Shell Scripts – AIX mksysb, filesystems and Informix DB

# Script /insight/local/backup/sysbkup.ksh

#####################################################################

#!/bin/ksh

################################################################################

#

# Name: sysbkup.ksh

#

# Reference: n/a

#

# Description: system backup using mksysb

#

# Parameters: sysbkup.ksh <tape device>

# tape device /dev/rmt0

#

# #

################################################################################

set -v

set -x

# script library

PATH=$PATH:/insight/local/backup/sitelib:.

cd /dmqjtmp/archiveSysbkupLog

backup\_tape=/dev/$1

backup\_lisfile=sysbkup\_lis.

backup\_errfile=sysbkup\_err.

backup\_logfile=sysbkup\_log.

backup\_date=`date +%Y%m%d%H%M`

lisfile=$backup\_lisfile$backup\_date

errfile=$backup\_errfile$backup\_date

logfile=$backup\_logfile$backup\_date

aixsupport="chenliru@yahoo.com"

date >$logfile

Check\_Tape()

{

# rewind the tape;

tctl -f $backup\_tape rewind

if [[ $? -eq 0 ]]

then

return 0 #tape is ready;

else

date > $errfile

echo "\nError: tape is not ready" >> $errfile

mail -s "Tape is not ready for SYSTEM backup on IFX01 @ $backup\_date" computerops@livingstonintl.com < $errfile

return 1

fi

}

count=3

while [[ $count -gt 0 ]]; do

Check\_Tape

if [[ $? -eq 0 ]]; then

break #tape ready, continue to do backup;

else

count=$(($count-1))

banner "Tapes" Please!!!

if [[ $count -eq 0 ]]

then

mail -s "Ifx01 Sysbackup Failed Due to Tape not ready ..." computerops@livingstonintl.com < /dev/null

mail -s "Ifx01 Sysbackup Failed Due to Tape not ready ..." $aixsupport < /dev/null

exit 1

fi

fi

sleep 120

done

# backup of the operating system (that is, the root volume group)

mksysb -e -p -i $backup\_tape 1>>$logfile 2>&1

errsts=$?

if (($errsts != 0))

then

errevent $logfile "<error = $errsts> error on mksysb command:"

mail -s "System backup failed (IFX01): $backup\_date" $aixsupport <$logfile

<$logfile

tctl -f $backup\_tape offline

exit 1

fi

date >> $logfile

# rewind the tape

bot.check $backup\_tape $logfile

# finally list all the files on tape

eventlog $logfile "----------------------------------------------------"

eventlog $logfile "Listing of the root volume group:" | tee -a $lisfile

eventlog $logfile "----------------------------------------------------"

/usr/sbin/restore -Tqs4 -f $backup\_tape.1 >> $lisfile 2>> $logfile

errsts=$?

if (($errsts != 0))

then

errevent $logfile "\t <$errsts> error on readcheck of system backup" | tee -a $lisfile

eventlog $logfile "\tDumping the contents of error file:"

mail -s "System backup failed (IFX01): $backup\_date" $aixsupport

mail -s "System backup failed (IFX01): $backup\_date" computerops@livingstonintl.com

tctl -f $backup\_tape offline

exit 1

fi

cat $errfile | tee -a $logfile

eventlog $logfile "----------------------------------------------------"

#rm $errfile

eventlog $logfile "SYSTEM BACKUP task has been completed"

eventlog $logfile "----------------------------------------------------"

sleep 3

# dismount the tape

tctl -f $backup\_tape offline

sleep 3

#Mail to Administrator;

mail -s "System backup successful (IFX01): $backup\_date" $aixsupport <$logfile

#Mail to operator;

mail -s "System backup successful (IFX01): $backup\_date" computerops@livingstonintl.com <$logfile

exit 0

#####################################################################

# Script /home/dguo/script/check\_apptape.ksh

#####################################################################

#####################################################################

# Script /insight/local/backup/appbkup.ksh

#####################################################################

#!/bin/ksh

################################################################################

#

# Name: appbkup.ksh

#

# Reference: n/a

#

# Description: application backup using muilt-backup

# backup all filesystems except database filesystems

#

# Parameters: appbkup <tape device>

# tape device /dev/rmt1

#

################################################################################

set -v

set -x

# script library

PATH=/insight/local/backup/sitelib:$PATH:.

# log file directory

cd /dmqjtmp/archiveAppbkupLog

# database filesystems

ixFS=/ix\_root:/ix\_plog:/ix\_llog:/ix\_dat1:/ix\_dat2:/ix\_dat3:/ix\_idx1:/ix\_idx2:/ix\_idx3:/ix\_temp

arFS=/ach\_root:/ach\_plog:/ach\_llog:/ach\_dat1:/ach\_dat2:/ach\_dat2\_2:/ach\_dat1\_2:/ach\_idx1:/ach\_idx2:/ach\_temp

dbFS=${ixFS}:${arFS}

# exclude filesystems

xList=`echo ${dbFS} | sed "s/:/:|/g"`":"

# backup level and tape driv

backup\_level=0

backup\_tape=/dev/$1

# log files

backup\_lisfile=level0\_lis.

backup\_errfile=level0\_err.

backup\_logfile=level0\_log.

backup\_date=`date +%Y%m%d%H%M`

lisfile=$backup\_lisfile$backup\_date

errfile=$backup\_errfile$backup\_date

logfile=$backup\_logfile$backup\_date

aixsupport="lchen@livingstonintl.com"

Check\_Tape()

{

# rewind the tape

tctl -f $backup\_tape rewind

if [[ $? -eq 0 ]]

then

return 0 #tape is ready fro backup;

else

date > $errfile

echo "\nError: tape is not ready" >> $errfile

mail -s "Tape is not ready for APPLICATION backup on IFX01 @ $backup\_date" computerops@livingstonintl.com < $errfile

return 1

fi

}

count=3

while [[ $count -gt 0 ]]; do

Check\_Tape

if [[ $? -eq 0 ]]; then

break #tape ready, continue to do the backup;

else

count=$(($count-1))

banner "Tapes Please!!!"

if [[ $count -eq 0 ]]

then

mail -s "Ifx01 Application Backup Failed Due to Tape not ready ..." $aixsupport < /dev/null

mail -s "Ifx01 Apllication Backup Failed Due to Tape not ready ..." computerops@livingstonintl.com < /dev/null

exit 1

fi

fi

sleep 120

done

# get a list of the mounted "jfs" filesystems exclude database filesystems

filesys=`lsfs -c -v jfs2 | tail +2 | grep -v -E "$xList" | cut -f1 -d":"`

# log the archive filesystems

eventlog $logfile "Backup filesystem listing=$filesys"

# get number of filesystem in the string "filesys"

set $filesys

integer fsCount=$#

# do not use <\*> inside argument values : <\*> is translated as <ls \*>

eventlog $lisfile "-------------------------------------------------------"

eventlog $lisfile "BACKUP LEVEL: $backup\_level"

eventlog $lisfile "BACKUP DATE : `date`"

eventlog $lisfile "-------------------------------------------------------"

# Backup the file systems in the listing

integer xCount=0

line=$filesys

set $line

while (( $xCount < $fsCount ))

do

eventlog $logfile "----------------------------------------------------"

eventlog $logfile "Backing up filesystem: $1"

eventlog $logfile "----------------------------------------------------"

sync

sleep 5

backup -$backup\_level -uf $backup\_tape.1 $1 2>&1 | tee -a $logfile

errsts=$?

if (($errsts != 0))

then

errevent $logfile "<error = $errsts> error on backing up filesystem: $1"

mail -s "Application backup failed (IFX01): $backup\_date" $aixsupport <$logfile

mail -s "Application backup failed (IFX01): $backup\_date" computerops@livingstonintl.com <$logfile

tctl -f $backup\_tape offline

exit 1

fi

shift 1

xCount=xCount+1

done

# rewind the tape

bot.check $backup\_tape $logfile

# finally list all the files on tape

integer xCount=1

line=$filesys

set $line

while (( $xCount <= $fsCount ))

do

eventlog $logfile "----------------------------------------------------"

eventlog $logfile "Listing of filesystem: $1 File number: $xCount" | tee -a $lisfile

eventlog $logfile "----------------------------------------------------"

restore -s1 -qvTf $backup\_tape.1 >>$lisfile 2>$errfile

errsts=$?

eventlog $logfile "Dumping the contents of error file:"

cat $errfile | tee -a $logfile

if (($errsts != 0))

then

errevent $logfile "<error = $errsts> error on reading filesystem: $1" | tee -a $lisfile

mail -s "Application backup failed (IFX01): $backup\_date" $aixsupport

<$logfile

tctl -f $backup\_tape offline

exit 1

fi

shift 1

xCount=xCount+1

done

#rm $errfile

eventlog $logfile "/etc/dumpdates at the close of this backup:"

sort /etc/dumpdates | tee -a $logfile

eventlog $logfile "BACKUP task has been completed"

sleep 5

# dismount the tape

tctl -f $backup\_tape offline

sleep 5

#Mail to Administrator;

mail -s "Application backup successful (IFX01): $backup\_date" $aixsupport <$logfile

#Mail to operator!

mail -s "Application backup successful (IFX01): $backup\_date" computerops@livingstonintl.com <$logfile

exit 0

#####################################################################

# Script /home/dguo/script/check\_dbstape.ksh

#####################################################################

#####################################################################

# Script /insight/local/backup/dbsbkup.ksh

#####################################################################

#!/bin/ksh

################################################################################

#

# Name: dbsbkup.ksh

#

# Reference: n/a

#

# Description: a production database backup

# a level-0 backup using ontape

#

# Parameters: dbsbkup.ksh <tape device>

# tape device /dev/rmt1

#

#

################################################################################

set -v

set -x

date

su - informix -c "/insight/local/backup/infbkup.ksh" > /dmqjtmp/archiveDbsbkupLog/infbkup.out 2>&1

exit 0

#####################################################################

# Script /insight/local/scripts/getTxnRpt.pl

#####################################################################

#!/usr/bin/perl -w

@QueList = qw(10 21 22 31 32 34 41 46 51 52 70 71 81);

$FF\_all = 0;

$Total\_all = 0;

$Errors\_all = 0;

$Records\_all = 0;

$Messages\_all = 0;

$Bytes\_all = 0;

$c = "|";

$DmqlogDir = "/usr/apps/dmq/beta/LOGS/LOGS.0";

#$DmqlogDir = "/usr/apps/dmq/beta/LOGS/LOGS.1";

#$DmqlogDir = "/usr/apps/dmq/beta/LOGS/LOGS.2";

#$DmqlogDir = "/login/dguo/temp";

#$ReportFile = "/login/dguo/report/trans.rpt";

$stamp = `date +%Y%m%d`;

$ReportFile = "/dmqjtmp/archiveFfileLog/getTxnRpt.${stamp}";

open (OUTFILE,">$ReportFile");

# report for transactions

$DATE = `date`;

format OUTFILE\_TOP =

Daily Transaction Report

@<<<<<<<<<<

$DATE

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

|QNum|Total\_FF| Transaction | Messages | Errors| Records | Bytes |

.

format OUTFILE =

|----|--------|-------------|------------|-------|-------------|---------------|

@<@||@<@||||||@<@>>>>>>>>>>>@<@>>>>>>>>>>@<@>>>>>@<@>>>>>>>>>>>@<@>>>>>>>>>>>>>@<

$c,$qname,$c,$FF,$c,$Total,$c,$Messages,$c,$Errors,$c,$Records,$c,$Bytes,$c

.

sub convert

{

($num) = @\_;

if ( $num > 999999999999 )

{

print "Exceed Max Number, quit!\n";

$num = 999999999999;

exit;

}

$num =

( $num =~ /(\d{1,3})(\d{3})(\d{3})(\d{3})$/ ) ?

sprintf "%d,%3s,%3s,%3s\n",$1,$2,$3,$4 :

( $num =~ /(\d{1,3})(\d{3})(\d{3})$/ ) ?

sprintf "%d,%3s,%3s\n",$1,$2,$3 :

( $num =~ /(\d{1,3})(\d{3})$/ ) ?

sprintf "%d,%3s\n",$1,$2 : $num ;

return $num;

}

$i = 0;

foreach (@QueList)

{

$qname = $\_;

$FF = 0;

$Total = 0;

$Errors = 0;

$Records = 0;

$Messages = 0;

$Bytes = 0;

$i++;

next if ( ! -e "$DmqlogDir/dmqlog.0${qname}");

open (LOGFILE,"$DmqlogDir/dmqlog.0${qname}")||die " Open Files Failed ... ";

while (<LOGFILE>)

{

next if ( !/Transactions/ );

(undef,undef,undef,$num,$err,undef) = split(' ',$\_);

$Total += $num;

$Errors += $err;

$FF++;

$\_ = <LOGFILE>;

(undef,undef,undef,$num,undef) = split(' ',$\_);

$Messages += $num;

$\_ = <LOGFILE>;

(undef,undef,undef,$num,undef) = split(' ',$\_);

$Records += $num;

$\_ = <LOGFILE>;

(undef,undef,undef,$num,undef) = split(' ',$\_);

$Bytes += $num;

}

close (LOGFILE);

$FF\_all += $FF;

$Total\_all += $Total;

$Records\_all += $Records;

$Errors\_all += $Errors;

$Messages\_all += $Messages;

$Bytes\_all += $Bytes;

$FF = convert($FF);

$Total = convert($Total);

$Records = convert($Records);

$Errors = convert($Errors);

$Messages = convert($Messages);

$Bytes = convert($Bytes);

write OUTFILE;

}

close(OUTFILE);

$FF\_all = convert($FF\_all);

$Total\_all = convert($Total\_all);

$Records\_all = convert($Records\_all);

$Errors\_all = convert($Errors\_all);

$Messages\_all = convert($Messages\_all);

$Bytes\_all = convert($Bytes\_all);

open (ENDFILE,">>$ReportFile");

format ENDFILE =

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

Total:

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

@<@||@<@||||||@<@>>>>>>>>>>>@<@>>>>>>>>>>@<@>>>>>@<@>>>>>>>>>>>@<@>>>>>>>>>>>>>@<

$c,$i,$c,$FF\_all,$c,$Total\_all,$c,$Messages\_all,$c,$Errors\_all,$c,$Records\_all,$c,$Bytes\_all,$c

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

.

write ENDFILE;

close (ENDFILE);

#####################################################################

# Script /insight/local/scripts/cron\_bkup.ksh

#####################################################################

#!/usr/bin/ksh

#Save cron jobs for everybody

cd /insight/local/crontabs

cp -p /var/spool/cron/crontabs/\* .

#####################################################################

# Script /sitemgr/b3\_arch/run\_autoarchive.ksh

#####################################################################

#####################################################################

# Script /insight/local/b3\_arch/run\_autoarchive.ksh

#####################################################################

#!/bin/ksh

#####################################################################

#Archive B3 data from Production instance to archive instance #

#Purge will done manually after the verification # #

#####################################################################

umask 0000

INFORMIXSERVER="ardb"

INFORMIXDIR="/usr/apps/inf/ver115UC3"

GL\_DATETIME="%iY/%m/%d %H:%M:%S"

PATH=$INFORMIXDIR/bin:$PATH

export INFORMIXDIR INFORMIXSERVER PATH GL\_DATETIME

local\_dir=/insight/local/b3\_arch

log\_dir=/dmqjtmp/archiveB3Log

week\_no=`date +%w`

year\_no=`date +%Y`

month\_no=`date +%m`

day\_no=`date +%d`

logfile=${log\_dir}/${year\_no}${month\_no}${day\_no}archive.log

aixsupport="lchen@livingstonintl.com"

cd $local\_dir

echo "\nStarts monthly archive and (no purge) program on `date`"

echo "-----------------------------------------------------------------"

#Start archive and Purge job

su - informix -c ${local\_dir}/autoArchive.ksh >> $logfile

echo "End of archive and (no purge) program - `date`"

echo "======================================================================"

echo "Please Stop the Cron Job and Verify the ARCHIVE!!!"| \

mail -s "Monthly B3 Archive Done @ `date`." $aixsupport

#!/bin/ksh

##################################################################################

#

# purpose: run update statistics medium

#

##################################################################################

export INFORMIXDIR=/usr/apps/inf/ver115UC3

export INFORMIXSERVER=ipdb

export PATH=$INFORMIXDIR/bin:$PATH

SQLDIR=/usr/apps/inf/bob/upstat

echo

date

time dbaccess < $SQLDIR/tbls\_med.sql > $SQLDIR/tbls\_med.out 2>&1

time dbaccess < $SQLDIR/tbls\_high.sql > $SQLDIR/tbls\_high.out 2>&1

time dbaccess < $SQLDIR/proc.sql > $SQLDIR/proc.out 2>&1

exit 0

database ip\_0p@ipdb ;

-- update for b3 table;

update statistics high for table b3(b3iid) ;

update statistics high for table b3(transno) ;

update statistics high for table b3(reldate) ;

update statistics high for table b3(approveddate) ;

update statistics high for table b3(createdate) ;

update statistics high for table b3(cargcntrlno) ;

update statistics high for table b3(custoff) ;

update statistics high for table b3(usportexit) ;

update statistics high for table b3(carriercode) ;

update statistics high for table b3(modetransp) ;

update statistics high for table b3(status) ;

update statistics high for table b3(liibrchno) ;

update statistics low for table b3(liibrchno,liirefno) ;

update statistics high for table b3(liiclientno) ;

update statistics low for table b3(liiclientno,liiaccountno) ;

-- update for status\_history table;

update statistics high for table status\_history(b3iid) ;

update statistics low for table status\_history(b3iid,status) ;

-- Update for b3\_subheader table;

update statistics high for table b3\_subheader(b3subiid) ;

update statistics high for table b3\_subheader(b3iid) ;

-- Update for b3\_line table;

update statistics high for table b3\_line(b3lineiid) ;

update statistics high for table b3\_line(b3subiid) ;

update statistics high for table b3\_line(hsno) ;

-- Update for b3\_recap\_details table;

update statistics high for table b3\_recap\_details(b3recapiid) ;

update statistics high for table b3\_recap\_details(b3lineiid) ;

update statistics high for table b3\_recap\_details(proddesc) ;

update statistics high for table b3\_recap\_details(detailponumber) ;

-- Update for tariff table;

update statistics high for table tariff(liiclientno) ;

update statistics high for table tariff(createdate) ;

update statistics high for table tariff(tariffcode) ;

update statistics high for table tariff(lastuseddate) ;

update statistics high for table tariff(tarifftrtmnt) ;

update statistics high for table tariff(hsno) ;

update statistics high for table tariff(remarks) ;

update statistics high for table tariff(moddate) ;

update statistics high for table tariff(b3description) ;

update statistics high for table tariff(productkeyword) ;

update statistics low for table tariff(liiclientno,createdate) ;

update statistics low for table tariff(liiclientno,lastuseddate) ;

update statistics low for table tariff(liiclientno,moddate) ;

update statistics low for table tariff(liiclientno,tarifftrtmnt) ;

update statistics low for table tariff(liiclientno,vendorname,productkeyword,productsufx) ;

-- Update for client\_invoice table;

update statistics high for table client\_invoice(liiclientno);

update statistics low for table client\_invoice(liiclientno,liiaccountno,liibrchno,liirefno,liireftext);

update statistics high for table client\_invoice(itemtypecode);

update statistics high for table client\_invoice(balance);

update statistics high for table client\_invoice(itemstatus);

update statistics high for table client\_invoice(itemdate);

update statistics high for table client\_invoice(totduty);

update statistics high for table client\_invoice(liiaccountno);

update statistics high for table client\_invoice(liirefno);

update statistics high for table client\_invoice(liibrchno);

-- Update for claim\_log table;

update statistics high for table claim\_log(claimlogiid);

update statistics high for table claim\_log(b3acctsecurno);

update statistics low for table claim\_log(b3acctsecurno,b3transno,b3transseqno);

update statistics high for table claim\_log(b2brchno);

update statistics low for table claim\_log(b2brchno,b2refno);

update statistics high for table claim\_log(claimstatus);

update statistics high for table claim\_log(claimcode);

update statistics high for table claim\_log(claimvendorname);

update statistics high for table claim\_log(b3transno);

update statistics high for table claim\_log(claimrefno);

-- Update for as\_accounted table;

update statistics high for table as\_accounted(asacctiid);

update statistics high for table as\_accounted(claimlogiid);

update statistics low for table as\_accounted(claimlogiid,b2subhdrno,b3lineno,b2lineno);

update statistics high for table as\_accounted(b3lineno);

update statistics high for table as\_accounted(b2lineno);

update statistics high for table as\_accounted(hsno);

-- Update for as\_claimed table;

update statistics high for table as\_claimed(asclaimediid);

update statistics high for table as\_claimed(claimlogiid);

update statistics high for table as\_claimed(b3lineno);

update statistics high for table as\_claimed(b2lineno);

update statistics high for table as\_claimed(hsno);

update statistics low for table as\_claimed(claimlogiid,b2subhdrno,b3lineno,b2lineno);

-- Update for carrier table;

update statistics high for table carrier(carriercode);

-- Update for hs\_uom table;

update statistics high for table hs\_uom(hsno);

update statistics low for table hs\_uom(hsno,effdate);

-- Update for hs\_duty\_rate table;

update statistics high for table hs\_duty\_rate(hsno);

update statistics low for table hs\_duty\_rate(hsno,hstarifftrtmnt,effdate);

-- Update for user\_locus\_xref table;

update statistics high for table user\_locus\_xref(userlocusxrefiid);

-- Update for tariff\_code table;

update statistics high for table tariff\_code(tariffcode);

update statistics low for table tariff\_code(tariffcode,effdate,hstarifftrtmnt);

-- Update for lii\_client table;

update statistics high for table lii\_client(liiclientno);

-- Update for securuser table;

update statistics high for table securuser(useriid);

update statistics high for table securuser(username);

-- Update for lii\_account table;

update statistics high for table lii\_account(liiclientno);

update statistics low for table lii\_account(liiclientno,liiaccountno);

-- Update for account\_contact table;

update statistics high for table account\_contact(liiclientno);

update statistics low for table account\_contact(liiclientno,liiaccountno);

close database ;