DB2 Database server and daily maintenance

## Start, access and stop db2 server

Firstly, you may need to know the instance(s) installed on your server, and current database manager instance name:

[/root@admsrv1:/home/db2inst1] **db2ilist**

db2inst1

db2inst2

[/root@admsrv1:/home/db2inst1] **db2 get instance**

The current database manager instance is: db2inst1

DB2® database products provide a number of registry variables and environment variables that you might need to know about to get up and running. You must set values for registry variables that you want to update before you execute the db2start command:

[/root@admsrv1:/home/db2inst1] **db2set -lr**

DB2\_OVERRIDE\_BPF

DB2\_PARALLEL\_IO

DB2ACCOUNT

DB2ADMINSERVER

DB2BQTIME

DB2BQTRY

DB2CHKPTR

DB2CLIINIPATH

DB2CODEPAGE

DB2COMM

DB2COUNTRY

DB2DBDFT

DB2DBMSADDR

DB2DEFPREP

DB2DMNBCKCTLR

DB2INCLUDE

DB2INSTDEF

DB2INSTPROF

DB2IQTIME

DB2LOADREC

DB2LOCK\_TO\_RB

DB2NTNOCACHE

DB2NTPRICLASS

DB2NTWORKSET

DB2OPTIONS

DB2PATH

DB2PRIORITIES

DB2REMOTEPREG

DB2RQTIME

DB2SORCVBUF

DB2SORT

DB2SOSNDBUF

DB2SYSTEM

DB2\_FORCE\_NLS\_CACHE

DB2YIELD

DB2\_AVOID\_PREFETCH

DB2\_COLLECT\_TS\_REC\_INFO

DB2\_GRP\_LOOKUP

DB2\_INDEX\_FREE

DB2\_MMAP\_READ

DB2\_MMAP\_WRITE

DB2DISCOVERYTIME

DB2ENVLIST

DB2MEMDISCLAIM

DB2LIBPATH

DB2CHKSQLDA

DB2PORTRANGE

DB2INSTOWNER

DB2NOEXITLIST

DB2LOADFLAGS

DB2\_HASH\_JOIN

DB2NTMEMSIZE

DB2CHECKCLIENTINTERVAL

DB2\_FALLBACK

DB2PROCESSORS

DB2ATLD\_PORTS

DB2\_SORT\_AFTER\_TQ

DB2\_LIKE\_VARCHAR

DB2ASSUMEUPDATE

DB2MAXFSCRSEARCH

DB2BIDI

DB2\_NEW\_CORR\_SQ\_FF

DB2CHGPWD\_EEE

DB2LOCALE

DB2\_SKIPDELETED

DB2LDAPHOST

DB2LDAPCACHE

DB2LDAP\_CLIENT\_PROVIDER

DB2LDAP\_BASEDN

DB2\_ENABLE\_LDAP

DB2\_SYSTEM\_MONITOR\_SETTINGS

DB2\_FCM\_SETTINGS

DB2SATELLITEID

DB2\_LIC\_STAT\_SIZE

DB2CONNECT\_IN\_APP\_PROCESS

DB2\_NUM\_FAILOVER\_NODES

DB2ROUTINE\_DEBUG

DB2\_DJ\_INI

DB2\_DJ\_COMM

DB2TCPCONNMGRS

DB2\_SQLROUTINE\_PREPOPTS

DB2\_ANTIJOIN

DB2\_DISABLE\_FLUSH\_LOG

DB2\_SELECTIVITY

DB2\_EXTENDED\_OPTIMIZATION

DB2\_ENABLE\_SINGLE\_NIS\_GROUP

DB2\_PINNED\_BP

DB2\_APM\_PERFORMANCE

DB2\_XBSA\_LIBRARY

DB2\_VENDOR\_INI

DB2DOMAINLIST

DB2\_FMP\_COMM\_HEAPSZ

DB2\_SNAPSHOT\_NOAUTH

DB2\_LOGGER\_NON\_BUFFERED\_IO

DB2\_EVALUNCOMMITTED

DB2TERRITORY

DB2\_PARTITIONEDLOAD\_DEFAULT

DB2\_ALLOCATION\_SIZE

DB2\_NO\_FORK\_CHECK

DB2\_REDUCED\_OPTIMIZATION

DB2\_USE\_PAGE\_CONTAINER\_TAG

DB2\_NUM\_CKPW\_DAEMONS

DB2\_KEEPTABLELOCK

DB2GRAPHICUNICODESERVER

DB2\_MINIMIZE\_LISTPREFETCH

DB2\_INLIST\_TO\_NLJN

DB2\_MEM\_TUNING\_RANGE

DB2\_TRUSTED\_BINDIN

DB2\_CLPPROMPT

DB2\_FORCE\_APP\_ON\_MAX\_LOG

DB2\_CLP\_EDITOR

DB2\_CLP\_HISTSIZE

DB2LOGINRESTRICTIONS

DB2\_LOAD\_COPY\_NO\_OVERRIDE

DB2\_USE\_DB2JCCT2\_JROUTINE

DB2\_MAX\_NON\_TABLE\_LOCKS

DB2\_SMS\_TRUNC\_TMPTABLE\_THRESH

DB2\_USE\_ALTERNATE\_PAGE\_CLEANING

DB2\_HADR\_BUF\_SIZE

DB2\_MAX\_CLIENT\_CONNRETRIES

DB2\_CONNRETRIES\_INTERVAL

DB2\_DOCHOST

DB2\_DOCPORT

DB2\_TAPEMGR\_TAPE\_EXPIRATION

DB2\_OBJECT\_TABLE\_ENTRIES

DB2\_LOGGING\_DETAIL

DB2\_VIEW\_REOPT\_VALUES

DB2\_SELUDI\_COMM\_BUFFER

DB2\_RESOURCE\_POLICY

DB2TCP\_CLIENT\_RCVTIMEOUT

DB2\_SKIPINSERTED

DB2CONNECT\_DISCONNECT\_ON\_INTERRUPT

DB2\_LARGE\_PAGE\_MEM

DB2\_ALTERNATE\_GROUP\_LOOKUP

DB2AUTH

DB2FFDC

DB2FODC

DB2\_ASYNC\_IO\_MAXFILOP

DB2RSHCMD

DB2RSHTIMEOUT

DB2\_MDC\_ROLLOUT

DB2\_TRUNCATE\_REUSESTORAGE

DB2\_WORKLOAD

DB2\_DXX\_PATHS\_ALLOWED\_READ

DB2\_DXX\_PATHS\_ALLOWED\_WRITE

DB2TCP\_CLIENT\_CONTIMEOUT

DB2\_MAX\_INACT\_STMTS

DB2FCMCOMM

DB2\_EXTENDED\_IO\_FEATURES

DB2\_UTIL\_MSGPATH

DB2\_ENABLE\_AUTOCONFIG\_DEFAULT

DB2\_MAP\_XML\_AS\_CLOB\_FOR\_DLC

DB2\_OPT\_MAX\_TEMP\_SIZE

DB2\_MAX\_LOB\_BLOCK\_SIZE

DB2\_MINIMUM\_CLIENT\_LEVEL

DB2CONNECT\_ENABLE\_EURO\_CODEPAGE

DB2TRC\_DEF\_BUFFSIZE

DB2\_RESOLVE\_CALL\_CONFLICT

DB2\_IO\_PRIORITY\_SETTING

DB2\_EVMON\_STMT\_FILTER

DB2\_SERVER\_CONTIMEOUT

DB2\_DISPATCHER\_PEEKTIMEOUT

DB2\_EVMON\_EVENT\_LIST\_SIZE

DB2\_MEMORY\_PROTECT

DB2\_SET\_MAX\_CONTAINER\_SIZE

DB2\_UPDDBCFG\_SINGLE\_DBPARTITION

DB2\_LIMIT\_FENCED\_GROUP

DB2\_CAPTURE\_LOCKTIMEOUT

DB2\_HADR\_NO\_IP\_CHECK

DB2\_HADR\_PEER\_WAIT\_LIMIT

DB2\_THREAD\_SUSPENSION

DB2\_OPTSTATS\_LOG

DB2\_ATS\_ENABLE

DB2\_ALLOW\_PUREXML\_IN\_DPF

DB2\_KEEP\_AS\_AND\_DMS\_CONTAINERS\_OPEN

DB2\_HADR\_SOSNDBUF

DB2\_HADR\_SORCVBUF

DB2\_USE\_IOCP

DB2\_SERVER\_ENCALG

[/root@admsrv1:/home/db2inst1 > **db2set**

DB2\_CAPTURE\_LOCKTIMEOUT=ON

DB2\_SKIPINSERTED=ON

DB2\_EVALUNCOMMITTED=YES

DB2\_FMP\_COMM\_HEAPSZ=12000

DB2\_SKIPDELETED=ON

DB2\_HASH\_JOIN=YES

DB2LIBPATH=/usr/lib:/opt/IBM/db2cmv8/lib

DB2ENVLIST=LIBPATH IBMCMROOT ICMDLL EXTSHM

DB2\_RR\_TO\_RS=YES

**DB2COMM=tcpip**

DB2AUTOSTART=NO

[/root@admsrv1:/home/db2inst1] **db2start**

03/14/2013 12:12:05 0 0 SQL1063N DB2START processing was successful.

SQL1063N DB2START processing was successful.

DB2 reports the service name for your instance, then you can identify the tcpip service port of this instance in /etc/services:

[/root@admsrv1:/home/db2inst1] **get dbm cfg | grep SVCE**

TCP/IP Service name (SVCENAME) = db2c\_db2inst1

[/root@admsrv1:/home/db2inst1] **grep db2c\_db2inst1 /etc/services**

db2c\_db2inst1 50000/tcp

Locate the name of the administration database you want to connect to. Make a note of the DB2 instance that the database is installed on, because different instances can have different connection port numbers.

[/root@admsrv1:/home/db2inst1] **db2 list db directory**

System Database Directory

Number of entries in the directory = 3

Database 1 entry:

Database alias = TOOLSDB

Database name = TOOLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 2 entry:

Database alias = ICMNLSDB

Database name = ICMNLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 3 entry:

Database alias = RMDB

Database name = RMDB

Node name = RM\_NODE

Database release level = c.00

Comment =

Directory entry type = Remote

Catalog database partition number = -1

Alternate server hostname =

Alternate server port number =

A list of the local and remote databases displays. Local databases are labeled: ***indirect***. Remote databases under other instances are labeled: ***Remote***.

Connect to database before you run any SQL:

[/root@admsrv1:/home/db2inst1] db2 connect to icmnlsdb user *db2inst1* using *yahoo*

Database Connection Information

Database server = DB2/AIX64 9.5.5

SQL authorization ID = DB2INST1

Local database alias = ICMNLSDB

[/root@admsrv1:/home/db2inst1 > **db2 list application**

[/root@admsrv1:/home/db2inst1 > **db2 force applications all**

DB20000I The FORCE APPLICATION command completed successfully.

DB21024I This command is asynchronous and may not be effective immediately.

//root@admsrv1:/home/db2inst1 **> db2 list tables for schema icmadmin**

A list of database tables, and the schema name associated with each table displays. Make a note of the database schema name, which is required by the server configuration utility.

Table/View Schema Type Creation time

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

ADVISE\_INDEX DB2INST1 T 2013-01-06-09.20.34.765421

ADVISE\_INSTANCE DB2INST1 T 2013-01-06-09.20.33.797574

ADVISE\_MQT DB2INST1 T 2013-01-06-09.20.36.423539

ADVISE\_PARTITION DB2INST1 T 2013-01-06-09.20.37.748243

ADVISE\_TABLE DB2INST1 T 2013-01-06-09.20.38.867968

ADVISE\_WORKLOAD DB2INST1 T 2013-01-06-09.20.35.476043

…….

XACT\_DEADLOCK\_ON\_TRANSACTIONS DB2INST1 T 2013-01-07-09.22.05.246008

XACT\_LMS\_NO\_RESPONSE DB2INST1 T 2013-01-04-09.07.05.426242

55 record(s) selected.

//root@admsrv1:/home/db2inst1 **> db2 list node directory**

Node names and other data for all databases installed or defined on the remote server display. Locate the connection port number associated with the remote system administration database.

Attention: The procedure for identifying the port number varies by operating system. Choose the procedure for the operating system that the remote database is on; For Unix, 2. Enter cd /usr/etc, cat services, Scroll through the list of services until you find the connection port number for the database instance of the remote database. The instance name is usually listed as a comment.

Node Directory

Number of entries in the directory = 2

Node 1 entry:

Node name = LSLBNODE

Comment =

Directory entry type = LOCAL

Protocol = TCPIP

Hostname = 127.0.0.1

Service name = 50000

Node 2 entry:

Node name = RM\_NODE

Comment =

Directory entry type = LOCAL

Protocol = TCPIP

Hostname = 127.0.0.1

Service name = 50001

**db2 => uncatalog database icmnlsdb**

DB20000I The UNCATALOG DATABASE command completed successfully.

DB21056W Directory changes may not be effective until the directory cache is refreshed.

**db2 => catalog db icmnlsdb as lnxls**

DB20000I The CATALOG DATABASE command completed successfully.

DB21056W Directory changes may not be effective until the directory cache is refreshed.

**db2 => list database directory**

System Database Directory

Number of entries in the directory = 3

Database 1 entry:

Database alias = TOOLSDB

Database name = TOOLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 2 entry:

Database alias = LNXLS

Database name = ICMNLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 3 entry:

Database alias = RMDB

Database name = RMDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

**db2 => uncatalog db rmdb**

DB20000I The UNCATALOG DATABASE command completed successfully.

DB21056W Directory changes may not be effective until the directory cache is refreshed.

**db2 => catalog db rmdb as lnxrm**

DB20000I The CATALOG DATABASE command completed successfully.

DB21056W Directory changes may not be effective until the directory cache is refreshed.

db2 => **terminate**

DB20000I The TERMINATE command completed successfully.

[/root@admsrv1:/home/db2inst1 > **db2stop**

03/14/2013 12:11:57 0 0 SQL1064N DB2STOP processing was successful.

SQL1064N DB2STOP processing was successful.

When the resource manager Web application starts, it attempts to make three connections to the resource manager database. If the Web application cannot connect to the resource manager database, the resource manager cannot process client requests.

The following steps explain how to validate the connection between the Web application and resource manager database.

1. If the resource manager database is on a remote server and you have not cataloged the database, either locally catalog the database or log on to the remote server where you installed the database. Whether the database is local or remote, you must log on with, or have the authority to switch to, a user ID that has db2admin privileges.

2. At a DB2 command prompt, enter list applications The system displays a table of all DB2 applications.

* If the table lists three resource manager applications, then the connections are working correctly.
* If the connections do not appear, then the resource manager Web application is having a problem connecting to the database. Usually, this problem occurs when the user ID and password used by the resource manager to connect to the database are invalid.

Configuring the DB2 database manager and/or database with configuration parameters

The disk space and memory allocated by the database manager on the basis of default values of the parameters might be sufficient to meet your needs. In some situations, however, you might not be able to achieve maximum performance using these default values.

**About this task**

Since the default values are oriented towards machines that have relatively small memory resources and are dedicated as database servers, you might need to modify these values if your environment has:

* Large databases
* Large numbers of connections
* High performance requirements for a specific application
* Unique query or transaction loads or types

Each transaction processing environment is unique in one or more aspects. These differences can have a profound impact on the performance of the database manager when using the default configuration. For this reason, you are strongly advised to tune your configuration for your environment.

A good starting point for tuning your configuration is to use the Configuration Advisor or the AUTOCONFIGURE command which will generate values for parameters based on your responses to questions about workload characteristics.

Some configuration parameters can be set to AUTOMATIC, allowing the database manager to automatically manage these parameters to reflect the current resource requirements. To turn off the AUTOMATIC setting of a configuration parameter while maintaining the current internal setting, use the MANUAL keyword with the UPDATE DATABASE CONFIGURATION command. If the database manager updates the value of these parameters, the ***GET DB CFG SHOW DETAIL*** and ***GET DBM CFG SHOW DETAIL*** commands will show the new value.

Parameters for an individual database are stored in a configuration file named SQLDBCONF. This file is stored along with other control files for the database in the SQLnnnnn directory, where nnnnn is a number assigned when the database was created. **Each database has its own configuration file**, and most of the parameters in the file specify the amount of resources allocated to that database. The file also contains descriptive information, as well as flags that indicate the status of the database.

**Attention:** DO NOT edit db2systm, SQLDBCON, or SQLDBCONF using a method other than those provided by the database manager, you might make the database unusable. Do not change these files using methods other than those documented and supported by the database manager.

In a partitioned database environment, a separate SQLDBCONF file exists for each database partition. The values in the SQLDBCONF file may be the same or different at each database partition, but the recommendation is that in a homogeneous environment, the configuration parameter values should be the same on all database partitions. Typically, there could be a catalog node needing different database configuration parameters setting, while the other data partitions have different values again, depending on their machine types, and other information.

Note: You can update configuration parameters or see their values using IBM® Data Studio. For more information, follow the Data Studio related link.

**Procedure**

Update configuration parameters.

Using the command line processor:

Commands to change the settings can be entered as follows:

For database manager configuration parameters:

*GET DATABASE MANAGER CONFIGURATION (or GET DBM CFG)*

*UPDATE DATABASE MANAGER CONFIGURATION (or UPDATE DBM CFG)*

*RESET DATABASE MANAGER CONFIGURATION (or RESET DBM CFG)*

to reset all database manager parameters to their default values

*AUTOCONFIGURE*

For database configuration parameters:

*GET DATABASE CONFIGURATION (or GET DB CFG)*

*UPDATE DATABASE CONFIGURATION (or UPDATE DB CFG)*

RESET DATABASE CONFIGURATION (or RESET DB CFG) to reset *all* database parameters to their default values

AUTOCONFIGURE

Using application programming interfaces (APIs):

The APIs can be called from an application or a host-language program. Call the following DB2® APIs to view or update configuration parameters:

*db2AutoConfig - Access the Configuration Advisor*

*db2CfgGet - Get the database manager or database configuration parameters*

*db2CfgSet - Set the database manager or database configuration parameters*

Using common SQL application programming interface (API) procedures:

You can call the common SQL API procedures from an SQL-based application, a DB2 command line, or a command script. Call the following procedures to view or update configuration parameters:

GET\_CONFIG - Get the database manager or database configuration parameters

SET\_CONFIG - Set the database manager or database configuration parameters

Using IBM Data Studio, right-click the instance to open the task assistant to update the database manager configuration parameters.

Using the Configuration Assistant

The Configuration Assistant can also be used to set the database manager configuration parameters on a client. Other parameters can be changed online; these are called configurable online configuration parameters.

View updated configuration values.

For some database manager configuration parameters, the database manager must be stopped (db2stop) and then restarted (db2start) for the new parameter values to take effect.

For some database parameters, changes will only take effect when the database is reactivated, or switched from offline to online. In these cases, all applications (sessions) must first disconnect from the database. (If the database was activated, or switched from offline to online, then it must be deactivated and reactivated.) Then, at the first new connect to the database, the changes will take effect.

If you change the setting of a configurable online database manager configuration parameter while you are attached to an instance, the default behavior of the UPDATE DBM CFG command will be to apply the change immediately. If you do not want the change applied immediately, use the DEFERRED option on the UPDATE DBM CFG command.

To change a database manager configuration parameter online:

db2 attach to *instance-name*

db2 update dbm cfg using *parameter-name* *value*

db2 detach

For clients, changes to the database manager configuration parameters t**ake effect the next time the client connects to a server.**

If you change a configurable online database configuration parameter while connected, the default behavior is to apply the change online, wherever possible. You should note that some parameter changes might take a noticeable amount of time to take effect due to the overhead associated with allocating space. To change configuration parameters online from the command line processor, a connection to the database is required.

To change a database configuration parameter online:

db2 connect to *dbname*

db2 update db cfg using *parameter-name* *parameter-value*

db2 connect reset

Each configurable online configuration parameter has a propagation class associated with it. The propagation class indicates when you can expect a change to the configuration parameter to take effect. There are four propagation classes:

**Immediate**: Parameters that change immediately upon command or API invocation. For example, diaglevel has a propagation class of immediate.

**Statement boundary**: Parameters that change on statement and statement-like boundaries. For example, if you change the value of sortheap, all new requests will start using the new value.

**Transaction boundary**: Parameters that change on transaction boundaries. For example, a new value for dl\_expint is updated after a COMMIT statement.

**Connection**: Parameters that change on new connection to the database. For example, a new value for dft\_degree takes effect for new applications connecting to the database.

While new parameter values might not be immediately effective, viewing the parameter settings (using the GET DATABASE MANAGER CONFIGURATION or GET DATABASE CONFIGURATION command) will always show the latest updates. Viewing the parameter settings using the SHOW DETAIL clause on these commands will show both the latest updates and the values in memory.

Rebind applications after updating database configuration parameters.

Changing some database configuration parameters can influence the access plan chosen by the SQL and XQuery optimizer. After changing any of these parameters, you should consider rebinding your applications to ensure the best access plan is being used for your SQL and XQuery statements. Any parameters that were modified online (for example, by using the UPDATE DATABASE CONFIGURATION IMMEDIATE command) will cause the SQL and XQuery optimizer to choose new access plans for new query statements. However, the query statement cache will not be purged of existing entries. To clear the contents of the query cache, use the FLUSH PACKAGE CACHE statement.

Note: A number of configuration parameters (for example, userexit) are described as having acceptable values of either Yes or No, or On or Off in the help and other DB2 documentation. To clarify, Yes should be considered equivalent to On and No should be considered equivalent to Off.

db2inst1@admsrv1$ db2 get db cfg

SQL1024N A database connection does not exist. SQLSTATE=08003

db2inst1@admsrv1$ db2 connect to icmnlsdb

Database Connection Information

Database server = DB2/AIX64 9.5.7

SQL authorization ID = DB2INST1

Local database alias = ICMNLSDB

db2inst1@admsrv1$ db2 get db cfg

Database Configuration for Database

Database configuration release level = 0x0c00

Database release level = 0x0c00

Database territory = US

Database code page = 819

Database code set = ISO8859-1

Database country/region code = 1

Database collating sequence = UNIQUE

Alternate collating sequence (ALT\_COLLATE) =

Number compatibility = OFF

Varchar2 compatibility = OFF

Database page size = 4096

Dynamic SQL Query management (DYN\_QUERY\_MGMT) = DISABLE

Discovery support for this database (DISCOVER\_DB) = ENABLE

Restrict access = NO

Default query optimization class (DFT\_QUERYOPT) = 2

Degree of parallelism (DFT\_DEGREE) = 1

Continue upon arithmetic exceptions (DFT\_SQLMATHWARN) = NO

Default refresh age (DFT\_REFRESH\_AGE) = 0

Default maintained table types for opt (DFT\_MTTB\_TYPES) = SYSTEM

Number of frequent values retained (NUM\_FREQVALUES) = 10

Number of quantiles retained (NUM\_QUANTILES) = 20

Decimal floating point rounding mode (DECFLT\_ROUNDING) = ROUND\_HALF\_EVEN

Backup pending = NO

Database is consistent = NO

Rollforward pending = NO

Restore pending = NO

Multi-page file allocation enabled = YES

Log retain for recovery status = RECOVERY

User exit for logging status = YES

Self tuning memory (SELF\_TUNING\_MEM) = OFF

Size of database shared memory (4KB) (DATABASE\_MEMORY) = AUTOMATIC(131168)

Database memory threshold (DB\_MEM\_THRESH) = 10

Max storage for lock list (4KB) (LOCKLIST) = 10000

Percent. of lock lists per application (MAXLOCKS) = 10

Package cache size (4KB) (PCKCACHESZ) = (MAXAPPLS\*8)

Sort heap thres for shared sorts (4KB) (SHEAPTHRES\_SHR) = 79000

Sort list heap (4KB) (SORTHEAP) = 6528

Database heap (4KB) (DBHEAP) = AUTOMATIC(2400)

Catalog cache size (4KB) (CATALOGCACHE\_SZ) = (MAXAPPLS\*5)

Log buffer size (4KB) (LOGBUFSZ) = 32

Utilities heap size (4KB) (UTIL\_HEAP\_SZ) = 5000

Buffer pool size (pages) (BUFFPAGE) = 1000

SQL statement heap (4KB) (STMTHEAP) = 8192

Default application heap (4KB) (APPLHEAPSZ) = AUTOMATIC(1689)

Application Memory Size (4KB) (APPL\_MEMORY) = AUTOMATIC(40000)

Statistics heap size (4KB) (STAT\_HEAP\_SZ) = AUTOMATIC(4384)

Interval for checking deadlock (ms) (DLCHKTIME) = 10000

Lock timeout (sec) (LOCKTIMEOUT) = 300

Changed pages threshold (CHNGPGS\_THRESH) = 60

Number of asynchronous page cleaners (NUM\_IOCLEANERS) = 1

Number of I/O servers (NUM\_IOSERVERS) = 3

Index sort flag (INDEXSORT) = YES

Sequential detect flag (SEQDETECT) = YES

Default prefetch size (pages) (DFT\_PREFETCH\_SZ) = AUTOMATIC

Track modified pages (TRACKMOD) = OFF

Default number of containers = 1

Default tablespace extentsize (pages) (DFT\_EXTENT\_SZ) = 32

Max number of active applications (MAXAPPLS) = 320

Average number of active applications (AVG\_APPLS) = 5

Max DB files open per application (MAXFILOP) = 61440

Log file size (4KB) (LOGFILSIZ) = 50000

Number of primary log files (LOGPRIMARY) = 10

Number of secondary log files (LOGSECOND) = 20

Changed path to log files (NEWLOGPATH) =

Path to log files = /lsactivelogs/ls/NODE0000/

Overflow log path (OVERFLOWLOGPATH) =

Mirror log path (MIRRORLOGPATH) =

First active log file = S0128782.LOG

Block log on disk full (BLK\_LOG\_DSK\_FUL) = NO

Block non logged operations (BLOCKNONLOGGED) = NO

Percent max primary log space by transaction (MAX\_LOG) = 0

Num. of active log files for 1 active UOW(NUM\_LOG\_SPAN) = 0

Group commit count (MINCOMMIT) = 1

Percent log file reclaimed before soft chckpt (SOFTMAX) = 100

Log retain for recovery enabled (LOGRETAIN) = RECOVERY

User exit for logging enabled (USEREXIT) = OFF

HADR database role = STANDARD

HADR local host name (HADR\_LOCAL\_HOST) =

HADR local service name (HADR\_LOCAL\_SVC) =

HADR remote host name (HADR\_REMOTE\_HOST) =

HADR remote service name (HADR\_REMOTE\_SVC) =

HADR instance name of remote server (HADR\_REMOTE\_INST) =

HADR timeout value (HADR\_TIMEOUT) = 120

HADR log write synchronization mode (HADR\_SYNCMODE) = NEARSYNC

HADR peer window duration (seconds) (HADR\_PEER\_WINDOW) = 0

First log archive method (LOGARCHMETH1) = DISK:/lsarchivelogs/ls/

Options for logarchmeth1 (LOGARCHOPT1) =

Second log archive method (LOGARCHMETH2) = OFF

Options for logarchmeth2 (LOGARCHOPT2) =

Failover log archive path (FAILARCHPATH) =

Number of log archive retries on error (NUMARCHRETRY) = 5

Log archive retry Delay (secs) (ARCHRETRYDELAY) = 20

Vendor options (VENDOROPT) =

Auto restart enabled (AUTORESTART) = ON

Index re-creation time and redo index build (INDEXREC) = SYSTEM (RESTART)

Log pages during index build (LOGINDEXBUILD) = OFF

Default number of loadrec sessions (DFT\_LOADREC\_SES) = 1

Number of database backups to retain (NUM\_DB\_BACKUPS) = 7

Recovery history retention (days) (REC\_HIS\_RETENTN) = 7

Auto deletion of recovery objects (AUTO\_DEL\_REC\_OBJ) = ON

TSM management class (TSM\_MGMTCLASS) =

TSM node name (TSM\_NODENAME) =

TSM owner (TSM\_OWNER) =

TSM password (TSM\_PASSWORD) =

Automatic maintenance (AUTO\_MAINT) = OFF

Automatic database backup (AUTO\_DB\_BACKUP) = OFF

Automatic table maintenance (AUTO\_TBL\_MAINT) = OFF

Automatic runstats (AUTO\_RUNSTATS) = OFF

Automatic statement statistics (AUTO\_STMT\_STATS) = OFF

Automatic statistics profiling (AUTO\_STATS\_PROF) = OFF

Automatic profile updates (AUTO\_PROF\_UPD) = OFF

Automatic reorganization (AUTO\_REORG) = OFF

Enable XML Character operations (ENABLE\_XMLCHAR) = YES

WLM Collection Interval (minutes) (WLM\_COLLECT\_INT) = 0

db2inst1@admsrv1$ db2 get dbm cfg

Database Manager Configuration

Node type = Enterprise Server Edition with local and remote clients

Database manager configuration release level = 0x0c00

CPU speed (millisec/instruction) (CPUSPEED) = 2.834065e-07

Communications bandwidth (MB/sec) (COMM\_BANDWIDTH) = 1.000000e+02

Max number of concurrently active databases (NUMDB) = 8

Federated Database System Support (FEDERATED) = NO

Transaction processor monitor name (TP\_MON\_NAME) =

Default charge-back account (DFT\_ACCOUNT\_STR) =

Java Development Kit installation path (JDK\_PATH) = /home/db2inst1/sqllib/java/jdk64

Diagnostic error capture level (DIAGLEVEL) = 3

Notify Level (NOTIFYLEVEL) = 3

Diagnostic data directory path (DIAGPATH) = /home/db2inst1/sqllib/db2dump

Default database monitor switches

Buffer pool (DFT\_MON\_BUFPOOL) = OFF

Lock (DFT\_MON\_LOCK) = ON

Sort (DFT\_MON\_SORT) = OFF

Statement (DFT\_MON\_STMT) = OFF

Table (DFT\_MON\_TABLE) = OFF

Timestamp (DFT\_MON\_TIMESTAMP) = ON

Unit of work (DFT\_MON\_UOW) = OFF

Monitor health of instance and databases (HEALTH\_MON) = ON

SYSADM group name (SYSADM\_GROUP) = DB2GRP1

SYSCTRL group name (SYSCTRL\_GROUP) =

SYSMAINT group name (SYSMAINT\_GROUP) =

SYSMON group name (SYSMON\_GROUP) =

Client Userid-Password Plugin (CLNT\_PW\_PLUGIN) =

Client Kerberos Plugin (CLNT\_KRB\_PLUGIN) =

Group Plugin (GROUP\_PLUGIN) =

GSS Plugin for Local Authorization (LOCAL\_GSSPLUGIN) =

Server Plugin Mode (SRV\_PLUGIN\_MODE) = UNFENCED

Server List of GSS Plugins (SRVCON\_GSSPLUGIN\_LIST) =

Server Userid-Password Plugin (SRVCON\_PW\_PLUGIN) =

Server Connection Authentication (SRVCON\_AUTH) = NOT\_SPECIFIED

Cluster manager (CLUSTER\_MGR) =

Database manager authentication (AUTHENTICATION) = SERVER

Cataloging allowed without authority (CATALOG\_NOAUTH) = NO

Trust all clients (TRUST\_ALLCLNTS) = YES

Trusted client authentication (TRUST\_CLNTAUTH) = CLIENT

Bypass federated authentication (FED\_NOAUTH) = NO

Default database path (DFTDBPATH) = /home/db2inst1

Database monitor heap size (4KB) (MON\_HEAP\_SZ) = AUTOMATIC(256)

Java Virtual Machine heap size (4KB) (JAVA\_HEAP\_SZ) = 2048

Audit buffer size (4KB) (AUDIT\_BUF\_SZ) = 0

Size of instance shared memory (4KB) (INSTANCE\_MEMORY) = AUTOMATIC(3601140)

Backup buffer default size (4KB) (BACKBUFSZ) = 1024

Restore buffer default size (4KB) (RESTBUFSZ) = 1024

Agent stack size (AGENT\_STACK\_SZ) = 1024

Sort heap threshold (4KB) (SHEAPTHRES) = 79000

Directory cache support (DIR\_CACHE) = YES

Application support layer heap size (4KB) (ASLHEAPSZ) = 15

Max requester I/O block size (bytes) (RQRIOBLK) = 32767

Query heap size (4KB) (QUERY\_HEAP\_SZ) = 32768

Workload impact by throttled utilities(UTIL\_IMPACT\_LIM) = 10

Priority of agents (AGENTPRI) = SYSTEM

Agent pool size (NUM\_POOLAGENTS) = AUTOMATIC(250)

Initial number of agents in pool (NUM\_INITAGENTS) = 0

Max number of coordinating agents (MAX\_COORDAGENTS) = AUTOMATIC(500)

Max number of client connections (MAX\_CONNECTIONS) = AUTOMATIC(MAX\_COORDAGENTS)

Keep fenced process (KEEPFENCED) = YES

Number of pooled fenced processes (FENCED\_POOL) = AUTOMATIC(MAX\_COORDAGENTS)

Initial number of fenced processes (NUM\_INITFENCED) = 0

Index re-creation time and redo index build (INDEXREC) = RESTART

Transaction manager database name (TM\_DATABASE) = 1ST\_CONN

Transaction resync interval (sec) (RESYNC\_INTERVAL) = 180

SPM name (SPM\_NAME) = admsrv1

SPM log size (SPM\_LOG\_FILE\_SZ) = 256

SPM resync agent limit (SPM\_MAX\_RESYNC) = 20

SPM log path (SPM\_LOG\_PATH) =

TCP/IP Service name (SVCENAME) = db2c\_db2inst1

Discovery mode (DISCOVER) = SEARCH

Discover server instance (DISCOVER\_INST) = ENABLE

Maximum query degree of parallelism (MAX\_QUERYDEGREE) = ANY

Enable intra-partition parallelism (INTRA\_PARALLEL) = NO

Maximum Asynchronous TQs per query (FEDERATED\_ASYNC) = 0

No. of int. communication buffers(4KB)(FCM\_NUM\_BUFFERS) = AUTOMATIC(4096)

No. of int. communication channels (FCM\_NUM\_CHANNELS) = AUTOMATIC(2048)

Node connection elapse time (sec) (CONN\_ELAPSE) = 10

Max number of node connection retries (MAX\_CONNRETRIES) = 5

Max time difference between nodes (min) (MAX\_TIME\_DIFF) = 60

db2start/db2stop timeout (min) (START\_STOP\_TIME) = 10

## Error LS RC 7015 SQL RC=-911 linked to concurrency control in DB2 Content Manager Database

When multiple users concurrently access the DB2® Content Manager database for operations such as retrieval, insertion, update, and deletion, you might get the SQL error RC=-911 (SQL0911N) because of database lock contention.

In a concurrent environment, *lock contention* occurs because the database manager must ensure data integrity.

**Possible causes**

Lock contention might occur because of a timeout (reason code 68) or deadlock.

*Timeout* means that DB2 was not able to lock a resource within the time specified by the LOCKTIMEOUT parameter. The DB2 Content Manager Default value for LOCKTIMEOUT is 30 seconds.

*Deadlock* means that one application is waiting for another application to release the lock. The lingering application is locking the resource needed by the other.

**Actions**

Search the **ICMSERVER.LOG** for SQL0911N and identify the reason code. You can detect SQL0911N and avoid lock contention by performing one of the following steps:

Update the library server and resource manager database statistics and execution utilities ***REORG, RUNSTATS, REBIND*** to maintain good performance. You must bind the application again after successfully performing ***RUNSTATS***.

Ensure that your application has **short transactions** (long transactions are NOT welcome in database application design).

When you define an item type, **create an index for attributes** that will be searched often. DB2 Universal Database uses indexes to retrieve the correct table row. When an index is absent, DB2 Universal Database must scan a table to meet the search criteria. Other applications can run concurrently, accessing the table being scanned, which could result in concurrency control issues.

If two transactions attempt to operate on the same row, locking can occur. This can happen from a variety of functions. For example, if one user is creating a document (with a long-running transaction) and another user performs a search that checks that record, the second transaction will be locked out until the first transaction is completed. To determine whether this is the cause of an error:

Run the application with the library server trace level set to -15.

Find the SQL error that reports the lock, and then find the item ID being accessed.

Search further up in the log file to see if another user session is also operating on that item ID. In the server log, each session is identified by a unique string such as "?05161633031148".

Ensure that all users have and use unique user IDs. If two users attempt to use the same user ID, locking can occur in functions such as check out or document routing APIs.

Set the following DB2 variable to avoid concurrency problems and improve performance of SQL update statements:

db2set DB2\_EVALUNCOMMITTED=YES

Run the DB2 utilities ***REGOR, RUNSTATS and REBIND*** for this variable to take effect. This variable helps prevent deadlocks on DB2 Universal Database.

**Add/change row STATIC\_FILE\_PERMISSION on rmdb database table: RMCONFIGURATION**

Reserved for Content Manager EE resource manager configuration.

| *Table 1. RMCONFIGURATION* | | |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Attribute** |
| PROPERTYNAME | VARCHAR(256) | NOT NULL |
| PROPERTYVALUE | VARCHAR(1536) FOR BIT DATA | NOT NULL |
| PROPERTYBINARY | VARCHAR(1536) DB2 ONLY RAW(1536) ORACLE ONLY | NULLABLE |

When a file is written to file system storage, the resource manager runs the **chmod** command to set the default permission for the file when it is not being created, updated, or deleted. The **chmod** system call uses the value from the **STATIC\_FILE\_PERMISSION** parameter to set the file permissions on the file. The default file permission is 400, a value that permits read by user and denies group and others from accessing the file. Other possible values include 440 (read by user and group), 404 (read by user and others), and 444 (read by user, group, and others).

**To count the registered users, enter the following command at a DB2**

db2=> select count(\*) from icmstusers where userid not in (’ICMCONCT’,’ICMPUBLIC’) and userkind=0

db2 => list database directory

System Database Directory

Number of entries in the directory = 3

Database 1 entry:

Database alias = TOOLSDB

Database name = TOOLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 2 entry:

Database alias = LNXRM

Database name = RMDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

Database 3 entry:

Database alias = LNXLS

Database name = ICMNLSDB

Local database directory = /home/db2inst1

Database release level = c.00

Comment =

Directory entry type = Indirect

Catalog database partition number = 0

Alternate server hostname =

Alternate server port number =

C:\Program Files\IBM\db2cmv8\cmgmt\connectors

ICMSERVER=LINUXLS

ICMSERVERREPTYPE=DB2

ICMSCHEMA=icmadmin

ICMSSO=FALSE

ICMDBAUTH=SERVER

ICMREMOTE=TRUE

ICMHOSTNAME=ibmserver

ICMPORT=50000

ICMREMOTEDB=icmnlsdb

ICMNODENAME=LINUXLS

ICMOSTYPE=LINUX

ICMSERVER=CM07LS

ICMSERVERREPTYPE=DB2

ICMSCHEMA=icmadmin

ICMSSO=FALSE

ICMDBAUTH=SERVER

ICMREMOTE=TRUE

ICMHOSTNAME=cm07

ICMPORT=50000

ICMREMOTEDB=icmnlsdb

ICMNODENAME=CM07LS

ICMOSTYPE=AIX

ICMSERVER=LMSLS

ICMSERVERREPTYPE=DB2

ICMSCHEMA=icmadmin

ICMSSO=FALSE

ICMDBAUTH=SERVER

ICMREMOTE=TRUE

ICMHOSTNAME=admsrv1\_svc

ICMPORT=50000

ICMREMOTEDB=icmnlsdb

ICMNODENAME=LMSLS

ICMOSTYPE=AIX

SERVERREPTYPE

SERVERREPTYPE is a parameter in the cmbicmsrvs.ini file. This file resides on the same workstation as your client. One of the following values indicates how the client connects to the IBM Content Manager library server.

DB2 Tells the API to use the user ID and password that is entered in the login window to connect to DB2 on the server. If the DB2 connection fails, the shared connection ID and password are used in a second attempt to connect.

DB2CON Tells the API to use the shared client ID and password on the first connection. Therefore, the user is a nonadministrative user and

## Snapshot monitor CLP commands for db2 performance analysis

The following table lists all the supported snapshot request types. For certain request types, some information is returned only if the associated monitor switch is set ON. See the individual monitor elements to determine if a required element is under switch control.

TIPS: Use the db2pd -db <database name> -locks -transactions -applications -dynamic command to get the following results

| **Monitor level** | **CLP command** | **Information returned** |
| --- | --- | --- |
| Connections list | list applications [show detail] | Application identification information for all applications currently connected to a database that is managed by the DB2® instance on the partition where snapshot is taken. |
| Connections list | list applications for database dbname [show detail] | Application identification information for each application currently connected to the specified database. |
| Connections list | list dcs applications | Application identification information for all DCS applications currently connected to a database that is managed by the DB2 instance on the partition where snapshot is taken. |
| Database manager | get snapshot for dbm | Database manager level information, including instance-level monitor switch settings. |
| Database manager | get dbm monitor switches | Instance-level monitor switch settings. |
| Database | get snapshot for database on dbname | Database level information and counters for a database. Information is returned only if there is at least one application connected to the database. |
| Database | get snapshot for all databases | Database level information and counters for each database active on the partition. Information is returned only if there is at least one application connected to the database. |
| Database | list active databases | The number of connections to each active database. Includes databases that were started using the ACTIVATE DATABASE command, but have no connections. |
| Database | get snapshot for dcs database on dbname | Database level information and counters for a specific DCS database. Information is returned only if there is at least one application connected to the database. |
| Database | get snapshot for remote database on dbname | Database level information and counters for a specific federated system database. Information is returned only if there is at least one application connected to the database. |
| Database | get snapshot for all remote databases | Database level information and counters for each active federated system database on the partition. Information is returned only if there is at least one application connected to the database. |
| Application | get snapshot for application applid appl-id | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for application agentid appl-handle | Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for applications on dbname | Application level information for each application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for all applications | Application level information for each application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for dcs application applid appl-id | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for all dcs applications | Application level information for each DCS application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for dcs application agentid appl-handle | Application level information, including cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for dcs applications on dbname | Application level information for each DCS application that is connected to the database on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for remote applications on dbname | Application level information, includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Application | get snapshot for all remote applications | Application level information for each federated system application that is active on the partition. This includes cumulative counters, status information, and most recent SQL statement executed (if statement switch is set). |
| Table | get snapshot for tables on dbname | Table activity information at the database and application level for each application connected to the database. Table activity information at the table level for each table that was accessed by an application connected to the database. Requires the table switch. |
| Lock | get snapshot for locks for application applid appl-id | List of locks held by the application. Lock wait information requires the lock switch. |
| Lock | get snapshot for locks for application agentid appl-handle | List of locks held by the application. Lock wait information requires the lock switch. |
| Lock | get snapshot for locks on dbname | Lock information at the database level, and application level for each application connected to the database. Requires the lock switch. |
| Table space | get snapshot for tablespaces on dbname | Information about table space activity for a database. Requires the buffer pool switch. Also included is information on containers, quiescers, and ranges. This information is not under switch control. |
| Buffer pool | get snapshot for all bufferpools | Buffer pool activity counters. Requires the buffer pool switch. |
| Buffer pool | get snapshot for bufferpools on dbname | Buffer pool activity counters for the specified database. Requires the buffer pool switch. |
| Dynamic SQL | get snapshot for dynamic sql on dbname | Point-in-time statement information from the SQL statement cache for the database. The information can also be from a remote data source. |

## DB2 9.5 deletes previous backup images, log files and copies of load images automatically

This alleviates lots of pain for maintaining DB2 backups and log files. Prior to DB2 9.5, these tasks were usually attained by the operating systems scripts. For example,

**Taking an online backup and retaining only 5 most recent backups**

$ db2 "BACKUP DATABASE dbname ONLINE TO /bkp/db2 COMPRESS UTIL\_IMPACT\_PRIORITY 50 INCLUDE LOGS WITHOUT PROMPTING"

$ find /bkp/db2 -mtime +5 | xargs rm

You should know what you are doing with the above command before you use it as it will delete backups older than 5 days. The above is the brute force method and I personally do not like such methods even though I have used them.

A nicer way will be to allow DB2 9.5 to do the job for you.

You can do this in 2 ways.

1. Let DB2 delete old backups, logs and copy of LOAD files automatically.  
2. Let DB2 delete old backups, logs and copy of LOAD files when you want it to be done.

**Let DB2 delete old backups automatically**

To allow DB2 to delete old backups, old log files and old copies of LOAD images, you will need to turn parameter AUTO\_DEL\_REC\_OBJ to ON.

db2 connect to icmnlsdb

db2 update db cfg using AUTO\_DEL\_REC\_OBJ on

db2 terminate

For automatic deletion, you will have to decide what value parameter REC\_HIS\_RETENTN should hold. The parameter REC\_HIS\_RETENTN tells to DB2 to prune the history file after 'n' number of days. The default value set is 366 days. If I take online backups every day, I may set this value to be 2 days. If I take off-line backup every Sunday followed by the delta backups every day, I may set this value to 8 days to make sure that my history file has information for the last 8 days (7 days + 1). Please remember that history file is very important for DB2 recovery operations. If by any chance you prune history, you can always restore history file from the the backup image.

Let us assume the value of REC\_HIS\_RETENTN to be 8.

db2inst1@cm07$ db2 update db cfg for icmnlsbd using REC\_HIS\_RETENTN 8

db2inst1@cm07$ db2 force applications all

DB20000I The FORCE APPLICATION command completed successfully.

DB21024I This command is asynchronous and may not be effective immediately.

db2inst1@cm07$ db2 terminate

DB20000I The TERMINATE command completed successfully.

db2inst1@cm07$ db2 attach to db2inst1

Instance Attachment Information

Instance server = DB2/AIX64 9.5.5

Authorization ID = DB2INST1

Local instance alias = DB2INST1

db2inst1@cm07$ db2 connect to icmnlsdb

Database Connection Information

Database server = DB2/AIX64 9.5.5

SQL authorization ID = DB2INST1

Local database alias = ICMNLSDB

DB2 will start pruning history file automatically after 8 days and since parameter AUTO\_DEL\_REC\_OBJ is set to ON, the recovery objects (backup, logs and load images) will be deleted automatically.

**Let DB2 delete old backups automatically when you say it so**

If you want to control when DB2 should delete the old recovery objects, you still need to set parameter AUTO\_DEL\_REC\_OBJ to ON and DB2 will delete the old recovery objects when you use PRUNE command to delete entries from the history file.

db2 connect to icmnlsdb

db2 update db cfg using AUTO\_DEL\_REC\_OBJ ON

db2 terminate

DB2 will delete the old recovery objects when you issue PRUNE HISTORY command. For example,

To remove the entries for all restores, loads, table space backups, and full database backups taken before and including December 14, 2012 from the recovery history file, enter:

$ db2 prune history 20121214

Since you are deleting entries from history file, DB2 will also delete backups, logs and load copy images prior to Dec 14st, 2012.

This way, you are controlling the deletion of old recovery objects when you issue PRUNE HISTORY command.

The above is a much better method than old brute force method.

## Commands to generate a script for runstats and rebind

db2 connect to db user userid using password

echo db2 connect to db user userid using password > fname.bat

db2 –x "select 'db2 runstats on table 'concat tabschema concat'.' concat tabname concat 'with distribution and detailed indexes all' from syscat.tables where tabschema='schema' and type='T'">> fname.bat

echo db2 connect reset >> fname.bat

db2 connect reset

echo db2rbind db –l bind.log all /u userid /p password >> fname.bat

Change ***db*** to the name of your database and change ***userid*** and ***password*** for your system values. Change the ***schema*** name based on your system and be sure to use capital letters.

## Reorg required

The DB2 command reorgchk can be used to suggest if a reorganization of the tables and indexes is warranted. You can run the command from a script and schedule to run the script when the system usage is low. You can use reorgchk to recalculate the table statistics using the “update statistics” option but it does not give the level of control over recalculating the index statistics that the

runstats command does. Run runstats before running a reorgchk. To see whether you need to reorganize a table, use the following command from a DB2 command line window (after connecting to the database):

db2 reorgchk current statistics on table all > reorgchk.txt

Attention: Library Server relies heavily on DB2 stored procedures and precompiled access modules to perform its functions. This is why runstats is so important for maintaining the performance of a Content Manager.

statistics for all of the tables followed by statistics for all of the indexes are included. The last column in the output (REORG) is the column that indicates by the presence of one or more asterisks whether a reorganization might be necessary (for example, -\*- or --\*).

To reorganize a specific table, use the following command from a DB2 command line window:

db2 reorg table <table name>

In this example, <table name> is the specific table you want to reorganize, such as RMADMIN.RMPARTS.

To reorganize the indexes for a specific table, use the following command from a DB2 command line window:

db2 reorg indexes all for table <table name>

Again, <table name> is the specific table you want to reorganize

## Recovering db2 database to new Server from Tivoli® Storage Manager (TSM)

This cross-node recovery example shows how to set up two computers so that you can recover data from one computer to another when log archives and backups are stored on a TSM server. Cross-node recovery using the **db2adutl** command, **logarchopt1** and **vendoropt** database configuration parameters.

DB2 backup db rmdb use tsmunder TSM node **db2node**, where passwords are managed using the PASSWORDACCESS=GENERATE option. node password file: **TSM.PWD** stored under /etc/security/adsm, node configuration files: **dsm.opt** and **dsm.sys** are under /usr/tivoli/tsm/client/api/bin64.

In LMS, admsrv1 is running the AIX® operating system. The db2instance on this machine is **db2inst2**. The database is called **rmdb**. The LMS development Server is called cm07, is also running the AIX operating system, and the db2instance for resore testing is db2rins1.

Copy all TSM related configuration files from production server: admsrv1 to cm07 under the same directories.

/etc/security/adsm/TSM.PWD

/usr/tivoli/tsm/client/api/bin64/dsm.opt

/usr/tivoli/tsm/client/api/bin64/dsm.sys

On cm07:

Create new db2 user on OS:

db2rins1

db2rfen1

Home directory: /restore/db2rins1, /restore/db2rfen1

Primary group: db2grp1

Create new db2instance

root@cm07$ $DB2DIR/instance/db2icrt -a server -u db2rfen1 db2rins1

DBI1070I Program db2icrt completed successfully.

Note: You can drop Instance:

root@cm07# cd /opt/IBM/V9.5/instance

root@cm07# ./db2idrop -f db2rins1

Check the new db2 instance:

root@cm07$ db2ilist

db2inst1

db2inst2

db2rins1

Setup db2rins1 running environment same as db2inst2 on admsrv1:

root@cm07$ rcp admsrv1:/home/db2inst2/sqllib/userprofile /restore/db2rins1/sqllib/userprofile

DB2 use db2uext2 to achive/retrieve logs to/from TSM:

root@cm07$ rcp admsrv1:/home/db2inst2/sqllib/adm/db2uext2 /restore/db2rins1/sqllib/adm

# su – db2rins1; db2set db2comm=tcpip

On admsrv1: To enable cross-node recovery, you must give access to the objects associated with the admsrv1 computer to another computer and account. In this example, give access to the computer cm07 and the user db2inst1 using the following command:

$ admsrv1:/home/db2inst2/sqllib/adsm> db2adutl grant all on all for db rmdb

Successfully added permissions for all to access rmdb on all

Note: You can confirm the results of the db2adutl grant operation by issuing the following command to retrieve the current access list for the current node:

$ admsrv1:/home/db2inst2/sqllib/adsm> db2adutl queryaccess

The following information is returned:

Node Username Database Name Type

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

all all RMDB A

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

Access Types: B - backup images L - logs A - both

On cm07, Verify that there is no data associated with this user and computer on the TSM server using the following command:

cm07:/restore/db2rins1/sqllib/adsm> db2adutl query db rmdb

The following information is returned:

--- Database directory is empty ---

Warning: There are no file spaces created by DB2 on the ADSM server

Warning: No DB2 backup images found in ADSM for any alias.

Query the TSM server for a list of objects for the icmnlsdb database associated with user db2inst2 and computer cm07 using the following command:

cm07:/restore/db2rins1/sqllib/adsm> db2adutl query db rmdb nodename db2node owner db2inst2

The following information is returned:

--- Database directory is empty ---

Query for database ZAMPLE

Retrieving FULL DATABASE BACKUP information.

1 Time: 20121216151025 Oldest log: S0000000.LOG DB Partition Number: 0

Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.

No INCREMENTAL DATABASE BACKUP images found for ICMNLSDB

Retrieving DELTA DATABASE BACKUP information.

No DELTA DATABASE BACKUP images found for ICMNLSDB

Retrieving TABLESPACE BACKUP information.

No TABLESPACE BACKUP images found for ICMNLSDB E

Retrieving INCREMENTAL TABLESPACE BACKUP information.

No INCREMENTAL TABLESPACE BACKUP images found for ICMNLSDB

Retrieving DELTA TABLESPACE BACKUP information.

No DELTA TABLESPACE BACKUP images found for ICMNLSDB

Retrieving LOAD COPY information.

1 Time: 20121216151213

Retrieving LOG ARCHIVE information.

Log file: S0000000.LOG, Chain Num: 0, DB Partition Number: 0,

Taken at: 2012-12-16-15.10.38

This information matches the TSM information that was generated previously and confirms that you can restore this image onto the cm07 computer.

Restore the rmdb database from the TSM server to the cm07 computer using the following command:

# cm07:/restore/db2rins1> db2 restore db rmdb use tsm options "'-fromnode=db2node -fromowner=db2inst2'"

DB20000I The RESTORE DATABASE command completed successfully.

Note: If the rmdb database already existed on cm07, the OPTIONS parameter would be omitted, and the database configuration parameter vendoropt would be used. This configuration parameter overrides the OPTIONS parameter for a backup or restores operation.

Perform a roll-forward operation to apply the transactions recorded in the rmdb database log file when a new table was created and new data loaded. In this example, the following attempt for the roll-forward operation will fail because the roll-forward utility cannot find the log files because the user and computer information is not specified:

# cm07:/restore/db2rins1> db2 rollforward db rmdb to end of logs and stop

The command returns the following error:

SQL4970N Roll-forward recovery on database "RMDB" cannot reach the specified stop point (end-of-log or point-in-time) because of missing log file(s) on node(s) "0".

Force the roll-forward utility to look for log files associated with another computer using the proper logarchopt value. In this example, use the following command to set the logarchopt1 database configuration parameter and search for log files associated with user db2inst2 and computer admsrv1:

# cm07:/restore/db2rins1> db2 update db cfg for rmdb using logarchopt1 "'-fromnode=db2node -fromowner=db2inst2'"

Enable the roll-forward utility to use the backup and load copy images by setting the vendoropt database configuration parameter using the following command:

# cm07:/restore/db2rins1> db2 update db cfg for rmdb using VENDOROPT "'-fromnode=db2node -fromowner=db2inst2'"

You can finish the cross-node data recovery by applying the transactions recorded in the rmdb database log file using the following command:

# cm07:/restore/db2rins1> db2 rollforward db rmdb to end of logs and stop

The following information is returned:

Rollforward Status

Input database alias = rmdb

Number of nodes have returned status = 1

Node number = 0

Rollforward status = not pending

Next log file to be read =

Log files processed = S0000000.LOG - S0000000.LOG

Last committed transaction = 2012-12-16-20.10.38.000000 UTC

DB20000I The ROLLFORWARD command completed successfully.

The database rmdb on computer admsrv1 under user db2inst2 has been recovered to the same point as the database on computer cm07 under user db2rins1.

Following practices are for achive logs retrieve, db2 CANNOT rollforward due to NO logs found under it archive log directory, I have to retrieve all needed archieved logs from TSM using db2adutl utility:

$ su root

root's Password:

$ db2adutl extract logs between S0014067 and S0014068

Query for database RMDB

Retrieving LOG ARCHIVE information.

LOG ARCHIVE image:

Log file: S0014067.LOG, Chain Num: -1, DB Partition Number: 0, Taken at: 2013-06-11-4.38.09

Do you want to extract this log image (Y/N)? y

Writing to file:

./S0014067.LOG

LOG ARCHIVE image:

Log file: S0014068.LOG, Chain Num: -1, DB Partition Number: 0, Taken at: 2013-06-11-4.41.02

Do you want to extract this log image (Y/N)? y

Writing to file:

./S0014068.LOG

Query for database RMDBLB

Retrieving LOG ARCHIVE information.

No LOG ARCHIVE images found for RMDBLB

Query for database TOOLSDB

Retrieving LOG ARCHIVE information.

No LOG ARCHIVE images found for TOOLSDB

# pwd

/tsmha1/db2inst2/log

# ls -l

total 0

-rw-r--r-- 1 db2rins1 db2grp1 6440 Jun 11 10:01 RETRIEVE.LOG

-rw-r----- 1 root system 528384 Jun 11 10:13 S0014067.LOG

-rw-r----- 1 root system 12288 Jun 11 10:14 S0014068.LOG

-rw-r--r-- 1 db2rins1 db2grp1 0 Jun 10 16:08 dsierror.log

# pwd

/tsmha1/db2inst2/log

# db2 get db cfg for rmdb

SQL1013N The database alias name or database name "RMDB" could not be found.

SQLSTATE=42705

# pwd

/tsmha1/db2inst2/log

# set -o vi

# db2 get db cfg for rmdb

Database Configuration for Database rmdb

Database configuration release level = 0x0c00

Database release level = 0x0c00

Database territory = US

Database code page = 1208

Database code set = UTF-8

Database country/region code = 1

Database collating sequence = IDENTITY

Alternate collating sequence (ALT\_COLLATE) =

Number compatibility = OFF

Varchar2 compatibility = OFF

Database page size = 4096

Dynamic SQL Query management (DYN\_QUERY\_MGMT) = DISABLE

Discovery support for this database (DISCOVER\_DB) = ENABLE

Restrict access = NO

Default query optimization class (DFT\_QUERYOPT) = 2

Degree of parallelism (DFT\_DEGREE) = 1

Continue upon arithmetic exceptions (DFT\_SQLMATHWARN) = NO

Default refresh age (DFT\_REFRESH\_AGE) = 0

Default maintained table types for opt (DFT\_MTTB\_TYPES) = SYSTEM

Number of frequent values retained (NUM\_FREQVALUES) = 10

Number of quantiles retained (NUM\_QUANTILES) = 20

Decimal floating point rounding mode (DECFLT\_ROUNDING) = ROUND\_HALF\_EVEN

Backup pending = NO

Database is consistent = NO

Rollforward pending = DATABASE

Restore pending = NO

Multi-page file allocation enabled = YES

Log retain for recovery status = RECOVERY

User exit for logging status = YES

Self tuning memory (SELF\_TUNING\_MEM) = OFF

Size of database shared memory (4KB) (DATABASE\_MEMORY) = AUTOMATIC(114400)

Database memory threshold (DB\_MEM\_THRESH) = 10

Max storage for lock list (4KB) (LOCKLIST) = 1000

Percent. of lock lists per application (MAXLOCKS) = 10

Package cache size (4KB) (PCKCACHESZ) = (MAXAPPLS\*8)

Sort heap thres for shared sorts (4KB) (SHEAPTHRES\_SHR) = 20000

Sort list heap (4KB) (SORTHEAP) = 1280

Database heap (4KB) (DBHEAP) = AUTOMATIC(2560)

Catalog cache size (4KB) (CATALOGCACHE\_SZ) = (MAXAPPLS\*5)

Log buffer size (4KB) (LOGBUFSZ) = 8

Utilities heap size (4KB) (UTIL\_HEAP\_SZ) = 5000

Buffer pool size (pages) (BUFFPAGE) = 1000

SQL statement heap (4KB) (STMTHEAP) = 2048

Default application heap (4KB) (APPLHEAPSZ) = AUTOMATIC(1024)

Application Memory Size (4KB) (APPL\_MEMORY) = AUTOMATIC(40000)

Statistics heap size (4KB) (STAT\_HEAP\_SZ) = AUTOMATIC(4384)

Interval for checking deadlock (ms) (DLCHKTIME) = 10000

Lock timeout (sec) (LOCKTIMEOUT) = -1

Changed pages threshold (CHNGPGS\_THRESH) = 60

Number of asynchronous page cleaners (NUM\_IOCLEANERS) = 1

Number of I/O servers (NUM\_IOSERVERS) = 3

Index sort flag (INDEXSORT) = YES

Sequential detect flag (SEQDETECT) = YES

Default prefetch size (pages) (DFT\_PREFETCH\_SZ) = AUTOMATIC

Track modified pages (TRACKMOD) = OFF

Default number of containers = 1

Default tablespace extentsize (pages) (DFT\_EXTENT\_SZ) = 32

Max number of active applications (MAXAPPLS) = 512

Average number of active applications (AVG\_APPLS) = 1

Max DB files open per application (MAXFILOP) = 61440

Log file size (4KB) (LOGFILSIZ) = 10000

Number of primary log files (LOGPRIMARY) = 50

Number of secondary log files (LOGSECOND) = 100

Changed path to log files (NEWLOGPATH) =

Path to log files = /restore/db2rins1/db2rins1/NODE0000/SQL00001/SQLOGDIR/

Overflow log path (OVERFLOWLOGPATH) =

Mirror log path (MIRRORLOGPATH) =

First active log file = S0014067.LOG

Block log on disk full (BLK\_LOG\_DSK\_FUL) = NO

Block non logged operations (BLOCKNONLOGGED) = NO

Percent max primary log space by transaction (MAX\_LOG) = 0

Num. of active log files for 1 active UOW(NUM\_LOG\_SPAN) = 0

Group commit count (MINCOMMIT) = 1

Percent log file reclaimed before soft chckpt (SOFTMAX) = 100

Log retain for recovery enabled (LOGRETAIN) = RECOVERY

User exit for logging enabled (USEREXIT) = ON

HADR database role = STANDARD

HADR local host name (HADR\_LOCAL\_HOST) =

HADR local service name (HADR\_LOCAL\_SVC) =

HADR remote host name (HADR\_REMOTE\_HOST) =

HADR remote service name (HADR\_REMOTE\_SVC) =

HADR instance name of remote server (HADR\_REMOTE\_INST) =

HADR timeout value (HADR\_TIMEOUT) = 120

HADR log write synchronization mode (HADR\_SYNCMODE) = NEARSYNC

HADR peer window duration (seconds) (HADR\_PEER\_WINDOW) = 0

First log archive method (LOGARCHMETH1) = USEREXIT

Options for logarchmeth1 (LOGARCHOPT1) = -fromnode=db2node -fromowner=db2inst2

Second log archive method (LOGARCHMETH2) = OFF

Options for logarchmeth2 (LOGARCHOPT2) =

Failover log archive path (FAILARCHPATH) =

Number of log archive retries on error (NUMARCHRETRY) = 5

Log archive retry Delay (secs) (ARCHRETRYDELAY) = 20

Vendor options (VENDOROPT) = -fromnode=db2node -fromowner=db2inst2

Auto restart enabled (AUTORESTART) = ON

Index re-creation time and redo index build (INDEXREC) = SYSTEM (RESTART)

Log pages during index build (LOGINDEXBUILD) = OFF

Default number of loadrec sessions (DFT\_LOADREC\_SES) = 1

Number of database backups to retain (NUM\_DB\_BACKUPS) = 10

Recovery history retention (days) (REC\_HIS\_RETENTN) = 366

Auto deletion of recovery objects (AUTO\_DEL\_REC\_OBJ) = ON

TSM management class (TSM\_MGMTCLASS) =

TSM node name (TSM\_NODENAME) = db2node

TSM owner (TSM\_OWNER) =

TSM password (TSM\_PASSWORD) =

Automatic maintenance (AUTO\_MAINT) = OFF

Automatic database backup (AUTO\_DB\_BACKUP) = OFF

Automatic table maintenance (AUTO\_TBL\_MAINT) = OFF

Automatic runstats (AUTO\_RUNSTATS) = OFF

Automatic statement statistics (AUTO\_STMT\_STATS) = OFF

Automatic statistics profiling (AUTO\_STATS\_PROF) = OFF

Automatic profile updates (AUTO\_PROF\_UPD) = OFF

Automatic reorganization (AUTO\_REORG) = OFF

Enable XML Character operations (ENABLE\_XMLCHAR) = YES

WLM Collection Interval (minutes) (WLM\_COLLECT\_INT) = 0

# cd /restore/db2rins1/db2rins1/NODE0000/SQL00001/SQLOGDIR/

# ls -l

total 0

# env

\_=/usr/bin/env

LANG=en\_US

LOGIN=root

PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:/restore/db2rins1/bin:/usr/bin/X11:/sbin:.:/restore/db2rins1/sqllib/bin:/restore/db2rins1/sqllib/adm:/restore/db2rins1/sqllib/misc:/restore/db2rins1/sqllib/db2tss/bin

LC\_\_FASTMSG=true

CLASSPATH=/restore/db2rins1/sqllib/java/db2java.zip:/restore/db2rins1/sqllib/java/db2jcc.jar:/restore/db2rins1/sqllib/function:/restore/db2rins1/sqllib/java/db2jcc\_license\_cu.jar:.

LOGNAME=db2rins1

MAIL=/usr/spool/mail/db2rins1

DSMI\_CONFIG=/usr/tivoli/tsm/client/api/bin64/dsm.opt

LOCPATH=/usr/lib/nls/loc

USER=db2rins1

AUTHSTATE=files

DEFAULT\_BROWSER=/usr/bin/mozilla

SHELL=/usr/bin/ksh

ODMDIR=/etc/objrepos

DSMI\_LOG=/tsmha1/db2inst2/log

PAM\_SERVICE=su

IBMCMROOT=/opt/IBM/db2cmv8

HOME=/restore/db2rins1

DB2INSTANCE=db2rins1

TERM=xterm

MAILMSG=[YOU HAVE NEW MAIL]

PWD=/restore/db2rins1/db2rins1/NODE0000/SQL00001/SQLOGDIR

TZ=EST5EDT,M3.2.5,M11.1.0

ICMDLL=/home/db2fenc1

DSMI\_DIR=/usr/tivoli/tsm/client/api/bin64

A\_\_z=! LOGNAME

# cp /tsmha1/db2inst2/log/\*.LOG .

# ls -l

total 0

-rw-r--r-- 1 root system 6440 Jun 11 10:20 RETRIEVE.LOG

-rw-r----- 1 root system 528384 Jun 11 10:20 S0014067.LOG

-rw-r----- 1 root system 12288 Jun 11 10:20 S0014068.LOG

# chown db2rins1:db2grp1 \*.LOG

# ls -l

total 0

-rw-r--r-- 1 db2rins1 db2grp1 6440 Jun 11 10:20 RETRIEVE.LOG

-rw-r----- 1 db2rins1 db2grp1 528384 Jun 11 10:20 S0014067.LOG

-rw-r----- 1 db2rins1 db2grp1 12288 Jun 11 10:20 S0014068.LOG

# exit

$ db2 rollforward db rmdb to end of logs and stop

Rollforward Status

Input database alias = rmdb

Number of nodes have returned status = 1

Node number = 0

Rollforward status = not pending

Next log file to be read =

Log files processed = S0014067.LOG - S0014068.LOG

Last committed transaction = 2013-06-11-08.40.07.000000 UTC

DB20000I The ROLLFORWARD command completed successfully.

$ db2 connect to rmdb

Database Connection Information

Database server = DB2/AIX64 9.5.5

SQL authorization ID = DB2RINS1

Local database alias = RMDB

$ pwd

/tsmha1/db2inst2/log

$ ls

ARCHIVE.LOG RETRIEVE.LOG S0014067.LOG S0014068.LOG dsierror.log

$ cd

$ pwd

/restore/db2rins1

$ ls

db2rins1 sqllib

$ ls -l

total 0

drwxrwxr-x 3 db2rins1 db2grp1 256 Jun 10 16:08 db2rins1

drwxrwsr-t 20 db2rins1 db2grp1 4096 Jun 10 15:33 sqllib

$ du -k db2rins1

80540 db2rins1/NODE0000/SQL00001/SQLOGDIR

127280 db2rins1/NODE0000/SQL00001/SQLT0000.0

4 db2rins1/NODE0000/SQL00001/SQLT0001.0

9988 db2rins1/NODE0000/SQL00001/SQLT0002.0

416 db2rins1/NODE0000/SQL00001/SYSTOOLSPACE

4 db2rins1/NODE0000/SQL00001/SYSTOOLSTMPSPACE

4 db2rins1/NODE0000/SQL00001/TEMPSPACE2

264 db2rins1/NODE0000/SQL00001/blobs

0 db2rins1/NODE0000/SQL00001/db2event/db2detaildeadlock

0 db2rins1/NODE0000/SQL00001/db2event

3961016 db2rins1/NODE0000/SQL00001/objects

164 db2rins1/NODE0000/SQL00001/objparts

452 db2rins1/NODE0000/SQL00001/replicas

416 db2rins1/NODE0000/SQL00001/sms

184 db2rins1/NODE0000/SQL00001/tracking

452 db2rins1/NODE0000/SQL00001/validateitm

4196444 db2rins1/NODE0000/SQL00001

12 db2rins1/NODE0000/sqldbdir

4196456 db2rins1/NODE0000

4196456 db2rins1

$

**Example 2: Passwords are user-managed (PASSWORDACCESS option set to PROMPT)**

This cross-node recovery example shows how to set up two computers so that you can recover data from one computer to another when log archives and backups are stored on a TSM server and where passwords are managed by the users. In these environments, extra information is required, specifically the TSM nodename and password of the computer where the objects were created.

Update the client dsm.sys file by adding the following line because computer admsrv1 is the name of the source computer

NODENAME db2node

Note: On Windows operating systems, this file is called the dsm.opt file. When you update this file, you must reboot your system for the changes to take effect.

Query the TSM server for the list of objects associated with user db2inst1 and computer admsrv1 using the following command:

admsrv2:/home/db2inst1/sqllib/adsm> db2adutl query db rmdb nodename db2node owner db2inst2 password \*\*\*\*\*\*\*

The following information is returned:

Query for database RMDB

Retrieving FULL DATABASE BACKUP information.

1 Time: 2012126151025 Oldest log: S0000000.LOG DB Partition Number: 0

Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.

No INCREMENTAL DATABASE BACKUP images found for ICMNLSDB

Retrieving DELTA DATABASE BACKUP information.

No DELTA DATABASE BACKUP images found for ICMNLSDB

Retrieving TABLESPACE BACKUP information.

No TABLESPACE BACKUP images found for ICMNLSDB

Retrieving INCREMENTAL TABLESPACE BACKUP information.

No INCREMENTAL TABLESPACE BACKUP images found for ICMNLSDB

Retrieving DELTA TABLESPACE BACKUP information.

No DELTA TABLESPACE BACKUP images found for ICMNLSDB

Retrieving LOAD COPY information.

1 Time: 20090216151213

Retrieving LOG ARCHIVE information.

Log file: S0000000.LOG, Chain Num: 0, DB Partition Number: 0,

Taken at: 2012-12-16-15.10.38

If the rmdb database does not exist on computer cm07, perform the following steps:

Create an empty rmdb database using the following command:

cm07:/restore/db2rins1> db2 create db rmdb

Update the database configuration parameter tsm\_nodename using the following command:

cm07:/restore/db2rins1> db2 update db cfg for icmnlsdb using tsm\_nodename db2node

Update the database configuration parameter tsm\_password using the following command:

cm07:/restore/db2rins1> db2 update db cfg for rmdb using tsm\_password \*\*\*\*\*\*\*\*

Attempt to restore the icmnlsdb database using the following command:

cm07:/restore/db2rins1> db2 restore db icmnlsdb use tsm options "'-fromnode=db2node -fromowner=db2inst21'" without prompting

The restore operation completes successfully, but a warning is issued:

SQL2540W Restore is successful, however a warning "2523" was encountered during Database Restore while processing in No Interrupt mode.

Perform a roll-forward operation using the following command:

cm07:/restore/db2rins1> db2 rollforward db rmdb to end of logs and stop

In this example, because the restore operation replaced the database configuration file, the roll-forward utility cannot find the correct log files and the following error message is returned:

SQL1268N Roll-forward recovery stopped due to error "-2112880618" while retrieving log file "S0000000.LOG" for database "ICMNLSDB" on node "0".

Reset the following TSM database configuration values to the correct values:

Set the tsm\_nodename configuration parameter using the following command:

cm07:/restore/db2rins1> db2 update db cfg for rmdb using tsm\_nodename db2node

Set the tsm\_password database configuration parameter using the following command:

cm07:/restore/db2rins1> db2 update db cfg for rmdb using tsm\_password \*\*\*\*\*\*\*

Set the logarchopt1 database configuration parameter so that the roll-forward utility can find the correct log files using the following command:

cm07:/restore/db2rins1> db2 update db cfg for rmdb using logarchopt1 "'-fromnode=db2node -fromowner=db2inst2'"

Set the vendoropt database configuration parameter so that the load recovery file can also be used during the roll-forward operation using the following command:

cm07:/restore/db2rins1> db2 update db cfg for rmdb using VENDOROPT "'-fromnode=db2node -fromowner=db2inst2'"

You can finish the cross-node recovery by performing the roll-forward operation using the following command:

cm07:/restore/db2rins1> db2 rollforward db rmdb to end of logs and stop

The following information is returned:

Rollforward Status

Input database alias = rmdb

Number of nodes have returned status = 1

Node number = 0

Rollforward status = not pending

Next log file to be read =

Log files processed = S0000000.LOG - S0000000.LOG

Last committed transaction = 2012-12-16-20.10.38.000000 UTC

DB20000I The ROLLFORWARD command completed successfully.

The database icmnlsdb on computer dps under user regress9 has been recovered to the same point as the database on computerbar under user roecken

## Backup icmstitemevents data before 2010-01-01 00:00:00.000000 for table purge

db2inst1@admsrv1$ *db2 attach to db2inst1*

db2inst1@admsrv1$ *db2 connect to icmnlsdb*

db2inst1@admsrv1$ *db2 "export to icmstitemevents.20110101.ixf of ixf select \* from icmadmin.icmstitemevents where created < '2011-01-01 00:00:00.000000'"*

db2inst1@cm07$ *db2 "import from /restore/db2rins1/icmstitemevents.20100101.ixf of ixf commitcount automatic insert into icmadmin.icmstitemevents"*

SQL3150N The H record in the PC/IXF file has product "DB2 02.00", date

"20130614", and time "134540".

SQL3153N The T record in the PC/IXF file has name

"icmstitemevents.20100101.ixf", qualifier "", and source " ".

SQL3109N The utility is beginning to load data from file

"/restore/db2rins1/icmstitemevents.20100101.ixf".

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "157207".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "310580".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "466658".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "620706".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "778993".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "933274".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1094883".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1252266".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1408475".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1565520".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1726870".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "1883563".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2039398".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2204844".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2381190".

SQL3222W ...COMMIT of any database changes was successful.

SQL3186W Data was not loaded into the database, because the log was full or

the lock space was exhausted. SQLCODE "-964" was returned. A commit will be

attempted and the operation will continue if the commit is successful.

SQL0964C The transaction log for the database is full. SQLSTATE=57011

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2560074".

SQL3222W ...COMMIT of any database changes was successful.

SQL3110N The utility has completed processing. "2679843" rows were read from

the input file.

SQL3221W ...Begin COMMIT WORK. Input Record Count = "2679843".

SQL3222W ...COMMIT of any database changes was successful.

SQL3149N "2679843" rows were processed from the input file. "2679843" rows

were successfully inserted into the table. "0" rows were rejected.

Number of rows read = 2679843

Number of rows skipped = 0

Number of rows inserted = 2679843

Number of rows updated = 0

Number of rows rejected = 0

Number of rows committed = 2679843

db2inst1@cm07$ db2 list active databases

Active Databases

Database name = ICMNLSDB

Applications connected currently = 2

Database path = /home/db2inst1/db2inst1/NODE0000/SQL00002/

db2inst1@cm07$ db2 force applications all

DB20000I The FORCE APPLICATION command completed successfully.

DB21024I This command is asynchronous and may not be effective immediately.

db2inst1@cm07$ db2 backup db icmnlsdb to /home/db2inst1/dbbkup

Backup successful. The timestamp for this backup image is : 20130614151855

CONNECT TO ICMNLSDB;

QUIESCE DATABASE IMMEDIATE FORCE CONNECTIONS;

UNQUIESCE DATABASE;

CONNECT RESET;

UPDATE DB CFG FOR ICMNLSDB USING logarchmeth1 "DISK:/db2lslogging" logprimary 10 logsecond 20 logfilsiz 1000;

BACKUP DATABASE ICMNLSDB TO "C:\Document" WITH 2 BUFFERS BUFFER 1024 PARALLELISM 1 WITHOUT PROMPTING;

Start Collect content of scripts listed in file SCRIPTS

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

# Script /admsrv/drmgr/aix/db2delinst1.ksh

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

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

# Script /home/db2inst1/snapshot/Database\_key\_snapshot\_icmnlsdb.ksh

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

snap\_time=`date +%H%M`

. ~/.profile

db2 connect to icmnlsdb

db2 get snapshot for applications on icmnlsdb > /home/lchen/snapshot/icmnls\_key\_snapshot.$snap\_time

sleep 1

db2 get snapshot for locks on icmnlsdb >> /home/lchen/snapshot/icmnls\_key\_snapshot.$snap\_time

sleep 1

grep Lock-wait /home/lchen/snapshot/icmnls\_key\_snapshot.$snap\_time > /home/lchen/snapshot/Lock\_wait\_start

if [ -s /home/lchen/snapshot/Lock\_wait\_start ]; then

mail -s " LMS Lock wait start now " lchen@yahoo.com < /home/lchen/snapshot/icmnls\_key\_snapshot.$snap\_time

fi

rm -f /home/lchen/snapshot/Lock\_wait\_start

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

# Script /admsrv/db2/icmnlsdb/db2stats/ls\_stats.ksh

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

echo " == Run Statistics for LS started at @ `date` ==\n"

. /home/db2inst1/sqllib/db2profile

db2 connect to icmnlsdb user icmadmin using cmls83

db2 runstats on table ICMADMIN.CARRIERCODENB with distribution and detailed indexes all

db2 runstats on table ICMADMIN.CLIENTADMNB with distribution and detailed indexes all

db2 runstats on table ICMADMIN.ICMUT04228001 with distribution and detailed indexes all

db2 runstats on table ICMADMIN.ICMUT04229001 with distribution and detailed indexes all

db2 connect reset

echo "\n== Run Statistics for LS completed @ `date`==\n\n"

echo "\*\* Run binding for LS started @ `date` \*\* \n"

db2rbind icmnlsdb -l /admsrv/db2/icmnlsdb/db2stats/ls\_stats.log all -u icmadmin -p cmls83

echo "\*\* Run binding for LS finished @ `date` \*\* \n"

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

# Script /admsrv/db2/icmnlsdb/db2mgmt/reorgchk/ls\_reorgchk.ksh

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

echo "== Reorgchk for LS started at @ `date` == \n"

. /home/db2inst1/sqllib/db2profile

cd /admsrv/db2/icmnlsdb/db2mgmt/reorgchk

db2 -v "connect to icmnlsdb user db2inst1 using yahoo"

db2 -v "reorgchk current statistics on table all" > ./ls\_reorgchk.log.`date +%Y%m%d` 2>&1

db2 -v "connect reset"

echo "== Reorgchk for LS completed at @ `date` == \n"

exit 0

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

# Script /admsrv/db2/icmnlsdb/db2mgmt/reorgtab/ls\_reorg.ksh

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

root@admsrv1:/admsrv/db2/icmnlsdb/db2mgmt/reorgtab # cat lsreorg.ksh

#!/bin/ksh

echo "Reorg is started on ICMNLSDB database @ `date`"

. /home/db2inst1/sqllib/db2profile

db2 -v "connect to icmnlsdb user icmadmin using cmls83"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMSTRESOURCEMGR CLEANUP ONLY ALL"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMSTUSERGROUPS"

db2 -v "REORG TABLE ICMADMIN.ICMSTCOMPILEDACL"

db2 -v "REORG TABLE ICMADMIN.ICMSTCOMPILEDPERM"

PAGES"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMUT04159001 CLEANUP ONLY ALL"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMUT04160001 CLEANUP ONLY ALL"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMUT04163001 CLEANUP ONLY ALL"

db2 -v "REORG INDEXES ALL FOR TABLE ICMADMIN.ICMUT04164001 CLEANUP ONLY ALL"

db2 -v "REORG TABLE ICMADMIN.CARRIERCODENB INDEX ICMADMIN.CC1329165922993"

db2 -v "connect reset"

echo "Reorg is completed on ICMNLSDB database @ `date`"

exit 0

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

# Script /home/db2inst1/export/broker.export.ksh

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

#!/usr/bin/ksh

. /home/db2inst1/sqllib/db2profile

cd /home/db2inst1/export/today

cp -p /home/db2inst1/export/today-1/\* /home/db2inst1/export/today-2

cp -p /home/db2inst1/export/today/\* /home/db2inst1/export/today-1

echo "Start to export key tables .... @ `date`\n\n"

db2 connect to icmnlsdb

#-- db2 "export to myfile.ixf of ixf messages msgs.txt select \* from staff "

db2 "export to CLIENTADM.ixf of ixf select \* from ICMADMIN.CLIENTADM"

db2 "export to XREFCLIENTBROKERADM.ixf of ixf select \* from ICMADMIN.XREFCLIENTBROKERADM"

db2 "export to BROKERADMSB.ixf of ixf select \* from ICMADMIN.BROKERADMSB"

db2 "export to CARRIERCODE.ixf of ixf select \* from ICMADMIN.CARRIERCODE"

db2 "export to CLIENTADMSB.ixf of ixf select \* from ICMADMIN.CLIENTADMSB"

db2 terminate

echo "\n\nExport key tables is completed @ `date`"

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

# Script /home/db2inst1/spe\_bkup.ksh

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

#!/usr/bin/ksh

. $HOME/sqllib/db2profile

cd $HOME/dbbackup

db2 backup database icmnlsdb online > spe\_bkup.out 2>&1

mail -s "special icmnlsdb backup is completed @ `date`" dguo@yahoo.com < spe\_bkup.out

exit

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

# Script /admsrv/drmgr/aix/db2delinst2.ksh

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

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

# Script /home/db2inst2/snapshot/Database\_key\_snapshot\_rmdb.ksh

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

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

# Script /admsrv/db2/rmdb/db2stats/rm\_stats.ksh

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

#!/usr/bin/ksh

echo " == Run Statistics for RM started at @ `date` ==\n"

. /home/db2inst2/sqllib/db2profile

db2 connect to rmdblb user rmadmin using yahoo

db2 runstats on table RMADMIN.RMVALREPORT with distribution and detailed indexes all

db2 runstats on table RMADMIN.RMVERSION with distribution and detailed indexes all

db2 runstats on table RMADMIN.RMVOLUMES with distribution and detailed indexes all

db2 connect reset

echo "\n== Run Statistics for RM completed @ `date`==\n\n"

echo "\*\* Run binding for RM started @ `date` \*\* \n"

db2rbind rmdblb -l /admsrv/db2/rmdb/db2stats/rm\_stats.log all -u rmadmin -p yahoo

echo "\n\*\* Run binding for RM finished @ `date` \*\* \n"

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

# Script /admsrv/db2/rmdb/db2mgmt/reorgchk/rm\_reorgchk.ksh

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

echo " == Reorgchk for RM started @ `date` == \n"

. /home/db2inst2/sqllib/db2profile

cd /admsrv/db2/rmdb/db2mgmt/reorgchk

db2 -v "connect to rmdblb user db2inst2 using yahoo"

db2 -v "reorgchk current statistics on table all" > ./rm\_reorgchk.log.`date +%Y%m%d` 2>&1

db2 -v "connect reset"

echo "\n == Reorgchk for RM completed @ `date` == \n"

exit 0

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

# Script /home/db2inst2/spe\_bkup.ksh

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

#!/usr/bin/ksh

cd $HOME/dbbackup

. $HOME/sqllib/db2profile

db2 backup database rmdb online > spe\_bkup.out 2>&1

mail -s "special rmdb backup is completed @ `date`" chenliru@yahoo.com < spe\_bkup.out

exit

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

# Script /home/lchen/scripts/errMail

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

#!/bin/ksh

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

#

#

# Yahoo AIX environment Monitor

#

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

set -v

set -x

# errpt -a | mail -s " System `hostname` Error Messages " lchen@yahoo.com

# mail -s " System $1 CANNOT be connected ! " lchen@yahoo.com < /dev/null

errpt -a > errLog

if [ -s errLog ]; then

mail -s " System `hostname` Error Messages " lchen@yahoo.com < errLog

fi

rm -f errLog

exit 0

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

# Script /home/lchen/scripts/dsmMail

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

#!/bin/ksh

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

#

#

# Yahoo TSM environment Monitor

#

#

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

set -v

set -x

dsmadmc -id=tsmtape -pass=tsm567 <<EOF

q eve \* \* > dsm.out

q eve \* t=a >> dsm.out

q stg >> dsm.out

q db >> dsm.out

q log >> dsm.out

q libv >> dsm.out

q actlog begint=-24 search=anr????e >> dsm.out

q actlog begint=-24 >> dsm.out

EOF

mail -s " System `hostname` TSM Messages " lchen@yahoo.com < dsm.out

rm -f dsm.out

exit 0

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

# Script /home/lchen/scripts/perfMail

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

#!/bin/ksh

#

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

# A system administrator should intuitively know when the system has gone into the "red zone". This is usually

# accompanied by their phone ringing as users call to complain about system performance. But there are more

# empirical measurements that an administrator can look for to show that the system is in imminent danger.

#

# 1. Average processor utilization exceeds 80%.

# 2. Network utilization exceeds 50%

# 3. Available real memory starts Pages In and Pages Out,

# Any substantive paging activity is occurring

# 4. Disk activity exceeds 60% (this is cumulative activity, or the ?tm acct? column from iostat).

#

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

set -v

set -x

DATE=`date +%Y%m%d%H%M`

cd /home/lchen/scripts/perf

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

# Program : cpuuse

# Purpose : Script to use sar and find out CPU usage on a system.

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

. ./cpuuse 2 10 > cpuusage

average\_cpu\_idle=$( grep Average cpuusage | awk ' {print $2} ' )

if [ $average\_cpu\_idle -lt 10 ]; then

mail -s " System `hostname` CPUs are Busy ! " lchen@yahoo.com < cpuusage

fi

rm -f cpuusage

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

# end: cpuuse

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

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

# Program : diskuse

# Purpose : Script to use iostat and find out disk usage on a system.

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

. ./diskuse 2 10 > diskusage

set -A average\_disk\_usage $( grep Average diskusage | awk ' {print $7} ' )

for item in ${average\_disk\_usage[@]}

do

if [ $item -gt 80 ]; then

mail -s " System `hostname` DISKs are Busy ! " lchen@yahoo.com < diskusage

fi

done

rm -f diskusage

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

# end: diskuse

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

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

# Program : memuse

# Purpose : Script to use vmstat and find out free memory on a system.

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

. ./memuse 2 10 > memusage

pi\_po\_usage=$( grep Average memusage | awk ' {print $2} ' )

if [ $pi\_po\_usage -gt 50 ]; then

mail -s " System `hostname` Memorys are Busy ! " lchen@yahoo.com < memusage

fi

rm -f memusage

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

# end: memuse

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

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

# Program : netuse

# Purpose : Script to use netstat to find network traffic

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

. ./netuse 2 10 > netusage

set -A average\_net\_usage $( grep Average netusage | awk ' {print $4} ' )

for item in ${average\_net\_usage[@]}

do

if [ $item -gt 62500 ]; then

mail -s " System `hostname` NETWORKs are Busy ! " lchen@yahoo.com < netusage

fi

done

rm -f netusage

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

# end: netuse

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

exit 0

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

# Script /home/ptang/admin/get\_drpplan.ksh

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

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

cd /admsrv/drmgr/drp

Drpplan=`find . -name "plan.\*" -mtime 1`

mail -s "$send\_date: Daily DRP plan file for AdminServ" aixsupport < $Drpplan

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

print "$send\_date: DRP plan has been sent out!"

fi

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

# Script /admsrv/local/apps/rns/stopprocessWB1.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessWB1" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessWB1 daemon is NOT running !\n"

else

echo "\ProcessWB1 daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessWB2.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessWB2" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessWB2 daemon is NOT running !\n"

else

echo "\ProcessWB2 daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startprocessWB1cron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessWB1.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessWB1.log ProcessWB1.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessWB1 >> /admsrv/local/apps/rns/logs/ProcessWB1.log 2>&1 &

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

# Script /admsrv/local/apps/rns/startprocessWB2cron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessWB2.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessWB2.log ProcessWB2.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessWB2 >> /admsrv/local/apps/rns/logs/ProcessWB2.log 2>&1 &

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

# Script /admsrv/local/apps/rns/stopprocessRNSFiles.sh

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

#!/usr/bin/ksh

print "COUNT=0" > /admsrv/local/apps/rns/rnsdowncounter.ini

PID=`ps -ef | grep "ProcessRNSFiles" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessRNFiles daemon is NOT running !\n"

else

echo "\ProcessRNFiles daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startprocessRNSFilescron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l Processrns.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv Processrns.log Processrns.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessRNSFiles >> /admsrv/local/apps/rns/logs/Processrns.log 2>&1 &

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

# Script /admsrv/local/apps/rns/stopprobillvalidate.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProbillandPortValidationProcess" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProbillandPortValidationProcess daemon is NOT running !\n"

else

echo "\ProbillandPortValdationProcess daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startprobillvalidatecron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l probillvalidate.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv probillvalidate.log probillvalidate.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProbillandPortValidationProcess >> /admsrv/local/apps/rns/logs/probillvalidate.log 2>&1 &

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

# Script /admsrv/local/apps/rns/stopprocessWBSB1.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessWBSB1" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessWBSB1 daemon is NOT running !\n"

else

echo "\ProcessWBSB1 daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessWBSB2.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessWBSB2" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessWBSB2 daemon is NOT running !\n"

else

echo "\ProcessWBSB2 daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessShipXMLtoSB.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessShipXMLtoSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ShipXMLtoSB daemon is NOT running !\n"

else

echo "\ShipXMLtoSB daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessFTPtoSB.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessFTPtoSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The FTPtoSB daemon is NOT running !\n"

else

echo "\FTPtoSB daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessStatUpdSB.sh

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

#!/usr/bin/ksh

print "COUNT=0" > /admsrv/local/apps/rns/sbStatdowncounter.ini

PID=`ps -ef | grep "ProcessStatUpdSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The StatUpdSB daemon is NOT running !\n"

else

echo "\StatUpdSB daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessWBNB.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessWBNB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The ProcessWBNB daemon is NOT running !\n"

else

echo "\ProcessWBNB daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startprocessWBSB1cron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

#Check log size;

cd /admsrv/local/apps/rns/logs

size=`ls -l ProcessWBSB1.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessWBSB1.log ProcessWBSB1.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessWBSB1 >> /admsrv/local/apps/rns/logs/ProcessWBSB1.log 2>&1 &

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

# Script /admsrv/local/apps/rns/startprocessWBSB2cron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

#Check log size;

cd /admsrv/local/apps/rns/logs

size=`ls -l ProcessWBSB2.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessWBSB2.log ProcessWBSB2.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessWBSB2 >> /admsrv/local/apps/rns/logs/ProcessWBSB2.log 2>&1 &

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

# Script /admsrv/local/apps/rns/startprocessShipXMLtoSBcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessShipXMLtoSB.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessShipXMLtoSB.log ProcessShipXMLtoSB.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessShipXMLtoSB >> /admsrv/local/apps/rns/logs/ProcessShipXMLtoSB.log 2>1 &

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

# Script /admsrv/local/apps/rns/startprocessFTPtoSBcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessFTPtoSB.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessFTPtoSB.log ProcessFTPtoSB.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessFTPtoSB >> /admsrv/local/apps/rns/logs/ProcessFTPtoSB.log 2>1 &

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

# Script /admsrv/local/apps/rns/startprocessStatUpdSBcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessStatUpdSB.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessStatUpdSB.log ProcessStatUpdSB.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessStatUpdSB >> /admsrv/local/apps/rns/logs/ProcessStatUpdSB.log 2>1 &

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

# Script /admsrv/local/apps/rns/startprocessWBNBcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns

#Check log size;

cd /admsrv/local/apps/rns/logs

size=`ls -l ProcessWBNB.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessWBNB.log ProcessWBNB.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessWBNB >> /admsrv/local/apps/rns/logs/ProcessWBNB.log 2>&1 &

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

# Script /admsrv/local/apps/rns/stopprocessFedExXMLOutput.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessFedExXMLOutput" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The FedExXMLOutput daemon is NOT running !\n"

else

echo "\FedExXMLOutput daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/stopprocessFedExFTPOutput.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "ProcessFedExFTPOutput" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\nERROR: The FedExFTPOutput daemon is NOT running !\n"

else

echo "\FedExFTPOutput daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startprocessFedExXMLOutputcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessFedExXMLOutput.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessFedExXMLOutput.log ProcessFedExXMLOutput.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessFedExXMLOutput >> /admsrv/local/apps/rns/logs/ProcessFedExXMLOutput.log 2>&1 &

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

# Script /admsrv/local/apps/rns/startprocessFedExFTPOutputcron.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ProcessFedExFTPOutput.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ProcessFedExFTPOutput.log ProcessFedExFTPOutput.log.${time\_stamp}

fi

cd /admsrv/local/apps/rns

nohup java ProcessFedExFTPOutput >> /admsrv/local/apps/rns/logs/ProcessFedExFTPOutput.log 2>&1 &

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

# Script /admsrv/local/apps/rns/stopshadowRouter.sh

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

#!/usr/bin/ksh

PID=`ps -ef | grep "java ShadowRouter" | grep -v "grep" | awk '{print $2}'`

#if [ ${#PID} -eq 0 ]

if [ $? -eq 1 ]

then

echo "\nERROR: The ShadowRouter daemon is NOT running !\n"

else

echo "\ShadowRouter daemon is running...killing PID: ${PID}\n"

kill ${PID}

fi

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

# Script /admsrv/local/apps/rns/startshadowRouter.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/rns/logs

#Check log size;

size=`ls -l ShadowRouter.log|awk '{print $5}'`

if [[ $size -gt 20000000 ]]

then

time\_stamp=`date +%Y%m%d`

mv ShadowRouter.log ShadowRouter.log.${time\_stamp}

fi

echo

date

echo "Change to working directory ...... \n"

cd /admsrv/local/apps/rns

echo "Start ShadowRouter Daemon ....\n"

nohup java ShadowRouter >> /admsrv/local/apps/rns/logs/ShadowRouter.log &

sleep 5

PID=`ps -ef | grep "java ShadowRouter" | grep -v "grep" | awk '{print $2}'`

if [ $? -eq 1 ]

then

echo "\n \*\* ERROR: The ShadowRouter daemon has NOT been started ! \*\*\n"

else

echo "Daemon has been started successfully ....\n"

fi

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

# Script /admsrv/local/apps/loisapps/lois\_import.sh

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

cd /admsrv/local/apps/loisapps

. ./lois\_profile

echo >> lois\_import.log

echo "sYear=`date`" >> lois\_import.log

nohup java com/yahoo/lois/ImportProcess >> lois\_import.log 2>&1

YEAR=`date +%Y`

MON=`date +%m`

DAY=`date +%d`

cd /arstmp/in/LOIS\_backup/$YEAR-$MON-$DAY

ls -l|grep 'ImportBusiness'

if [[ $? -eq 0 ]]

then

echo "There is error message ....\n"

file=`ls -l|grep 'ImportBusiness'|awk '{print $9}'`

mail -s "LOIS import to LMS Process Error Report @ `date`" cstanciu@yahoo.com < $file

fi

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

# Script /admsrv/local/apps/rns/restartRNSFiles.sh

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

#!/usr/bin/ksh

. $HOME/.profile

cd /admsrv/local/apps/rns

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

PID=`ps -ef | grep "ProcessRNSFiles" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

cat rnscontrol.ini | grep "SHUTDOWN" | cut -c 10 | read SHUTDOWN

cat rnsdowncounter.ini | grep "COUNT" | cut -c 7 | read COUNT

else

exit 0

fi

if [[ $SHUTDOWN = 'Y' ]];

then

mail -s "$send\_date: The ProcessRNFiles daemon exit gracefully!!" dguo@yahoo.com < rnscontrol.ini

mail -s "$send\_date: The ProcessRNFiles daemon exit gracefully!!" bchong@yahoo.com < rnscontrol.ini

sleep 69

/admsrv/local/apps/rns/startprocessRNSFilescron.sh

print "SHUTDOWN=N" > rnscontrol.ini

print "COUNT=$COUNT" > rnsdowncounter.ini

exit 0

fi

if [[ $SHUTDOWN = 'N' && $COUNT -lt 10 ]];

then

mail -s "$send\_date: The ProcessRNFiles daemon is down!!" dguo@yahoo.com < rnscontrol.ini

mail -s "$send\_date: The ProcessRNFiles daemon is down!!" bchong@yahoo.com < rnscontrol.ini

sleep 69

/admsrv/local/apps/rns/startprocessRNSFilescron.sh

let "COUNT=COUNT+1" ;

print "SHUTDOWN=N" > rnscontrol.ini

print "COUNT=$COUNT" > rnsdowncounter.ini

else

mail -s "$send\_date: The ProcessRNFiles daemon is down 9 times!!" dguo@yahoo.com < /dev/null

#mail -s "$send\_date: The ProcessRNFiles daemon is down 9 times!!" bboyczuk@yahoo.com < /dev/null

mail -s "$send\_date: The ProcessRNFiles daemon is down 9 times!!" bchong@yahoo.com < /dev/null

fi

exit 0

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

# Script /admsrv/local/apps/rns/restartStatUpdSB.sh

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

#!/usr/bin/ksh

cd /admsrv/local/apps/rns

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

PID=`ps -ef | grep "ProcessStatUpdSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

cat sbstatupdcontrol.ini | grep "SHUTDOWN" | cut -c 10 | read SHUTDOWN

cat sbStatdowncounter.ini | grep "COUNT" | cut -c 7 | read COUNT

else

exit 0

fi

if [[ $SHUTDOWN = 'Y' ]];

then

sleep 29

/admsrv/local/apps/rns/startprocessStatUpdSBcron.sh

print "SHUTDOWN=N" > sbstatupdcontrol.ini

print "COUNT=$COUNT" > sbStatdowncounter.ini

exit 0

fi

if [[ $SHUTDOWN = 'N' && $COUNT -lt 9 ]];

then

sleep 29

mail -s "$send\_date: The ProcessStatUpdSB daemon is down!!" dguo@yahoo.com < /dev/null

/admsrv/local/apps/rns/startprocessStatUpdSBcron.sh

let "COUNT=COUNT+1" ;

print "SHUTDOWN=N" > sbstatupdcontrol.ini

print "COUNT=$COUNT" > sbStatdowncounter.ini

else

mail -s "$send\_date: The ProcessStatUpdSB daemon is down 9 times!!" dguo@yahoo.com < /dev/null

# mail -s "$send\_date: The ProcessStatUpdSB daemon is down 9 times!!" bboyczuk@yahoo.com < /dev/null

fi

exit 0

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

# Script /admsrv/local/apps/rns/restart\_WBSB.sh

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

#!/usr/bin/ksh

#set -A AlertList dguo@yahoo.com \

# tliu@yahoo.com \

# cstanciu@yahoo.com

set -A AlertList dguo@yahoo.com \

tliu@yahoo.com

#Get today's WBSB log, output will be "today\_WBSB.log";

cd /admsrv/local/apps/rns

./get\_WBSB.pl

#Search for "JVMST109" which means memory allocation failure;

ErrLog=/admsrv/local/apps/rns/logs/WBSB\_alarm.log

TodayLOG=/admsrv/local/apps/rns/logs/today\_WBSB.log

grep 'JVMST109' $TodayLOG > $ErrLog

grep 'JVMST109' $TodayLOG

if [ $? -eq 0 ]

then

mail -s "WBSB Memory Alarm on LMS @ `date` !!!" ${AlertList[\*]}<$ErrLog

date

echo "We found error message ...\n"

echo "We will try to restart the process ....\n"

#Make sure we stop the process first if exist ...

PID=`ps -ef | grep "ProcessWBSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\n ProcessWBSB daemon is NOT running !\n"

else

echo "\n ProcessWBSB daemon is still running...\n"

echo "killing process: ${PID}\n"

echo "\n We will kill the process ...\n"

kill ${PID}

fi

sleep 60

#Verify ProcessWBSB has been shutdown;

PID=`ps -ef | grep "ProcessWBSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

echo "\n ProcessWBSB has been successfully shutdown ... \n"

else

echo "\n ProcessWBSB daemon has not been shutdown ...\n"

mail -s "ProcessWBSB has not been shutdown propertly, please investigate ASAP ..." ${AlertList[\*]} < /dev/null

exit 1

fi

else

echo "No memory error been found ...\n"

#We also check if the daemon is still running ....

PID=`ps -ef | grep "ProcessWBSB" | grep -v grep | awk '{print $2}'`

if [ ${#PID} -eq 0 ]

then

cd /admsrv/local/apps/rns

echo "No Daemon found .....\n"

echo "Restart the Daemon now .....\n"

./startprocessWBSBcron.sh

fi

exit 0

fi

Dstamp=`date +%Y +%m +%d`

#Rename current WBSB log

cd /admsrv/local/apps/rns/logs

echo "Rename current WBSB log file ...\n"

mv ProcessWBSB.log ProcessWBSB.log.$Dstamp

cd /admsrv/local/apps/rns

echo "Restart the process now .....\n"

./startprocessWBSBcron.sh

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

# Script /admsrv/local/apps/rns/reStartshadowRouter.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

echo

date

PID=`ps -ef | grep "java ShadowRouter" | grep -v "grep" | awk '{print $2}'`

if [ -n "$PID" ]

then

echo "ShadowRouter Daemon is still Running ....\n"

else

echo "\n \*\* ERROR: The ShadowRouter daemon is not Running ! \*\*\n"

echo "Change to working directory ...... \n"

cd /admsrv/local/apps/rns

echo "== reStart ShadowRouter Daemon .... == \n"

nohup java ShadowRouter >> /admsrv/local/apps/rns/logs/ShadowRouter.log &

fi

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

# Script /admsrv/local/apps/ASCleanCheckedOut/startprocessCleanCheckedOut.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/ASCleanCheckedOut

nohup java ASLauncher >> ./logs/ProcessCleanCheckedOut.log 2>&1 &

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

# Script /admsrv/local/apps/rns/logClean.ksh

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

#!/usr/bin/ksh

#Archive logs older than 1 day to /admsrv/local/apps/rns/archive\_logs;

cd /admsrv/local/apps/rns

find ./ \( ! -name . -prune \) -name "\*log" -type f -mtime +1 -exec mv -f {} /admsrv/local/apps/rns/archive\_logs \;

#Purge logs older than 30 days;

cd /admsrv/local/apps/rns/archive\_logs

find ./ -name "\*.log" -mtime +30 -exec rm -f {} \;

#Purge process log which is older than 60 days;

cd /admsrv/local/apps/rns/logs

find ./ -name "\*.log.\*" -mtime +60 -exec rm -f {} \;

#Purge ReportService log which is older than 7 days;

cd /admsrv/local/apps/rns/AdminservReportsService

find ./ -name "\*.log" -mtime +7 -exec rm -f {} \;

#Archive old heapdump to heapdump directory;

cd /admsrv/local/apps/rns

find ./ \( ! -name . -prune \) -name "heapdump\*" -type f -mtime +1 -exec rm -f {} \;

find ./ \( ! -name . -prune \) -name "javacore\*" -type f -mtime +1 -exec rm -f {} \;

#Purge old files for fedex application;

cd /cmapp/fedex/processed

find ./ -name "\*.xml" -mtime +7 -exec rm -f {} \;

#Purge old files for RNSFiles application;

cd /cmapp/rns/backup

find ./ -name "\*.TXT" -mtime +7 -exec rm -f {} \;

#Purge old files for SB application;

cd /cmapp/sb/processed

find ./ -name "\*.xml" -mtime +7 -exec rm -f {} \;

#Purge old files for WATKINS application;

cd /cmapp/watkins/214/backup

find ./ -name "\*.txt" -mtime +7 -exec rm -f {} \;

cd /cmapp/watkins/212/backup

find ./ -name "\*.txt" -mtime +7 -exec rm -f {} \;

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

# Script /admsrv/local/apps/rns/AdminservReportsService/startprocessPrintEmailReports.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

export CLASSPATH=$CLASSPATH:.:/admsrv/local/apps/rns/AdminservReportsService/lib/poi38.jar

cd /admsrv/local/apps/rns/AdminservReportsService

nohup java ProcessPrintEmailReports >> /admsrv/local/apps/rns/AdminservReportsService/ProcessPrintEmailReports.log &

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

# Script /admsrv/local/apps/monthlyprofile/ProcessMonthlyProfile.sh

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

#!/usr/bin/ksh

. /home/rns/.profile

cd /admsrv/local/apps/monthlyprofile

PATH=/usr/bin:/etc:/usr/sbin:/usr/ucb:$HOME/bin:/usr/bin/X11:/sbin:/opt/IBM/db2cmv8/java/jre/bin:.

export PATH

CLASSPATH=.:/opt/IBM/db2cmv8/lib/cmb81.jar:/opt/IBM/db2cmv8/lib/cmbsdk81.jar:/opt/IBM/db2cmv8/lib/cmbview81.jar:$CLASSPATH

export CLASSPATH

# The following three lines have been added by UDB DB2.

#if [ -f /home/db2inst1/sqllib/db2profile ]; then

# . /home/db2inst1/sqllib/db2profile

#fi

# The following three lines have been added by IBM CM.

#if [ -f /opt/IBM/db2cmv8/bin/cmbenv81.sh ]; then

# . /opt/IBM/db2cmv8/bin/cmbenv81.sh

#fi

#IBMCMROOT=/opt/IBM/db2cmv8

#export IBMCMROOT

#CMCOMMON=/opt/IBM/db2cmv8/cmgmt

#export CMCOMMON

echo

date

echo "Start Process JP Morgan Monthly Profile ....\n"

#CLASSPATH=$CLASSPATH:.:JPProfile.jar

nohup java -Xms128M -Xmx1650M MonthlyProfile >> /admsrv/local/apps/monthlyprofile/log/JPMonthlyProfile.log &

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

# Script /admsrv/admin/bin/sysbkup.ksh

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

#!/bin/ksh

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

#

# Name: sysbkup.ksh

#

# Reference: n/a

#

# Description: system backup using mksysb

#

# Parameters: sysbkup.ksh <tape device>

# tape device /dev/rmt0

#

# Modification History:

#

# Date Name Description

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

# 2004-05-15 Bob Chong Original

#

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

set -v

set -x

# script library

PATH=/admsrv/admin/lib:$PATH:.

cd /admsrv/admin/log/sysbkupLog

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

# rewind the tape

tctl -f $backup\_tape rewind

if [ $? != 0 ]

then

date > $errfile

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

mail -s "Sysbkup failed (admsrv1) due to tape not ready : `date`" lchen@yahoo.com < $errfile

exit 1

fi

# 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 "Sysbkup failed (admsrv1): `date`" lchen@yahoo.com < $logfile

tctl -f $backup\_tape offline

exit 1

fi

# rewind the tape

bot.check $backup\_tape $logfile

# finally list all the files on tape

logevent $logfile "----------------------------------------------------"

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

logevent $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: $1" | tee -a $lisfile

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

mail -s "Sysbkup failed (admsrv1): `date`" lchen@yahoo.com < $lisfile

tctl -f $backup\_tape offline

exit 1

fi

logevent $logfile "----------------------------------------------------"

#rm $errfile

logevent $logfile "SYSTEM BACKUP task has been completed"

logevent $logfile "----------------------------------------------------"

mail -s "Sysbkup successful (admsrv1): `date`" lchen@yahoo.com < $logfile

mail -s "Sysbkup successful (admsrv1): `date`" computerops@yahoo.com < $logfile

sleep 3

# dismount the tape

#tctl -f $backup\_tape offline

exit 0

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

# Script /admsrv/admin/bin/alertDog.ksh

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

#!/bin/ksh

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

#

# Name: alertDog.ksh

#

# Reference: n/a

#

# Description: monitor the errpt message

#

# Parameters: None

#

# Modification History:

#

# Date Name Description

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

# 2004-05-15 Bob Chong Original

#

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

set -v

set -x

# log and reference files

msgLog=/admsrv/admin/log/monitorLog/alertDog.log

errRpt=/admsrv/admin/log/monitorLog/syserr.rpt

reFile=/admsrv/admin/log/monitorLog/alertDog.ref

# email user list

set -A AlertList dguo@yahoo.com

msgLog(){

set -x

print `date` "$1" >> $msgLog

}

msgAlert(){

set -x

echo "URGENT: please call the LMS Unix administrator immediately!" > $errRpt

errpt -a >> $errRpt

mail -s "System Error Reported on Admsrv1!" ${AlertList[\*]} < $errRpt

}

### check the system error message

anyErrpt() {

set -x

typeset integer errptCnt0=0

errptCnt1=`errpt | wc -l`

(( $errptCnt0 == $errptCnt1 ))

}

### main

# check the control reference file

[ -f $reFile ] || {

set -x

msgLog "Error: no control file"

msgAlert

exit 1

}

anyErrpt || {

set -x

msgLog "Error: see errpt message"

msgAlert

exit 1

}

msgLog "Message: no errpt message"

touch $reFile

exit 0

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

# Script /admsrv/nmon/startnmon.ksh

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

#!/bin/ksh

date

cd /admsrv/nmon

nmon -f -t -r admsrv1 -s900 -c90 -D

exit 0

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

# Script /admsrv/drmgr/aix/db2del01.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

date

. /home/db2inst1/sqllib/db2profile

#su - db2inst1 >> /admsrv/drmgr/aix/db2del01.out.$timestamp 2>&1 <<EOF

db2adutl delete full keep 10 db icmnlsdb without prompting >> /admsrv/drmgr/aix/db2del01.out.$timestamp 2>&1

#EOF

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2del01.out.\*" -mtime +10 -exec rm -f {} \;

exit 0

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

# Script /admsrv/drmgr/aix/db2del02.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

date

. /home/db2inst2/sqllib/db2profile

#su - db2inst2 >> /admsrv/drmgr/aix/db2del02.out.$timestamp 2>&1 <<EOF

db2adutl delete full keep 10 db rmdblb without prompting >> /admsrv/drmgr/aix/db2del02.out.$timestamp 2>&1

#EOF

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2del02.out.\*" -mtime +10 -exec rm -f {} \;

exit 0

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

# Script /admsrv/drmgr/aix/db2rec01.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

. /home/db2inst1/sqllib/db2profile

date >> /admsrv/drmgr/aix/db2rec01.out.$timestamp

db2adutl query db icmnlsdb | head -10 >> /admsrv/drmgr/aix/db2rec01.out.$timestamp

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2rec01.out.\*" -mtime +10 -exec rm -f {} \;

mail -s "LMS ICMNLSDB backup reports @ `date`" lchen@yahoo.com </admsrv/drmgr/aix/db2rec01.out.$timestamp

exit 0

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

# Script /admsrv/drmgr/aix/db2rec02.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

. /home/db2inst2/sqllib/db2profile

date >> /admsrv/drmgr/aix/db2rec02.out.$timestamp

db2adutl query db rmdblb | head -10 >> /admsrv/drmgr/aix/db2rec02.out.$timestamp

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2rec02.out.\*" -mtime +10 -exec rm -f {} \;

mail -s "LMS RMDB backup reports @ `date`" lchen@yahoo.com </admsrv/drmgr/aix/db2rec02.out.$timestamp

exit 0

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

# Script /admsrv/admin/bin/cleanup.ksh

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

#!/usr/bin/ksh

#

#

set -x

date

find /tsmha1/preschedule -name "db2\*bkup\*06\*" -type f -mtime +2 -exec /usr/bin/rm -f {} \; \

>> /admsrv/admin/log/cleanupLog/cleanup.log 2>&1

cat /dev/null > /usr/IBMHttpServer/logs/access\_log

cat /dev/null > /usr/IBMHttpServer/logs/error\_log

#Clean system backup logs;

find /admsrv/admin/log/sysbkupLog -mtime +30 -exec /usr/bin/rm -f {} \; \

1>/dev/null 2>&1

exit 0

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

# Script /admsrv/admin/bin/cm\_uncomptx\_cleanup.ksh

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

export IBMCMROOT=/opt/IBM/db2cmv8

cd /opt/IBM/db2cmv8/bin

./icmrmtx.sh

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

# Script /etc/rc.cmrmproc2

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

#!/bin/sh

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

# Licensed Materials - Property of IBM #

# #

# IBM Content Manager for Multiplatforms V8.2 (program number 5724-B19)#

# (c) Copyright IBM Corp. 1994, 2002, 2003. All Rights Reserved. #

# #

# US Government Users Restricted Rights - #

# Use, duplication or disclosure restricted by GSA ADP Schedule #

# Contract with IBM Corporation #

# #

# Program name : rm.cmrmproc #

# #

# Description : Auto start all ContentManager Resource Manager #

# processes on system boot and also enable selective #

# stop / start of components as required #

# #

# NOTE: To avoid your system from failing on reboot, do not change #

# this file in any way. #

# #

# This script is designed to be executed on reboot. Do the #

# following to enable auto-starting all Content Manager 8.1 #

# Resource Manager processes on boot #

# #

# 1) copy this file as /etc/rc.cmrmproc #

# 2) add the following line to /etc/inittab: #

# #

# cm:2:once:/etc/rc.cmrmproc > /dev/console 2>&1 #CM AIX #

# cm:3:once:/etc/rc.cmrmproc > /dev/console 2>&1 #CM SUN #

# #

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

init()

{

sunawk=/usr/xpg4/bin/awk

IBMCMROOT=/opt/IBM/db2cmv8

hostname=`hostname`

caller\_script=rc.cmrmproc

if [ `uname` = AIX ] || [ `uname` = SunOS ] || [ `uname` = Linux ] ; then

if [ -f $IBMCMROOT/config/rmutil\_common.sh ] ; then

. $IBMCMROOT/config/rmutil\_common.sh $caller\_script $\*

else

exit

fi

fi

JAVA=$WAS\_HOME/java/bin/java

$JAVA NLVLog ICMRM COPYRIGHT

}

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

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

getPortNum()

{

if [ `uname` = SunOS ]; then

PortNum=`"$sunawk" "/^$RM\_COMPONENT[\t| ]/" /etc/services | tail -1 | cut -f1 -d"/" | "$sunawk" '{print $2}'`

else

PortNum=`awk "/^$RM\_COMPONENT[\t| ]/" /etc/services | tail -1 | cut -f1 -d"/" | awk '{print $2}'`

fi

if [ -z $PortNum ] ; then

$JAVA NLVLog ICMRM INVALID\_INPUT\_PARM $RM\_COMPONENT

$JAVA NLVLog ICMRM INVALID\_INPUT\_PARM $RM\_COMPONENT >>$CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

exit

fi

}

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

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

isRunning()

{

baseComp=$1

echo "$baseComp --> $dbname"

if [ `uname` = AIX ] ; then

ps -ef | grep java | awk "/$dbname/" | grep $baseComp > /dev/null 2>&1

if [ $? -eq 0 ] ; then

if [ "$procAction" = "start" ] ; then

$JAVA NLVLog ICMRM RM\_PROC\_RUNNING $baseComp $dbname >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

runproc=no

else

runproc=yes

fi

elif [ `uname` = SunOS ] ; then

/usr/ucb/ps auxww | grep java | "$sunawk" "/$dbname/" | grep $baseComp > /dev/null 2>&1

if [ $? -eq 0 ] ; then

if [ "$procAction" = "start" ] ; then

$JAVA NLVLog ICMRM RM\_PROC\_RUNNING $baseComp $dbname >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

runproc=no

else

runproc=yes

fi

elif [ `uname` = Linux ] ; then

/bin/ps auxww | grep java | awk "/$dbname/" | grep $baseComp > /dev/null 2>&1

if [ $? -eq 0 ] ; then

if [ "$procAction" = "start" ] ; then

$JAVA NLVLog ICMRM RM\_PROC\_RUNNING $baseComp $dbname >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

runproc=no

else

runproc=yes

fi

fi

}

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

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

getValue()

{

if [ `uname` = SunOS ]; then

parm1=`"$sunawk" "/RMMigrator\_$dbname[\t| ]/" /etc/services | tail -1`

parm2=`"$sunawk" "/RMPurger\_$dbname[\t| ]/" /etc/services | tail -1`

parm3=`"$sunawk" "/RMReplicator\_$dbname[\t| ]/" /etc/services | tail -1`

parm4=`"$sunawk" "/RMStager\_$dbname[\t| ]/" /etc/services | tail -1`

else

parm1=`awk "/RMMigrator\_$dbname[\t| ]/" /etc/services | tail -1`

parm2=`awk "/RMPurger\_$dbname[\t| ]/" /etc/services | tail -1`

parm3=`awk "/RMReplicator\_$dbname[\t| ]/" /etc/services | tail -1`

parm4=`awk "/RMStager\_$dbname[\t| ]/" /etc/services | tail -1`

fi

var1="${parm1} ${parm2} ${parm3} ${parm4}"

}

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

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

showUsage()

{

$JAVA RMUtilHelp $caller\_script

}

checkInput()

{

## Special for -proc <procname>

## add support for "migrator", "replicator",etc. instead of "RMMigrator", "RMReplicator"..

if [ -n "$process" ]; then

if [ "$process" = "migrator" ] ; then

process=RMMigrator

elif [ "$process" = "replicator" ] ; then

process=RMReplicator

elif [ "$process" = "purger" ] ; then

process=RMPurger

elif [ "$process" = "stager" ] ; then

process=RMStager

fi

fi

if [ -z $dbname ] || [ -z $rmappname ]; then

showUsage

echo "RMDBNAME: $dbname --> RMAPPNAME: $rmappname" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

exit

fi

if [ ! -d $fullpath ] ; then

$JAVA NLVLog ICMRM RM\_DEPLOY\_ENV\_PROBLEM >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA NLVLog ICMRM RM\_DEPLOY\_APP\_PROBLEM >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA NLVLog ICMRM RM\_DEPLOY\_DIR\_PROBLEM >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

exit

fi

if [ "$procAction" = "start" ] ; then

$JAVA NLVLog ICMRM RM\_PROC\_STARTING 8.3 >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

elif [ "$procAction" = "stop" ] ; then

$JAVA NLVLog ICMRM RM\_PROC\_STOPPING 8.3 >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

else

procAction=start

$JAVA NLVLog ICMRM RM\_PROC\_STARTING 8.3 >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

}

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

cleanupProcess()

{

baseComp=$1

echo "$baseComp --> $dbname"

if [ `uname` = AIX ] ; then

isRunning $baseComp

if [ "$runproc" = "no" ] ; then

ps -ef | grep java | awk "/$dbname/" | grep $baseComp | grep "$PortNum" | awk '{print $2}' | xargs kill

fi

elif [ `uname` = SunOS ] ; then

isRunning $baseComp

if [ "$runproc" = "no" ] ; then

/usr/ucb/ps auxww | grep java | "$sunawk" "/$dbname/" | grep $baseComp | grep "$PortNum" | "$sunawk" '{print $2}' | xargs kill

fi

elif [ `uname` = Linux ] ; then

isRunning $baseComp

if [ "$runproc" = "no" ] ; then

/bin/ps auxww | grep java | awk "/$dbname/" | grep $baseComp | grep "$PortNum" | awk '{print $2}' | xargs kill

fi

fi

}

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

# Main()

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

init $\*

fullpath=$rmappdir/icmrm.war/WEB-INF/classes

checkInput

if [ -n "$process" ]; then

if [ `uname` = SunOS ]; then

var1=`"$sunawk" "/$process\\_$dbname[\t| ]/" /etc/services | tail -1`

else

var1=`awk "/$process\\_$dbname[\t| ]/" /etc/services | tail -1`

fi

else

getValue

fi

for i in set $var1

do

case $i in

RMMigrator\_$dbname)

RM\_COMPONENT=RMMigrator\_$dbname

isRunning RMMigrator

getPortNum

if [ "$runproc" = "yes" ] ; then

if [ "$procAction" = "start" ]; then

cd $fullpath

echo "$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMMigratorControl $PortNum $waittime $dbname" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA -Xms256M -Xmx768M com.ibm.mm.icmrm.process.RMMigratorControl $PortNum $waittime $dbname & > /dev/null 2>&1

$JAVA NLVLog ICMRM MIG\_STARTED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

elif [ "$runproc" = "no" ] ; then

if [ "$procAction" = "stop" ]; then

cd $fullpath

echo "$JAVA com.ibm.mm.icmrm.process.RMProcessClient $hostname $PortNum shutdown $RM\_COMPONENT" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA com.ibm.mm.icmrm.process.RMProcessClient `hostname` $PortNum shutdown $RM\_COMPONENT & > /dev/null 2>&1

# Now have configurable shutdown time parameters. See setprocenv for timewait and sleeptime

isRunning RMMigrator

startcount=0

while [ $runproc = "no" ]; do

startcount=`expr $startcount + $sleeptime`

sleep $sleeptime

isRunning RMMigrator

if [ $startcount -ge $waittime ] ; then

runproc=yes

fi

done

#Sometimes the child java process under Linux lose their connectivity. Make sure they all go down

cleanupProcess RMMigrator

$JAVA NLVLog ICMRM MIG\_STOPPED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

else

showUsage

fi

;;

RMPurger\_$dbname)

RM\_COMPONENT=RMPurger\_$dbname

isRunning RMPurger

getPortNum

if [ "$runproc" = "yes" ] ; then

if [ "$procAction" = "start" ]; then

cd $fullpath

echo "$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMPurgerControl $PortNum $waittime $dbname" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMPurgerControl $PortNum $waittime $dbname & >/dev/null 2>&1

$JAVA NLVLog ICMRM PURGER\_STARTED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

elif [ "$runproc" = "no" ] ; then

if [ "$procAction" = "stop" ]; then

cd $fullpath

echo "$JAVA com.ibm.mm.icmrm.process.RMProcessClient $hostname $PortNum shutdown $RM\_COMPONENT" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA com.ibm.mm.icmrm.process.RMProcessClient `hostname` $PortNum shutdown $RM\_COMPONENT & > /dev/null 2>&1

# Now have configurable shutdown time parameters. See setprocenv for timewait and sleeptime

isRunning RMPurger

startcount=0

while [ $runproc = "no" ]; do

startcount=`expr $startcount + $sleeptime`

sleep $sleeptime

isRunning RMPurger

if [ $startcount -ge $waittime ] ; then

runproc=yes

fi

done

#Sometimes the child java process under Linux lose their connectivity. Make sure they all go down

cleanupProcess RMPurger

$JAVA NLVLog ICMRM PURGER\_STOPPED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

else

showUsage

fi

;;

RMReplicator\_$dbname)

RM\_COMPONENT=RMReplicator\_$dbname

isRunning RMReplica

getPortNum

echo " runproc $runproc procAction $procAction" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

if [ "$runproc" = "yes" ] ; then

if [ "$procAction" = "start" ]; then

cd $fullpath

echo "$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMReplicaControl $PortNum $waittime $dbname" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMReplicaControl $PortNum $waittime $dbname & >/dev/null 2>&1

$JAVA NLVLog ICMRM REPLICATOR\_STARTED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

elif [ "$runproc" = "no" ] ; then

if [ "$procAction" = "stop" ]; then

cd $fullpath

echo "$JAVA com.ibm.mm.icmrm.process.RMProcessClient $hostname $PortNum shutdown $RM\_COMPONENT" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA com.ibm.mm.icmrm.process.RMProcessClient `hostname` $PortNum shutdown $RM\_COMPONENT & >/dev/null 2>&1

# Now have configurable shutdown time parameters. See setprocenv for timewait and sleeptime

isRunning RMReplica

startcount=0

while [ $runproc = "no" ]; do

startcount=`expr $startcount + $sleeptime`

sleep $sleeptime

isRunning RMReplica

if [ $startcount -ge $waittime ] ; then

runproc=yes

fi

done

#Sometimes the child java process under Linux lose their connectivity. Make sure they all go down

cleanupProcess RMReplica

$WAS\_HOME/java/bin/java NLVLog ICMRM REPLICATOR\_STOPPED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

else

showUsage

fi

;;

RMStager\_$dbname)

RM\_COMPONENT=RMStager\_$dbname

isRunning RMStager

getPortNum

if [ "$runproc" = "yes" ] ; then

if [ "$procAction" = "start" ]; then

cd $fullpath

echo "$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMStagerControl $PortNum $waittime $dbname" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA -Xms$initjavaheap -Xmx$maxjavaheap com.ibm.mm.icmrm.process.RMStagerControl $PortNum $waittime $dbname & >/dev/null 2>&1

$JAVA NLVLog ICMRM STAGER\_STARTED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

elif [ "$runproc" = "no" ] ; then

if [ "$procAction" = "stop" ]; then

cd $fullpath

echo "$JAVA com.ibm.mm.icmrm.process.RMProcessClient $hostname $PortNum shutdown $RM\_COMPONENT" >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

$JAVA com.ibm.mm.icmrm.process.RMProcessClient `hostname` $PortNum shutdown $RM\_COMPONENT & > /dev/null 2>&1

# Now have configurable shutdown time parameters. See setprocenv for timewait and sleeptime

isRunning RMStager

startcount=0

while [ $runproc = "no" ]; do

startcount=`expr $startcount + $sleeptime`

sleep $sleeptime

isRunning RMStager

if [ $startcount -ge $waittime ] ; then

runproc=yes

fi

done

#Sometimes the child java process under Linux lose their connectivity. Make sure they all go down

cleanupProcess RMStager

$JAVA NLVLog ICMRM STAGER\_STOPPED >> $CMRM\_LOG\_DIR/$CMRM\_LOG\_FILE

fi

else

showUsage

fi

;;

esac

done

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

#End Main()

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

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

# Script /usr/es/sbin/cluster/utilities/clcycle

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

#!/bin/ksh

# IBM\_PROLOG\_BEGIN\_TAG

# This is an automatically generated prolog.

#

# 53haes\_r560 src/43haes/usr/sbin/cluster/utilities/clcycle.sh 1.7.2.20

#

# Licensed Materials - Property of IBM

#

# COPYRIGHT International Business Machines Corp. 1990,2008

# All Rights Reserved

#

# US Government Users Restricted Rights - Use, duplication or

# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

#

# IBM\_PROLOG\_END\_TAG

#

# If about to cycle through hacmp.out files, append all Event Summaries from

# hacmp.out file to a cl\_event\_summaries.txt file.

# @(#)62 1.7.2.20 src/43haes/usr/sbin/cluster/utilities/clcycle.sh, hacmp.utils, 53haes\_r560 6/11/08 12:56:15

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

#

# COMPONENT\_NAME: UTILITIES

#

# FUNCTIONS: none

#

# Name: clcycle

#

# This program saves the the LOGFILE regularly

#

# Arguments: start - cycle the clstrmgr.debug log file

# ||

# $@ - cycle the list of log files

#

# Returns: 0 - success

#

# Environment:

#

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

PROGNAME=$(basename ${0})

export PATH="$($(dirname ${0})/cl\_get\_path all)"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

[[ "$VERBOSE\_LOGGING" = "high" ]] && version='1.7.2.20'

HA\_DIR="$(cl\_get\_path)"

#

# save\_log will save 7 consecutive versions of the

# logfile which is passed.

#

save\_log() {

typeset PS4\_FUNC="save\_log"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

LOGFILE=$1

mv $LOGFILE.6 $LOGFILE.7 2> /dev/null

mv $LOGFILE.5 $LOGFILE.6 2> /dev/null

mv $LOGFILE.4 $LOGFILE.5 2> /dev/null

mv $LOGFILE.3 $LOGFILE.4 2> /dev/null

mv $LOGFILE.2 $LOGFILE.3 2> /dev/null

mv $LOGFILE.1 $LOGFILE.2 2> /dev/null

mv $LOGFILE $LOGFILE.1 2> /dev/null

touch $LOGFILE 2> /dev/null

}

#

# The clstrmgr.debug file is different. While the clstrmgr is

# running, it holds a file descriptor to it. So, just moving it

# does not work.

# There is a "-c" option on cl\_src\_cmd to contact the clstrmgr and

# have it cycle this particular logfile.

cycle\_clstrmgr\_log() {

typeset PS4\_FUNC="cycle\_clstrmgr\_log"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

STANZA=$(odmget -q"name = clstrmgr.debug" HACMPlogs)

if [ "$STANZA" != "" ]

then

/usr/es/sbin/cluster/diag/cl\_src\_cmd -c

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clstrmgr.debug"

fi

}

# The clavan.log is processed in the same way as clstrmgr.debug

# because clstrmgr holds a file descriptor to it.

cycle\_clavan\_log() {

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

STANZA=$(odmget -q"name = clavan.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

/usr/es/sbin/cluster/diag/cl\_src\_cmd -cl

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clavan.log"

fi

}

ODMDIR="/etc/${HA\_DIR}/objrepos"

DEFAULTLOGDIR="/var/hacmp/log"

LOG\_LIST=""

# We need to determine which logs need to be cycled.

#

# If the first option is "startup", then we are being

# called by clstart, and only want to cycle the

# clstrmgr.debug file. Since the clstrmgr is not

# running at this time, we can use the same save\_log

# procedure that all the other logs use.

#

if [[ $1 = startup ]]

then

STANZA=$(odmget -q"name = clstrmgr.debug" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSTRMGR\_OUT\_FILE="$DESTDIR/clstrmgr.debug"

save\_log $CLSTRMGR\_OUT\_FILE

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clstrmgr.debug"

fi

exit 0

fi

LOG\_LIST="$\*"

# WebSMIT logs. Cycle them only if no logs are

# specified on the command line; We want them to be cycled from cron, but not

# necessarily every time this script is run. If needed, they can be specified

# on the command line.

if [[ -z $LOG\_LIST ]] ; then

/usr/es/sbin/cluster/wsm/websmitctl clcycle

fi

# Otherwise, we are called from cron, or from the command

# line. In that case, hacmp.out and clinfo.rc.out get

# cycled by default.

#

# If the name of a logfile is given on invocation of this script,

# it gets cycled.

#

# Other than clinfo.rc.out , we need to check

# if the location has been changed in the odm.

#

LOG\_LIST="$LOG\_LIST $DEFAULTLOGDIR/clinfo.rc.out"

#

# For the rest, read the HACMPlogs ODM for the pathname of the file.

# If the ODM is empty or corrupted, use its default location.

#

# We always do hacmp.out

#

STANZA=$(odmget -q"name = hacmp.out" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

HACMP\_OUT\_FILE="$DESTDIR/hacmp.out"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "hacmp.out"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "hacmp.out"

HACMP\_OUT\_FILE="$DEFAULTLOGDIR/hacmp.out"

fi

LOG\_LIST="$LOG\_LIST $HACMP\_OUT\_FILE"

if [[ $\* = \*cluster.log\* ]]

then

STANZA=$(odmget -q"name = cluster.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSTRLOG\_OUT\_FILE="$DESTDIR/cluster.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cluster.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cluster.log"

CLSTRLOG\_OUT\_FILE="$DEFAULTLOGDIR/cluster.log"

fi

LOG\_LIST="$LOG\_LIST $CLSTRLOG\_OUT\_FILE"

fi

if [[ $\* = \*cl\_sm.log\* ]]

then

STANZA=$(odmget -q"name = cl\_sm.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSM\_OUT\_FILE="$DESTDIR/cl\_sm.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cl\_sm.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cl\_sm.log"

CLSM\_OUT\_FILE="$DEFAULTLOGDIR/cl\_sm.log"

fi

LOG\_LIST="$LOG\_LIST $CLSM\_OUT\_FILE"

fi

if [[ $\* = \*cspoc.log ]]

then

STANZA=$(odmget -q"name = cspoc.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CSPOC\_OUT\_FILE="$DESTDIR/cspoc.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cspoc.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cspoc.log"

CSPOC\_OUT\_FILE="$DEFAULTLOGDIR/cspoc.log"

fi

LOG\_LIST="$LOG\_LIST $CSPOC\_OUT\_FILE"

fi

if [[ $\* = \*cspoc.log.long ]]

then

STANZA=$(odmget -q"name = cspoc.log.long" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CSPOCLONG\_OUT\_FILE="$DESTDIR/cspoc.log.long"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cspoc.log.long"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cspoc.log.long"

CSPOCLONG\_OUT\_FILE="$DEFAULTLOGDIR/cspoc.log.long"

fi

LOG\_LIST="$LOG\_LIST $CSPOCLONG\_OUT\_FILE"

fi

if [[ $\* = \*emuhacmp.out\* ]]

then

STANZA=$(odmget -q"name = emuhacmp.out" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

EMU\_OUT\_FILE="$DESTDIR/emuhacmp.out"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "emuhacmp.out"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "emuhacmp.out"

EMU\_OUT\_FILE="$DEFAULTLOGDIR/emuhacmp.out"

fi

LOG\_LIST="$LOG\_LIST $EMU\_OUT\_FILE"

fi

if [[ $\* = \*clavan.log\* ]]

then

STANZA=$(odmget -q"name = clavan.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLUMT\_OUT\_FILE="$DESTDIR/clavan.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clavan.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clavan.log"

CLUMT\_OUT\_FILE="$DEFAULTLOGDIR/clavan.log"

fi

LOG\_LIST="$LOG\_LIST $CLUMT\_OUT\_FILE"

fi

if [[ $\* = \*clinfo.log\* ]]

then

STANZA=$(odmget -q"name = clinfo.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLINFO\_OUT\_FILE="$DESTDIR/clinfo.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clinfo.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clinfo.log"

CLINFO\_OUT\_FILE="$DEFAULTLOGDIR/clinfo.log"

fi

LOG\_LIST="$LOG\_LIST $CLINFO\_OUT\_FILE"

fi

# We need to check the size of clutils.log, if it's greater than 1MB

# in size then we will rotate it, even if it wasn't specified by the user.

# Of course if the user specifies it, then we will force the rotation

# regardless of the size of the file.

STANZA=$(odmget -q"name = clutils.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLUTILS\_LOG\_FILE="$DESTDIR/clutils.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clutils.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clutils.log"

CLUTILS\_LOG\_FILE="$DEFAULTLOGDIR/clutils.log"

fi

if [ -f "$CLUTILS\_LOG\_FILE" ];then

CLUTILS\_SIZE=$(ls -l $CLUTILS\_LOG\_FILE | awk '{print $5}')

else

touch "$CLUTILS\_LOG\_FILE"

CLUTILS\_SIZE=0

fi

# If clutils.log has been specified OR if clutils.log size is greater than 1MB

if [[ $\* = \*clutils.log\* ]] || (( $CLUTILS\_SIZE > 1000000 )); then

LOG\_LIST="$LOG\_LIST $CLUTILS\_LOG\_FILE"

fi

#

# Now, cycle the selected log files.

#

#

# If about to cycle through hacmp.out files, append all Event Summaries from

# hacmp.out file to a /var/hacmp/log/cl\_event\_summaries.txt file.

for log in $LOG\_LIST ; do

if [ "$log" = "$HACMP\_OUT\_FILE" ]

then

cl\_extract\_evsum -w

fi

[[ -s $log ]] && save\_log $log

done

# The clstrmgr.debug file is different. Since the clstrmgr always

# holds a file descriptor to it, just moving it does not work.

if [[ $\* = \*clstrmgr.debug\* ]]

then

cycle\_clstrmgr\_log

fi

# The cluster.log file is maintained through syslogd, and syslogd

# holds a file descriptor on it, so we need to refresh the syslogd

# daemon to open the new empty cluster.log file and use that.

if [[ $\* = \*cluster.log\* ]]

then

refresh -s syslogd

fi

# Clinfo has to be sent SIGUSR2 in order for the log to be cycled.

if [[ $\* = \*clinfo.log\* ]]

then

CLINFOPID=$(lssrc -s clinfoES | tail -1 | awk '{print $3}' | sed -e 's/[^0-9]//g')

[[ -n $CLINFOPID ]] && kill -31 $CLINFOPID

fi

# If the file is clavan.log then do this

if [[ $\* = \*clavan.log\* ]]

then

cycle\_clavan\_log

fi

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

# Script /admsrv/local/apps/isisapps/isis\_import.sh

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

#!/bin/ksh

#set -v

#set -x

cd /admsrv/local/apps/isisapps

. ./isis\_profile

echo > isis\_import.log

echo "sYear=`date`" >> isis\_import.log

fileNo=`ls -l /arstmp/in/ISIS\_import|grep -v 'total'|wc -l`

#To confirm we do receive files from ISIS in order to process it.

if [[ $fileNo -eq 0 ]]

then

echo "`date`: there is no Image Files from ISIS ...\n" >> isis\_import.log

tail -3 isis\_import.log | mail -s "Missing ISIS files @ `date`" isis\_tech@yahoo.com

else

echo "`date`: we have received some files and will process it soon ...\n"

nohup java -Xmx512m com/yahoo/isis/ImportProcess >> isis\_import.log 2>&1

fi

cd /admsrv/local/apps/isisapps

grep ImportProcess isis\_import.log

if [[ $? -eq 0 ]]

then

echo "`date`: there is error message (Job failed) ... \n"

tail -3 isis\_import.log | mail -s "ImportPorcess failed @ `date`" isis\_tech@yahoo.com

fi

#YEAR=`date +%Y`

#MON=`date +%m`

#DAY=`date +%d`

#cd /arstmp/in/backup/$YEAR-$MON-$DAY

#ls -l | grep 'ImportBusiness'

#if [[ $? -eq 0 ]]

#then

# echo "There is error message (Backup failed) ... \n"

# file=`ls -l|grep 'ImportBusiness'|awk '{print $9}'`

# mail -s "ISISFTP Process Backup Error Report @ `date`" isis\_tech@yahoo.com < $file

#fi

exit 0

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

# Script /admsrv/admin/bin/sysbkup.ksh

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

#!/bin/ksh

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

#

# Name: sysbkup.ksh

#

# Reference: n/a

#

# Description: system backup using mksysb

#

# Parameters: sysbkup.ksh <tape device>

# tape device /dev/rmt0

#

# Modification History:

#

# Date Name Description

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

# 2004-05-15 Bob Chong Original

#

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

set -v

set -x

# script library

PATH=/admsrv/admin/lib:$PATH:.

cd /admsrv/admin/log/sysbkupLog

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

# rewind the tape

tctl -f $backup\_tape rewind

if [ $? != 0 ]

then

date > $errfile

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

mail -s "Sysbkup failed (admsrv1) due to tape not ready : `date`" lchen@yahoo.com < $errfile

exit 1

fi

# 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 "Sysbkup failed (admsrv1): `date`" lchen@yahoo.com < $logfile

tctl -f $backup\_tape offline

exit 1

fi

# rewind the tape

bot.check $backup\_tape $logfile

# finally list all the files on tape

logevent $logfile "----------------------------------------------------"

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

logevent $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: $1" | tee -a $lisfile

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

mail -s "Sysbkup failed (admsrv1): `date`" lchen@yahoo.com < $lisfile

tctl -f $backup\_tape offline

exit 1

fi

logevent $logfile "----------------------------------------------------"

#rm $errfile

logevent $logfile "SYSTEM BACKUP task has been completed"

logevent $logfile "----------------------------------------------------"

mail -s "Sysbkup successful (admsrv1): `date`" lchen@yahoo.com < $logfile

mail -s "Sysbkup successful (admsrv1): `date`" computerops@yahoo.com < $logfile

sleep 3

# dismount the tape

#tctl -f $backup\_tape offline

exit 0

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

# Script /admsrv/admin/bin/alertDog.ksh

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

#!/bin/ksh

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

#

# Name: alertDog.ksh

#

# Reference: n/a

#

# Description: monitor the errpt message

#

# Parameters: None

#

# Modification History:

#

# Date Name Description

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

# 2004-05-15 Bob Chong Original

#

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

set -v

set -x

# log and reference files

msgLog=/admsrv/admin/log/monitorLog/alertDog.log

errRpt=/admsrv/admin/log/monitorLog/syserr.rpt

reFile=/admsrv/admin/log/monitorLog/alertDog.ref

# email user list

set -A AlertList dguo@yahoo.com

msgLog(){

set -x

print `date` "$1" >> $msgLog

}

msgAlert(){

set -x

echo "URGENT: please call the LMS Unix administrator immediately!" > $errRpt

errpt -a >> $errRpt

mail -s "System Error Reported on Admsrv1!" ${AlertList[\*]} < $errRpt

}

### check the system error message

anyErrpt() {

set -x

typeset integer errptCnt0=0

errptCnt1=`errpt | wc -l`

(( $errptCnt0 == $errptCnt1 ))

}

### main

# check the control reference file

[ -f $reFile ] || {

set -x

msgLog "Error: no control file"

msgAlert

exit 1

}

anyErrpt || {

set -x

msgLog "Error: see errpt message"

msgAlert

exit 1

}

msgLog "Message: no errpt message"

touch $reFile

exit 0

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

# Script /admsrv/nmon/startnmon.ksh

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

#!/bin/ksh

date

cd /admsrv/nmon

nmon -f -t -r admsrv1 -s900 -c90 -D

exit 0

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

# Script /admsrv/drmgr/aix/db2del01.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

date

. /home/db2inst1/sqllib/db2profile

#su - db2inst1 >> /admsrv/drmgr/aix/db2del01.out.$timestamp 2>&1 <<EOF

db2adutl delete full keep 10 db icmnlsdb without prompting >> /admsrv/drmgr/aix/db2del01.out.$timestamp 2>&1

#EOF

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2del01.out.\*" -mtime +10 -exec rm -f {} \;

exit 0

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

# Script /admsrv/drmgr/aix/db2del02.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

date

. /home/db2inst2/sqllib/db2profile

#su - db2inst2 >> /admsrv/drmgr/aix/db2del02.out.$timestamp 2>&1 <<EOF

db2adutl delete full keep 10 db rmdblb without prompting >> /admsrv/drmgr/aix/db2del02.out.$timestamp 2>&1

#EOF

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2del02.out.\*" -mtime +10 -exec rm -f {} \;

exit 0

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

# Script /admsrv/drmgr/aix/db2rec01.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

. /home/db2inst1/sqllib/db2profile

date >> /admsrv/drmgr/aix/db2rec01.out.$timestamp

db2adutl query db icmnlsdb | head -10 >> /admsrv/drmgr/aix/db2rec01.out.$timestamp

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2rec01.out.\*" -mtime +10 -exec rm -f {} \;

mail -s "LMS ICMNLSDB backup reports @ `date`" lchen@yahoo.com </admsrv/drmgr/aix/db2rec01.out.$timestamp

exit 0

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

# Script /admsrv/drmgr/aix/db2rec02.ksh

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

#!/bin/ksh

#set -x

timestamp=`date +%Y%m%d\_%H%M%S`

. /home/db2inst2/sqllib/db2profile

date >> /admsrv/drmgr/aix/db2rec02.out.$timestamp

db2adutl query db rmdblb | head -10 >> /admsrv/drmgr/aix/db2rec02.out.$timestamp

#Clean old output file;

cd /admsrv/drmgr/aix

find ./ -name "db2rec02.out.\*" -mtime +10 -exec rm -f {} \;

mail -s "LMS RMDB backup reports @ `date`" lchen@yahoo.com </admsrv/drmgr/aix/db2rec02.out.$timestamp

exit 0

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

# Script /admsrv/admin/bin/cleanup.ksh

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

#!/usr/bin/ksh

#

#

set -x

date

find /tsmha1/preschedule -name "db2\*bkup\*06\*" -type f -mtime +2 -exec /usr/bin/rm -f {} \; \

>> /admsrv/admin/log/cleanupLog/cleanup.log 2>&1

cat /dev/null > /usr/IBMHttpServer/logs/access\_log

cat /dev/null > /usr/IBMHttpServer/logs/error\_log

#Clean system backup logs;

find /admsrv/admin/log/sysbkupLog -mtime +30 -exec /usr/bin/rm -f {} \; \

1>/dev/null 2>&1

exit 0

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

# Script /admsrv/admin/bin/restart\_SNMPD.ksh

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

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

# Script /usr/es/sbin/cluster/utilities/clcycle

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

#!/bin/ksh

# IBM\_PROLOG\_BEGIN\_TAG

# This is an automatically generated prolog.

#

# 53haes\_r560 src/43haes/usr/sbin/cluster/utilities/clcycle.sh 1.7.2.20

#

# Licensed Materials - Property of IBM

#

# COPYRIGHT International Business Machines Corp. 1990,2008

# All Rights Reserved

#

# US Government Users Restricted Rights - Use, duplication or

# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

#

# IBM\_PROLOG\_END\_TAG

#

# If about to cycle through hacmp.out files, append all Event Summaries from

# hacmp.out file to a cl\_event\_summaries.txt file.

# @(#)62 1.7.2.20 src/43haes/usr/sbin/cluster/utilities/clcycle.sh, hacmp.utils, 53haes\_r560 6/11/08 12:56:15

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

#

# COMPONENT\_NAME: UTILITIES

#

# FUNCTIONS: none

#

# Name: clcycle

#

# This program saves the the LOGFILE regularly

#

# Arguments: start - cycle the clstrmgr.debug log file

# ||

# $@ - cycle the list of log files

#

# Returns: 0 - success

#

# Environment:

#

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

PROGNAME=$(basename ${0})

export PATH="$($(dirname ${0})/cl\_get\_path all)"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

[[ "$VERBOSE\_LOGGING" = "high" ]] && version='1.7.2.20'

HA\_DIR="$(cl\_get\_path)"

#

# save\_log will save 7 consecutive versions of the

# logfile which is passed.

#

save\_log() {

typeset PS4\_FUNC="save\_log"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

LOGFILE=$1

mv $LOGFILE.6 $LOGFILE.7 2> /dev/null

mv $LOGFILE.5 $LOGFILE.6 2> /dev/null

mv $LOGFILE.4 $LOGFILE.5 2> /dev/null

mv $LOGFILE.3 $LOGFILE.4 2> /dev/null

mv $LOGFILE.2 $LOGFILE.3 2> /dev/null

mv $LOGFILE.1 $LOGFILE.2 2> /dev/null

mv $LOGFILE $LOGFILE.1 2> /dev/null

touch $LOGFILE 2> /dev/null

}

#

# The clstrmgr.debug file is different. While the clstrmgr is

# running, it holds a file descriptor to it. So, just moving it

# does not work.

# There is a "-c" option on cl\_src\_cmd to contact the clstrmgr and

# have it cycle this particular logfile.

cycle\_clstrmgr\_log() {

typeset PS4\_FUNC="cycle\_clstrmgr\_log"

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

STANZA=$(odmget -q"name = clstrmgr.debug" HACMPlogs)

if [ "$STANZA" != "" ]

then

/usr/es/sbin/cluster/diag/cl\_src\_cmd -c

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clstrmgr.debug"

fi

}

# The clavan.log is processed in the same way as clstrmgr.debug

# because clstrmgr holds a file descriptor to it.

cycle\_clavan\_log() {

[[ "$VERBOSE\_LOGGING" = "high" ]] && set -x

STANZA=$(odmget -q"name = clavan.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

/usr/es/sbin/cluster/diag/cl\_src\_cmd -cl

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clavan.log"

fi

}

ODMDIR="/etc/${HA\_DIR}/objrepos"

DEFAULTLOGDIR="/var/hacmp/log"

LOG\_LIST=""

# We need to determine which logs need to be cycled.

#

# If the first option is "startup", then we are being

# called by clstart, and only want to cycle the

# clstrmgr.debug file. Since the clstrmgr is not

# running at this time, we can use the same save\_log

# procedure that all the other logs use.

#

if [[ $1 = startup ]]

then

STANZA=$(odmget -q"name = clstrmgr.debug" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSTRMGR\_OUT\_FILE="$DESTDIR/clstrmgr.debug"

save\_log $CLSTRMGR\_OUT\_FILE

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clstrmgr.debug"

fi

exit 0

fi

LOG\_LIST="$\*"

# WebSMIT logs. Cycle them only if no logs are

# specified on the command line; We want them to be cycled from cron, but not

# necessarily every time this script is run. If needed, they can be specified

# on the command line.

if [[ -z $LOG\_LIST ]] ; then

/usr/es/sbin/cluster/wsm/websmitctl clcycle

fi

# Otherwise, we are called from cron, or from the command

# line. In that case, hacmp.out and clinfo.rc.out get

# cycled by default.

#

# If the name of a logfile is given on invocation of this script,

# it gets cycled.

#

# Other than clinfo.rc.out , we need to check

# if the location has been changed in the odm.

#

LOG\_LIST="$LOG\_LIST $DEFAULTLOGDIR/clinfo.rc.out"

#

# For the rest, read the HACMPlogs ODM for the pathname of the file.

# If the ODM is empty or corrupted, use its default location.

#

# We always do hacmp.out

#

STANZA=$(odmget -q"name = hacmp.out" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

HACMP\_OUT\_FILE="$DESTDIR/hacmp.out"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "hacmp.out"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "hacmp.out"

HACMP\_OUT\_FILE="$DEFAULTLOGDIR/hacmp.out"

fi

LOG\_LIST="$LOG\_LIST $HACMP\_OUT\_FILE"

if [[ $\* = \*cluster.log\* ]]

then

STANZA=$(odmget -q"name = cluster.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSTRLOG\_OUT\_FILE="$DESTDIR/cluster.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cluster.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cluster.log"

CLSTRLOG\_OUT\_FILE="$DEFAULTLOGDIR/cluster.log"

fi

LOG\_LIST="$LOG\_LIST $CLSTRLOG\_OUT\_FILE"

fi

if [[ $\* = \*cl\_sm.log\* ]]

then

STANZA=$(odmget -q"name = cl\_sm.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLSM\_OUT\_FILE="$DESTDIR/cl\_sm.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cl\_sm.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cl\_sm.log"

CLSM\_OUT\_FILE="$DEFAULTLOGDIR/cl\_sm.log"

fi

LOG\_LIST="$LOG\_LIST $CLSM\_OUT\_FILE"

fi

if [[ $\* = \*cspoc.log ]]

then

STANZA=$(odmget -q"name = cspoc.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CSPOC\_OUT\_FILE="$DESTDIR/cspoc.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cspoc.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cspoc.log"

CSPOC\_OUT\_FILE="$DEFAULTLOGDIR/cspoc.log"

fi

LOG\_LIST="$LOG\_LIST $CSPOC\_OUT\_FILE"

fi

if [[ $\* = \*cspoc.log.long ]]

then

STANZA=$(odmget -q"name = cspoc.log.long" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CSPOCLONG\_OUT\_FILE="$DESTDIR/cspoc.log.long"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "cspoc.log.long"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "cspoc.log.long"

CSPOCLONG\_OUT\_FILE="$DEFAULTLOGDIR/cspoc.log.long"

fi

LOG\_LIST="$LOG\_LIST $CSPOCLONG\_OUT\_FILE"

fi

if [[ $\* = \*emuhacmp.out\* ]]

then

STANZA=$(odmget -q"name = emuhacmp.out" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

EMU\_OUT\_FILE="$DESTDIR/emuhacmp.out"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "emuhacmp.out"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "emuhacmp.out"

EMU\_OUT\_FILE="$DEFAULTLOGDIR/emuhacmp.out"

fi

LOG\_LIST="$LOG\_LIST $EMU\_OUT\_FILE"

fi

if [[ $\* = \*clavan.log\* ]]

then

STANZA=$(odmget -q"name = clavan.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLUMT\_OUT\_FILE="$DESTDIR/clavan.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clavan.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clavan.log"

CLUMT\_OUT\_FILE="$DEFAULTLOGDIR/clavan.log"

fi

LOG\_LIST="$LOG\_LIST $CLUMT\_OUT\_FILE"

fi

if [[ $\* = \*clinfo.log\* ]]

then

STANZA=$(odmget -q"name = clinfo.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLINFO\_OUT\_FILE="$DESTDIR/clinfo.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clinfo.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clinfo.log"

CLINFO\_OUT\_FILE="$DEFAULTLOGDIR/clinfo.log"

fi

LOG\_LIST="$LOG\_LIST $CLINFO\_OUT\_FILE"

fi

# We need to check the size of clutils.log, if it's greater than 1MB

# in size then we will rotate it, even if it wasn't specified by the user.

# Of course if the user specifies it, then we will force the rotation

# regardless of the size of the file.

STANZA=$(odmget -q"name = clutils.log" HACMPlogs)

if [ "$STANZA" != "" ]

then

DESTDIR=`echo $STANZA | cut -d'"' -f8`

CLUTILS\_LOG\_FILE="$DESTDIR/clutils.log"

else

dspmsg scripts.cat 463 "The cluster log entry for %s could not be found in the HACMPlogs ODM.\n" "clutils.log"

dspmsg scripts.cat 464 "Defaulting to log directory %s for log file %s.\n" "$DEFAULTLOGDIR" "clutils.log"

CLUTILS\_LOG\_FILE="$DEFAULTLOGDIR/clutils.log"

fi

if [ -f "$CLUTILS\_LOG\_FILE" ];then

CLUTILS\_SIZE=$(ls -l $CLUTILS\_LOG\_FILE | awk '{print $5}')

else

touch "$CLUTILS\_LOG\_FILE"

CLUTILS\_SIZE=0

fi

# If clutils.log has been specified OR if clutils.log size is greater than 1MB

if [[ $\* = \*clutils.log\* ]] || (( $CLUTILS\_SIZE > 1000000 )); then

LOG\_LIST="$LOG\_LIST $CLUTILS\_LOG\_FILE"

fi

#

# Now, cycle the selected log files.

#

#

# If about to cycle through hacmp.out files, append all Event Summaries from

# hacmp.out file to a /var/hacmp/log/cl\_event\_summaries.txt file.

for log in $LOG\_LIST ; do

if [ "$log" = "$HACMP\_OUT\_FILE" ]

then

cl\_extract\_evsum -w

fi

[[ -s $log ]] && save\_log $log

done

# The clstrmgr.debug file is different. Since the clstrmgr always

# holds a file descriptor to it, just moving it does not work.

if [[ $\* = \*clstrmgr.debug\* ]]

then

cycle\_clstrmgr\_log

fi

# The cluster.log file is maintained through syslogd, and syslogd

# holds a file descriptor on it, so we need to refresh the syslogd

# daemon to open the new empty cluster.log file and use that.

if [[ $\* = \*cluster.log\* ]]

then

refresh -s syslogd

fi

# Clinfo has to be sent SIGUSR2 in order for the log to be cycled.

if [[ $\* = \*clinfo.log\* ]]

then

CLINFOPID=$(lssrc -s clinfoES | tail -1 | awk '{print $3}' | sed -e 's/[^0-9]//g')

[[ -n $CLINFOPID ]] && kill -31 $CLINFOPID

fi

# If the file is clavan.log then do this

if [[ $\* = \*clavan.log\* ]]

then

cycle\_clavan\_log

fi

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

# Script

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