\*)

;;

esac

if [[ -z ${conr} ]]

then

break

else

case $conr in

DP|dp)

t=DP

read dpn?"

Enter DP Number >> "

read cba?"

Enter CBA Number >> "

dd=DP$dpn

conr=$center$dd

dp=y

v=0

;;

SS|ss)

t=SS

read ssn?"

Enter SS Number >> "

read cba?"

Enter CBA Number >> "

ss=SS$ssn

conr=$center$ss

ss=y

v=0

;;

LBS|lbs)

t=lbs

read conr?"

Enter Number Of The LBS >> "

read lbs\_type?"

Enter Type Of LBS ( KO , POS ) >> "

read rout?"

Enter Number in the form [ DP5 or SS5 ] >> "

read arrang?"

Enter The arrang In LBS >> "

dd=$rout

lbs=y

v=0

;;

TR|tr) v=1

;;

\*)v=0

;;

esac

cont\_room=`expr $cont\_room + 1`

case $sel in

2)

read rarrang?"

Enter The rarrang In LBS >> "

if [[ ${rlbs\_type} = KO ]]

then

sw=KO-$rname-S$rarrang

discrp=SW$rarrang-$conr'\*KIOSK '$rname' LBS'

igname=$center-DP$rrout

data\_catagr=TCOPCL

sw\_type=LBS\_SF6

ctgr\_lbs=ok

else

sw=$rname'M'$rarrang'\_POS'

igname=$center-DP$rrout

discrp\_1=$conr' IM'$arrang' CLOSE'

discrp\_2=OPEN

discrp=$conr' IM'$arrang' CLOSE/'OPEN

sw\_type=ISO

data\_catagr=OPNCLS

ctgr\_lbs=N

replecs=1

fi

;;

\*)

sw=SW-$rname-$conr

igname=$center-SWITCH

discrp=SW-$conr\*$rname

sw\_type=ISO

data\_catagr=OPNCLS

;;

esac

if [[ ${v} = 1 ]]

then

case $ctgr\_lbs in

ok)

data\_catagr=TCOPCL

;;

\*)

data\_catagr=OPNCLS

;;

esac

remark=SW' 'TRANSFORMER\*$rname

discrp=SW-TRANSFORMER\*$rname

fi

conr\_data ${sw}

co1b="$co1a"$var

node\_nm=$co1b

while true

do

cheak\_dev\_node

if [[ ${s} = 'Not Exists' ]]

then

break

else

loop\_room=`expr $loop\_room + 1`

case $loop\_room in

1)var=A

;;

2)var=B

;;

3)var=C

;;

4)var=D

;;

5)var=E

;;

6)var=F

;;

7)var=G

;;

\*)

;;

esac

node\_nm="$co1a"$var

fi

done

loop\_room=`expr $loop\_room + 1`

co1b=$node\_nm

#igname=$center-SWITCH

aojd=$aoj

tlgd=$tlg

co1=$co1a

co2=$co1b

# sw\_type=ISO

# data\_catagr=OPNCLS

if [[ ${v} = 1 ]]

then

display\_sw\_tr\_bar

temppdb=temp\_tr\_bar

else

display\_sw

temppdb=temp\_pdb

fi

clear

tempecs=temp\_ecs

#data\_catagr=OPNCLS

replase\_data

temppdb=temp\_pdb

tempecs=temp\_ecs

if [[ ${v} = 0 ]]

then

case $t in

DP)

sw=CBA$cba

gdp=DP$dpn

igname=$center-$gdp

sw\_type=CB\_DIST

discrp=CB' '$rname

read\_sw

co1=$node

N2=\_A$cba

co2=$co1$N2

echo " $co2 "

co2b=$co2

t=0

temppdb=temp\_cba\_pdb

;;

SS)

sw=CBA$cba

gss=SS$ssn

igname=$center-$gss

sw\_part2=P

sw\_part1=CB\_O

sw\_type=CB\_O/P

echo " $sw\_type"

discrp=CB' '$rname

read\_sw

co1=$node

N2=\_A$cba

co2=$co1$N2

echo " $co2 "

co2b=$co2

t=0

temppdb=temp\_cba\_pdb

repl=1

;;

lbs)

if [[ ${lbs\_type} = KO ]]

then

sw=KO-$conr-S$arrang

discrp=SW$arrang-$rname'\*KIOSK '$conr' LBS'

echo " $discrp "

data\_catagr=TCOPCL

igname=$center-$dd

sw\_type=LBS\_SF6

co1=$center-K$conr

cheak\_scnd\_node

co2b=$co2

else

sw=$conr'M'$arrang'\_POS'

discrp\_1=$conr' IM'$arrang' CLOSE'

discrp\_2=OPEN

discrp=$conr' IM'$arrang' CLOSE/'OPEN

echo " $discrp "

igname=$center-$dd

sw\_type=ISO

#data\_catagr=OPNCLS

co1=$center-K$conr

cheak\_scnd\_node

co2b=$co2

replecs=1

fi

;;

\*) echo " "

sw=SW-$conr-$rname

discrp=SW-$rname\*$conr

igname=$center-SWITCH

sw\_type=ISO

data\_catagr=OPNCLS

co2a=$center-K$conr

node\_nm=$co2a

Room=$conr

cheak\_frist\_node ${node\_nm} ${Room}

co2a=$node\_nm

co1=$node\_nm

cheak\_scnd\_node

echo "

The Second Connected Node For Switch ( $sw ) Will Be $co2

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"

co2b=$co2

co1=$co2a

co2=$co2b

aojd=$aoj

tlgd=$tlg

;;

esac

clear

if [[ ${temppdb} = temp\_cba\_pdb ]]

then

read print?"

Would You Like To Add To File ( No Y )? "

if [ ${print} = Y ]

then

conr\_data ${sw}

display\_sw\_cba

replase\_data

fi

else

conr\_data ${sw}

display\_sw

replase\_data

data\_catagr=OPNCLS

fi

ca=CA-$rname-$conr

dev\_len=1001

dev\_typ=3X240AL\_X

ddd=""

display\_cable

sed "s/casw/$ca/" $path/temp\_cab |sed "s/gnname/$center-CABLE/" |sed "s/c1/$co1b/" |sed "s/c2/$co2b/" |sed "s/len/$dev\_len/" |sed "s/type/$dev\_typ/" |sed "s-xx-$ddd-" >> $path\_final/pdb-$rname

fi

fi

done

read ld?"

Do You Want Load In Room ( Y , N ) >> "

case $ld in

Y|y)

load\_rat=505

display\_load

sed "s/room/$rname/" $path/temp\_load |sed "s/dccl/$center-LOAD/" |sed "s/con1/$co1a/" |sed "s/rate/$load\_rat/" >> $path\_final/pdb-$rname

;;

\*)

;;

esac

if [ ${barno} = ok ]

then

sw=SW-$rname

cob2=$co1a

if [ ${bar} = 2 ]

then

remark=BETWEEN' BAR1&2'\*$rname

discrp=SW-$remark

conr\_data

co1=$cob1

co2=$cob2

sw\_type=ISO

display\_sw\_tr\_bar

sed "s/SWitch/$sw/" $path/temp\_ecs |sed "s/gname/$center-SWITCH/" |sed "s/discr/$dis/" |sed "s/aso/$tlgd/" |sed "s/area/$aojd/" |sed "s/pointn/$PNo/" >> $path\_final/ecs-$rname

sed "s/device/$sw/" $path/temp\_tr\_bar |sed "s/gname/$igname/" |sed "s/swtype/$sw\_type/" |sed "s/rmark/$remark/" |sed "s/connode1/$co1/" |sed "s/connode2/$co2/" >> $path\_final/pdb-$rname

echo " $cob2 $cob1 "

fi

cob1=$co1a

fi

}

function normal\_room

{

while [ ${loop} -lt 3 ]

do

enter\_data

conectr

bar=`expr $bar + 1`

loop=`expr $loop + 1`

done

}

function lbs\_room

{

enter\_data\_lbs

conectr

}

while true

do

clear

read sel?"

AEDC ADDITION OF NEW ROOM

===========================

\*\*\* CHIOCES \*\*\*

1. Room Consistes Of More Than One bar

2. Load Break Switches (LBS)

( 1,2,<CR> For One Room ) >> "

case $sel in

1)loop=1

barno=ok

bar=1

nrml\_room=ok

;;

2)nrml\_room=n

;;

\*)loop=2

barno=n

nrml\_room=ok

;;

esac

enter\_dcc

aoj

read tlg?"

Enter Associated Display >> "

read rname?"

Enter Room Name >> "

#path\_final=/home/sis/SOFWARE\_STAFF/mohaned/script/result/

path\_final=$path\_file/$center\_file

sed "s/centerdefault/$centdflt/" $path/temp-ecs-default >> $path\_final/ecs-$rname

echo "!!" >> $path\_final/pdb-$rname

if [[ ${nrml\_room} = ok ]]

then

normal\_room

else

lbs\_room

fi

done