CAPITA

Scripts Help MS Word Edition: October 2013

Table Of Contents

The cron 9

Introduction to the cron 9

Cron command syntax 9

Cron lines 11

Standard cron file 11

Editing the cron 16

The 'at' command 16

Scripts 19

Introduction to scripts 19

Executing scripts 19

Standard arguments used in scripts 19

Getting help from the UNIX prompt 20

Database Management 21

archive_trandumps 21

checkalloc 22

checkdb 22

checkstorage 22

data_backup 23

data_restore 23

full_dbdump 24

full_dbrestore 25

full_softdump 25

full_softrestore 26

kill_opac 26

kill_process 26

save_master 26

save_sybprocs 27

top5 27

trandump 27

update_stats 27

Item usage scripts 28

ite_usg_add_period.pl 28

ite_usg_upd_current.pl 29

ite_usg_delete_period.pl 30

ite_usg_merge_periods.pl 30

ite_usg_set_display.pl 31
ite_usg_set_available.pl 31
ite_usg_total_loans.pl 32
Utilities 33
assign_rtn_pln.pl 33
authority_build 34
authority_load 35
auto_access_points 36
bor_add_pin 37
bor_anon_delete.pl 37
bor_block.pl 42
bor_name_list_build.pl 42
borr_import 43
borr_type_updt.pl 53
borrower_image_import.pl 54
cad_dup_sans_list 55
chk_seq_reset 55
clear_search_works.pl 56
dedup_works.pl 57
edi_inv_delete.pl 58
email_post.pl 58
email_xfer.pl 59
ffl_assign_links.pl 60
findlock 61
fun_tot_base_exp.pl 61
fun_totals.pl 62
grp_course_import.pl 63
ill_art_intray.pl 64
inv_status_upd.pl 66
imp_modify 67
irs_compress 68
ite_labels.pl 68
itp_seq_reset 68
lo_compress.pl 69
loc_add_insert 70
itu_compress.pl 70
itu_update_wku.pl 71

linkuk_cat_update.pl 72
loa_plr_retrieve.pl 74
loa_plr_retrieve_lyra 76
loa_plr_tape 79
load_authority_tags.pl 80
loan_select 80
marcdiag 82
new_item_exp 83
oll_pass_reset.pl 84
oclc_pica_update.pl 84
orr_ack_imp 86
orr_confirm.pl 87
orr_import 89
oor_subcost_upd.pl 90
orr_pot_ords_del 91
orr_price_upd.pl 92
orr_unverified.pl 93
pay_inv_exp.pl 94
pay_prev_run 96
res_add_itms 97
res_item_rotate.pl 99
resupdate.pl 99
rlb_non_isbn.pl 100
roll_aggfunds_run.pl 103
roll_basefunds_run.pl 103
roll_fyr_backup 103
roll_fyr_drop 103
roll_fyr_recover 103
sel_works 103
ser_qty 104
site_parameter_transfer 104
soc_seq_reset 104
std_prnt_cleanup 105
sup_totals.pl 105
unlock 106
unlocker 106
upd_ser_cns 106

update_daily_access_points 107
wel_update 107
wku_compress.pl 108
wku_update.pl 108
work_logdelete.pl 109
wrk_class_reset 110
wrk_counts 111
wrk_disp_reset 112
wrk_rbn_exp.pl 113
wrk_unsuppress_M21.pl 115
Perl MIS reports 116
Introduction to Perl MIS reports 116
About the MIS server 116
Perl MIS Report Structure 117
Running MIS reports 118
Tailoring reports 119
Tailoring output formats 120
Tailoring selection criteria 121
bor_charge_history 122
bor_charge_stats 123
bor_loan_history 124
bor_ite_charge_stats 125
bor_loc_stats 126
bor_mes_finedays_ins 126
col_shelf_list 127
cop_dates_upd 127
conversionMopUp.pl 128
edi_orders_list 129
fun_ill_list 130
fun_order_audit 131
fun_user_links 132
fun_orders_list 133
ill_art_cancel 133
ill_art_cancel_v2 135
ill_art_chase 136
ill_art_chase_v2 137
ill_art_new 139

ill_art_new_v2 140
ill_art_reapply 141
ill_art_reapply_v2 143
ill_art_renew 144
ill_art_renew_v2 145
ill_letter_chase 147
ill_letter_chase_v2 147
ill_letter_new 148
ill_letter_new_v2 149
ill_letter_reapply 150
ill_letter_reapply_v2 151
ill_letter_renew 152
ill_letter_reapply_v2 153
ill_memo_arrival 154
ill_memo_arrival_v2 155
ill_memo_cancel 156
ill_memo_cancel_v2 156
ill_memo_confirm 157
ill_memo_confirm_v2 158
ill_memo_delay 159
ill_memo_delay_v2 160
ill_memo_form 161
ill_memo_form_v2 162
ill_memo_overdue 163
ill_memo_overdue_v2 163
ill_memo_recall 164
ill_memo_recall_v2 165
ill_memo_refuse 166
ill_memo_refuse_v2 167
ill_memo_source 168
ill_memo_source_v2 169
ill_memo_uncollected 170
ill_memo_uncollected_v2 170
ill_unverified 171
IS_loa_odue_letter 172
ite_miss_del 172
ite_wrk_upd 173

iti_transit 174
iti_transit_del 174
iti_transit_stats 175
itm_onloan_stats 176
itm_rotate 176
loa_borr_loan_stats 177
loa_ite_fmt_loan_stat 179
loa_fine_rate_upd 181
loa_ite_loan_stats 181
loa_long_odue 183
loa_odue_charges 185
loa_odue_loclev 187
loa_odue_letter 189
loa_odue_letter_ftr 190
oor_claim_letter 192
oor_order_letter 193
orr_can_ords_list 194
orr_imp 194
orr_chaser 195
orr_ite_returns 197
orr_order_letter 198
pay_inv_list 199
pay_sup_charges 200
rec_loa_letter 201
rec_loa_letter_dmail 202
rec_long_soon_letter 204
res_dem_od_list 206
res_itm_noloan 206
res_outstanding 207
res_query_letter 209
res_shelf_upd 211
res_wait_list 212
res_waiting_letter 213
res_work_list 213
rlt_works_list 214
SK_loa_school_letter 215
soc_claim_letter 216

soc_no_receipt 217
SelfServ 218
SIP2 scripts 218
ILL Manager 218
ill_caretaker 218
ill_mail 218
ill_manager 219
ill_send 219
ill_stub 219
Import work 219
item_imp 219
build 220
convert 220
ite_wrk_update 220
import_MARC21 221
import_oclc 223
svol_imp 224
work_imp 225
wrk_upd_imp 226
wwl_imp 227

The cron

Introduction to the cron

Many processing jobs, particularly those batch jobs affecting the database, need to be scheduled to run at quieter periods. Some need to be run at regular intervals throughout the working day. The UNIX system process known as cron can be used to automate repetitive tasks.

Cron is a system facility that enables you to schedule the regular or repetitive execution of operations on a time and date basis. Cron is started automatically at system boot. You can also use it to automate daily or weekly operations such as backup and disk clean-up.

- The cron process enables the system manager to set up jobs that will run automatically on pre-determined days and times. It may be used to schedule overnight or weekend jobs, or jobs that need to run at regular intervals during the day.
- The cron process reads a file known as the root cron file into memory and executes the jobs it contains at the times that are specified.
- cron files can potentially exist for each UNIX account on the system. However, for ease of maintenance and support, Capita recommends that a cron file exists only for the root user (or superuser).

Note

An alternative solution is to use the UNIX command known as **at**, as this allows one-off jobs to be scheduled.

Cron command syntax

A global cron file is provided with the UNIX system. This has been edited to include LMS housekeeping operations such as producing system logs, cleaning up the disk, clearing out error logs and some database related operations.

You can view the contents of the global cron file by logging in as **root** and typing:

 # crontab -l

Cron command syntax

The cron process executes the jobs it contains at the times that are specified. Each "cron job" is composed of six fields. Fields one to five contain clock information which specify when the command (given in the sixth field) should be executed.

Field	1	2	3	4	5	6
Function	Minute	Hour	Day of month	Month of year	Day of week	Command
Ranges	0-59	0-23	1-31	1-12	0-6 Begins with Sunday (0) and ends Saturday (6).	

Clock information

- None of the fields can be left blank. For fields 1 through 5 an asterisk must be entered to signify every day or every minute if no specific range or value is required.

- An asterisk can be used to signify all values in a field. For example, by placing an asterisk in the third field, the cron daemon will run a job every day. Use asterisks with caution - you may impact on system performance.
- Ranges can also be defined by using the minus "-" symbol. For example, if you wanted to run a job for the first four months of the year, you could add **1-4** to the fourth field.
- A comma can also be used to select certain values in the field. If you wanted to run a report every 1st and 15th of the month, you could add **1, 15** to the third field.

Commands

In the sixth field of each cron line, commands can be added. The Capita convention is to use the **/bin/su** syntax to run the job as a specified user. For example to run a UNIX script called **daliystatus.sc** as the user called **talis**, the following syntax would be appropriate:

/bin/su – talis –c "/users/talis/dailystatus.sc"

Note that the full pathname to the required script is being cited. This is good practice. You may wish to consider saving such commands in a separate file where they can be commented and pointed to from the cron line. Jobs will be much easier to document, test and maintain and will run sequentially.

Output files

Reference to an output or report file can be made within the quotes surrounding the command. This will generate output just as if the command had been run directly from the UNIX command prompt. For example, to create a report from the dailystatus.sc script, the output could be re-directed thus:

/bin/su – talis –c "/users/talis/dailystatus.sc >/scratch/dailystatus.out"

Note that the full pathname to the output target is used, in this case re-directing the file into the **/scratch** directory.

 Redirecting output files

In addition to the output from the script, output can be created that relates to the running of the cron line itself. For example:

/bin/su – talis –c "/users/talis/dailystatus.sc >/scratch/dailystatus.out"
1>/var/tmp/dailystatus.cron 2>/var/tmp/dailystatus.err

The 1> is cron output and the 2> is error information. Typically this is re-directed into the /var/tmp directory. You may also see lines that force any cron error output into the same file as the cron output. This done using the syntax 2>&1 for example:

/bin/su – talis –c "/users/talis/dailystatus.sc >/scratch/dailystatus.out"
1>/var/tmp/dailystatus.cron 2>&1

As quite often the cron output is not required, it can be re-directed into /dev/null. This effectively suppresses the output. For example:

/bin/su – talis –c "/users/talis/dailystatus.sc >/scratch/dailystatus.out" 1>/dev/null 2>&1

If the cron outputs are not directed to a file or to /dev/null then such output will be sent to the UNIX mailbox for the root account. Use re-direction to limit the growth of this mailbox.

Example cron command line

The following line would run the dailystatus.sc script as a user called talis, at 8am Monday to Friday. It disregards the date. The script output would be placed into the /scratch directory and the cron output into /var/tmp whilst the cron error output is suppressed.

00 08 * * 1-5 /bin/su – talis –c "/users/talis/dailystatus.sc >/scratch/dailystatus.out"
1>/var/tmp/dailystatus.cronl 2>/dev/null

Cron lines

- The order of the lines in the cron is not significant. Lines can be entered anywhere in the file, but it is sensible to group related lines together (for example, all daily jobs).
- Any line in the cron beginning with "#" is treated as a comment, and is not executed as a command. Liberal commenting of the root cron file will can help others understand the function of each line.
- Be careful not to add blank lines to improve the readability of your file. This can result in problems to your scheduled cron jobs. Instead, precede blank lines with the pound symbol so that the rest of the line will not be interpreted.
- The following diagram shows a typical cron file:

Example cron file

```
#############################################################################
#
#                     Talis client cron file
#                     ======================
# value 1 is minutes
# value 2 is hours
# value 3 is day of month (e.g. 31st)
# value 4 is month
# value 5 is day of week (Sunday=0, Monday=1, Saturday=6)
#
# Any output from the command is written to the file specified after the "1>".
# Should the command fail errors will be written to the file specified after
# the "2>".  If no 2> is specified, error messages will be sent to mail,
# which is stored in /var/mail/root.
#
#
#############################################################################
#=========================
# Generate system activity reports
#=========================
0 * * * 0,6             /bin/su - sys -c "/usr/lib/sa/sa1"
0,20,40 8-18 * * 1-5    /bin/su - sys -c "/usr/lib/sa/sa1"
#=========================
# full_dbdump (secures database)
#=========================
0 5 * * 1-6 /bin/su - ops -c "full_dbdump >/var/tmp/full_dbdump.log" 2>/var/tmp/full_dbdump.err
#==========
# Weekly jobs
#==========
30 8 * * 0 /bin/su - talis -c "update_stats prod_talis >/var/tmp/update_stats.log" 2>/var/tmp/update_stats.err
30 23 * * 6 /etc/shutdown -g0 -y -i6
30 13 * * 0 /bin/su - ops -c "checkdb >/var/tmp/checkdb.log" 2>/var/tmp/checkdb.err
0 18 * * 0 /bin/su - ops -c "checkalloc >/var/tmp/checkalloc.log" 2>/var/tmp/checkalloc.err
#==========
# Daily Jobs
#==========
15 23 * * 1-6 /usr/opt/blcmp/backup/bin/kill_opac cron 1>/var/tmp/kill_opac.log
2>/var/tmp/kill_opac.err
0 23 * * 1-6 /bin/su - ops -c "update_daily_access_points >/var/tmp/update_daily_access_points.log"
2>/var/tmp/update_daily_access_points.err
# update reservation status
0 1 * * 1-6 /bin/su - talis -c "resupdate prod_talis >/var/tmp/resupdate.log" 2>/var/tmp/resupdate.err
##########################################
# Stopping and starting daemons
##########################################
0 21 * * * /bin/su - ops -c "dae_term authorisor_dae" 1>/dev/null 2>/dev/null
0 21 * * * /bin/su - ops -c "dae_term work_exp_dae" 1>/dev/null 2>/dev/null
0 21 * * * /bin/su - ops -c "dae_term ord_exp_dae" 1>/dev/null 2>/dev/null
#=========================
```

Standard cron file

The following topic shows the standard cron file supplied with the Capita LMS. If you need to change the cron, please refer to the topic 'Editing the cron'.

Note: you may want to maximise this window to view the contents more easily.

```
#############################################################
#
#                       Standard cron file
#                       ==================================
# Notes:
#
# value 1 is minutes
# value 2 is hours
# value 3 is day of month (e.g. 31st)
# value 4 is month
# value 5 is day of week (Sunday=0, Monday=1, Saturday=6)
#
# Any output from the command is written the file specified after the "1>".
# Should the command fail errors will be written to the file specified after
```

```
# the "2>".  If no 2> is specified, error messages will be sent to mail,
# which is stored in /var/mail/root.
#
# 1> output is generally written to files ending ".log" in /var/tmp
# 2> output is generally written to files ending ".err" in /var/tmp
# >/dev/null means that outputs do not get written to a file
# 2>&1 indicates that errors should be written to the same file as the normal
# output file.
#
# When a job runs it is recorded in the file /var/cron/log
#
# Note that not all of the jobs in this cron exist as shipped scripts
# and may require setting up manually
##############################################################################
#
#============================================================================
# Status Reports
#============================================================================
#=========================
# System activity reports
#=========================
#0 * * * 0,6             /bin/su - sys -c "/usr/lib/sa/sa1" >/dev/null 2>&1
#0 19-7 * * 1-5          /bin/su - sys -c "/usr/lib/sa/sa1" >/dev/null 2>&1
#0,20,40 8-18 * * 1-5    /bin/su - sys -c "/usr/lib/sa/sa1" >/dev/null 2>&1
#==============================
# Run and print status reports
# Note that dailystatus.sc and weeklystatus.sc are not part of the standard ship
# and would need to be set up at installation if required
#==============================
#07 09 * * 2-5 /bin/su - talis -c "dailystatus.sc > /scratch/dailystatus.out" >/dev/null
2>&1
#15 09 * * 2-5 /bin/su - talis -c "lp /scratch/dailystatus.out" >/dev/null 2>&1
#07 09 * * 1 /bin/su - talis -c "weeklystatus.sc > /scratch/weeklystatus.out" >/dev/null
2>&1
#15 09 * * 1 /bin/su - talis -c "lp /scratch/weeklystatus.out" >/dev/null 2>&1
#=============
# top5 report - use to monitor database space
#=============
#00 4 * * 1 /bin/su - sybase -c "/usr/opt/blcmp/backup/bin/top5 >/scratch/top5.txt"
>/dev/null  2>/dev/null
#============
# sys_config scripts
# These gather system information to aid supporting the system
#============
#10 4 * * 1 /bin/su - root -c "/usr/opt/blcmp/backup/bin/sys_config" >/dev/null 2>&1
#15 4 * * 1 /bin/su - ops -c "/usr/opt/blcmp/backup/bin/sys_config_collate" >/dev/null
2>&1
#============================================================================
# Securities and database jobs
#============================================================================
# full_dbdump of all databases (to disk or tape)
#==============================
# to tape
#0 6 * * 1-6 /bin/su - ops -c "full_dbdump -tALL -y" >/var/tmp/full_dbdump.cron 2>&1
# alternative: dump to disk
#0 6 * * * /bin/su - ops -c "dump2disk.pl > /scratch/dump2disk.out"
>/var/tmp/dump2disk.cron 2>/var/tmp/dump2disk.err
#==========
# trandump
# These timings may need to be amended, e.g. for 24/7 running
#==========
#59,29 7-18 * * 1-6 chgrp blcmp /dev/console
#0 8-21 * * 1-5  /bin/su - ops -c "trandump >/var/tmp/trandump.log" >/dev/null
2>/var/tmp/trandump.err
#======================
# Full software backup
#======================
#0 11 * * 2 /bin/su - ops -c "full_softdump > /var/tmp/full_softdump.log" 1>/dev/null
2>/var/tmp/f
ull_softdump.err
# full_ufsdump of all filesystems. This script is normally only avaiable on "hardened"
```

```
systems
#30 1 * * 2 /usr/opt/blcmp/backup/bin/full_ufsdump.sh >/var/tmp/full_ufsdump.log
2>/var/tmp/full_u
fsdump.err
#==============
# update_stats
#==============
#30 8 * * 0 /bin/su - talis -c "update_stats prod_talis >/var/tmp/update_stats.log"
>/dev/null 2>/
var/tmp/update_stats.err
#===================
# Database checker
#===================
#00 10 * * 0 /bin/su - ops -c "checkstorage >/var/tmp/checkstorage.log" >/dev/null
2>/var/tmp/chec
kstorage.err
#===================
# archive_trandumps
#===================
#0 10 * * 1 /bin/su - ops -c "archive_trandumps >/var/tmp/archive_trandumps.log"
>/dev/null >/var/
tmp/archive_trandumps.err
#========================
# Save databases to disk
#========================
#0 5 * * 1 /bin/su - sybase -c "/usr/opt/blcmp/backup/bin/save_master"
>/var/tmp/save_master.cron
2>/var/tmp/save_master.err
#sybsystemprocs db save to disk
#30 5 * * 1 /bin/su - sybase -c "/usr/opt/blcmp/backup/bin/save_sybprocs"
>/var/tmp/save_sybprocs.
cron 2>/var/tmp/save_sybprocs.err
#=========================================================================
# Cleanup Jobs
#=========================================================================
#30 3 * * 2-6 /bin/find / -name core -exec rm -f {} \; >/dev/null 2>&1
#30 4 * * 2-6 /bin/find / -name nohup.out -exec rm -f {} \; >/dev/null 2>&1
# clear_logs script rotates frequently-used log files
# To specify logs to clear, edit /users/talis/admin/logs_to_clear
#30 0 * * 0  /users/talis/admin/clear_logs >/tmp/clear_logs.log 2>&1
#0 7 * * 2 /usr/bin/find /usr/opt/blcmp/talis/reports -mtime +30 -exec rm -f {} \;
>/dev/null 2>&1
# Additional - amend as required for filename(s) and number of days to keep
#49 3 * * 0 /usr/bin/find  /scratch/xxxxx.rep.* -mtime +7 -exec rm -f {} \; >/dev/null
2>&1
#49 3 * * 1 /usr/bin/find  /scratch/blcmp/mis/xxxxx.rep.* -mtime +28 -exec rm -f {} \;
>/dev/null2>&1
# Clear out listener logs for users of mobile - note that the directory may differ for
some sites
# 00 6 * * 1 /usr/bin/find  /scratch/blcmp/mobile/listener* -mtime +30 -exec rm -f {} \;
>/dev/null 2>&1
#=========================================================================
# Application-Related
#=========================================================================
#==============
# Compressions
#==============
# Drops out-of-date rows in WORK_UPDATE
#35 2 * * 2 /bin/su - talis -c "wku_compress.pl -e30" 1>/dev/null
2>/var/tmp/wku_compress.pl.err
# Drops out-of-date rows in ITEM_UPDATE
#40 2 * * 2 /bin/su - talis -c "itu_compress.pl -e30" 1>/dev/null
2>/var/tmp/itu_compress.pl.err
# Drops obsolete ILL request sequences
#45 2 * * 2 /bin/su - talis -c "irs_compress >
/usr/opt/blcmp/data/utils/irs_compress.out" 1>/dev/null 2>/var/tmp/irs_compress.err
#=================
# Loan compress
#================
#00 18 * * 6 /bin/su - talis -c "lo_compress.pl -q1 -u -dprod_talis -r/scratch -z >
/scratch/lo_compress.out"  >/var/tmp/lo_comp.run 2>&1
```

```
#==============
# Reservation jobs
#==============
#Set Reservation Flags for newly-acquired items (add in stock to active res's)
#25 23 * * 1 /bin/su - talis -c "res_add_itms -dprod_talis -tINTRAN" 1>/dev/null
2>/var/tmp/res_add_itms.err
#25 23 * * * /bin/su - talis -c "res_add_itms -dprod_talis -tINTRAN -b20000" 1>/dev/null
2>/var/tmp/res_add_itms.err
#30 0 * * 1-6 /bin/su - talis -c "resupdate.pl -dprod_talis -tALL
>/var/tmp/resupdate.log" 2>/var/tmp/resupdate.err
#=============
# Cataloguing
#=============
#0 22 * * 1-6 /bin/su - ops -c "unlocker >/var/tmp/unlocker.log" >/dev/null
2>/var/tmp/unlocker.err
#0 18 * * 1-5 /bin/su - ops -c "wku_update.pl -dprod_talis -u" >/dev/null 2>&1
# Logically delete works with no items
#00 18 * * 0-6  /bin/su - report -c "work_logdelete.pl -m20000 -r/users/report -
s/users/report -u -v > /scratch/logdelete.rep" 2>/scratch/logdelete.err
#==============
# Acquisitions
#==============
#00 2 * * 0 /bin/su - talis -c "fun_totals.pl -pfun_totals.param -u -v" >/dev/null
2>/var/tmp/fun_totals.err
#00 2 * * 0 /bin/su - talis -c "sup_totals.pl -psup_totals.param -u -v" >/dev/null
2>/var/tmp/sup_totals.err
# EDI template lines
#5 8 * * 1-5 /bin/su - talis -c "/usr/opt/blcmp/talis/bin/orr_ack_imp" 1>/dev/null
2>/var/tmp/orr_ack_imp.err
#30 02 * * 1-6 /bin/su - ops -c "/usr/opt/blcmp/data/impdir/XX_order_import" 1>/dev/null
2>/var/tmp/XX_order_import.err
#45 04 * * 1-6 /bin/su - ops -c "/usr/opt/blcmp/data/impdir/XX_invoice_import"
1>/dev/null 2>/var/tmp/XX_invoice_import.err
# Drops unwanted potential orders
#50 2 * * 2 /bin/su - talis -c "orr_pot_ords_del -m5000" 1>/dev/null
2>/var/tmp/orr_pot_ords_del.err
#==================
# Borrower scripts
#==================
#30 23 31 07 4 /bin/su - talis -c "/usr/opt/blcmp/talis/utils/bin/bor_anon_delete.pl -
pbor_anon_delete.param -dprod_talis -tLASTTRANS -v -u" 1>/var/tmp/bor_anon_delete.log
2>/var/tmp/bor_anon_delete.err
#30 23 08 08 5 /bin/su - talis -c "/usr/opt/blcmp/talis/utils/bin/bor_anon_delete.pl -
pbor_anon_delete.param -dprod_talis -tDELETED -u -v -z" 1>/var/tmp/bor_anon_delete.log
2>/var/tmp/bor_anon_delete.err
#30 2 * * 1-6 /bin/su - talis -c "/usr/opt/blcmp/talis/utils/bin/bor_block.pl -v"
>/dev/null 2>&1
# Template borrower import line. Amend for library code, barcode length (and hemis
argument if necessary)
# 30 2 * * 2 /bin/su - talis -c "borr_import -a xx 8 /scratch >
/scratch/borrower_import.out"  >/var/tmp/borrower_import.cron  2>&1
#======
# OPAC
#======
# Update the05 22 * *  catalogue
#0 23 * * 1-5 /bin/su - ops -c "update_daily_access_points
>/var/tmp/update_daily_access_points.log" >/dev/null
2>/var/tmp/update_daily_access_points.err
#0 1 * * 1-5 /bin/su - ops -c "/usr/opt/blcmp/talis/access_points/authority_ap
>/var/tmp/authority_ap.log" >/var/tmp/authority_ap.err 2>&1
# Update item-based Mindexes
# 2 /bin/su - ops -c "itu_update_wku.pl" 1>/dev/null 2>/var/tmp/itu_update_wku.pl.err
# Lines to stop/restart grabber overnight
# 05 22 * * 1-6 /bin/su - root -c "/usr/local/marcgrabber/bin/marcgrabberctl.sh stop"
>/var/tmp/grabberstop.log 2>/var/tmp/grabberstop.err
# 00 08 * * 1-6 /bin/su - root -c "/usr/local/marcgrabber/bin/marcgrabberctl.sh start"
>/var/tmp/grabberstart.log 2>/var/tmp/grabberstart.err
#====================================================================
# Talis Daemons
#====================================================================
# Stop/start daemons and rename reports
```

```
# 0 8 * * 1-5 /bin/su - ops -c "archive_daemon_reports
>/var/tmp/archive_daemon_reports.log" >/dev/null 2>/var/tmp/archive_daemon_reports.err
# Ensure daemons don't run overnight
# 0 20 * * 1-6 /bin/su - ops -c "dae_term work_exp_dae" >/dev/null 2>&1
# 0 20 * * 1-6 /bin/su - ops -c "dae_term authorisor_dae" >/dev/null 2>&1
# 0 20 * * 1-6 /bin/su - ops -c "dae_term ord_exp_dae"  >/dev/null 2>&1
# 0 20 * * 1-6 /bin/su - ops -c "dae_term work_merge_dae"  >/dev/null 2>&1
# Start sip2 processes
#00 07 * * * /bin/su - talis -c "/usr/opt/blcmp/sip2/bin/start_sip2_server" >/dev/null
2>&1
#=========================================================================
# Management information reports template commands
# These scripts are not part of the standard ship and would need to be set up at
installation if required
#=========================================================================
#00 22 * * 1 /bin/su - report -c "REPORTS_MON >/var/tmp/reports_mon.log" >/dev/null
2>/var/tmp/reports_mon.err
#00 22 * * 2 /bin/su - report -c "REPORTS_TUE >/var/tmp/reports_tue.log" >/dev/null
2>/var/tmp/reports_tue.err
#00 22 * * 3 /bin/su - report -c "REPORTS_WED >/var/tmp/reports_wed.log" >/dev/null
2>/var/tmp/reports_wed.err
#00 22 * * 4 /bin/su - report -c "REPORTS_THU >/var/tmp/reports_thu.log" >/dev/null
2>/var/tmp/reports_thu.err
#00 22 * * 5 /bin/su - report -c "REPORTS_FRI >/var/tmp/reports_fri.log" >/dev/null
2>/var/tmp/reports_fri.err
#00 22 * * 6 /bin/su - report -c "REPORTS_SAT >/var/tmp/reports_sat.log" >/dev/null
2>/var/tmp/reports_sat.err
#00 22 * * 0 /bin/su - report -c "REPORTS_SUN >/var/tmp/reports_sun.log" >/dev/null
2>/var/tmp/reports_sun.err
#==================
# RLB notification - use to export details of holdings for contribution to ILL union
catalogues
#==================
#30 23 * * 5 /bin/su - talis -c "rlb_non_isbn.pl -dprod_talis -prlb_non_isbn.param"
>/var/tmp/rlb_non_isbn.log 2>/var/tmp/rlb_non_isbn
#==================
# PLR (Public Lending Right) extract script
#==================
#40 05 * * 3 /bin/su - talis -c "/usr/opt/blcmp/talis/utils/bin/loa_plr_retrieve.pl -
b01/07/06 -e31/08/06 -ploa_plr_retrieve.param" 2>/var/tmp/loa_plr_retrieve.err
#=========================================================================
# ILL jobs
#=========================================================================
#15,45 9-17 * * 1-5 su - ill -c "/usr/opt/blcmp/talis/ill_manager/bin/ill_send ALL"
>/var/tmp/ill_send.cron 2>&1
#20,50 9-17 * * 1-5 su - ill -c "/usr/opt/blcmp/talis/ill_manager/bin/ill_mail DEFAULT"
>/var/tmp/ill_mail.cron 2>&1
#=========================================================================
# Request rotation
#=========================================================================
#30 10 * * 1-5 /bin/su - talis -c "res_item_rotate.pl" >/var/tmp/res_item_rotate.log 2>&1
#30 15 * * 1-5 /bin/su - talis -c "res_item_rotate.pl" >/var/tmp/res_item_rotate.log 2>&1
#
#=====================================================
# talislist scripts
#================================
# TalisList to Talis Sync script
#================================
#30 5 * * 1-6 /bin/su - talislist -c "date;cd
/usr/opt/blcmp/talislist/Sync;TalisListTalisSync.sh"
>/usr/opt/blcmp/talislist/Sync/Sync.rep 2> /usr/opt/blcmp/talislist/Sync/Sync.err
#================================
# Talislist - stop and start tomcat weekly
# Amend if $TOMCAT_HOME is not held under /usr/local
#================================
#00 6 * * 1-6 /bin/su - talislist -c "/usr/opt/blcmp/talislist/bin/restart.sh"
1>/var/tmp/talislist_restart.cron 2>&1
#================================
# trandump for talislist
#================================
#10 8 * * 1-6 /bin/su - ops -c "trandump prod_list >/var/tmp/trandump_prodlist.log"
```

```
>/dev/null 2>&1
#
#============================
# Customer-specific scripts to be entered below
```

Editing the cron

In order to make additions or changes to scheduled tasks, the root cron file must be edited and loaded into memory. Editing can be carried out using a standard text editor such as vi.

A standard root cron file is provided with the LMS system. This resides in the **/var/spool/cron/crontabs** directory. You should tailor this file to meet your own requirements.

To edit the cron file

1. Log into UNIX **root** account, using the **su** command and the correct password.
2. Make a backup of the cron file:

   cd /var/spool/cron/crontabs
   cp root root.[date]

3. Edit the cron file:

   crontab -e

   This command will invoke the UNIX vi text editor. Make your required changes then exit the editor using the command **:wq!** at the command line prompt. This will update the information held by the cron process.

The 'at' command

at is a UNIX command which allows a script to be executed once at a specified time and/or date. This is an alternative to running one-off jobs in the cron. In using it, you do not need to remember to remove the cron line once it is no longer required.

Scheduling a job

The following syntax used at the UNIX command prompt, allows a job to be scheduled:

at {time-to-run) {date-to-run)

For example entering

at 0800 Feb 23

will produce the **at>** prompt, from where you should enter the command to be run at this time. Note that:

- If no date is specified, today is assumed.
- If no date is specified and the time is less than now, tomorrow is assumed.
- If no month is specified, the current month is assumed.
- If a month earlier than the current month is given, next year is assumed.

Using **CTRL-D** will exit the prompt and offer you a job reference number. When an at job has run, a message will be written to the UNIX mailbox of the account name running the job.

Checking and cancelling jobs

Your queued at jobs can be viewed using the **atq** command at the UNIX prompt. This will reveal the job reference number, owners and the date and time that jobs are due to run.

To cancel a job use the command:

atrm {job reference}

at the UNIX prompt. You must have appropriate permissions to do this. The command:

atrm –a

will cancel all jobs for your current UNIX account.

Note
When using these commands, the root user account has the privilege to view or cancel all at jobs.

Scripts

Introduction to scripts

This section contains general information about running scripts. Please read this section before consulting the documentation for a particular script.

Scripts are files on the LMS machine that can contain a sequence of UNIX commands or programs. Typing the name of the script in at the UNIX prompt will cause the contained commands to be executed in sequence. They are used to run regular or repetitive tasks (usually against the **prod_talis** database) and can be configured to run automatically.

The scripts have been organised by function, and comprise:

- **Database management -** Scripts concerning database backup and restoration.
- **Utilities -** General utility scripts covering a range of business areas.
- **Perl MIS reports -** Reports written in Perl. For more information refer to the topic 'Introduction to Perl MIS reports'
- **Import work -** Scripts concerning the import of work data into the local database

For completion, **ILL** and **SelfServ** scripts are also referenced.

Executing scripts

- To execute most scripts you must either be in the directory that contains the script file, or a path to the file must have been set up as an environment variable.
- In order to run *most* scripts you must be logged in as **talis**. If a specific login or location is required to run a particular script, this will be stated within the script details.
- Where tasks might have a detrimental affect on performance, scripts should be run at quiet times or when there is no on-line activity.
- The following example shows how the syntax for the script **archive_trandumps** should be applied.

Example

The **archive_trandumps** script transfers trandump files from the **/scratch/prod_talis** directory to tape. The syntax for the script is as follows.

archive_trandumps <database> <days to retain> |tee<filename>

Where:

- the <**database**> argument specifies the database whose log files you wish to transfer for example prod_talis).
- the <**days to retain**> argument specifies the number of days files to keep
- the <**filename>** argument is a print file which should be printed after completion and stored with the tape

So to keep up transaction logs of **prod_talis** for the most recent three day period, you would enter the command:

archive_trandumps prod_talis 3 |tee archivelist

Note

The <**database**> argument is a standard argument. Since standard arguments appear in a number scripts, they are described in the following topic. Note that all scripts that contain standard arguments contain a link to the "Standard arguments" topic.

Standard arguments used in scripts

Scripts are usually run with series of arguments that modify how the script is executed. The arguments described in this section are common in a large number of Capita LMS scripts.

Where an argument performs a different function to the one described here, the details will be stated within the script details.

Argument	Description
-a	The -a argument allows you to append to the output file produced by the script, rather than overwriting it.
-h	The -h argument displays brief help on-screen about the arguments supported by the script. Note that the help text will also be displayed if an invalid command is given.
-d	The -d argument allows you to specify the database against which the report is to be run. If this argument is not given, then the default is defined by the TAL_DEFAULT_DBNAME environment variable. This will default to prod_talis if it has not been set already.
-i	The -i argument allows you specify the name of an input file to be used by the script.
-o	The -o argument may be used to specify the name of the output file generated by the script.
-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.
-s	The -s argument may be used to specify the data directory where the process will write its output file. If an output file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the default directory specified in the help.
-u	The -u argument instructs the script to update the database. If this argument is not specified, the script will run in report mode , and the database will not be updated. In report mode, the script merely identifies the updates which would occur; that is a report file is generated but the database is not updated.
-v	The -v argument specifies that a verbose report file should be generated. Verbose report files contain more detailed information than standard reports.
report	This is the Perl report generator, which runs against a set of pre-defined parameters for all MIS reports.

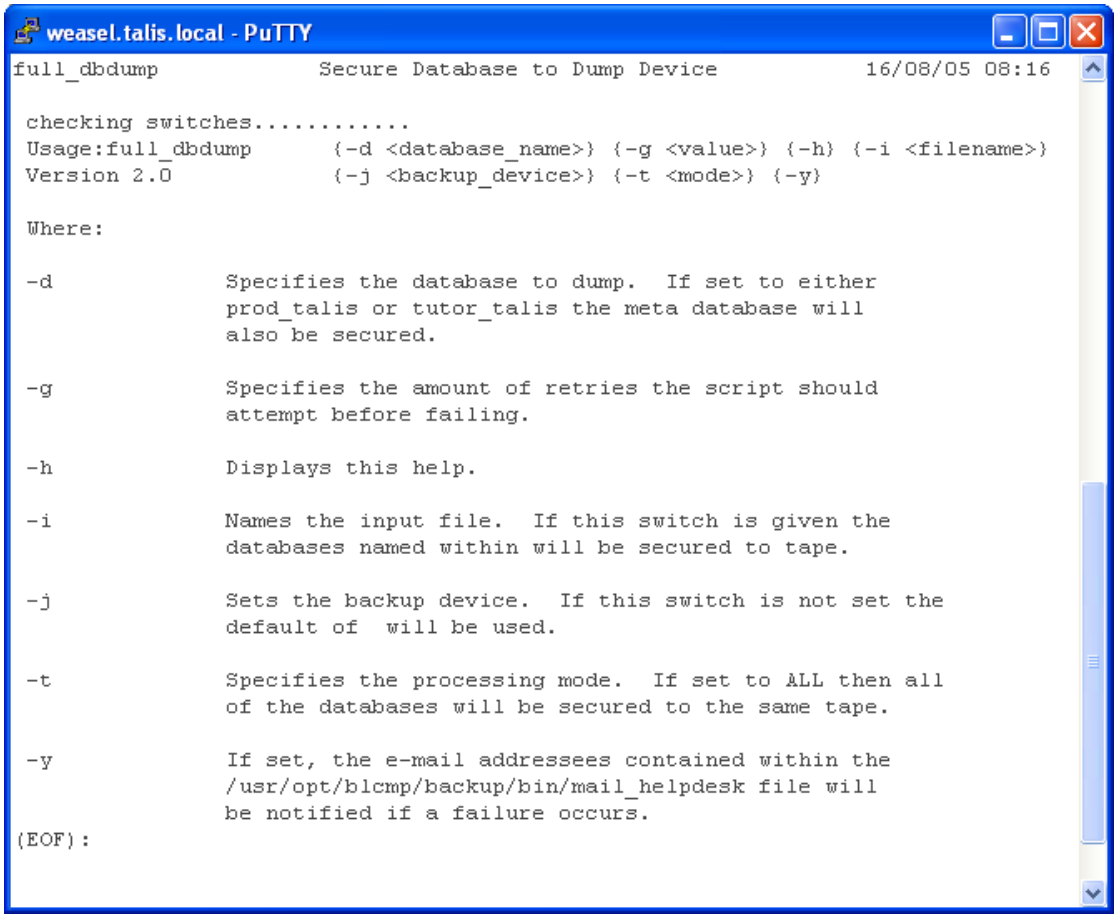
Getting help from the UNIX prompt

Most scripts contain 'help text' that can be displayed within the UNIX prompt. The help text describes the arguments supported by the script, and is therefore a valuable reference tool if documentation is not to hand.

To display the help text, simply enter the name of the script followed by **-h** argument. For example, to check the arguments supported by the script **full_dbdump**, you would navigate to the **/usr/opt/blcmp/backup/bin** directory and type the following command:

full_dbdump -h

The screen displays the supported arguments, as shown below:



Database Management

archive_trandumps

When free disk space runs low, a warning message will be displayed on the console. If this happens, the script **archive_trandumps** should be executed. The **archive_trandumps** script transfers the trandump files from the **/scratch/prod_talis** directory to tape.

Usage

The script supports the following arguments:

archive_trandumps <database> <days to retain> |tee<filename>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
<days to retain>	This argument specifies the number of days files to keep.
<filename>	Where the **filename** parameter is a print file which should be printed after completion and stored with the tape. For example, to keep up transaction logs of **prod_talis** for the most recent three day period, you would enter the command: **archive_trandumps prod_talis 3 |tee archivelist**

Notes

- The command can be entered without any values to accept the default values. Enter the command **archive_trandumps** to retain dump files less than one day from the **prod_talis** database.
- This script is usually executed manually.

checkalloc

The **checkdb** and **checkalloc** scripts check the internal integrity of the Sybase database, and should be run regularly when Alto is not in use. The scripts should be scheduled using the cron as the user **ops**.

Usage

Example crontab lines for automating "checkdb" and "checkalloc" are shown below:

```
5 * * 0 /bin/su - ops -c "checkalloc  /var/tmp/checkalloc.log"
2/var/tmp/checkalloc.err
5 * * 0 /bin/su - ops -c "checkalloc  /var/tmp/checkalloc.run" 2/var/tmp/checkalloc
```

The first line executes checkdb at 12:30AM on Sunday mornings.

The second line executes checkalloc at 5:00AM on Sunday mornings.

Notes

- On older systems these scripts can take many hours to run. Sybase 12 offers an alternate script called checkstorage which is faster and more efficient.
- The **checkdb** and **checkalloc** outputs must be checked after each run. Use the **grep** command to search for Sybase error messages. Report any error messages to Capita Support.

checkdb

The **checkdb** and **checkalloc** scripts check the internal integrity of the Sybase database, and should be run regularly when Alto is not in use. The scripts should be scheduled using the cron as the user **ops**.

Usage

Example cron tab lines for automating "checkdb" and "checkalloc" are shown below:

```
30 0 * * 0 /bin/su - ops -c "checkdb /var/tmp/checkdb.run" 2/var/tmp/checkdb.err
5 * * 0 /bin/su - ops -c "checkalloc  /var/tmp/checkalloc.run" 2/var/tmp/checkalloc
```

The first line executes checkdb at 12:30AM on Sunday mornings.

The second line executes checkalloc at 5:00AM on Sunday mornings.

Notes

- On older systems these scripts can take many hours to run. Sybase 12 offers an alternate script called checkstorage which is faster and more efficient.
- The **checkdb** and **checkalloc** outputs must be checked after each run. Use the **grep** command to search for Sybase error messages. For example:
  grep Msg /var/tmp/checkalloc.log
- Report any error messages to Capita Support.

checkstorage

Like the checkdb and checkalloc scripts, checkstorage checks the internal integrity of the Sybase database, and should be run weekly.

Usage

Log on as **ops** and enter the command:

checkstorage

The command takes no switches and should only br run against **prod_talis** which it treats as the default database. To check the consistency of other databases, use the checkdb and checkalloc scripts.

Notes

Checkstorage creates an output file called **checkstorage.md.log** which is housed in the **/var/tmp not /var/temp** directory. (where **m** is month and **d** is date).

data_backup

BRS™ server software must have been configured and started for end-users to be able to perform a Z39.50 search against the Talis target. Use the **data_backup** script to back up the entire file system under which the BRS server software resides. The **data_backup** backup script calls the UNIX file system dump utility **ufsdump** (for more information about **ufsdump**, refer to the UNIX man pages).

The script is normally scheduled using the cron. Backup should be run daily if you are updating the database daily.

Usage

Log in as **ops** and enter the following command:

data_backup -o<**backup_device**> **-p**<**filename**> **-r**<**directory**> **-x -h**

The arguments for this script are described in the following table.

Argument	Description
-o	This mandatory argument specifies the dump device. This is normally /dev/rmt/0. If this argument is not specified the script will display help text and terminate.
-p	This optional argument is used to give the name of parameter file to be used. The parameter file should contain a list of file systems to be backed-up. If none is specified, the default parameter file data_backup.param will be used. If a nonexistent parameter file is specified, the script will terminate stating the problem.
-r	This names the report directory where the process will create its report file. The default, if this argument is not given, is defined by the $TAL_REP_DIR environment variable. If not already set, this defaults to the data directory $BLCMP_HOME/data/utils. The report file takes the name "data_backup.rep".
-x	If this argument is not included, the script verifies the contents of the media against the source file. If this argument is included, the contents of the media are not verified.

Parameter file

The default parameter file, **data_backup.param** contains a list of file systems. If the parameter file contains no file systems the script will terminate stating that no valid file systems have been specified. In most cases only one file system will be specified.

data_restore

The **data_restore** script restores the BRS server software that has been backed up using the **data_backup**. The script is configured to restore the entire file system. This script is run by the root user. It should only be used when the Advanced OPAC system is offline. The **data_restore** restore script calls the UNIX file system restore utility **ufsrestore** (for more information about **ufsrestore**, refer to the UNIX man pages).

Usage

Log in as root and enter the following command:

data_restore -o<**backup_device**> **-p**<**filename**> **-r**<**directory**> **-j**

The arguments for this script are described in the following table.

Argument	Description
-o	This mandatory argument specifies the dump device. This is normally /dev/rmt/0. If this argument is not specified the script will display help text and terminate.
-p	This optional argument is used to give the name of parameter file to be used. The parameter file should contain a list of file systems to be restored. If none is specified, the default parameter file data_backup.param will be used. If a nonexistent parameter file is specified, the script will terminate stating the problem.
-r	This names the report directory where the process will create its report file. The default, if this argument is not given, is defined by the $TAL_REP_DIR environment variable. If not already set, this defaults to the data directory $BLCMP_HOME/data/utils. The report file takes the name "data_restore.rep".
-j	If this is include, existing files within the the file-system being restored will be deleted as part of the restoration process. Note that any files that have been created since the last backup will be deleted; do not use this swicth if these files need to be retained.

Parameter file

The default parameter file, **data_backup.param** contains a list of file systems. If the parameter file contains no file systems the script will terminate stating that no valid file systems have been specified. In most cases only one file system will be specified.

full_dbdump

Regular backups of the database need to be taken so that a full record of transactions may be maintained and restored in the event of a system crash. The **full_dbdump** script allows you to back up (or 'dump') the **prod_talis** database (and other databases) to tape cartridges.

The script is normally scheduled using the cron.

Usage

Log in as **ops** and enter the following command:

full_dbdump -d<database> -g<value> -h -i<filename> -j<backup_device> -tALL -y

The arguments for this script are described in the following table.

Argument	Description
-d	Specifies the database to dump. If set to either **prod_talis** or **tutor_talis** the meta database will also be secured.
-g	This optional argument is used to give the name of parameter file to be used. The parameter file should contain a list of file systems to be restored. If none is specified, the default parameter file data_backup.param will be used. If a nonexistent parameter file is specified, the script will terminate stating the problem.
-i	If this is included, existing files within the file-system being restored will be deleted as part of the restoration process. Note that any files that have been created since the last backup will be deleted; do not use this swicth if these files need to be retained.
-j	Defines the backup device. If not set then the default of $BACKUP_DEV is used.

-t	Sets the processing mode. With the current version of the script (version 2.0) only the ALL mode is supported. When the script is called in ALL mode each database present will be secured to the same tape in order of db_id.
-y	When the argument is set, an email will be sent (to the addressees contained within the mail_helpdesk file) if the script encounters a fatal error condition. The email informs the support of the problem encountered.

Notes

- Only one execution mode may be selected (that is, argument **-d** or **-i** or **-t**). If multiple modes have been set then the script will not be run.
- If the **-i** argument is set, the named list file is validated to ensure that it exists and is readable. If the **-t** argument is set, a validation check ensures that the **ALL** mode is given.
- The script can also be run without any command line switches and will default to using the **-d** mode with the target database defined by the **$BACKUP_DBNAME** environment variable. This functional inclusion was made for backwards compatibility issues.

The list file

The list file specified with the **-i** argument is required to name the databases that will be dumped when this mode is used. The order which they appear in the file is the order that they will be secured to the tape. Each named database should occupy one line in the file. For example, a list file containing:

- prod_talis
- prod_meta
- master

would secure the **prod_talis**, **prod_meta** and **master** databases to the tape in the stated order. Each entry in the list file is validated to ensure that the database exists; if it does not then it is skipped, otherwise it is backed up to the tape.

full_dbrestore

The **full_dbrestore** script allows you restore the **prod_talis** database that has been backed up using the full_dbdump script.

Usage

To run the script, ensure the backup tape is in the device is in place , log on as **ops**, and enter the following command:

full_dbrestore <dbname>

For example:

full_dbrestore prod_talis

or

full_dbrestore tutor_talis

Notes

- Ensure that the database reported by the product is **prod_talis**. If anything other than this database is named for restoration, immediately contact Capita Support. Failing to do so may jeopardise future restores.
- The message "Restore Completed" signifies the completion of the process.

full_softdump

The **full_softdump** script allows you to back up the entire software system, which includes all the Capita LMS, UNIX and user files.

Usage

To run the script, ensure the backup tape is in the device is in place , log in as **ops**, and enter the following command:

full_softdump

Notes

- Everything under the root directory (/)will be secured. That is, no files are excluded from the backup.
- You will be prompted to enter additional tapes if required.
- A message confirming the date and time of the backup signifies the completion of the process.

full_softrestore

The **full_softrestore** script allows you to restore the entire software system, if required, or selected files. It should be run once at least once a week.

Usage

To recover the entire software system, ensure the backup tape is in the device is in place , log in as **ops**, and enter the following command:

full_softrestore

If used without arguments, the recover command will attempt to restore all files from the software backup tape.

To recover specific files, add the full path and name of the file to be recovered, omitting the leading slash. Separate multiple files with spaces. For example, to recover the following files:

usr/opt/blcmp/talis/tmp/listfile
tmp/tempdata
etc/hosts.equiv

enter the command:

full_softrestore usr/opt/blcmp/talis/tmp/listfile tmp/tempdata etc/hosts.equiv

kill_opac

The **kill_opac** script is used to close down all OPAC processes.

Usage

Log in as root and enter the following command:

kill_opac

kill_process

The **kill_process** script is used to close down all LMS processes (except executing processes).

The script is normally scheduled using the cron.

Usage

Log in as root and enter the following command:

kill_opac

save_master

The **save_master** script improves the recoverability of the Sybase server by copying the 'master' database. This small database occupies less than 10MB and contains critical Sybase server configuration information.

The script is normally scheduled using the cron as the **ops** user.

Usage

Example crontab lines for automating **save_master** are shown below:

```
15 9 * * 5 /bin/su - ops -c "/usr/opt/blcmp/backup/bin/save_master"
>/var/tmp/save_master.cron 2>&1
30 5 * * 2 /bin/su - ops -c "/usr/opt/blcmp/backup/bin/save_sybprocs"
>/var/tmp/save_sybprocs.cron 2>&1
```

Notes

- The script takes less than 5 minutes to dump the database and requires less than 80MB of disk space. It has no impact on other jobs being run on the server.
- The copied file is called **masterdb** and is written to **/scratch/master**.

save_sybprocs

sybsystemprocs is a Sybase database where the system stored procedures are kept. The **save_sybrocs** script saves the sybsystemprocs database to the **/scratch** directory. This file is required in certain restore situations.

The script is normally scheduled using the cron.

Usage

Log on as **ops** and enter the following command:

save_sybrocs

top5

The **top5** script identifies the top5 largest tables in the database and indicates if any of the indexes on these tables cannot be re-indexed. In doing so, it checks that there is sufficient space to run the loan.index job. If there is insufficient space, a warning message is displayed and you should contact Capita Support.

Usage

Log on as **ops** and enter the following command:

top5

trandump

The trandump script copies the contents of the transaction log to the disk space in the **/scratch/prod_talis** directory. It then clears the transaction log area of the database so that Alto can accept further transactions.

The script is normally scheduled using the cron.

Usage

To perform a manual trandump, log on as **ops** and enter the following command:

trandump

Notes

The outcome of trandumps should be checked each day. A log of all trandumps is kept on the server machine in **/scratch/prod_talis/dump_log**.

update_stats

The **update_stats** script ensures that the information contained in the statistics pages of the tables is kept up to date. This prevents long response times and consequent degradation of system performance. It is strongly recommended that the script is run through the cron on a weekly basis, with the output directed into a log file. It should take less than two hours to complete.

Usage

An example crontab line for automating **update_stats** is shown below:

```
30 8 * * 0 /bin/su - talis -c "update_stats prod_talis >/var/tmp/update_stats.log"
2>/var/tmp/update_stats.err
```

Table names can be entered as arguments to the script if desired. If no table name is entered, statistics pages will be updated for all tables (including those which have not changed or which have been updated to only a minimum extent). This may be what is required; if so, table names need not be entered, and the command will be:

update_stats prod_talis

Several tables may be specified as arguments to the script if required; as follows:

update_stats prod_talis <table1> <table2> <table>

For example:

update_stats prod_talis BORROWER

If also re-directing output to a log file you might type in:

update_stats prod_talis > /scratch/update_stats.report

Note:

This script is held in **/usr/opt/blcmp/talis/bin** .

Utilities

Item usage MIS reports

Item usage scripts

Prism 2 includes item usage statistics to help Library Staff make decisions about Acquisitions and Collection Management. The item usage scripts can be found in **$TALIS_HOME/utils/bin**. To run these scripts log in as **talis**.

Warning

For Libraries wishing to implement item usage scripts, consultancy must be arranged by contacting Capita Support.

Item usage statistics: period for inclusion

A period is defined with a name and start/end dates using the script ite_usg_add_period.pl . Loan statistics are accumulated for the range of dates between the start/end dates. Any number of periods can be defined with up to 12 displaying on the Copies History page. The current period has a name and start date only.

Item usage statistics: historical data removal / consolidation

An the end of each period of loan activity, the statistics for the oldest period may be removed or merged, using the ite_usg_delete_period.pl and ite_usg_merge_periods.pl scripts respectively. The new current period has its count set to zero. The state of the Item at the time the update is run is also recorded.

A count of all issues and renewals since the Item became available is recorded and is updated as required. Each time an Item is loaned (issued/renewed), the loan counts for the current period and the total since the Item first became available are updated.

ite_usg_add_period.pl

The **ite_usg_add_period.pl** script creates a new statistics period with a specified name and date range. It calculates the Item Loan counts for each Item over the date range specified.

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on as **talis** and enter the following command:

ite_usg_add_period.pl –b<beginning of period> -d<database> –e<end of period> -h –n<period name> -q<existing period name> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies the beginning of the date range for the new usage statistics period. This can be specified in one of three ways.

	• It can be a literal date in the format DD/MM/YY or DD/MM/YYYY. • It can be specified as a positive or negative number, in which case it represents an offset from the current date (for example, use -7 to specify 7 days before today). • Finally it can be specified as "last", in which case it will take the first day after the end date of the most recent existing period.
-e	This mandatory argument specifies the end of the date range for the new usage statistics period. This can be specified in one of two ways. • It can be a literal date in the format DD/MM/YY or DD/MM/YYYY. • Alternatively it can be specified as a positive or negative number, in which case it represents an offset from the current date (for example, use -7 to specify 7 days before today).
-n	This mandatory argument specifies the Name for the new period.
-q	This optional argument specifies the name of an existing period which will be used to obtain the states of the Items. If this switch is given, then the Item states will be copied from the existing period. Otherwise Item states will be derived using the rules defined in the ITEM_STATE_MAP database table.

Notes

When processing is complete, the total Items rows processed, total period statistics rows successfully created, and total period statistics rows failed to be created are reported.

ite_usg_upd_current.pl

The **ite_usg_upd_current.pl** script re-calculates the loan count for the current period. The script can optionally amend the start date and name assigned to the current period.

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on **talis** and enter the following command:

ite_usg_upd_current.pl -b<beginning of period> -d<database> -h –n<period name> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies a new begin date for the current usage statistics period. This can be specified in one of four ways: • It can be a literal date in the format DD/MM/YY or DD/MM/YYYY. • It can be specified as a positive or negative number, in which case it represents and offset from the current date (for example, use –7 to specify 7 days before today) • It can be specified as the name of an existing period, in which case it takes the first day after the end date of the named period. • Finally it can be specified as "last", in which case it takes the first day after the end date of the most recent existing period.
-n	This mandatory argument specifies a new name for the current usage statistics period.

Notes

When processing is complete, the total rows processed, total period statistics successfully created and total period statistics rows failed to be created are reported.

ite_usg_delete_period.pl

The **ite_usg_delete_period.pl** script removes from the database a complete set of Item usage statistics corresponding to the period name given.

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on **talis** and enter the following command:

ite_usg_delete_period.pl -d<database> -h –n<period name> -q<period id> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument specifies the name of the period to be deleted. You must include either the -n or the -q argument (they are mutually exclusive).
-q	This argument specifies the PERIOD_ID of the period to be deleted. You must include either the -n or the -q argument (they are mutually exclusive).

Notes

When processing is complete, the total **ITEM_PERIOD_STATS** rows successfully deleted and total **ITEM_PERIOD_STATS** rows failed to be deleted are reported. The name of any periods which failed to be deleted is also reported.

ite_usg_merge_periods.pl

The **ite_usg_merge_periods.pl** script creates a new set of statistics by merging a specified set of existing statistics. Existing loan counts for all "old periods" specified are summed and the results stored in the new period.

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on **talis** and enter the following command:

ite_usg_merge_periods.pl -d<database> -h –n<period name> -q<existing periods> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the name of the new period.
-q	This optional switch specifies a list of existing periods. The statistics from these periods are merged to create the new period. Each period name in the list should be separated by a comma with no spaces between the names: for example, "–qperiod1,period2,period3".

Notes

When processing is complete, the total existing period statistics processed, total merge period statistics rows successfully created and total merged period statistics rows failed to be created are reported.

ite_usg_set_display.pl

The **ite_usg_set_display.pl** script defines the order in which the columns are displayed in the Prism 2 "Copies History" page.

For Libraries wishing to implement this, consultancy must be arranged bycontacting Capita Support.

Usage

Log on as **talis** and enter the following command:

ite_usg_set_display.pl -d<database> -h -r<report directory> column<column> <column>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
column	This mandatory switch specifies the order of the columns to be displayed and the text of each column heading. Each column is specified in the format: `<column_name>{=<heading>}` The column can be "Barcode", "Location", "Type", "Sequence", "Available", "Total" or it can be the name of an existing statistics period. A maximum of 12 period names can be specified in this way. The optional heading assignment can be used to specify the text of the column heading. If not specified, the heading defaults to the column name.

For example:

ite_usg_set_display.pl 1996=96/97 1997=97/98 1998=98/99 Aut98 Spr99 Sum99 barcode=ItemNo. sequence type

would set-up 9 display column headings as follows:

96/97	97/98	98/99	Aut98	Spr99	Sum99	ItemNo.	Sequence	Type

When processing is complete, the total existing display column setting deleted, the total new display column settings failed to be created and total new display column settings created are reported.

ite_usg_set_available.pl

The **ite_usg_set_available.pl** script sets the date which displays as the date the Item was available. It can be set to DATE_CREATED, i.e. setting the available date to the Item's creation date, or to STATUS_CHANGED (i.e. setting the available date to the date the Item Status was changed between two specified statuses).

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on **talis** and enter the following command:

ite_usg_set_available.pl -b<created after> -d<database> -e<created after> -h –n<status changes> -r<report directory> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-b	This optional argument is valid only in conjunction with "-tDATE_CREATED". If this option is used then only those Items created on or after the specified date are processed. This can be specified in one of two ways: • It can be a literal date in the format DD/MM/YY or DD/MM/YYYY. • Alternatively it can be specified as a positive or negative number, in which case it represents and offset from the current date (for example, use –7 to process only those Items created in the last 7 days).
-e	This optional argument is valid only in conjunction with "–tDATE_CREATED". If this option is used then only those Items created on or before the specified date are processed. This can be specified in one of two ways: • It can be a literal date in the format DD/MM/YY or DD/MM/YYYY. • Alternatively it can be specified as a positive or negative number, in which case it represents and offset from the current date (for example, use –8 to process only those Items created more than seven days ago).
-n	This optional argument is valid only in conjunction with "–tSTATUS_CHANGED". It specifies the pairs of Status changes that will trigger the process to update the available date of the Item. Each pair should be specified as the Code of the old Status and the Code of the new Status separated by a hyphen. For example, "–nREC-IS" specifies that an Item should be processed if its Status has changed from "REC" to "IS". Multiple pairs of statuses should be separated by commas without spaces. A question mark ("?") can be used as a wildcard character to represent any Status, for example "–n?-IS" specifies that an Item should be processed if its Status has changed to "IS" from any other Status.
-t	This optional switch specifies the type of processing to perform. The valid options are "–tDATE_CREATED" and "–tSTATUS_CHANGED". • If "–tDATE_CREATED" is used then the process uses the date each Item was created to set the available date for each Item. • If "–tSTATUS_CHANGED" is used, then the process looks for Items whose Status has changed between pairs of specified values since the process was last run in this mode and sets the available date for such Items to the date on which the Status change took place.

Notes

When processing is complete, the total Item rows processed, the total main statistical rows successfully updated and the total main statistics rows failed to be updated are reported.

ite_usg_total_loans.pl

The **ite_usg_total_loans** script sets a historical loan count for each Item.

For Libraries wishing to implement this, consultancy must be arranged by contacting Capita Support.

Usage

Log on **talis** and enter the following command:

ite_usg_total_loans.pl -b<begin at> -d<database> -e<end at> -h -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory switch specifies the beginning of a range of Items to be processed. This can be specified either as an ITEM_ID or a BARCODE. Precede the value with "code=" to specify a barcode or "id=" to specify an ITEM_ID (for example, "-bcode=18181813" specifies a barcode or "-bid=61423"

	specifies an ITEM_ID). If the value is preceded by neither an id or code, an ITEM_ID is assumed. If used in conjunction with "-e", the same type must be used for both switches.
-e	This optional switch specifies the end of a range of Items to be processed. This can be specified as either an ITEM_ID or a BARCODE. Precede the value with "code=" to specify a barcode and "id=" to specify an ITEM_ID (for example, "-bcode=18181813" specifies a barcode or "-bid=61423" specifies an ITEM_ID). If the value is not preceded by id or code, an ITEM_ID is assumed. If used in conjunction with "-b", the same type must be used for both switches.

Notes

When processing is complete, the total Item rows processed, the total main statistical rows successfully updated and the total main statistics rows failed to be updated are reported.

Utilities

assign_rtn_pln.pl

The **assign_rtn_pln** script facilitates the assignment of groups items to rotation plans, and allows for a plan to be set against items in bulk.

Usage

Log on as **ops** and enter the following command:

assign_rtn_pln.pl -d<database> -h -n<rotation plan> -p<filename> -r<report directory> -s<data directory> -u -v.

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	Specifies the rotation plan that the selected items will be assigned to. This argument is compulsory and will only accept a single rotation plan code.
-p	The -p argument allows a batch of items for assignment to a rotation plan to be selected. Refer to the section below for more details.

Parameter file

A parameter file allows a batch of items for assignment to a rotation plan to be selected. A sample parameter file called **assign_rtn_pln.pa.default** is supplied in the directory **/usr/opt/blcmp/data/utils**. An example file is shown below.

```
# assign_rtn_pln.pa.default
#
# Sample Parameter file for assign_rtn_pln.pl
#
#CLASS_NUMBER=
#ITEM_TYPE
#LOCATION=
#BEGIN_RECEIPT_DATE=
#END_RECEIPT_DATE=
#ROTATION_PLAN=
#SEQUENC_CODE=
#SIZE_CODE=
```

Accepted parameters are shown in the following table.

Parameter	Description

CLASS_NUMBER	Specifies the rotation plan that the selected items will be assigned to. This argument is compulsory and will only accept a single rotation plan code.
ITEM_TYPE	This parameter accepts item type codes, with any items of the type specified being selected for processing. You can separate multiple item type codes with a comma. For example, ITEM_TYPE=NORM,REF would include normal and reference item types.
LOCATION	This parameter accepts location codes, with any items located at the given site(s) being selected for processing. You can separate multiple location codes with a comma.
BEGIN_RECEIPT_DATE	This parameter accepts a date in the format DD/MM/YYYY. Any items which have a receipt date equal to or less than the specified date will be included in the processing.
END_RECEIPT_DATE	This parameter accepts a date in the format DD/MM/YYYY. Any items which have a receipt date equal to or greater than the specified date will be included in the processing.
ROTATION_PLAN	This parameter accepts item rotation plan codes, with any items that belong to the given plan(s) being selected for processing. You can separate multiple rotation plan codes with a comma (for example, ROTATION_PLAN=SR1,SR2).
SEQUENCE_CODE	This parameter accepts sequence codes, with any items of the sequences specified being selected for processing. You can separate multiple sequence codes with a comma (for example, SEQUENCE_CODE=AF,ANF).
SIZE_CODE	This parameter accepts size codes, with any items of the sizes specified being selected for processing. You can separate multiple size codes with a comma (for example, SIZE_CODE=OS,MINI).
SUFFIX	This parameter accepts suffixes, with any items with a matching suffix specified being selected for processing. You can separate multiple suffixes with a comma (for example, SIZE_CODE=ABC,ACL).

The script will only select items that match the selection criteria defined by the parameters, and assign them to the rotation plan specified.

- This action will insert a row for each item and plan into the **ITEM_ROTATION_LINK** table with the rotation period measured from the point that this update takes place.
- Any current active row in the **ITEM_ROTATION_LINK** table for the item will be updated so that the CURRENT flag is changed to FALSE (that is, if a selected item is currently assigned to a different rotation plan, it will be removed from that plan and assigned to the new one).
- If the **ITEM.ACTIVE_SITE_ID** of the item is present in the rotation pattern then the **ITEM_ROTATION_LINK.NEXT_SITE** value will be set to the next available in the pattern. If not present then the first site in the pattern will be used.

Notes

- The script will create a report file named (by default) **assign_rtn_pln.rep** in the directory specified by the **-r** argument. If a report already exists with the same name, it will be renamed with a date and time extension.

authority_build

The **authority_build** script runs the subordinate script **authority_build.pl**. It performs a number of functions:

- It extracts and retains existing Authority data for Authority Types which are not being rebuilt.
- It retains existing references, notes and code data for Authorities of the Type(s) specified (if required).
- It generates BACK_INDEX keys for any "See" references retained.
- It creates the command line for the second script. Effectively, the authority_load.pl script is created automatically by running authority_build.
- It runs the Authorisor daemon (**authorisor_dae**) in batch mode to extract data from **WORK_SUBFIELD** and create a data file to be used by the authority_load script.

The script is normally scheduled using the cron.

Usage

Log on as the operator usually used for running the Authorisor daemon and enter the following command:

authority_build -d<database> -h -l<lock database> -n<type codes> -r<report directory> -s<data directory>
-t<"overwrite" or **"retain"> -u**

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-l	This optional argument may be used to specify the name of the lock database name. If not specified, it defaults to the value of the TAL_META_DBNAME environment variable, and, if that is not set, to prod_meta.
-n	This mandatory argument specifies the Authority Type(s) to be built. The Authority Type Code is required. If more than one Type is to be built, the codes should be separated by commas. If all Authority Types are to be built in one run, the value "-n ALL_TYPES" may be used. If the Types specified already exist, the "-t" argument must also be given.
-t	This optional argument specifies how existing authorities of the Authority Type(s) specified are to be handled. There are two options; "overwrite" which deletes all existing data for the Authority Type(s) specified, or "retain" which saves existing references, notes, and codes data and relinks them to the new Authority data. If the argument is not given, the existence of Authorities of the Type specified by the "-n" argument causes the script to terminate. **-toverwrite** This deletes all existing Authorities of the Type specified, and performs a complete Authority build based on WORK_SUBFIELD. **-tretain** This retains references, notes and code data for Authorities of the Authority Type specified, and re-links them after building Authorities from WORK_SUBFIELD data.

Notes

A report file, **authority_build.rep**, is created. This indicates the times the script started and stopped, the command line used, and reports of the interim stages. These interim stage reports comprise a message indicating the stage being started, a progress report as each 10000 rows processed, and counts of the number of rows processed at each stage.

A list of the files in the data directory available for use by the **authority_load** script appears at the end of the report. The **authorisor_dae** daemon generates the standard **.err**, **.log** and **.rep** files.

authority_load

The **authority_load** script is created by the authority_build script. The script estimates the amount of space required for running **authority_load**, based on the size of the files produced by **authority_build**. It then checks the amount of space available, and reports the figures to the report file.

authority_load runs a subordinate script **authority_load.pl** with switches derived from values used for running authority_build. The possibility of operator error (i.e. specifying different, incompatible switches for the two scripts) has been reduced by constructing the command line for the second script automatically. As there is a possibility that, for example, environment variables may be re-set in between the running of the two scripts, most of the (hidden) mandatory switches for **authority_load** take the values used by authority_build without operator intervention.

Usage

Log on as the operator used for running the **authority_build** script and enter the following command:

authority_load -u

The **-u** argument is optional. Running the script without the argument simply performs a disk space check, before terminating. Running with the **-u** argument causes the script to perform the processing after the free disk space check.

Notes

- Unique Authority IDs are re-calculated and re-assigned.
- If **authority_build** is being run to build or rebuild some Authority Types, while leaving other type(s) unaffected, the data for unaffected Types is loaded back into the new tables.
- A list of the files in the data directory available for use by the **authority_load** script appears at the end of the report. The **authorisor_dae** daemon generates the standard **.err**, **.log** and **.rep** files.
- Any retained data (codes, notes and references from the Type(s) being rebuilt) is merged into the file of rebuilt Authorities, and the merged data loaded into AUTHORITY, AUTHORITY_CODE, AUTHORITY_NOTE, AUTHORITY_AUTHORITY_LNK and AUTHORITY_WORK_LINK.
- If any of the re-linking process fails to find a match, the filename(s) holding the unmatched data will be written to the report file.
- The **authority_load.rep** report file is created. This indicates the times the script started and stopped, the command line used, and reports of the interim stages. These interim stage reports indicate which stage is started in each case, with a progress report as each 10000 rows is processed, and counts of the total number of rows processed at each stage.

auto_access_points

auto_access_points is the process which builds OPAC indexes, according to rules defined by the Library (in the tag rules tables). It also builds separate collections, according to criteria defined by the Library. The script calls two main processes, the first to build the collections (**mcoll_dae**) and the second to rebuild the OPAC indexes (**access_points_dae**).

Make sure that sufficient time is available to complete the run as this process cannot easily be interrupted. Check the **access_points.report** file from the previous run for an indication of how long it took last time. Note that adding a new collection can make a significant difference to the time taken to complete this. Remove any processor-intensive jobs from the cron, ensure that sufficient disk space is available, and ensure that all Alto users are logged off.

The script is normally scheduled using the cron.

Usage

Log on as the operator usually used for running the **access_points** and enter the following command:

auto_access_points -t<processing_type>

The optional processing type can be any of the following.

Processing type	Description
STANDARD	This option builds the OPAC access_points tables as normal. If no option is specified, the default will be STANDARD.

AUTHORITY	Running auto_access_points with this option will add entries from Authority cross references to the existing OPAC Author table.
ALL	Running auto_access_points with this option will build the standard OPAC access_points tables, and then add entries from Authority cross references to the existing OPAC Author table.

Notes

- **auto_access_points** produces many reports in the working directory and reports directory during processing. On successful completion of a run these are amalgamated into a single file: "access_points.report" in the working directory (usually **/scratch**). Details are appended to this script each time **access_points** is run.
- It is important to check this file after running **auto_access_points** as any errors or failures in the process will be reported to this file. Certain errors can be ignored, others must be acted on. If there is insufficient space the run will terminate, in which case the figure produced by the report should be used as a guide when clearing sufficient space in /scratch for a successful run.
- Please do not delete this file after running **auto_access_points** as it provides a useful record for predicting how much disk space and time should be allocated for future runs.

bor_add_pin

The **bor_add_pin** script allocates randomly generated Personal Identification Numbers (PINs) to Borrowers who are already on the system and have their PIN field currently set to null. Existing Borrowers who already have PIN numbers will not be affected.

Usage

Log on as **talis** and enter the following command:

bor_add_pin -h -u -v -d<database> -r<report directory> -b<begin_id> -e<end_id> -m<max_borrs>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	The **begin_id** argument may be used to specify the BORROWER_ID of the Borrower from which to commence processing. If this argument is not given then all Borrowers up to that identified by the **end_id** argument (-e) will be processed. That is the process will commence from the Borrower where BORROWER_ID=1.
-e	The **end_id** argument may be used to specify the BORROWER_ID of the Borrower at which to finish processing. If this argument is not given then the process will end processing from the Borrower with the highest BORROWER_ID. If a **begin_id** and **end_id** range is not specified then all Borrowers on the system will be processed.
-m	As an alternative to using the begin_id and end_id pair of arguments, the **max_borrs** argument can be used to specify the maximum number of Borrowers to be processed in the current run. If this option is used, the process will commence processing from the next BORROWER_ID from where the previous run finished, and process the specified number of Borrowers. For this option to function correctly it is essential that the report from the previous run is left intact. This option cannot be used either with the **-b** and **-e** options, or if the previous run used a different database.

bor_anon_delete.pl

The Data Protection Act restricts the retention of information relating to individuals, to a limited period after it is no longer necessary. The **bor_anon_delete.pl** script enables libraries to erase the personal details of borrowers that are no longer considered as active, while retaining the statistical information relating to them and the integrity of the database.

Note: this script has been amended in Alto 5.5 so that it will now remove the content of both ad hoc letters and the new-style notifications when a borrower's details are anonymised.

Usage

Log on as **talis** and enter the following command:

bor_anon_delete.pl -p<filename> -b<date> -d<database> -h -n<site,site> -r<report directory>
-s<data directory> -t<processing_type> -u -v -z

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the date to be used for processing. It can be used with a processing type of 'EXPIRY' or 'DELETED'. If specified it must be a date in the past and a borrower must have an expiry date (when -tEXPIRY is used) or an edit date (when –tDELETED is used) before this date to be included in the processing. If the date is not specified the system date will be used.
-n	An optional argument that specifies which site codes are to be included in the processing. If multiple sites are specified they must be separated by a comma. If omitted, all sites will be selected.
-p	This argument is mandatory. The parameter file named here should be located in the data directory.
-t	An optional argument that specifies the processing type. The valid options are 'EXPIRY', 'LASTTRANS' and 'DELETED'. If not specified the processing type will default to 'EXPIRY'. EXPIRY – The script will select active borrowers on the basis of their expiry date. A borrower will be included in the processing if he/she has an expiry date before the date specified in the –b<date> argument or before the run date if the –b argument is not specified. DELETED – The argument will select borrowers of deleted status. A borrower will be included in the processing if he/she has a 'deleted' status and has an edit date before the date specified in the –b<date> argument or before the run date if the –b argument is not specified.
-z	An optional argument that if specified will cause the selected borrower details to be anonymised. If not specified the borrower status will be set to 'deleted' but the details will remain.

Example:

bor_anon_delete.pl –tEXPIRY –b01/01/2004 –v –u

This would set the status of borrowers with an expiry date before 01/01/2004 to 'deleted' but would not remove details (as no **–z** argument has been specified). The edit date of these borrowers would be set to the run date which in this example is 31/1/2004.

Parameter file

The following parameters may optionally be specified in the parameter file:

Parameter	Description

months_since_last_trans=<value>	<value> is a whole number less than 999. This allows the user to specify the period of time in months that must have elapsed since a borrower's last transaction took place. This parameter is only used if the –t<processing type> argument has a value of 'LASTTRANS'. The default value of 24 months will be used if the parameter is not specified.
borrower_types_in=<value>,<value>	<value> must be a valid borrower type code. It defaults to all borrower types. Only borrowers of the nominated types are selected. This parameter cannot be used with the borrower_types_out parameter.
borrower_types_out=<value>,<value>	<value> must be a valid borrower type code. It defaults to no borrower types. Only borrowers not of the nominated types are selected. This parameter cannot be used with the borrower_types_in parameter.
override_guarantor=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes" allows a borrower who is a guarantor to be included in the processing. The details of the guarantor are removed from the borrowers they guarantee. If this parameter is set to "no" or is not specified, any borrower who is a guarantor for another borrower will not have their personal details removed.
override_loans=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes", allows a borrower with current loans to be included in the processing. Details of the current loans are written to the report before being discharged. Any queries attached to the loans will be resolved. If this parameter is set to "no" or is not specified, any borrowers with current loans will not have their personal details removed.
override_reservations=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes", allows a borrower with current reservations to be included in the processing. Details of the current reservations are written to the report before being cancelled. If this parameter is set to "no" or is not specified, any borrowers with current reservations will not have their personal details removed.
override_interloans=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes", allows a borrower with current interloans to be included in the processing. Only interloans of 'Pending' status are cancelled. Interloans of a status other than 'Pending' have to be dealt with manually by the user before the borrower may be anonymised. Details of any interloans cancelled by the script are written to the report file. If this parameter is set to "no" or is not specified, any borrowers with current interloans will not have their personal details removed.
override_bookings=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes", allows a borrower with current bookings to be included in the processing. Details of the current bookings are written to the report before being cancelled. If this parameter is set to "no" or is not specified, any borrowers with current bookings will not

	have their personal details removed.
waive_charge_amount=<value>	where <value> is a positive monetary value, with two decimal places between 0.00 and 999.99. e.g. £5 would be entered as 5.00, fifty pence would be entered as 0.50. This parameter allows the user to specify a monetary limit that is used to determine if a borrower with outstanding charges should be included in the processing. If a value is specified here and a borrower has outstanding charges that exceed the value, the borrower's details are not removed. If a borrower has outstanding charges that are less than or equal to the value specified here the borrower's details are removed provided no other conditions prevent it. A value of 999.99 indicates that there is no limit to the outstanding charges that can be waived.
waive_charge_years=<value>	<value> is a whole number. This parameter allows the user to specify the number of years that will be used to determine if a borrower with outstanding charges should be processed. If a value is specified here and a borrower has outstanding charges that were incurred within the period specified, the borrower's details are not removed. If the latest charge was incurred before the period specified here, his/her personal details are removed if no other condition prevents it. The waive_charge_amount and waive_charge_years parameters can be used in a number of ways. If neither parameter is specified, borrowers with outstanding charges do not have their personal details removed. If the waive_charge_amount parameter is used alone, the period since the charge was incurred is not taken into account. If outstanding charges are to be waived only on the basis of the period since they were incurred the waive_charge_amount parameter should be set to a value representing 'no limit' (i.e 999.99) and the waive_charge_years parameter to the required period. If both parameters are specified, an outstanding charge must satisfy both conditions to be overridden.
stop_messages=<value>,<value>	<value> is a valid borrower message id or "all" (any case). This parameter allows the user to specify the borrower message ids (including the borrower block message) that if attached to a borrower, exclude the borrower from the processing. If this parameter is not specified, no messages cause the borrower to be excluded from the processing. If the parameter is specified, a borrower is excluded if he/she has any of the specified messages attached. If the parameter is specified with the value "all" (any case), any messages attached to a borrower prevents the removal of their personal details.
anonymise_barcode=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "yes", causes the borrower barcode field to be anonymised. The barcode will only be anonymised if the command line option –z has also been specified.
anonymise_postcode=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "no", prevents the postcode field of the borrower address being anonymised. If the parameter is not specified or if it is specified with a value of "yes", the postcode field of the borrower address is anonymised. This parameter is only effective if the command line option -z has also been specified.

anonymise_date_of_birth=<value>	<value> is "yes" or "no" (any case). This parameter, if set to "no", prevents the date of birth field of the borrower being anonymised. If the parameter is not specified or if it is specified with a value of "yes", the date of birth field of the borrower is anonymised. This parameter is only effective if the command line option -z has also been specified.

Processing

Records are selected from the BORROWER table on the basis of the **–t** <processing type> argument used.

If no processing type or a processing type of 'EXPIRY' is specified, active borrower records are selected if they have an EXPIRY_DATE that is before the date specified via the command line **–b**<date> argument , or is earlier than today's date if a date was not specified.

If a processing type of 'DELETED' is specified, deleted borrower records not already anonymised are selected if they have an EDIT_DATE that is before the date specified via the command line **–b**<date> argument , or is earlier than today's date if a date was not specified.

If a processing type of 'LASTTRANS' is specified, active borrowers are selected if they have no transactions with a date that falls within the number of months period prior to the run date (as specified in the parameter file). A transaction could be a loan, a reservation, a booking or an inter-library loan. If the **–n**<site> argument is specified borrower records with a home site other than the one specified are excluded.

If the borrower_types_in or borrower_types_out parameter is specified, borrower records that are not of a required type are excluded. Checks are made throughout the database to determine if a selected borrower has any database conditions that would prevent further processing. If any of the following conditions are found, and the user has not set a specific parameter to override the condition, the borrower record is excluded from further processing. Details of the reason(s) for the exclusion are written to the report file, and can include.

- Borrower is a guarantor
- Borrower is blocked
- Borrower has current loan(s)
- Borrower has current reservation(s)
- Borrower has current inter-loan(s)
- Borrower has current booking(s)
- Borrower has attached message(s)
- Borrower has unpaid charge(s)

If the script is run in 'update' mode by specifying the **–u** argument on the command line, borrower records that are deemed eligible for processing are updated. The status is set to 'deleted' and the index name entry removed, if active borrowers were selected. The appropriate action is taken to remove conditions that have been overridden. Further database changes depend on the setting of the **–z** 'anonymise' argument . If the **–z** 'anonymise' argument has not been specified no further database updates are carried out. If this argument has been specified, the borrower details held in the database are 'anonymised'.

Report file

A report file, named bor_anon_delete.rep is created each time the script is run. The location of the file is taken from the **–r** argument , if used. If the –r argument is not specified on the command line, the report is located in the directory specified by the TAL_REP_DIR or if not set to the default data directory **/usr/opt/blcmp/data/utils**. If a report file of the same name already exists, it is renamed with a date/time extension.

The report file contains a progress report of every 1000 rows processed and a count of the number of borrowers selected for processing.

The barcode of all borrowers processed are reported if the **–v** argument has been used. The wording of the report also depends on whether the **–u** argument is specified on the command line. If it is not

specified, the report contains the line "Script is running in report mode – no borrowers will be updated". The count text specified in the report will also depend on this setting. If the **–u** argument is not set, the report shows "Number of borrowers… would be…". If the **–u** argument is set, the report shows "Number of borrowers updated …".

If the –**z** argument has been used the wording of the report indicates that the action being taken is to "delete and anonymise".

bor_block.pl

Libraries may need to prevent Borrowers who have transgressed in some way from actively using their services. The **bor_block.pl** script places a blocking message against Borrowers based on the criteria specified in the Borrower Blocking Rules Form. It may optionally be used to report the barcode numbers (and from Alto 5.2, the expiry dates) of Borrowers who have been blocked.

Usage

Log on as **talis** and enter the following command:

bor_block.pl -b<begin id> -d<database> -e<end id> -h -q<number of days> -r<report directory> -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	The "-b" argument is optional and specifies the BORROWER_ID from which to begin processing. If this argument is not given then processing will commence from the first Borrower on the database (i.e. where BORROWER_ID=1).
-e	The argument "-e" may be used to specify the ID of the Borrower with which to stop processing. If not given, processing stops after the Borrower with the highest BORROWER_ID.
-q	This option is only available in Alto 5.2 and above. It may be used to specify a number of days. Any borrowers whose expiry date is more than that number of days ago will be excluded from processing. If not set, all expired borrowers will be included

Notes

- The "bor_block" script places the blocking message specified on the Borrower Blocking Rules Form against any active Borrower found to have exceeded any of the limits specified in the Borrower Blocking Rules. It does this based on the criterion that the Borrowers Type is one of those specified for inclusion, and provided he or she does not already have the message set.
- If the fines outstanding limit is set, the message "Borrowers with fines exceeding n will be blocked" appears, where "n" is the amount specified.
- If the days outstanding limit is set, the message "Borrowers with fines more than n days old" appears, where "n" is the number of days specified.
- If the Items overdue limit is set, the message "Borrowers with more than n items overdue will be blocked" appears, where "n" is the number specified.
- If the days overdue limit is set, the message "Borrowers with an item more than n days overdue will be blocked" appears, where "n" is the number of days specified.

bor_name_list_build.pl

The **bor_name_list_build.pl** script re-builds the NAME_LIST_DISPLAY in Borrower records using all information in the Borrower's default address, including Postcode details. These changes maximise the data available for searching with the Address and Postcode restrictors.

Usage

Log on as **talis** and enter the following command:

bor_name_list_build.pl -b<begin id> -d<database> -e<end id> -h -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	The "-b" argument specifies the BORROWER_ID from which to begin processing. If this argument is not given then processing will commence from the first Borrower on the database (i.e. where BORROWER_ID=1).
-e	The argument "-e" may be used to specify the ID of the Borrower with which to finish processing. If this argument is not given then processing will finish after the Borrower with the highest BORROWER_ID.

Notes

- The script obtains the default Address details of each Borrower in the range specified on the command line and builds a NAME_LIST_DISPLAY in the new format. If run with the **-u** argument it replaces the existing NAME_LIST _DISPLAY in each Borrower's record with the new one.

borr_import

The Borrower Importer is a batch program **borr_import** which facilitates the import of Borrower records onto the LMS database from an external source. Typically this utility is used by academic libraries, and the external source of Borrower data is normally the institution's registry. The origin of the Borrower records determines the way in which the input file is processed; this is specified on the command line as either:

- "bbreg" or
- "hemis".

If "hemis" is specified, incoming records are matched against the LMS database by Registration Number. If a record already exists for a Borrower, the incoming details are used to update the record. If a record does not exist, a new record is added to the database. Each new record is normally given a dummy barcode number, but it is possible to use the barcode number in the incoming record, instead of a dummy number, when creating a Borrower record or updating an existing record.

The default email address now has the following tags available:

- 600 Email address name
- 601 Email address
- 602 Email start date
- 603 Email end date

For second and third email addresses, tags in the range 610-613 and 620-623 respectively should be available. Email address and Email address name are mandatory for each set of tags where data exists. If the borrower being imported exists on the database and there is a row in the IMPORT_PARAMETER table where TYPE_ID = 117 and VALUE_1 = EMAILx (where x represents an email address number in the range 1-3), then any email address with a corresponding name for that borrower will not be updated. If there is no master borrower parameter present for a particular email address, then any existing email address with a corresponding name is deleted and email address in the incoming record is inserted.

Email validation

- An email address name and an email address must be specified. However, start and end dates are optional.
- If an email address name to be imported is duplicated, the duplicate email address name should not be imported

- An email address must not contain spaces
- An email address must contain a single '@' symbol.

Usage

Log on as **talis** and enter the following command:

borr_import -x -a -z -m <libcode> <borrower code length> <data directory> <origin> <type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-x	The argument "-x" requests generation of the barcode check digit. This option is only necessary when importing "BBREG" format records. When you run with the check digit argument, Borrower Import will generate a check digit for all barcodes that have one less digit than the number specified in the barcode length argument.
-z	The argument "-z" specifies that the script should use the borrower barcode validate routine used by online Talis. This is required by some libraries with non-standard borrower barcode validates. The validate expects barcodes in incoming records to be in the format in which they are entered in Alto i.e. it will not handle barcodes without check digits. If this argument is not specified borr_import uses its own validate routine, which will accept standard barcodes with or without check digits plus the barcodes of several non-standard libraries.
-m	The argument "-m" allows you to import multiple courses per borrower to be imported. If the argument is specified on the command line, borr_import will attempt to import a borrower's default course from tag 080 in the BBREG record and other courses from tags 081 to 089 inclusive. It is not necessary to specify these tags in the Borrower Attribute Parameters (TYPE_ID = 108). ▪ Course codes in tags 081 to 089 will be rejected if the BBREG record does not contain a default course code in tag 080. The message: No default course: other courses rejected is written to borr_imp.rep. ▪ If the course code in tag 080 is not a valid course code in the GROUPING table (i.e. it does not appear in the course types list under Utilities | Parameters | Names | Circulation) and there are other courses in the record, none will be linked to the borrower record. The message: Invalid default course <code>: all courses rejected is written to borr_imp.rep, where <code> is the invalid code. ▪ If the course code in tag 080 is the only course in the record and it is not a valid course code in the GROUPING table, it will not be linked to the Borrower record. The message: Invalid course <code> rejected is written to borr_imp.rep. ▪ If a course code in tag 081 to 089 is invalid, it will not be linked to the Borrower record. The message: Invalid course <code> rejected is written to borr_imp.rep. Any valid courses in the same record will be

	linked in the normal way. ▪ If the same course code is present in more than one tag no courses will be linked to the borrower record. The message: Duplicate course codes: all courses rejected is written to borr_imp.rep. ▪ If an incoming BBREG record containing course data matches an existing borrower record, any courses attached to the existing record will be deleted and the incoming course(s) added. ▪ If you wish to import one course per borrower, you can run the borr_import script without the -m argument. However you must continue to update the Borrower Course (TYPE_ID 110) parameters in the IMPORT_PARAMETER table. Note: It is not necessary to update the Borrower Course Parameters if you use the -m argument.
libcode	The Library's Libcode. This should be supplied on the command line exactly in the form used by the Library in lower case. No prefix is required.
barcode length	It is mandatory to specify the Borrower barcode length as used by the Library; for example "8".
data directory	The data directory represents the directory path to which the BBREG input records should be written ready for import and conversion (for example, "/scratch"). Note: This directory will also be used for the report file and audit file generated as a result of running "borr_import". Intermediate files will also be written to this directory.
record origin	This argument is optional but, if present, must be supplied at the end of the command line after the mandatory arguments. One of two valid values may be supplied to indicate the origin of the records to be imported; either "bbreg" or "hemis" (a database system used by certain academic registries). If no value is supplied for Record Origin the system will default to "bbreg".
type of processing	The argument "-t" may be specified as either "DUMMY" or "BARCODE". The default is "DUMMY". The values "DUMMY" and "BARCODE" may be entered in upper or lower case. The "-t" argument may only be used with the "Record Origin" argument specified as "hemis". If specified, it must follow the origin on the command line. If "hemis" is not supplied, the script will terminate with the message: "ERROR: 'hemis' must be specified if -t option used" If an invalid "-t" option is specified, the script will terminate with the message: "ERROR: type of processing must be DUMMY or BARCODE

The Borrower Import audit file (borr_imp.au)maintains a record of the Borrower records which have been processed, showing the Barcode, Registration Number, Borrower Name (both Surname and Forenames) and the processing performed on that record (recorded in the Action column). The audit file is written to the data directory as specified when running "borr_import". The default is /scratch unless a different directory has been specified.

Standard BBREG Processing

The barcode will be extracted from Tag 000 in every record found in the import file. The barcode will be used to see if it matches against the existing database and thereby determine whether the

Borrower already exists on the system. If the Borrower already exists then the system will treat this Borrower record as an "Update".

Conversely, if the Borrower barcode does not yet exist on the Talis database then the current Borrower record will be treated as an "Insert"; i.e. a new Borrower record will be added to the database. All of the data found in the Borrower record to be imported is processed and used either to update an existing Borrower record or create a new Borrower record.

HEMIS processing

Normal processing of data in HEMIS format attempts to match incoming records against existing LMS Borrowers by their Registration Numbers. If a Borrower with this Registration Number already exists then his/her imported details will be treated as an "Update" to the database. There is no default mapping for the Borrower Registration Number, so one must be specified in TYPE_ID 108 (see below). If a particular Borrower Registration Number does not yet exist on the Talis database then the current Borrower record will normally be treated as an "Insert"; i.e. a new Borrower will normally be added to the database.

Type of processing

Each new record is normally given a dummy barcode number, but optional use of the "-t" argument permits a barcode number in the incoming record to be used instead of a dummy number when creating a Borrower record and in updating an existing record.

When "-tBARCODE" is specified, the Registration Number in the input record is matched against REGISTRATION_NUMBER in the BORROWER table. If a match is found, the input record is treated as an update of the existing Borrower. The barcode number in the input record will then replace that in the existing record if they are different, provided the incoming number is valid and not already being used by another Borrower. Details of the original number will be added to the BORROWER_OLD_BARCODE table.

If no match is found, a new Borrower will be added. The barcode number in the input record will be included if this is valid and not already in use. A check digit will be generated for an incoming barcode if "-x" has been specified on the command line.

The audit file is produced if "-a" has been specified, listing the records processed with their barcode numbers. An input record will be rejected if it does not contain a barcode or if the barcode number fails validation or is already in use. An appropriate error message will be output to "borr_import.rep".

Dummy barcodes

If "-tDUMMY" is specified or "-t" is not specified on the command line, the basic "HEMIS" functionality is applied.

HEMIS import records do not have to have a Borrower barcode because the matching criterion used during the import (insert/update) process is the Borrower Registration Number. The LMS requires barcodes, however, so a dummy barcode will be generated for each Borrower by the data conversion to LMS format.

Format

The format of the dummy barcode is capital "H" followed by numerics up to the barcode length used by the Library. For example the sequence might start "H0000001", "H0000002", "H0000003", i.e. incrementing by one with each additional Borrower.

Manual Modification

These barcodes will remain useless dummies until they are changed manually to something meaningful by Library staff. When Borrowers go into their Library with their identification cards Library staff will be alerted to the fact that this is the first time they have come to use the Library and that their records need functional barcodes inserting manually. This is achieved by:

1. Calling up the Borrower record using a Borrower Name search.
2. Editing the dummy barcode to a real LMS barcode for use with the Borrower's Library membership card.

Borrower Import Parameters

These parameters govern the conversion of both "bbreg" and "hemis" format data into LMS format. They are held in the IMPORT_PARAMETER table. (Import Works uses this table too). This table has three attributes:

- TYPE_ID

- VALUE_1
- VALUE_2

The TYPE_ID indicates the type of parameter. For example, Borrower Attribute parameters have a TYPE_ID of 108. The contents of VALUE_1 and VALUE_2 for each TYPE_ID are described in the **Borrower Default Data Parameters** section below.

For more information about parameters, click on the appropriate section:

Borrower Attribute Parameters (TYPE_ID = 108)

Borrower attribute parameters used by the Borrower import process are held in the IMPORT_PARAMETER table and have a TYPE_ID of 108.

Each parameter maps an LMS Borrower-related attribute to a field in the incoming BBREG borrower data. As explained above, the LMS attribute is named in VALUE_1 of the parameter, and the tag value of the corresponding BBREG field is named in VALUE_2 of the parameter.

VALUE_1 (Valid Parameters)	VALUE_2 Default Tag Values
BORROWER.BARCODE	0
BORROWER.SURNAME	10
BORROWER.FIRST_NAMES	30
BORROWER.TYPE_ID	1
BORROWER.PIN	130
BORROWER.STYLE	20
BORROWER.DATE_OF_BIRTH	Null
BORROWER.REGISTRATION_DATE	120
BORROWER.REGISTRATION	Null
_NUMBER	
BORROWER.EXPIRY_DATE	2
BORROWER.HOME_SITE	3
BORROWER.DEPARTMENT	Null
BORROWER.NOTE	100
BORROWER.GUARANTOR	Null
BORROWER_OLD_BARCODE	110
.BARCODE	
ADDRESS.LINE1/1	50_1
ADDRESS.LINE2/1	50_2
ADDRESS.LINE3/1	50_3
ADDRESS.LINE4/1	Null
ADDRESS.LINE5/1	Null

ADDRESS.POSTCODE/1	Null
ADDRESS.TELEPHONE/1	Null
ADDRESS.FAX/1	Null
ADDRESS.EMAIL/1	Null
ADDRESS.LINE1/2	60_1
ADDRESS.LINE2/2	60_2
ADDRESS.LINE3/2	60_3
ADDRESS.LINE4/2	Null
ADDRESS.LINE5/2	Null
ADDRESS.POSTCODE/2	Null
ADDRESS.TELEPHONE/2	Null
ADDRESS.FAX/2	Null
ADDRESS.EMAIL/2	Null
ADDRESS.LINE1/3	70_1
ADDRESS.LINE2/3	70_2
ADDRESS.LINE3/3	70_3
ADDRESS.LINE4/3	Null
ADDRESS.LINE5/3	Null
ADDRESS.POSTCODE/3	Null
ADDRESS.TELEPHONE/3	Null
ADDRESS.EXT/3	Null
ADDRESS.FAX/3	Null
ADDRESS.EMAIL/3	Null
ADDRESS.NAME/1	Null
ADDRESS.NAME/2	Null
ADDRESS.NAME/3	Null
ADDRESS.START_DATE/1	Null
ADDRESS.START_DATE/2	Null
ADDRESS.START_DATE/3	Null
ADDRESS.END_DATE/1	Null
ADDRESS.END_DATE/2	Null

ADDRESS.END_DATE/3	Null
ADDRESS.COUNT	Null
GROUPING.GROUPING_ID/COURSE	Null
ADDRESS.EMAIL/1	600,601.602,603
ADDRESS.EMAIL/2	610,611,612,613
ADDRESS.EMAIL/3	620,621,622,623

- Column 1 in the above table shows the valid attribute names that may occur in VALUE_1.
- The Borrower import process assumes that the incoming record can contain up to 3 addresses.
- Column 2 shows the default Tag values which the Borrower import process will use. For example, if a row is present in IMPORT_PARAMETER with TYPE_ID = 108 and VALUE_1 = "BORROWER.SURNAME" and VALUE_2= "25" then the Borrower import process will use the data in Tag 25 of the BBREG input records to generate the SURNAME attribute of the BORROWER rows. If this parameter is not present, then borrower import will default to Tag 10 for generating Surnames.
- It is advisable to use the default tags only for the data to which they default.

No Default Tags

Where no default Tag value ("Null") is shown in the table above, then the Borrower import process will not attempt to generate this attribute unless the user has set up the parameter. This is because when the VALUE_2 column shows a "Null" value (i.e. no default Tag) this indicates that the field is not required because the incoming records do not contain that particular type of data (or because this data is not required on Talis).

Multiple Addresses

The import record can contain up to three Addresses, with 5 lines for each Address. Different Addresses are specified in the parameters by the "/#" at the end of the VALUE_1. For example "ADDRESS.POSTCODE/2" informs Borrower import where to look for the Postcode of the second Address. Similarly "ADDRESS. LINE5/3" informs Borrower import where to look for the 5th line of the third Address.

Address Lines

Alto Borrower import is compatible with BLS Borrower import so that ex-BLS customers can continue with their existing procedures. Since the BBREG format used by the Alto Borrower import process was inherited from BLCMP's BLS system, the process can cope with all the lines of an Address being present in the same tag, with each line being delimited by a carriage return (hex "OD") character. This is the expected default unless told otherwise by the parameters.

The default tag values for the lines of each Address are specified in the table above as, for example, "50_1", "50_2" and "50_3". This indicates that if the parameters are absent the import process will look for three address lines, each separated by a carriage return.

Default Address

A tag may be present in each BBREG input record to specify which of the Addresses in the record is the default Address at the time of running the import process. The "ADDRESS.COUNT" parameter indicates which tag specifies the default address. For example, if VALUE_2 = "7", then the process will look for Tag 7 in the input record and expect it to contain either the number "1", "2" or "3", specifying whether the first, second or third Address is the default.

Borrower course

The Borrower's Course (in the case of academic libraries) is held as a GROUPING_ID in the GROUPING table on Talis; hence the complex syntax of the "GROUPING.GROUPING_ID/COURSE" parameter used to specify the Tag containing the Borrower's Course.

Dates

Where the year component of a date is specified as just two digits in the incoming data, "borr_import" will apply the following rules:

- For Dates of Birth, all two digit years will be expanded to be in the 20th century.
- For all other dates occurring in other fields, two digit values in the range "00" to "09" will be treated as occurring in the year 2000 and beyond. Two digit values outside this range (i.e. "10" to "99") will be treated as occurring in the 20th century.

Borrower Default Data Parameters (TYPE_ID = 111)

The Borrower default data parameters used by the Borrower import process are held in the IMPORT_PARAMETER table. They have a TYPE_ID of 111. Each of these parameters defines the default data to be used by the Borrower import process for specific LMS Borrower-related attributes in cases where the data is not present in the BBREG input record. There are 14 of these parameters, all of which are mandatory. It is essential for Talis to have some data against these 14 attributes.

VALUE_1 (Valid parameters)	VALUE_2
BORROWER.TYPE_ID	A valid TYPE_ID
BORROWER.DEPARTMENT_ID	A valid LOCATION_ID
BORROWER.EXPIRY_DATE	DD/MM/YY
BORROWER.HOME_SITE_ID	A valid LOCATION_ID
ADDRESS.NAME/1	Name of Address 1
ADDRESS.NAME/2	Name of Address 2
ADDRESS.NAME/3	Name of Address 3
ADDRESS.START_DATE/1	DD/MM/YY
ADDRESS.START_DATE/2	DD/MM/YY
ADDRESS.START_DATE/3	DD/MM/YY
ADDRESS.END_DATE/1	DD/MM/YY
ADDRESS.END_DATE/2	DD/MM/YY
ADDRESS.END_DATE/3	DD/MM/YY
ADDRESS.COUNT	1, 2 or 3

As before, the LMS attribute is named in the VALUE_1 column of the parameter, and the corresponding LMS default data is named in the VALUE_2 column. The Registry office can supply the Library with Borrower records each having up to three Addresses (to cater for Home Address, Term Address etc.), and all three Addresses having their respective Start and End Dates specified. The "ADDRESS.COUNT" parameter indicates the default address.

Site Parameters (TYPE_ID = 100)

The VALUE_1 column of Site parameters contains the data as found in the incoming record, while the VALUE_2 column informs the Borrower Import process the Site Names/Codes into which this Site information should be translated for LOCATION_ID.

If there is no recognisable Site data in the imported record which matches that in the Site parameters then the record will retain its current value. (For example, if the Registry is going to supply suitable Site information in the first instance there is no need to translate this on import).

Borrower Department Attribute (TYPE_ID = 109)

This works the same way as Site Parameters above; the VALUE_1 column of Borrower Department parameters contains the data as found in the incoming records, and the VALUE_2 column contains the correct LMS equivalent to be inserted by the Borrower Import process.

Borrower Course (TYPE_ID 110)

As before, the VALUE_1 column of Borrower Course parameters (for academic libraries) will hold data as found in the incoming records while VALUE_2 will hold the corresponding GROUPING_ID of Sub_type 6 (i.e. a Course).

VALUE_1 should not be duplicated or contain wildcard characters. VALUE_2 should be a valid GROUPING_ID otherwise the course will be rejected. The message:

```
Invalid course <code> rejected
```

will be written to borr_imp.rep

Master Borrower Parameters

It is possible to set up Master Borrower Parameters which specify the data which should not be overwritten by the incoming Borrower records. "borr_import" reads the import parameters (i.e. the Master Borrower Parameters) from the IMPORT_PARAMETER table and uses these to check which data should be loaded from the imported Borrower records into the LMS database.

Borrower Attributes

The following lists indicate the Master Borrower Parameters that may be set-up. They have a have a TYPE_ID of "117" and VALUE_1 is set to the entry indicated on the list. The following are Master Borrower Parameters that correspond to attributes in the BORROWER table:

- BORROWER.BARCODE
- BORROWER.SURNAME#
- BORROWER.FIRST_NAMES#
- BORROWER.TYPE_ID
- BORROWER.STATUS
- BORROWER.PIN
- BORROWER.STYLE
- BORROWER.DATE_OF_BIRTH
- BORROWER.REGISTRATION_DATE
- BORROWER.REGISTRATION_NUMBER
- BORROWER.EXPIRY_DATE
- BORROWER.HOME_SITE_ID
- BORROWER.DEPARTMENT_ID
- BORROWER.NOTE
- BORROWER.INDEX_NAME#
- BORROWER.NAME_LIST_DISPLAY*

When any of the above are set-up in Master Borrower Parameters then those attributes will not be updated/overwritten by BBREG data. Those marked with "#" are all related to the Borrower's name and should be implemented together or not at all.

BORROWER.NAME_LIST_DISPLAY is marked with a "*" because it relates to the ADDRESS Master Borrower Parameters. It should be used only in combination with one or more of these parameters which are described below.

Handling Addresses/Contact Points

The following are Address-related Master Borrower Parameters:

• ADDRESS.COUNT*

• ADDRESS1*

• ADDRESS2*

• ADDRESS3*

• BORROWER.NAME_LIST_DISPLAY*

There is one Master Borrower Parameter for each of the three potentially existing Address/Contact point pairs.

If the Borrower already exists on the database and there is a row in the IMPORT_ PARAMETER table where TYPE_ID = 117 and VALUE_1 = "ADDRESSX"(where "X" represents an Address number, in the range 1 to 3) then that address will not be updated.

If there is no Master Borrower Parameter present for a particular Address/ Contact point pair then the existing address is deleted and the one in the incoming record will be inserted (if present).

If one or more Address/Contact points exist then one must be "Currently flagged" (i.e. having the CURRENT_CONTACT_POINT attribute set to "T"rue). The flagged address is the one to which Overdues and other correspondence will be sent. The ADDRESS.COUNT Master Borrower Parameter, if present, ensures that the existing flagging will not be overwritten by the incoming record.

If there is no ADDRESS.COUNT Master Borrower Parameter, then the incoming BBREG record or defaults will be used to determine the current flagging.

The attribute BORROWER.NAME_LIST_DISPLAY contains part of the currently flagged Address, so the BORROWER.NAME_LIST_DISPLAY Master Borrower Parameter must be present if the Master Borrower Parameter for the current Address is present.

It is safest if all those Master Borrower Parameters marked with "*" are implemented together or not at all.

Handling email addresses

The following are the Email-related Master Borrower Parameters:

- EMAIL1
- EMAIL2
- EMAIL3
- EMAIL_PREFERRED

There is one Master Borrower Parameter for each of the potentially existing Email addresses.

If the Borrower already exists on the database and there is a row in the IMPORT_PARAMETER table where TYPE_ID=117 and VALUE_1=EMAILX (where X represents an Email Address number then that address will not be updated.

If there is no Master Borrower Parameter present for a particular email address then the existing address is deleted and replaced by the incoming email address.

If one or more Email addresses exists then one must be flagged as the default email address. If EMAIL_PREFERRED is present in IMPORT_PARAMETER with TYPE_ID=117 then the existing flag and associated Email address will not be overwritten by the incoming record.

If there is no EMAIL_PREFERRED Master Borrower Parameter, then the incoming BBREG record will be used to determine the flagging.

Old Barcode

If the Borrower already exists on the database and there is a row in the IMPORT_PARAMETER table where TYPE_ID = 117 and VALUE_1 = "BORROWER_OLD_BARCODE" then the old barcode will not be updated by the incoming record.

Course

There is no Master Borrower Parameter for Course. It is assumed that the BBREG data for Course will always be the most up-to-date available and so it will be used for updating by default.

Entering Master Borrower Parameters

Unfortunately, there is currently no user interface for entering import parameters. The System Manager may instead enter the relevant rows using the "imp_modify" script (which is more user-friendly than using isql). This general purpose utility for inserting or deleting rows in the IMPORT_PARAMETER table is documented fully in The imp_modify Utility section of the System Management Manual.

The following example shows how to enter the Master Borrower Parameter for ADDRESS1:

1. Log on as **talis**.
2. Change directory (if necessary):

   cd /usr/opt/blcmp/talis/utils/bin

3. Type in the following command, to run the "imp_modify" utility with arguments to match your requirements. For example:

   imp_modify -u -tINSERT -j117 -kADDRESS1 -l+ <Enter>

borr_type_updt.pl

The **bor_type_updt.pl** script updates the borrower type for borrowers whose date of birth falls between a given date range that is calculated from the minimum and maximum ages specified in the parameter file, and whose current borrower type matches a borrower type specified in the parameter file. It optionally updates the expiry date of borrowers as well as their borrower type.

Due to the nature of the script is it best that it is not run during the daytime. Instead it should be scheduled using the cron and run in the evening when the libraries are all closed.

Usage

Log on as **talis** and enter the following command:

borr_type_updt.pl -a<appended filename> -d<database> -h -o<output filename> -p<filename> -r<directory>
-s<directory> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This names the parameter file. If the argument is not given, then the default is 'borr_type_updt.param'. The parameter file should be located in the data directory.

Parameter file

The name of the parameter file by default is borr_type_updt.param, and it will be stored in the directory **/usr/opt/blcmp/data/utils**. An example file is shown below:

```
CURR_BORR_TYPE=C16,C17,C18
NEW_BORR_TYPE=AD
STARTING_AGE=18
FINISHING_AGE=21
UPDATE_EXPIRY=yes
```

The parameter file contains the information that identifies which borrowers need to be updated, the new borrower type that will be assigned to the borrowers, and information informing the script whether the expiry date is to be updated.

Argument	Description
CURR_BORR_TYPE	Only borrowers whose current borrower type is one of the borrower types specified by this mandatory parameter, will be processed by the script. If multiple borrower types are present they must be separated by a comma.
NEW_BORR_TYPE	This mandatory parameter must contain a single borrower type code. The code entered for this parameter is the borrower type that all the borrowers that meet the selection criteria will be updated to.
STARTING_AGE	This mandatory parameter is used to specify the minimum age in years of the borrowers whose borrower type you wish to update.
FINISHING_AGE	This optional parameter is used to specify the maximum age in years of the borrowers whose borrower type you wish to update. It is an optional parameter, and if it is given the script will only process borrowers whose age is identical to that specified by STARTING_AGE.
UPDATE_EXPIRY	This is an optional parameter that specifies whether the script will update the borrower expiry date as well as the borrower type. If it not specified in the parameter file it will default to 'no'.

borrower_image_import.pl

The **borrower_image_import** script may be used by libraries to import images in bulk into the BORROWER_IMAGE table. The script will process each row in a specified input file in turn, checking for the existence of either the barcode or registration number (depending on the parameter) in the BORROWER table.

Where a match is found, the picture is retrieved from the .jpg file and processed as if it had been imported through the borrower screen in Alto. It is then added to the BORROWER_IMAGE table along with the appropriate BORROWER_ID.

The script is normally scheduled using the cron.

Usage

Log on as **talis** and enter the following command:

borrower_image_import.pl -d<database> -h -s<data directory> -i<filename> -t<REGISTRATION_NUMBER/BARCODE> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This optional argument allows you to specify the name of the input file. If this argument is not given, it will default to borrower_image_import.in.
-t	This optional argument specifies whether the input file contains the REGISTRATION_NUMBER or the borrower BARCODE. If this argument is not given, the default is REGISTRATION_NUMBER.

Input file

Rows in the input file must be in the format:

<barcode_number>|<Filename>

For example:

12345679|9905607.jpg

The filename is a file containing the photo. The filename must refer to a .jpg file. Both the input file and the .jpg file must be in directory specified by the **-s** argument (described above). If any other type of file is submitted, then the associated barcode is recorded in the report file and displayed on screen with the message "Format of the image not recognised".

Also note that:

- Both pieces of information are mandatory
- Any blank lines in the input file or lines with "#" in will be ignored

Notes

- If an image already exists in the database for the BORROWER_ID in question then it is replaced by the incoming image.
- Where no match for the barcode or registration number is found in the BORROWER table then the associated barcode/registration number is recorded in the report file and displayed on screen with the message "Borrower record not found".
- If the script cannot locate the image using the specified filename then the associated barcode/registration number is recorded in the report file and displayed on screen with the message "Photo not found".

cad_dup_sans_list

In order to set up a Parent / Base Supplier link, all of the Suppliers to be linked must have the same SAN (Standard Address Number) in the Address Form. If the Suppliers concerned receive Orders by EDI, this will be the ANA Number of the Supplier. The **cad_dup_sans_list** script report can be used to generate a list of Suppliers with the same SAN, thereby helping Libraries to re-create their Parent Suppliers where they may have set up more than one Supplier record for one "actual" Supplier.

Usage

Log on as **talis** and enter the following command:

cad_dup_sans_list.pl -d<database> -h<help> -r<report directory>

The script uses standard script arguments, as described here. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Notes

- A report file, named **cad_dup_sans_list.rep**, is created each time the script is run. If a report file of the same name already exists, it is re-named with a date/time extension. The report shows each SAN which is held for more than one Supplier. The matching Supplier Code(s) and Supplier Name(s) are reported below each SAN.
- Each Supplier can have more than one address. If multiple addresses for the same Supplier contain the same SAN, this permissible duplication is not reported. The report only lists different Suppliers sharing the same SAN.

chk_seq_reset

The script **chk_seq_reset** may need to be run as a result of multiple insertions in the issue Check-in List online in Alto. This is a batch script used to reset the interval between consecutive issue rows which have been added to Check-in List for a specified Work at a certain Delivery Site. When seven or more rows are inserted in this list (i.e. when checking-in additional issues in Acquisitions Open Orders) this will usually cause an overflow and the error message "No more space to insert rows for [Control Number]. Please see your System Manager "

Usage

Log on as **talis** and enter the following command:

chk_seq_reset -h -d<database> -n<control number> -l<delivery site>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-n	This argument is mandatory as the script will fail without a Control Number to specify the parent Work for which rows on the Check-in List are to be reset. If omitted, an error message will appear on the screen informing you that the Control Number must be supplied when running this script.
-l	This argument is mandatory as the script will fail if a Delivery Site is not specified to indicate where the operator was attempting to check-in the Work. If omitted, an error message will appear on the screen informing you that a Delivery Site must be specified when running this script.

clear_search_works.pl

The **clear_search_works.pl** script (available in Alto 5.2 and above) will clear out old rows from the SEARCH_WORKS table if they still have not been processed after a given number of days. Rows are inserted into this table by the item_imp_cat_serv daemon, which imports item fulfilment data for Cataloguing Service users. The table is then read by the Cataloguing Service MARC Import process, which searches for and imports MARC records for these items. If it cannot find a record to import, the row remains in the table and the Import process tries again the next time it runs. Over time, many rows can build up in this table, slowing down the process.

The script will remove rows older than a given number of days. The default is 90 days.

Usage

Log on as **talis** and enter the following command:

clear_search_works.pl –b<days> -d<database> -r<report directory> –u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the number of days. This is used to calculate the date before which rows should be deleted. If, for example, the date is calculated as 15/12/2010, rows added before (not on) that date will be deleted.
-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.

Notes

- The report file will show the number of rows that have been or would be deleted and the WORK_ID, ISBN/EAN and the date the row was added to the SEARCH_WORKS table for each "deleted" row.

Example report file:

```
clear_search_works.pl              commenced                  29/06/11 10:45:59
                                   ~~~~~~~~~

Command line: clear_search_works.pl -b179
Rows greater than 179 days will be deleted
Database will NOT be updated
BIB_ID      Control Number       Create Date
----------- -------------------- --------------------------
631195      9782137200355        26/07/06 12:00PM
635806      2001544061           26/07/06 12:00PM
```

```
637497      5409873645          26/07/06 12:00PM
637520      0967806186          26/07/06 12:00PM
637524      978-0-8493-9386-0   27/07/06 12:00PM
20689       029776928           24/08/07 11:00AM
Number of rows would be deleted  :       6
clear_search_works.pl              completed                  29/06/11 10:45:59
                                   ~~~~~~~~~
```

dedup_works.pl

Many Libraries have duplicate Works in their database. These duplicates may have been created by the Import Works **import_work** script, by migration, by local working practices or by defects. Duplicate Works may have different Items attached. They may have Orders, Interloan requests and reservations attached to one or more of the duplicates. The duplicate Works appear in OPAC, but only one version of the Work may be accessed from Cataloguing.

The "dedup_works.pl" script is used to:

- Identify duplicate Works.
- Assign new Local Control Numbers to the second and subsequent Works.
- Produce a report file and an output file giving details of Items, active reservations and Orders attached to each Work.
- Tidy up the CONTROL_NUMBER table where rows would otherwise remain for deleted Works.

Usage

Log on as **talis** and enter the following command:

dedup_works.pl -b<begin id> -d<database> -e<end id> -h -m<max rows to process> -o<output filename> -r<report directory> -s<data directory> -u.

Standard script arguments are described here. The remaining arguments for this script are described in the following table:

Argument	Description
-b	This optional argument specifies the WORK_ID in the WORKS table from which to begin processing. If this option is not given then processing commences from the lowest WORK_ID. It is usually used in conjunction with the -e argument (below).
-e	This optional argument specifies the row in the WORKS table at which to end processing. If this option is not given, processing continues to the last WORK_ID in the WORKS table. It is usually used in conjunction with the -b argument (above).
-m	This optional argument states the maximum number of WORK_IDs to process from the WORKS table. When used, the script only attempts to process the number of rows specified, continuing from the last run (using the LAST_PROCESSED from the UTILITY_LOG table). If the "-m" argument is not specified then the script attempts to process all rows from the WORKS table, continuing from where the last run left off.

Notes

- If none of the above switches are specified, this would process the entire WORKS table. In practice Libraries are advised to process their entire WORKS table in small stages (of approximately 2000 Work Ids) until all have been processed.
- Libraries are strongly advised to run in report mode (i.e. without the **-u** argument) in order to gain an accurate impression of the number of Work Ids to be processed, and to understand what will happen to the Works identified. Libraries may then choose to run in update mode on a small range (or, indeed, individual Work Ids) by use of the **-b** and **-e** switches.

edi_inv_delete.pl

EDI invoices are imported into the LMS from suppliers using the **inv_import** script. If this script fails to process the entire contents of an input file (or creates incomplete EDI invoices) then it may (when run a 2nd time) fail to create new invoices, which would duplicate existing invoices on the system from the original import.

The **edi_inv_delete** script deletes the invoices from the database so that the inv_import script can re-import the data, or so any errors associated with a particular invoice can be addressed.

Usage

Log on as **talis** and enter the following command:

edi_inv_delete.pl -d<database> -h -p<filename> -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	A mandatory argument which specifies the name of the parameter file to be used.

Parameter file

The parameter file **edi_inv_delete.param** is located in the **/usr/opt/blcmp/data/utils** directory. The script will only delete invoices that match the selection criteria defined by the parameters. Accepted parameters are defined in the following table.

Parameter	Description
INVOICE_NUMBER	This parameter accepts invoice numbers. Only invoices with matching invoice numbers will be included in the report. You can separate multiple invoice numbers with a comma (for example, INVOICE_NUMBER=1001085,1001086,1001092). If entering multiple invoice numbers, ensure they are from the same supplier.
SUPPLIER_CODE	This parameter accepts a single supplier code. Ensure the supplier code is for the supplier linked to the specified invoice number(s).

Notes

- Note that the script will fail if:
  - Any number of incorrect invoice numbers are specified (or if invoice numbers are omitted entirely)
  - More than one (or an incorrect) supplier code is specified
  - The supplier code specified does not match the supplied invoice number(s)
- The script will not process
  - Invoices that are not of level 0,1,2,3 (i.e. non EDI invoices)
  - Invoices that are not 'deleted' status
  - Invoices that do not have mandatory Invoice Number and Supplier Code values specified.

email_post.pl

The **email_post.pl** script can process the output file produced by any MIS Letters query. It extracts letters containing an email address and sends them via email using standard UNIX mail facilities. Letters that do not contain an email address are written to a file for printing and posting in the usual manner.

Usage

Log on as **talis** and enter the following command:

email_post.pl -a<filename> -h -i<filename> -o<output filename> -p<filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This mandatory argument names the input file, i.e. the file of MIS letters. The input file will be the output file from a MIS Letters script. The default is usually "[letters_script_name].out" and the file(s) will normally be found in the $BLCMP_HOME/data/mis directory unless the "-s" argument has been used to specify an alternative output directory for the letters script.
-p	This mandatory argument specifies the parameter file, which should be created in the data directory. If this argument is not given, the default is "email_post.pa".

Parameter file

The parameter file should be set up in the **/usr/opt/blcmp/data/mis** directory if the script is to be run without the **-s** option on the command line. If you intend to use **-s** on the command line to name an alternative directory for the input/output files then you should create the parameter file in this alternative directory. Accepted parameters are described in the following table:

Parameter	Description
EMAIL_SUBJECT=	This mandatory parameter describes the text to appear on the Subject line of each email letter.
PAGELENGTH=	This optional parameter defines the page length of the letters in the input file. If this is not specified, the default is 66. PAGELENGTH=0 should be specified if the letters are paginated by form feeds rather than page length.
MAILER_PATH=	This optional parameter defines the UNIX path to the mail facility to be used. If this is not specified the default will be usr/lib/sendmail.
EMAIL_REPLYTO=	If you are using mailx software to send letters, you should specify a new parameter in the parameter file to specify an email address to which bounced messages and replies should be sent. For example: MAIL_REPLYTO=bob@talis.com
MAILER_PATH=	This optional parameter defines the UNIX path to the mail facility to be used. If this is not specified the default will be usr/lib/sendmail.

Notes

- When writing/editing parameter files, remember:
  - All values may be entered in upper or lower case.
  - Any characters following a hash "#" will be treated as a comment.
  - Blank lines are permitted.
- The script produces a report file "email_post.rep", containing the following information:
  - A Header with start date and time.
  - The number of letters processed.
  - The number of letters sent via EMail.
  - The number of letters written to the output file.
  - Footer with end date and time and completion message.

email_xfer.pl

The **email_xfer.pl** script can be used to copy borrower email addresses from their postal address to the new database structure. The script checks all ADDRESS.LINE entries and the ADDRESS.NOTE for an @ symbol, and transfers any email addresses accordingly.

Usage

Log on as **talis** and enter the following command:

email_xfer.pl -d<database> -t<TRANSFER/RETAIN> -r<directory> -u<update> -v<verbose>

Standard script arguments are described here. The remaining arguments for this script are described in the following table. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Argument	Description
-t	This mandatory argument is used to determine whether email addresses retrieved from the postal address are removed when transferred. Use -tTRANSFER to remove addresses, and use -tRETAIN to retain them in the postal address.

Notes

- If there are missing characters before or after the '@' symbol (or if there is more than one '@' symbol in the address) the email address will not be transferred. In addition, a warning message is displayed identifying the invalid email address.
- All email addresses, regardless of start and end dates, are transferred.
- The associated values of ADDRESS.NAME, ADDRESS.START_DATE and ADDRESS.END_DATE are also transferred.
- If there is an email address transferred from the default postal address, then this will become the default email address . If there is no email address in the default postal address, an email address from the postal address with the narrowest date range is used as the default.
- The email address Note field is not over-written when an email address is updated.

ffl_assign_links.pl

Since some systems need to be able to limit the use of individual funds to specific users, you can create links between operators and funds via the fund user profile.

For this reason, a script **ffl_assign_links.pl** allows you to link funds to fund user profiles in batch mode. You can specify exact fund codes or to assign fund user profiles to a set of funds with codes based on the same stem.

Systems with joint working require functionality that limits the use of individual funds to specific user(s).

Usage

Log on as **talis** and enter the following command:

ffl_assign_links.pl -p<filename> -d<database> -h -r<report directory> -s<data directory> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	Names the parameter file to be used in the script.

Parameter file

The parameter file defines funds to be linked to fund user profiles. Each row in the parameter file must be in the following format:

```
FUND_USER_PROFILE=
FUND_CODE=
FINANCIAL_YEAR=
```

Separate multiple values for each parameter by using a comma. Note that you can use the percentage symbol with the FUND_CODE parameter as a wildcard to assign fund user profiles to a group of funds with the same stem.

Notes

- The standard report will include the command line entered to run the script with start and end times. If the script is not being run in 'Update' mode (i.e. the –u argument is not specified in the command line) the report will state "Script is running in report mode – no funds will be updated".
- The report file will detail the fund codes to which the fund user profiles are to be assigned. This will be sorted by fund code, then financial year.

findlock

To avoid problems arising from different persons or processes attempting to update the same record(s) at the same time, each user - or process - is given undivided access to relevant record(s) they are using for the duration of particular transaction(s). Those records are said to be locked. It may occasionally happen that records are left in a locked state unintentionally; for example in the event of a system crash. There are two utilities which allow the System Manager to look for locked records on the database (findlock), and to unlock those records (unlock) either individually, in multiples or altogether.

Usage

To fine locked records, log on as **talis** and enter the following command:

findlock

The script returns the control number(s) of any locked records.

Notes

Provided that all users are logged out of the system, output from the findlock command can be piped directly into the unlock command, in order to find and unlock all records at once. To do this, enter the following command:

findlock | unlock.

fun_tot_base_exp.pl

The **fun_tot_base_exp.pl** script enables a flat file of Fund data to be produced from the LMS for use in other databases. The records in the output file are variable length, delimited by the newline character (HEX "0A"). They contain a fixed number of fields, each field delimited by the pipe character (HEX "7C").

The script reads the FUND table. For each Base Fund containing the financial year (or years) specified on the command line it will output:

- Fund code
- Expenditure code
- Financial Year
- Allocation
- Amount Carried Forward
- Total Outstanding Commitment
- Number of Items Committed
- Total Spent Value and
- Number of Items Paid for

The fun_totals.pl script should be run prior to running **fun_tot_base_exp.pl**, to ensure that the Committed and Spent totals in the Funds are correct.

Usage

Log on as **talis** and enter the following command:

fun_tot_base_exp.pl -h -n<financial year> -d<database> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument allows you to specify a particular financial year for the selection of Fund data. The year must be specified in four figure format, i.e. 2003 for the year "2003/04". If not specified, all years will be processed by default.

Notes

- The output file, **fun_tot_base_exp.out**, contains a line for each fund processed in the format:

  fund_code| expenditure_code| financial_year| allocation| carried_forward| committed_total|items_committed| spent_total| items_paid

  For example:

```
CLAV|WBC96023|1995|3000.00|0.00|1900.45|65|20.00|1
CLHBR|WBC89662|1995|1000.00|0.00|0.00|0|0.00|0
CLJNF||1995|8500.00|0.00|1040.65|145|120.50|15
```

fun_totals.pl

The **fun_totals.pl** script is used to sum the Total Committed and Total Spent, including Invoice level charges and allowances, for all Base Funds dealing with Orders, Open Orders and Inter-Library Loans. Aggregate Funds are not affected. This script should be run following Order or Serials migration and after rolling forward to a new Financial Year. It should also be run regularly to re-calculate Fund values in case of online errors or updating defects.

For each Base Fund/Financial Year combination, the Committed values against the Fund will be summed together, (excluding Items which are "Deleted", "Cancelled", or "Potential" and not counting Items which have been paid already). Similarly, the Spent values of each Fund will be summed together, (excluding Items which are deleted).

Usage

Log on as **talis** and enter the following command:

fun_totals.pl -d<database> -h<help> -p<filename> -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Argument	Description
-p	This mandatory argument gives the pathname of the parameter file to be used. The default directory for the parameter file is the $BLCMP_HOME/data/utils directory.

Parameter file

The **fun_totals.pl** script uses a parameter file to specify a set of variables. These variables are used to select the Funds to be updated. A default parameter file (**fun_totals.param.default**) is found in the **$BLCMP_HOME/data/utils** directory. This should be copied to another file, for example **fun_totals.param**. Libraries need to edit the new parameter file, to uncomment the parameters needed for use, and insert the appropriate values for those parameters.

Argument	Description
FINANCIAL_YEAR	The "FINANCIAL_YEAR" parameter is optional and may be used to specify the financial year(s) to be included in the processing. The value(s) specified relate

	to the "Display as" value in Utilities, Parameters, Rules, Acquisitions, Financial years. The value required is the display value, i.e. the same as appears online in Alto. If more than one financial year is specified, these must be separated by a comma. If no financial year is specified, it defaults to all financial years. The financial year(s) specified must be valid financial year(s) used in the database specified. If any of the years entered are not valid, an error message is reported when the script is run: `Invalid FINANCIAL_YEAR [xxx] specified in the parameter file` where [xxx] is the invalid financial year.
FUND_CODE	The "FUND_CODE" parameter is optional and may be used to specify the Fund(s) to be included in the processing. The format required is the Fund Code(s), for example "HIST,GEOG". If more than one Fund Code is specified, they must be separated by a comma. If no Fund Code is specified or if this parameter is commented out, all Funds are included. Fund Codes should be given in upper case, and must be valid for the database specified. If any of the Fund Codes entered are not valid, an error message is reported: `Invalid FUND_CODE [xxx] specified in the parameter file` where [xxx] is the invalid Fund Code.

Notes

- Subscriptions and Interloan charges are included when calculating commitment and expenditure.
- Subscriptions will not be included in the statistics if they are "Potential" or "Closed".
- Cancelled Items with payments are included in the Spent values of each fund and the number of Items paid.

grp_course_import.pl

Since many students now study more than one course, Alto allows you to link multiple courses to a single borrower record. The **grp_course_import.pl** script allows course details to be imported into the GROUPING table from an input file.

Usage

Log on as **talis** and enter the following command:

grp_course_import.pl -d<database> -h -i<filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	Names the input file. If this argument is not given then the default is **grp_course_import.in**

Input file

Rows in the input file must be in the format:

```
Code | Name | Current | Note
```

For more information refer to the following table

Element	Description
Code	Code must be a valid course code. A course code must be between 1 and 20 characters in length, made up of any alphanumeric plus the following

	characters: / _ - &. The script will convert any lower case characters to upper case.
Name	Name may be blank if required. Any name longer than 60 characters will be truncated to 60 characters.
Current	Current must be either 'T' to indicate an active course or 'F' to indicate an inactive course.
Note	Any note longer than 200 characters will be truncated to 200 characters.

Notes

- The script will process each row in the input file in turn, checking if the Code matches an existing course code in the GROUPING table.
- If a match is found the existing Name and Current data will be replaced by the data in the input file.
- If there is a Note in the row this will overwrite any existing note, otherwise an existing note will be retained.
- If no match is found a new row will be added.

ill_art_intray.pl

Status reports from the BLDSC are routed automatically to the Replies Intray mailbox. Regular email messages from the BLDSC keep the Library informed of progress or problems with the supply of requested materials. The "ill_art_intray.pl" script processes ARTEmail Replies Intray message files received from the BLDSC. It extracts data from these files and updates the database, adding reports to the relevant requests. For each Replies Intray message file processed successfully, the report file is updated and an output file is created.

All unprocessed Replies Intray message files ending in _REPLY (case sensitive) in the data directory are processed by default. It is possible to restrict processing to either a Replies Intray message file specified by name using the "-i" argument , or to certain BLDSC User Code(s) specified using the **-n** argument . (The latter option, allows multi-site Libraries to process Replies Intray files separately for each site).

Libraries wishing to perform a trial run of the **ill_art_intray.pl** script before actually processing ARTEmail Replies Intray messages are able to run without the update **-u** argument first.

Usage

Log on as **ill** and enter the following command:

ill_art_intray.pl -d<database> -h<help> -i<filename> -n<user code, user code> -p<filename> -r<report directory>
-s<data directory> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This optional argument may be used to specify the name of a single Replies Intray message file to be processed. If omitted, all unprocessed Replies Intray message files are processed.
-n	This optional argument may be used to specify which BLDSC User Code(s) are to be included in processing. This allows multi-site Libraries, whose sites are registered as separate BLDSC customers, to process Replies Intray files separately for each site. If omitted, all unprocessed Replies Intray message files are processed. Multiple User Code(s) should be separated by commas, for example: -n01734,10516,12584

	The above example would process files such as: 01734.10_06_17_15_02_REPLY 10516.10_06_17_15_02_REPLY 12584.1 0_06_17_15_02_REPLY etc. Note that the "-i" and "-n" switches are mutually exclusive.
-p	The argument "-p" is mandatory and must be followed by the name of a parameter file.

Parameter file

The script will look for the filename specified by the **-p** argument .

Example parameter file

```
REPORT_CODES_NOT_ADD=NUKL, FICHE, FILM
NO_LETTERS=CONF,THESIS,DUE WAIT
SUPPLIER=BLDSC
```

Parameter	Description
FILE_TYPE	This optional parameter, introduced in Alto 5.3, describes the format of the incoming file. Before the introduction of the BLDSC's new BLDSS system in 2012, libraries could request files in the Standard format, but BLDSS always sends files in the WIDE format. The FILE_TYPE parameter should be set to WIDE when the library moves over to the BLDSS system.
REPORT_CODES_ADD **REPORT_CODES_NOT_ADD**	When defining the BLDSC codes which are to be added to the database, two mutually exclusive parameters can be used, REPORT_CODES_ADD or REPORT_CODES_NOT_ADD. PEORT_CODES takes Report Codes as arguments. When a report code appears as an argument to this parameter then for each report code identified in the ARTTel intray file(s), a report is added to the database. Otherwise, the report does *not* get added to the database. REPORT_CODES_NOT_ADD takes Report Codes as arguments. When a report code appears as an argument to this parameter then for each report code identified in the ARTTel intray file(s), a report is not added to the database. Otherwise the report gets added to the database. If neither is specified then all Report Codes are added to the database.
NO_LETTERS	This optional parameter determines whether the report suppresses a letter from being sent. It accepts Interloan Report Codes as arguments.
SUPPLIER	This compulsory parameter specifies the Supplier Code used to represent the BLDSC. It accepts a single Supplier Code. This Supplier is added to each Interloan report created by this utility.

Notes

- If no files are found then an appropriate message is written to the ".rep" report file. If files are identified they are re-named, with a ".inprog" extension, so that processing on these can continue whilst other files are being imported.
- If a file simply contains the text NO REPLIES IN THIS TRANSMISSION and does not contain any Reply Code information then an entry is made in **ill_art_intray.pl.rep**. If a Request Number is

not found in the database then no further processing of that line takes place and a message is written to the output file.

- If more than one row matches a Request Number in the database, no further processing of that line takes place and a message is written to the output file. Provided a single match is made against the database, each Report Code on the line is processed.
- Data is extracted from each Replies Intray message file based on the identification of Report Codes. Multiple Interloan Report Codes can exist on a single line. A report code is ignored if defined as such by the parameters REPORT_CODES_ADD and REPORT_CODES_NOT_ADD. When a Reply Code is excluded by the arguments to REPORT_CODES_ADD or REPORT_CODES_NOT_ADD then no database amendments are made. Provided the report code is not excluded and exists on Alto, the database is updated.
- A Report Code is inserted into the Report field (ILL_REPORT) and the Note field (ILL_REQUEST) may be updated. If the Report Code is either "LOC" or "TRY" then any explanatory note that follows the code is inserted into the Note. If the text to be inserted will not fit into the 200 character Note field (because of existing notes) then the new note is not added, but the text is written to the output file.
- For ARTTel intray files containing 100 replies the script **ill_art_intray.pl** should complete under 5 minutes. The operation of the ARTTel Intray Reports facility requires that a number of new Report Codes are added to the Interloan Report Codes table. These Report Codes, do not need to be added manually, as they are loaded automatically.

inv_status_upd.pl

The **inv_status_upd.pl** script is available to change the status of Invoices. It is possible to specify the old and new Invoice statuses using the scripts parameter file (**inv_status_upd.param**) and to limit processing to a selected date range or just the financial year(s) and/or supplier code(s) specified.

Usage

Log on as **talis** and enter the following command:

inv_status_upd.pl -d<database> -h -p<filename> -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Argument	Description
-p	This mandatory argument which specifies the name of the parameter file to be used. The default parameter file inv_status_upd.param is located in the directory $BLCMP_HOME/data/utils.

Parameter file

The parameter file is used to specify a set of variables which govern processing by selecting the invoices to be updated by the script. The default parameter file has all of the possible parameters commented out. This should be copied to **inv_status_upd.param**, which should then be edited to requirements, by uncommenting the parameters needed for use, and inserting the appropriate values for the parameters.

The PRESENT_STATUS and NEW_STATUS parameters must be uncommented out and allocated Invoice Status Codes:

Parameter	Description
PRESENT_STATUS	This specifies the old Invoice status to be changed. Only a single Invoice Status Code may be specified.
NEW_STATUS	This specifies the new Invoice status to be applied. Only a single Invoice Status Code may be specified and this must differ from the code entered against the PRESENT_STATUS parameter.

START_DATE	This may be used to specify the Invoice Start Date to be processed. Dates are entered as DD/MM/YYYY. Processing is inclusive of this date.
END_DATE	This may be used to specify the Invoice End Date to be processed. If the START_DATE parameter is specified, and the END_DATE omitted, this defaults to the current date. Processing is inclusive of this date.
FINANCIAL_YEAR	This may be used to specify the financial year(s) to be included in the processing. If more than one financial year is specified, these must be separated by a comma. The financial year(s) specified must be valid in the current database specified by the "-d" argument . The value specified must correspond to the display value of the financial year as given in the same format found on the Fund Prompt Bar and Payment screens (i.e. the ACQUISITION_RULE.DISPLAY_VALUE, for example FINANCIAL_YEAR=2001/02, 2003/04). If a value is entered in the parameter file for this parameter, it is not possible to specify the START_DATE / END_DATE parameters at the same time, because these approaches to selection are mutually exclusive.
SUPPLIER_CODE	This may be used to specify the Supplier(s) to be included in the processing. If not specified, or if this parameter is commented out, this parameter defaults to all Suppliers. Note: Multiple Supplier Codes must be separated by a comma, for example: SUPPLIER_CODE=BLAZEBKS,COV,TRFC,TML All Supplier Codes specified must be valid in the current database.

Notes

- This script may be run more than once, but it will only process "converted" invoices.

imp_modify

There are numerous occasions in the management of the LMS that require rows to be entered (or modified) in the IMPORT_PARAMETER table. The **imp_modify** script is a general purpose utility for inserting or deleting rows in the IMPORT_PARAMETER table.

Usage

Log on as **talis** or **ops** and enter the following command:

imp_modify -h -u -d<database> -r<report directory> -t<modification type> -j<TYPE ID>] -k<VALUE 1> -l<VALUE_2>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-t	This mandatory argument specifies the type of modification to be performed. The two options are "INSERT" (to insert a row) and "DELETE" (to delete a row).
-j	This mandatory argument specifies the contents of the TYPE_ID attribute for the IMPORT_PARAMETER row that is to be modified. The TYPE_ID must be a number. For example, -j10 will refer to the update of a row relating to a BLCMP database file.
-k	This mandatory argument specifies the contents of the VALUE_1 attribute of the IMPORT_PARAMETER row that is to be inserted or deleted.

-l	This mandatory argument specifies the contents of the VALUE_2 attribute of the IMPORT_PARAMETER row that is to be inserted or deleted.

Notes

- If more than one word is to be entered into either the **-k** or **-l** arguments, the words should be separated by the plus character ("+"). The script will substitute the necessary blanks during the update.
- The mandatory **-j** argument must be followed by **9** and the argument **-l** must be followed by the **+** symbol. For example, the user input:

  imp_modify **-u -tINSERT -j9 -kITEM.CLASS_ID -l+**

  would suppress update of the CLASS_ID attribute of each Item. This should be used with care as it will prevent Items from being updated with the specified data.

- The values of attributes to be excluded are specified using the **-k** argument . Any of the following can be excluded:
  - ITEM.BARCODE
  - ITEM.VALUE
  - ITEM.ITEM_WANTS_NOTE
  - ITEM.ITEM_DESC_NOTE
  - ITEM.ITEM_GEN_NOTE
  - ITEM.SIZE_ID
  - ITEM.FORMAT_ID
  - ITEM.SEQUENCE_ID
  - ITEM.CLASS_ID
  - ITEM.SUFFIX

irs_compress

After a period of time, there may be a build-up of "obsolete" Interloan request sequences which can no longer be used as all of the request numbers in their range (including any spares) have been used. A utility called **irs_compress** removes Interloan request sequences which have been exhausted. When run, it deletes request sequences from the ILL_REQUEST_SEQUENCE table when the sequences contain no unused request numbers or spares.

Usage

Log on as **talis** or **ops** and enter the following command:

irs_compress -h -v -d<database>

Standard script arguments are described here.

ite_labels.pl

The **ite_labels.pl** script performs the display and physical printing of the spine labels and/or book labels. This script is passed arguments by the online Web interface.

For more information, refer to the Book Label Printing Release Notice under Talis WebOPAC located at the Talis Documentation web pages.

itp_seq_reset

The script "itp_seq_reset" may need to be run as a result of multiple insertions in the Issue Prediction Rows List online in Atlo. This is a batch script used to reset the interval between rows newly added to the Issue Prediction Rows List (i.e. when defining an Issue Prediction Sequence in Online Utilities). If seven or more new rows have been inserted into the sequence this will typically cause an overflow problem in the Issue Prediction Rows List.

Usage

Log on as **talis** and enter the following command:

itp_seq_reset -h -d<database> -n<sequence name>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument is mandatory as the script will fail without a name specifying the ISSUE_SEQUENCE for which the ISSUE_TEMPLATE rows will be reset. An error message will appear on the screen if the name of the Issue Prediction Sequence is not given.
-l	This argument is mandatory as the script will fail if a Delivery Site is not specified to indicate where the operator was attempting to check-in the Work. If omitted, an error message will appear on the screen informing you that a Delivery Site must be specified when running this script.

lo_compress.pl

Online loan transactions (issue, discharge and renew) add rows to the LOAN table. Additionally, when a loan incurs fine or hire charges rows are added to the following three tables:

- CHARGE_INCURRED
- CREDIT_VS_INCURRED
- BORROWER_CREDIT

If an overdue or recall is generated the LETTER_SNT table is updated. The Loan Compressor script, **lo_compress.pl**, removes completed loan transactions (i.e. loans which are discharged and which do not have fines or hire charges outstanding) from these tables. Users decide the select criteria used for deletion, using the script **loan_select**.

Usage

Before running the script:

- Carry out a database backup using full_dbdump.
- Check whether the BORROWER_CREDIT table is indexed using the isql command: **sp_helpindex BORROWER_CREDIT**

You are advised to set up the **lo_compress.pl** script to run automatically from the cron. To run it manually, log on as **talis** and enter the following command:

lo_compress.pl -h -q<no.> -u -d<database> -r<report directory> -z

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-q	The -q argument is mandatory. It names the query number to be used. This must be in the range 1-32 as described in the instructions for loan_select.
-z	The -z argument instructs the script to rebuild the indexes on the LOAN and other tables. This option can only be used in conjunction with -u.

Notes

- It is important to ensure that all users are logged off and no other batch jobs are running when executing **lo_compress.pl**, to ensure that there are no conflicts and that it completes successfully. The "kill_talis" and "kill_opac" procedures can be used to make sure all users are logged off.
- **lo_compress.pl** should always be followed by full_dbdump.
- The duration taken by "lo_compress.pl" to run will depend on several factors, for example the size your machine and the size of the LOAN table.
- It is worthwhile doing a test on the MIS server if you have one. This will give indications of the runtime although it may be slower than your main system if it is a lower spec machine. It will also test the select statement.
    - If "lo_compress.pl" is run without the "-z" argument , further jobs need to be executed for the gains in free disk space to become available for use. The following scripts to do this are held in the $TALIS_HOME/database/index directory:

    loan.index

    letter_snt.index

    charge_incurred.index

    credit_vs_incurred.index

    borrower_credit.index

    You are advised to set these to run regularly in the "cron" as soon as possible after lo_compress.pl has run. A significant amount of free disk space is required to run the loan.index job. To establish if you have sufficient space, run the top5 script.

loc_add_insert

The **loc_add_insert** script is designed to enable Library addresses to be entered easily, instead of using isql. The script overwrites any previous occurrence of the LOCATION in the database. If the LOCATION_ID specified exists already then the relevant address information will be displayed on the screen. You will be asked if you wish to continue entering new address information, thereby overwriting the existing address. After all the relevant address data has been entered, you will be reminded of the data you have entered and asked to confirm whether you wish to insert it into the database

Usage

Log on as **talis** and enter the following command:

loc_add_insert -h -d<database> -r<report directory>

The script uses standard script arguments.

The script will ask will ask a number of questions about the Library.

```
Please enter (in upper case) the LOCATION_ID:
Please enter text for LINE_1:
Please enter text for LINE_2:
Please enter text for LINE_3:
Please enter text for LINE_4:
Please enter text for LINE_5:
Please enter the Postcode:
Please enter the Telephone number:
Please enter the Telephone ext.:
Please enter the Fax number:
and finally the EMail number:
```

- Type in the information required against each prompt. Any of the above questions can be left blank, except for the LOCATION_ID (which is mandatory and must be specified in UPPER CASE).
- When you have completed the required input prompts, the script will present the details you have entered for checking and request confirmation of your intention to save them. Press "Y" to save the details as shown, or "N" to exit the script.
- Pressing "Y" at this point will overwrite any previous address that may already exist on the database for this Site's LOCATION_ID. If an address already exists, you must confirm whether you wish to continue entering new address information, thereby overwriting the existing address. Select 'Y' to do so, or press "N" to terminate the program and leave the original address unchanged.

itu_compress.pl

The **itu_compress** script is a database compressor for the ITEM_UPDATE table. It is used to delete rows from the ITEM_UPDATE table based on their date of creation. Libraries should run this script on a regular basis by including it in a daily or weekly 'cron', to keep the size of the table manageable.

Usage

Log on as **talis** and enter the following command:

itu_compress.pl -d<database> -e<number of days> -r<report directory> -h

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-e	This is a mandatory argument . The "-e" argument is used to specify the age of Item edits (expressed as a number of days before this script is run). Any ITEM_UPDATE rows will be deleted if their CREATE_DATE in ITEM_UPDATE exceeds this age. For example, "-e30" deletes all ITEM_UPDATE rows which have a create date which is 30 days or more earlier than the current date. The maximum value allowed is 999 days. If an invalid "-e" value is entered, the script will abort processing and report: `ERROR: script aborted - -e(value given) invalid. Should be in the range 1-999`

Notes

- Even Libraries not utilising the ITEM_UPDATE table data should set up regular compression runs in order to keep this file to a manageable size, as it will be created automatically. The ITEM_UPDATE table grows at a rate similar to the ITEM table.

itu_update_wku.pl

All Item transactions (insert, update or delete) trigger the entry of a row in the ITEM_UPDATE table. The batch script **itu_update_wku.pl** is run against the ITEM_UPDATE table as a regular process (probably set up as an overnight 'cron' job) to generate rows in the WORK_UPDATE table. This allows the OPAC "make_collections" software to remove or add updated Items to specific local catalogues as appropriate.

Usage

Log on as **talis** and enter the following command:

itu_update_wku.pl -<database> -h -m<max to process> -r<report directory> -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-m	If the "-m" argument is used, the script only processes the specified number of rows, starting from where the previous run left-off. This information is held in the UTILITY_LOG table. The number selected continues from the last row processed in any previous run. All remaining rows are processed if the "-m" value specified is greater than the number of remaining unprocessed rows. If the "-m" argument is not given, the script processes all rows remaining in the ITEM_UPDATE table, continuing from the last run.

Notes

- If there is already a row in the WORK_UPDATE table with a Work status equal to any of the following statuses:
  `0, 10, 30, 40, 100, 130, 140`

a row will not be added to WORK_UPDATE. An Item will not be processed if it is for the same WORK_ID as a row processed previously; this enhances performance.

- The "itu_update_wku.pl" script is able to select and process 1,000 Works, of which 500 require WORK_UPDATE inserts, per minute.

linkuk_cat_update.pl

The linkuk_cat_update.pl script, which is derived from the rlb_non_isbn.pl script, allows libraries to create files of holdings records for export to LinkUK. Only monograph records with an ISBN, BNB or Library of Congress control number are reported. The script does not currently handle ISBN-13 control numbers.

The script should be run regularly to produce notifications of all stock changes where the first copy has been added or the last copy has been deleted since the last run. It can also be used to report on all items or to produce a subset of holdings data limited by site, item status and/or item type.

The records are output in the format specified by OCLC PICA in June 2005.

Usage

Log on as **talis** and enter the following command:

linkuk_cat_update.pl –d<database> -h –p<filename> -r<report directory> -s<output directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory switch specifies the parameter file. The parameter file must be located in the data directory (see below).

Parameter file

The parameter file must be located in the data directory. The parameters are case-insensitive. There is a default parameter file, linkuk_cat_update.param.default, in '/usr/opt/blcmp/data/expdir' which can be copied to create the file.

The valid parameters are described in the following table.

Parameter	Options
LIST=	This optional parameter controls how the LIST_VALUES= option will work. Set it to FI if output is to be created from a file of WORK_IDS. Set it to FC if output is to be created from a file of control numbers. Set it to I if output is to be created from a list of WORK_IDs specified in the LIST_VALUES= parameter. Set it to C if output is to be created from a list of control numbers specified in the LIST_VALUES= parameter. If any of these values is set the script will only process works in the file or list and will ignore all other optional parameters except REFERENCE_TYPES. If this parameter is set to N or is not set, the script will ignore the LIST_VALUES= parameter.
LIST_VALUES=	This parameter is used in conjunction with the LIST parameter above. ▪ If the value 'I' is specified in the LIST parameter, then a comma-separated list of WORK_IDs should be specified here. ▪ If the value 'C' is specified in the LIST parameter, then a comma-

	separated list of control numbers should be specified here. ▪ If the 'FI' value is given in the LIST parameter then the name of a file that contains a list of WORK_IDs or control numbers should be given here. This file should be located in the data directory. ▪ If the 'FC' value is given in the LIST parameter then the name of a file that contains a list of control numbers should be given here. This file should be located in the data directory. In all cases the output will only contain records with valid control numbers. Only 10-digit ISBNs should be entered; the script cannot currently handle ISBN-13 control numbers.
MODE=	This optional parameter determines the content of the output file to be created. There are three possible options: • ADL produces a file of holdings records for which the first item has been added or the last item has been either withdrawn or changed to a not in stock status since the script was last run. • FDA produces a full dump of holdings records for all Items • FLT produces a file of holdings records for all Items that match the Status, Type and Location specified using the LOCATION, ITEM_STATUS and ITEM_TYPE parameters. The default is ADL. When MODE=ADL, the statuses that represent 'in stock' or 'out of stock' should be specified using the TAL_IN_STOCK or TAL_NOT_IN_STOCK environment variable. The variable should be set in the .profile of the talis user. If neither is set, IS is assumed to be the only 'in stock' status. Note: A full database dump can be produced by specifying MODE=FDA in the parameter file. This will use the WORKS table. Note that only Works with Items in the ITEM table will be included in the output files. You should use the ITEM_STATUS parameter to exclude works that have only deleted items, for example, attached.
LOCATION=	This optional parameter can be used to restrict the selection to Items that belong to a specific site. A list of comma-separated site codes can be entered. If no sites are specified then all sites will be selected. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
ITEM_STATUS=	This optional parameter can be used to restrict selection to particular item statuses. A list of comma-separated status codes may be entered (e.g. REC,IS). If no statuses are specified then all statuses will be selected. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
ITEM_TYPE=	This optional parameter can be used to restrict selection to particular item types. A list of comma-separated type codes may be entered (e.g. AF,ANF,JF,JNF). If no types are specified then all types will be selected. The list must include any items types that may be listed under the REFERENCE_TYPES parameter. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
REFERENCE_TYPES=	This optional parameter may be used specify which item types should be treated as reference stock. Item Type codes should be specified,

	separated by a comma. If none is specified then all Item Types are assumed to be lending. If the ITEM_TYPE parameter is used to limit the items types processed by the script then the REFERENCE_TYPES parameter values must also be listed in the ITEM_TYPE parameter.
LOC_CODE=	This mandatory parameter specifies a four-digit library code. This comprises your one-character region code followed by your 3-digit library number. The region code character must be one of the following: D = West Midlands F = Original LASER area H = Wales C = South West

Notes

- The script will automatically exclude certain types of work. Namely:
  - Serial records
  - Analytical 'child' records
  - Multipart monograph 'parent' records
  - Series 'parent' records
  - ILL request records (i.e. Works where the WORK_ID exists in the ILL_REQUEST table)

loa_plr_retrieve.pl

The PLR organisation cumulates the information contained in issue data samples provided by a number of Public Libraries and makes appropriate payments to the authors. The PLR software comprises two scripts:

- loa_plr_retrieve.pl (described below)
- loa_plr_tape

The **loa_plr_retrieve.pl** script selects and processes data relating to loans and renewals that have occurred over a particular date range. The script produces three output files that have the extensions: **.1.out** , **.2.out** and **.3.out**. **.1.out** contains header information. **.2.out** consists of records that contain information on the number of times that Items linked to a particular Work have been issued during the date range specified. **.3.out** contains trailer information.

You should communicate with the PLR organisation to determine the frequency with which the script should be run and the size of the date range.

Usage

It is an offline script which will take up a considerable portion of the processing power of the server, so it is not advisable to run it while Alto is available, or while other offline scripts are running.

Log on as **talis** and enter the following command:

loa_plr_retrieve.pl -h -d<database> -b<begin date> -e<end date> -o<output file prefix> -p<filename>
-r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument is used to specify the begin date for loans and renewals to be included in the data sample. The date should be entered in the format "DD/MM/YY".

-e	This mandatory argument specifies the end date for loans and renewals to be included in the data sample. The date should also be entered in the format "DD/MM/YY".
-o	This argument names the output file prefix. Three output files will be produced. The name of each will consist of the output file prefix plus ".1.out", ".2.out" and "3.out". If not given then the default file prefix is "loa_plr_retrieve".
-p	The argument "-p" names the parameter file. The parameter file will be located in the data directory. This is a mandatory argument .

Parameter file

The script is case insensitive to the contents of the parameter file. All text following a hash character ("#") on a particular line will be treated as a comment. The parameter file may contain the following labels:

Parameter	Options
AUTHORITY_ CODE=	**Mandatory**. **Not repeatable**. **Contents**: A one or two character numeric code supplied by the PLR organisation, to identify the Library to the PLR organisation.
DELETE_PREVIOUS_OUTPUT=	**Mandatory.** **Not repeatable.** **Contents: "Y"** or **"YES"** or **"N"** or **"NO"**. If the label content is "Y" or "YES", files in the scratch directory with names that begin with the output file prefix and end with ".1.out", ".2.out" or ".3.out" will be recreated. If the label content is "N" or "NO", existing files matching the above description will have a date and time stamp appended to their filenames.
COPIES_IN_AUTHORITY=	**Mandatory.** **Not repeatable.** **Contents: "1"** or **"2".** This label controls the calculation method used to determine one of the total fields in the records in file "2".
BORR_TYPE_IN=	**Optional.** **Repeatable.** **Contents:** A valid Borrower type code. The script will only process issues relating to Borrowers of the type(s) indicated by the Borrower type codes entered in these labels. The use of this label cannot be combined with the use of the BORR_TYPE_OUT label.
BORR_TYPE_OUT=	**Optional.** **Repeatable.** **Contents:** A valid Borrower type code. The script will process issues relating to Borrowers of all types except those indicated by the Borrower type codes entered in these labels. The use of this label cannot be combined with the use of the BORR_TYPE_IN label.

ITEM_TYPE_IN=	**Optional.** **Repeatable.** **Contents:** A valid Item type code. The script will only process issues relating to Items of the type indicated by the Item type codes entered in these labels. The use of this label cannot be combined with the use of the ITEM_TYPE_OUT label.
ITEM_TYPE_OUT=	**Optional.** **Repeatable.** **Contents:** a valid Item type code. The script will process issues relating to Items of all types except those indicated by the Item type codes entered in these labels. The use of this label cannot be combined with the use of the ITEM_TYPE_IN label.
LOCATION_IN=	**Optional.** **Repeatable.** **Contents:** A valid location code. The script will only process issues relating to Items with "create" locations indicated by the location codes entered in these labels. The use of this label cannot be combined with the use of the LOCATION_OUT label.
LOCATION_OUT=	**Optional.** **Repeatable.** **Contents:** A valid location code. The use of this label cannot be combined with the use of the LOCATION_IN label.

Example

An example parameter file is shown below.

```
# Parameter file for the loa_plr_retrieve.pl script
# Created 10/01/1996NAB
# Edited 12/01/1996TWB

AUTHORITY_CODE=12
DELETE_PREVIOUS_OUTPUT=Y
COPIES_IN_AUTHORITY=1
BORR_TYPE_OUT=BIND# binding
ITEM_TYPE_OUT=CA1
ITEM_TYPE_OUT=CA2
LOCATION_IN=TR
LOCATION_IN=ST
LOCATION_IN=RS
```

Notes

- If neither BORR_TYPE_IN nor BORR_TYPE_OUT labels are entered in the parameter file, the selection will not be restricted according to the Borrower type. This principle also applies to ITEM_TYPE_IN and ITEM_TYPE_OUT and to LOCATION_IN and LOCATION_OUT.
- The Library should communicate with the PLR organisation in order to determine the contents of the following labels: AUTHORITY_CODE
  COPIES_IN_AUTHORITY
  BORR_TYPE_IN or BORR_TYPE_OUT
  ITEM_TYPE_IN or ITEM_TYPE_OUT
  LOCATION_IN or LOCATION_OUT

loa_plr_retrieve_lyra

The PLR organisation cumulates the information contained in issue data samples provided by a number of Public Libraries and makes appropriate payments to the authors.

The loa_plr_retrieve_lyra.pl script selects and processes data relating to loans and renewals that have occurred over a particular date range. The script produces an output file that can then be sent to the PLR Organisation.

The output file will contain a first line that contains the authority code and the start and end dates for the period.

Each work will be listed showing the ISBN, Issues in Period, Copies in Authority, Contributor Code and Item or Material type.

The ISBN displayed will only be a valid ISBN10 or ISBN13 number. If the TalisMARC control number for a work is not one of these two types of number then any other numbers associated with the work are checked. If one of these associated numbers is a valid ISBN10 or ISBN13 the first valid number is used. If no valid ISBN10 or ISBN13 is found then the work will not be reported.

Note

It is possible for this reason that the same ISBN will be reported more than once. Any duplication will be listed in the report file and duplicate numbers will be next to each other in the output file. The PLR organisation is aware of this possible duplication and will deal with any such duplication as part of their processing.

A Contributor Code will be retrieved where possible from the database or **** where no author name exists. The Material Type will be retrieved from the physical medium associated with the work.

The last row in the output file will contain the count of all records in the file and the count of all issues for the records.

All data on a row is separated by a | symbol and each row of data is ended with a Carriage Return. An example output would be:

24|01072003|31072003|
000642139X|94|5|SMIT|BK
1234567890|26|4|JONE|SPO
1234567891|2|1|ANDERSON|24
3|122|

You should communicate with the PLR organisation to determine the frequency with which the script should be run and the size of the date range.

Usage

It is an offline script which will take up a considerable portion of the processing power of the server, so it is not advisable to run it while Alto is available, or while other offline scripts are running.

Log on as talis and enter the following command:

loa_plr_retrieve_lyra.pl -h -d<database> -b<begin date> -e<end date> -o<output file prefix> -p<filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument is used to specify the begin date for loans and renewals to be included in the data sample. The date should be entered in the format "DD/MM/YY".
-e	This mandatory argument specifies the end date for loans and renewals to be included in the data sample. The date should also be entered in the format "DD/MM/YY".
-o	This argument names the output file. If not given then the default file name is "loa_plr_retrieve_lyra".
-p	The argument "-p" names the parameter file. The parameter file will be located in the data directory. This is a mandatory argument.

Parameter file

A default parameter file, **loa_plr_retrieve_lyra.param.default**, will be shipped to **$BLCMP_HOME/data/utils** directory. A copy of this file should be made called **loa_plr_retrieve_lyra.param** and this file updated with the required parameter values. The script is case insensitive to the contents of the parameter file. All text following a hash character ("#") on a particular line will be treated as a comment. The parameter file may contain the following labels:

Parameter	Options
AUTHORITY_ CODE=	**Mandatory.** **Not repeatable.** Contents: A one or two character numeric code supplied by the PLR organisation, to identify the Library to the PLR organisation.
DELETE_PREVIOUS_OUTPUT=	**Mandatory.** **Not repeatable.** Contents: "Y" or "YES" or "N" or "NO". If the label content is "N" or "NO", existing files matching the above description will have a date and time stamp appended to their filenames. It is advised to run with the "N" or "No" value.
COPIES_IN_AUTHORITY=	**Mandatory.** **Not repeatable.** Contents: "1" or "2". Value 1 should be used in all cases.
BORR_TYPE_IN=	**Optional.** **Repeatable.** Contents: A valid Borrower type code. The script will only process issues relating to Borrowers of the type(s) indicated by the Borrower type codes entered in these labels. The use of this label cannot be combined with the use of the BORR_TYPE_OUT label.
BORR_TYPE_OUT=	**Optional.** **Repeatable.** Contents: A valid Borrower type code. The script will process issues relating to Borrowers of all types except those indicated by the Borrower type codes entered in these labels. The use of this label cannot be combined with the use of the BORR_TYPE_IN label.
ITEM_TYPE_IN=	**Optional.** **Repeatable.** Contents: A valid Item type code. The script will only process issues relating to Items of the type

	indicated by the Item type codes entered in these labels. The use of this label cannot be combined with the use of the ITEM_TYPE_OUT label.
ITEM_TYPE_OUT=	**Optional.** **Repeatable.** Contents: a valid Item type code. The script will process issues relating to Items of all types except those indicated by the Item type codes entered in these labels. The use of this label cannot be combined with the use of the ITEM_TYPE_IN label.
LOCATION_IN=	**Optional.** **Repeatable.** Contents: A valid location code. The script will only process issues relating to Items with "create" locations indicated by the location codes entered in these labels. The use of this label cannot be combined with the use of the LOCATION_OUT label.
LOCATION_OUT=	**Optional.** **Repeatable.** Contents: A valid location code. The use of this label cannot be combined with the use of the LOCATION_IN label.

Example

An example parameter file is shown below.

```
# Parameter file for the loa_plr_retrieve.pl script
# Created 10/01/1996NAB
# Edited 12/01/1996TWB
AUTHORITY_CODE=12
DELETE_PREVIOUS_OUTPUT=N
COPIES_IN_AUTHORITY=1
BORR_TYPE_OUT=BIND# binding
ITEM_TYPE_OUT=CA1
ITEM_TYPE_OUT=CA2
LOCATION_IN=TR
LOCATION_IN=ST
LOCATION_IN=RS
```

Notes

- If neither BORR_TYPE_IN nor BORR_TYPE_OUT labels are entered in the parameter file, the selection will not be restricted according to the Borrower type. This principle also applies to ITEM_TYPE_IN and ITEM_TYPE_OUT and to LOCATION_IN and LOCATION_OUT.

- The Library should communicate with the PLR organisation in order to determine the contents of the following labels: AUTHORITY_CODE
  COPIES_IN_AUTHORITY
  BORR_TYPE_IN or BORR_TYPE_OUT
  ITEM_TYPE_IN or ITEM_TYPE_OUT
  LOCATION_IN or LOCATION_OUT

loa_plr_tape

The PLR organisation cumulates the information contained in issue data samples provided by a number of Public Libraries and makes appropriate payments to the authors. The PLR software comprises two scripts:

- loa_plr_retrieve.pl
- loa_plr_tape (described below)

The **loa_plr_tape** script writes the "1", "2" and "3" .out files to a cartridge, which should then be sent to the PLR organisation.

Usage

Log on as **talis** and enter the following command:

loa_plr_tape -h -o<output device> -r<directory> -s<directory> -i<input file prefix>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	If used, the argument "-i" names the file prefix used for the three output files produced by "loa_plr_retrieve.pl". If not given, this defaults to "loa_plr_retrieve".
-o	The argument "-o" names the output device. If not given, this defaults to "/dev/rmt/0".

load_authority_tags.pl

The AUTHORITY_TAG table must contain the tags which are to be authorised. The **load_authority_tags** script, located in **/usr/opt/blcmp/talis/database/scripts**, calls a subordinate Perl script. The switches given for load_authority_tags are applied to the Perl script.

The script must be run interactively from the **/usr/opt/blcmp/talis/database/scripts** directory, and never from the "cron", because operator input is required.

Usage

Log on as **talis** and enter the following command:

load_authority_tags -d<database> -h -n<type code> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This mandatory argument specifies the Authority Type to be applied to the tags added. The Authority Type Code should be given (not the Name). Only one Authority Type may be specified. The relevant Authority Type must have been set up already.

Notes

- The script determines the Authority format associated with the Authority Type specified and loads the corresponding file of default tag data. If the format is "Name" then **att.name.def.data** file is loaded. If the format is "Title" then **att.title.def.data** is loaded. If the format is "Subject" then **att.subject.def.data** is loaded.

loan_select

Online loan transactions (issue, discharge and renew) add rows to the LOAN table. The Loan Compressor script, **lo_compress.pl**, removes completed loan transactions (i.e. loans which are discharged and which do not have fines or hire charges outstanding) from these tables.

Users decide the select criteria used for deletion, using the script **loan_select**. The archiving and compression of data is achieved by running **lo_compress.pl**. The tables concerned must be re-indexed for the disk space freed by the removal of rows to be made available for use. This may be done at the time **lo_compress.pl** is run or as a separate job.

Note there are no arguments for this script.

Usage

To run loan_select, log on as **talis** and enter the following command:

loan_select

And follow the screen output.

Loan compression options

When the script is run, you are prompted to enter the database name (by default **prod_talis**). The Manage Loan Compression screen allows you to chose any of the following three options.

Option	Description

1) Create selection query	This option allows up to 32 selection queries to be created and subsequently edited. The procedure having selected this menu option is as follows:
	
	Enter the number of the query to edit (between 1 and 32).
	
	If the query already exists it will be displayed. You will be prompted:
	
	`Do you wish to change this query using vi ? (y or n):`
	
	Choosing "y" will place you in a "vi" edit session.
	
	If the query does not exist, the system may supply the first line:
	
	`select LOAN_ID from LOAN where LOAN_ID = @loan_id`
	
	If not supplied it needs to be entered exactly as shown above. The user-defined SELECT criteria must be appended to this text using an AND clause. It is most efficient to select on a range of LOAN_IDs. For example:
	
	`and LOAN_ID > 50000 and LOAN_ID < 1000000`
	
	This above example would delete loans between the two LOAN_IDs specified.
	
	After completing the "vi" edit session the next prompt will ask:
	
	`Do you wish to update your database ? (y or n):`
	
	Choose "y" to add the query to the database as a stored procedure.
	
	Select examples
	
	The select:
	
	`and CREATE_LOCATION='HW'`
	
	would delete all loans at the site whose code is "HW".
	
	The select:
	
	`and LOAN_TYPE in (select LOAN_TYPE from LOAN_TYPE_GROUP where NAME='Junior loan')`
	
	would delete all loan types with the name of the loan type specified.

2) Define query on database	Selecting option 2 prompts for a query number. The output then displays the following:
	
	`Query number: n`
	`Name:`
	`Procedure: loan_compress_n`
	
	Where "n" is the query number. It then prompts:

	`Do you wish to change this query? (y or n):`
	
	Entering "y" gives you the option to give the query a name and a note (which are useful when listing queries using menu Option 3). It also leads to the prompt:
	
	`Do you wish to update your database ? (y or n):`
	
	Choosing y updates the COMPRESSION table, making the query accessible to lo_compress.pl.

3) List queries on database	Choosing option 3 will list the number, name and note of each query.
-a	The -a argument allows you to append the output file produced by the script, rather than overwriting it. Validation i.e. the filename must already exist.
-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.

Notes

- It is possible to create select criteria which result in the Sybase runtime environment being unable to complete successfully. If this happens, **lo_compress.pl** will not drop any rows and will report an error message:

  `Insufficient SYSLOGS space : nnnnn bytes`

  Such messages may take some time to appear since **lo_compress** has to select the rows first before it can calculate how much log space is required. In such cases, the solution is to restrict the select criteria further.

marcdiag

Works may be printed offline from the LMS in MARC diagnostic format using the **marcdiag** script. Records may be printed individually by entering the required Control Number. Note that this does not print MARC 21 data. It only picks up the TalisMARC form of the record.

Usage

Log on as **talis** and enter the following command:

marcdiag -d<database> -s<server name> -w<page width> -v<level> <control number>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-s	This argument allows the user to specify an alternative server name.
-w	This argument may be used specify an alternative page width, given in characters.
-v	The Level argument is used to specify the Level of error messaging to be provided, with "-v0" (Level 0) being the lowest and "-v3" (Level 3) being the highest.
<control number>	Enter the control number of the work to be printed.

new_item_exp

THIS SCRIPT IN /usr/opt/blcmp/talis/bin

The **new_item_exp** script is used to export MARC records from the database. UK MARC standard records are output to tape in SPANNED format. The data extraction stage selects all Works from the database which have Items that have been either created or edited since the extract program was last run. Works will only be selected at the data extraction stage providing the Items attached to them have been included in a list of specifically named Sites.

The list of Sites from which records are to be included in the export process needs to be tailored to your local requirements. The file **new_item_exp_sites** is used to record the list of Site codes for the bibliographic records eligible for extraction from the database. This file has to be created in the **/scratch/uk_marc** directory.

Site codes should be stated in the same case as used on the local system (usually upper case), enclosed in single quotes, and separated by commas (optionally with a space following the commas). For example:

'AB', 'AC', 'AD', 'AE'

Running MARC export (first time only)

The MARC extract software selects Works from the database which have Items that have been created or edited since the extract program was last run. The date/time when extract was last run is held in a file called **last_new_item_exp** in the **/scratch/uk_marc** directory. This file is created when the extract software is run for the first time, and gets updated automatically each time the extract is run subsequently to reflect the current date/time. (Running the write to tape software does not influence this file).

When running the extract for the first time no records will be located, because **last_new_item_exp** is set to the current date/time. For this reason it is necessary to edit this file using "vi", the UNIX text editor in order to specify the appropriate date/time to be taken as the official starting point. You may then run the extract software again.

Avoid setting the date in the **last_new_item_exp** file to more than one week before the current date, in order to avoid hitting large number of records which need exporting.

Usage

Log on as **talis** and enter the following commands:

new_item_exp -d<database> -p<pathname> -e -w

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	Specifies the path for the extract reports and the output file "new_item_exp_extract". (The default is "/scratch/uk_marc")
-e	Extracts Work records only.
-w	Writes output to tape only.

Notes

- The script may be run without arguments. Running this script without parameters is equivalent to running "new_item_exp -e -w", in that records will be extracted from the database and output to tape.
- The system will retain two copies of the interim output file i.e. normally the last two days' output. The most recent output is always written to "new_item_exp_extract". When the extract software is run again, the new output will be written to the same file, over-writing the previous output. The previous version of this file will have been copied to the backup file first, called "new_item_exp_extract_old".
- The default path to which extract-related reports and the main output file "new_item_exp_extract" will be written is "/scratch/uk_marc. This subdirectory has to be created.

- Items with a status of "Deleted" or "Cancelled" will be excluded by the data extraction procedures.
- Extracted UK MARC records will be written to an interim file, called "new_item_exp_extract".

oll_pass_reset.pl

When passwords are forgotten, the only way forward is to reset them and re-allocate new ones. A utility called **oll_pass_reset.pl** removes redundant and forgotten passwords no longer required for LMS operator override.

Usage

Log on as **talis** or **ops**:

oll_pass_reset.pl -h -d<database> -n<operator profile code> -q<site profile code> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This mandatory argument requires a valid operator profile code.
-q	This mandatory argument requires a valid site profile code.

oclc_pica_update.pl

The **oclc_pica_update.pl** script, supersedes the linkuk_cat_update.pl script. It allows libraries to create files of holdings records for export to LinkUK and UnityUK.

Only monograph records with an ISBN, BNB or Library of Congress control number are reported. The script now handles ISBN-13 control numbers.

The script should be run regularly to produce notifications of all stock changes where the first copy has been added or the last copy has been deleted since the last run. It can also be used to report on all items or to produce a subset of holdings data limited by site, item status and/or item type.

The script produces an output file in the format

OCLC<location code><mode>.<submission number>

For example: OCLC2040FLT.001, OCLC2040FDA.002, OCLC2040ADL.003

The final name may need to be changed for submitting to LinkUK or UnityUK.

The records are output in the format specified by OCLC PICA in November 2007.

Usage

Log on as **talis** and enter the following command:

oclc_pica_update.pl –d<database> -h –p<filename> -r<report directory> -s<output directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory switch specifies the parameter file. The parameter file must be located in the data directory (see below).

Parameter file

The parameter file must be located in the data directory. The parameters are case-insensitive. There is a default parameter file, **oclc_pica_update.param.default** in **/usr/opt/blcmp/data/expdir** which should be copied to create the file **oclc_pica_update.param**.

The valid parameters are described in the following table.

Parameters	Description
LIST	This optional parameter controls how the LIST_VALUES= option will work. Set it to FI if output is to be created from a file of WORK_IDS. Set it to FC if output is to be created from a file of control numbers. Set it to I if output is to be created from a list of WORK_IDs specified in the LIST_VALUES= parameter. Set it to C if output is to be created from a list of control numbers specified in the LIST_VALUES= parameter. If any of these values is set the script will only process works in the file or list and will ignore all other optional parameters except REFERENCE_TYPES. If this parameter is set to N or is not set, the script will ignore the LIST_VALUES= parameter.
MODE	This optional parameter determines the content of the output file to be created. There are three possible options: • ADL produces a file of holdings records for which the first item has been added or the last item has been either withdrawn or changed to a not in stock status since the script was last run. • FDA produces a full dump of holdings records for all Items • FLT produces a file of holdings records for all Items that match the Status, Type and Location specified using the LOCATION, ITEM_STATUS and ITEM_TYPE parameters. The default is ADL. When MODE=ADL, the statuses that represent 'in stock' or 'out of stock' should be specified using the TAL_IN_STOCK or TAL_NOT_IN_STOCK environment variable. The variable should be set in the .profile of the talis user. If neither is set, IS is assumed to be the only 'in stock' status. Note: A full database dump can be produced by specifying MODE=FDA in the parameter file. This will use the WORKS table. Note that only Works with Items in the ITEM table will be included in the output files. You should use the ITEM_STATUS parameter to exclude works that have only deleted items, for example, attached.
LOCATION	This optional parameter can be used to restrict the selection to Items that belong to a specific site. A list of comma-separated site codes can be entered. If no sites are specified then all sites will be selected. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
ITEM_STATUS	This optional parameter can be used to restrict selection to particular item statuses. A list of comma-separated status codes may be entered (e.g. REC,IS). If no statuses are specified then all statuses will be selected. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
ITEM_TYPE	This optional parameter can be used to restrict selection to particular item types. A list of comma-separated type codes may be entered (e.g.

	AF,ANF,JF,JNF). If no types are specified then all types will be selected. The list must include any items types that may be listed under the REFERENCE_TYPES parameter. Note: This parameter should only be used in conjunction with the MODE=FLT option. If you use it with the FDA or ADL option the output will not be accurate.
REFERENCE_TYPES	This optional parameter may be used specify which item types should be treated as reference stock. Item Type codes should be specified, separated by a comma. If none is specified then all Item Types are assumed to be lending. If the ITEM_TYPE parameter is used to limit the items types processed by the script then the REFERENCE_TYPES parameter values must also be listed in the ITEM_TYPE parameter.
LOC_CODE=	This mandatory parameter specifies a four-character library code. This comprises your one alphanumeric character region code followed by your 3-digit library number.

Notes

The script will automatically exclude certain types of work. Namely:

- Serial records
- Analytical 'child' records
- Multipart monograph 'parent' records
- Series 'parent' records
- ILL request records (i.e. Works where the WORK_ID exists in the ILL_REQUEST table)

orr_ack_imp

This script is found in **/usr/opt/blcmp/talis/bin**

The **orr_ack_imp** utility enables Order acknowledgement reports from Book Suppliers to be imported into Alto and attached to Order records. Acknowledgement reports will be produced by a Book Supplier if an exceptional Order condition has been found, for example:

- Out of print
- Remaindered

The Book Supplier will send acknowledgements to BLCMP, who will collate all acknowledgements from all participating Suppliers and then send one file to each Library on a daily basis (providing there are acknowledgements for that Library).

All incoming acknowledgements are appended to the **orr_ack_imp.in** file. The **orr_ack_imp** utility processes each acknowledgement from the import file and creates a Report for each relevant Order.

Usage

Log on as **talis** and enter the following command:

orr_ack_imp -h -d<database> -i<filename> -m<max number to process> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This argument specifies the name of the import file, which defaults to "orr_ack_imp.in". When this option is not used, transmitted files matching the template "orr_ack_imp.trans." will be searched for in the directory pointed at

	by the symbolic link directory /usr/opt/blcmp/data/impdir. If more than one file is found, the transmitted files will be concatenated together. The resulting file will then be renamed to "orr_ack_imp.in.DD_MM_HH_MM_SS", where "DD_MM_HH_MM_SS" represents the date and time when the script was run. The report file created will have exactly the same suffix, namely "orr_ack_imp.rep.DD_MM_HH_MM_SS"
-m	This argument specifies the maximum number of normal Order Reports to process in this run. If this option is used, processing will commence from the next Report onwards from where the previous run of "orr_ack_imp" finished. This script will process the specified number of Reports. If not specified, this defaults to processing to the end of the file. For this option to function correctly it is essential that the report file ("orr_ack_imp.rep") from the previous run is left intact

Notes

- The "orr_ack_imp" script generates three reports, orr_ack_imp.rep, orr_ack_imp.log, and orr_ack_imp.err, which are written to the /usr/opt/blcmp/talis/reports directory. These should be checked regularly, at least daily to monitor processing.

orr_confirm.pl

The **orr_confirm.pl** script identifies Proposed Orders, verifies the Orders, performs the necessary updating of the database, and queues the Orders for transmission via EDI to the Supplier. It performs the same validations and database updates which would occur if an Alto user were to confirm a Proposed Order online, by verifying it and inserting an Order Date. It checks that the relevant Funds are active and are not overspent or overcommitted, and that the Supplier is active. Updates to several tables occur. Each Order is treated as a unit, and all database updates for each Order must be successful.

The **orr_confirm.pl** script uses a parameter file. The default parameter file, called **orr_confirm.param** is provided. The **orr_confirm.pl** script selects and updates Proposed Orders, using a combination of select criteria provided by the script, and criteria provided by the customer via the parameter file.

Usage

Log on as **talis** and enter the following command:

orr_confirm.pl -d<database> -h -p<parameter file> -r<report directory> -u<update>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory argument gives the pathname of the parameter file. This will normally be "-porr_confirm.param". The default directory for the parameter file is $BLCMP_HOME/data/impdir.

Parameter file

The parameter file specifies a set of variables used to select the Proposed Orders to be confirmed by the script. Optionally, the Order Date to be inserted may also be specified. A default parameter file (**orr_confirm.param.default**) is found in the **$BLCMP_HOME/data/impdir** directory. This should be copied to another file, for example "orr_confirm.param". Libraries need to edit the new parameter file, to uncomment the parameters needed for use, and insert the appropriate values for those parameters.

Orders are processed only if they satisfy certain criteria. The basic criteria apply when all "orr_confirm.param" are commented out. To be processed by the "orr_confirm.pl" script, Orders must:

- Be Order Status "Proposed".
- Be Order Type "Supplier Confirmatory".
- Be Unverified.
- Have no Order Date (or the Order Date "Jan 1 1970 12:00AM").
- Have at least one Item attached of "Potential" status.

Accepted parameters are described in the following table:

Parameters	Description
SUPPLIER_CODE	This parameter is mandatory. The value(s) input should be the Supplier Code. More than one Supplier Code may be input, separated by a comma, provided that the optional selection parameters are not used.
SUPPLIER_REFERENCE	This optional parameter may be used to specify one Supplier Reference Number. The Supplier Reference Number input is matched against the values held in the Proposed Orders. Each Supplier uses their own format for the Supplier Reference Number, but it is likely to be a batch number, followed by a running number for each Order, (for example, Order 1 = 0231/6486/1, Order 2 = 0231/6486/2). It is likely that Libraries will want to process the batch of Orders, so the Supplier Reference Number input is treated as a stem. The reference given in the parameter matches all Orders for that Supplier where the Supplier Reference Number begins with that string.
OFFICIAL_ORDER_NUMBER	This optional parameter may be used to specify one or more official Order Numbers, separated by a comma. An asterisk may be specified as a wildcard character at the end of the number. The number input is matched against the official Order Number values of the Proposed Orders.
BEGIN_ORDER_NUMBER / END_ORDER_NUMBER	The Order Numbers of the Proposed Orders may be known, as these numbers are reported in the Order, "orr_import.rep" and in "orr_imp.out". The first and last Order Numbers to be processed may be specified using these parameters. Both parameters must be used together. If a number is given in either BEGIN_ORDER_NUMBER or END_ORDER_NUMBER alone, the script aborts.
BEGIN_CREATE_DATE / END_CREATE_DATE	It is possible to process Orders created on, since or before a particular date. The date matched is the Order Date, which is usually the date that the "orr_import" utility was run to import the Proposed Orders. The BEGIN_CREATE_DATE and END_CREATE_DATE parameters may be used together or alone. The date should be given in "DD/MM/YYYY" format.
ORDER_DATE	The "orr_confirm" script inserts the current date into Order Date. If, instead, you want the Order Date to reflect the date when the Order was initiated (for example, the date of a showroom visit) then this parameter may be used. If an Order date is to be supplied by this parameter, the date should be specified in "DD/MM/YYYY" format. The Order Date may be in the past or in the future.

Example parameter file

```
#Mandatory selection parameter
  SUPPLIER_CODE=

#Optional selection parameters (may not be used if more than one supplier
code is given)

      #SUPPLIER_REFERENCE=
      #OFFICIAL_ORDER_NUMBER=
      #BEGIN_ORDER_NUMBER=
      #END_ORDER_NUMBER=
      #BEGIN_CREATE_DATE=dd/mm/yyyy
      #END_CREATE_DATE=dd/mm/yyyy

#Optional insert parameter
 #ORDER_DATE=dd/mm/yyyy
```

orr_import

The **orr_import** utility enables Orders generated Book Suppliers to be imported into Alto. This script should be scheduled when Alto is not running.

Usage

Log on as **talis** and enter the following command:

orr_import -d<database> -h -i<input file> -m<max number> -n<NOT AUTHORISE> -r<report directory> -t<processing type> -w

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This optional argument gives the name of the input file. This option is only used when continuing the processing of an input file "wrk_ite_csv.in.[datetime]" that has been processed previously using the "-m" option.
-m	This optional argument specifies the maximum number of Orders to process in this run. It should only be used where there is an exceptionally large file to be processed; if this parameter is absent the whole file is processed.
-n	This optional argument which, if used, takes the value "NOT_AUTHORISE". This value is passed to the "orr_import_dae" configuration file, and affects the WORK_UPDATE.STATUS value assigned. If "-nNOT_AUTHORISE" is specified, rows are added as status 30 and are not be processed by the Authorisor daemon ("authorisor_dae"). If "-n" is not specified, Works are added as status 30 are submitted for Authority Control. Note: Libraries which use the "TAL_QMW_ATY_EXCL" environment variable to prevent Works created or imported via Acquisitions from being authorised should import Orders with "-nNOT_AUTHORISE" set.
-t	This optional argument allows you to include item price adjustments by optionally including default supplier discount and service charges in the price per copy. If set to NOADJUST, the supplier default discount and service charges are not included in the Price per copy amount. If set to ADJUST the default discount and service charge (as specified on the supplier form for the supplier linked to the order) are applied. If not specified, it will default to NOADJUST.
-w	This optional argument , if used, instructs "orr_import" to apply the Main Classification Number of the Work, if it already exists, to the Order Items, in

preference to the classmark given by the Supplier.

Notes

- The Fund Code must be present and valid for an Item to be created. The Order record is still created if the Fund is missing or invalid, but the Item is not created.
- The Fund Code must be valid in the current Financial Year.
- The Fund must be a Base Fund and Active.
- Only valid ISBNs are accepted.

oor_subcost_upd.pl

Bulk update scripts allow you to update your existing commitments to apply the default servicing and discount from the supplier form. The scripts (orr_price_upd.pl for orders and **oor_subcost_upd.pl** for open orders) update the price per copy/subs cost pa and committed amounts for all items/subscriptions against which no payments have been made.

Usage

Log on as **talis** and enter the following command:

orr_subcost_upd.pl -d<database> -h -p<parameter file> -r<report directory> -s<filename> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory argument which specifies the name of the parameter file to be used. The contents of the parameter file are described below.

Parameter file

The script will only update open orders that match the selection criteria defined by the following parameters.

Parameter	Description
SUPPLIER_CODE	Allows you to limit processing to specific suppliers. If no supplier codes are specified, processing will default to all suppliers. Separate multiple supplier codes with a comma.
BEGIN_ORDER_NUMBER	Allows you to specify the first number in the range to be processed. Parameters BEGIN_ORDER_NUMBER and END_ORDER_NUMBER must be used together or not at all.
END_ORDER_NUMBER	Allows you to specify the last number in the range to be processed. Parameters BEGIN_ORDER_NUMBER and END_ORDER_NUMBER must be used together or not at all.
BEGIN_ORDER_DATE	Allows you to specify the start order date for processing in the format DD/MM/YYYY.
END_ORDER_DATE	Allows you to specify the end order date for processing in the format DD/MM/YYYY.
BEGIN_CREATE_DATE	Allows you to specify the begin order create date for processing in the format DD/MM/YYYY.

END_CREATE_DATE	Allows you to specify the end order create date for processing DD/MM/YYYY.

Notes

- The price per copy/subs cost pa breakdown will be updated unless:
  normal order has a type of pre-paid or a status of cancelled
  open order has a status of "closed".
- Closed status subscriptions will not be updated.
- Open order subscriptions will have their commitments adjusted regardless of whether payments have already been made against the open order subscription.
- Open order commitments will only be updated for the current financial year.
- The 'Items/Subs unpaid' column contains the total number of items on the order which have not had a payment made against them and therefore are being updated as part of the processing.
- The 'Price per copy/Sub cost pa previous' column contains the value of the Price per copy/Sub cost pa for each item/subscription in the order/open order prior to running the script. This will be in the currency of the order, not converted to the base currency amount.
- The 'Price per copy/Sub cost pa new' column contains the value of the Price per copy/Sub cost pa for each item/subscription in the order/open order after running the script. This will be in the currency of the order, not converted to the base currency amount.
- The 'Funds affected' column lists the funds that were linked to the order items that were affected by running the script. The adjustment will be the difference in the base currency before and after running the script.
- Where the report is not run in update mode, you are advised with the message "Funds would be affected" rather than "Funds affected".
- After the reporting of orders/open orders, a breakdown of the funds affected is displayed. The 'Adjustment' column contains the difference in the value of the fund commitments linked to the fund as a result of running this script. This could be a positive or negative value. For open orders, the FUND_DISTRIBUTION.SPENT value will need to be taken into consideration with this.
- Using this script to apply the default values will not zero the other fields of the price per copy and sub cost pa forms, i.e. Service VAT, VAT, Other and Other VAT.
- The user will need to run fun_totals.pl and sup_totals.pl after running the script for the amendments to be reflected in fund and supplier commitment totals.

orr_pot_ords_del

The Potential Order Compressor, **orr_pot_ords_del**, deletes unwanted Potential Orders from the database. Libraries have the option to delete all Orders of "Potential" status or to remove imported records only, retaining those created online.

Usage

Log on as **talis** and enter the following command:

orr_pot_ords_del -h -d<database> -b<begin date> -e<end date> -m<max records to process>
-r<report directory> -t<processing type> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Orders created on or since the date specified by "-b[begin date]" will be removed.
-e	Orders created on or before the date specified by "-e[end date]" will be

	removed.
-m	The "-m" specifies the maximum number of Orders to be processed in the current run. This argument cannot be used with the "-b" and "-e" options.
-t	The argument "-t[type_of_processing]" defines the type of Orders to be deleted. There are two options, either: ▪ "IMP_ONLY" to delete imported Orders only, or ▪ "ALL" to delete all Potential Orders matching the selection criteria.

Notes

- The script will report progress to screen (and to the report file) at an interval of every 1000 records processed.
- The **orr_pot_ords_del** script will create a new report file each time it runs. The report from the previous run will be renamed with a date and time extension, as illustrated below:

  orr_pot_ords_del.rep.[day]_[month]_[hour]_[min]_[sec]

orr_price_upd.pl

Bulk update scripts allow you to update your existing commitments to apply the default servicing and discount from the supplier form. The scripts (**orr_price_upd.pl** for orders and oor_subcost_upd.pl for open orders) update the price per copy/subs cost pa and committed amounts for all items/subscriptions against which no payments have been made.

Usage

Log on as **talis** and enter the following command:

orr_price_upd.pl -d<database name> -h -p<parameter file> -r<report directory> -s<data filename> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory argument which specifies the name of the parameter file to be used. The contents of the parameter file are described below.

Parameter file

The script will only update orders that match the selection criteria defined by the following parameters.

Parameter	Description
SUPPLIER_CODE	Allows you to limit processing to specific suppliers. If no supplier codes are specified, processing will default to all suppliers. Separate multiple supplier codes with a comma.
BEGIN_ORDER_NUMBER	Allows you to specify the first number in the range to be processed. Parameters BEGIN_ORDER_NUMBER and END_ORDER_NUMBER must be used together or not at all.
END_ORDER_NUMBER	Allows you to specify the last number in the range to be processed. Parameters BEGIN_ORDER_NUMBER and END_ORDER_NUMBER must be used together or not at all.

BEGIN_ORDER_DATE	Allows you to specify the start order date for processing in the format DD/MM/YYYY.
END_ORDER_DATE	Allows you to specify the end order date for processing in the format DD/MM/YYYY.
BEGIN_CREATE_DATE	Allows you to specify the begin order create date for processing in the format DD/MM/YYYY.
END_CREATE_DATE	Allows you to specify the end order create date for processing DD/MM/YYYY.

Notes

- The price per copy/subs cost pa breakdown will be updated unless:
  normal order has a type of pre-paid or a status of cancelled
  open order has a status of "closed".
- The 'Items/Subs unpaid' column contains the total number of items on the order which have not had a payment made against them and therefore are being updated as part of the processing.
- The 'Price per copy/Sub cost pa previous' column contains the value of the Price per copy/Sub cost pa for each item/subscription in the order/open order prior to running the script. This will be in the currency of the order, not converted to the base currency amount.
- The 'Price per copy/Sub cost pa new' column contains the value of the Price per copy/Sub cost pa for each item/subscription in the order/open order after running the script. This will be in the currency of the order, not converted to the base currency amount.
- The 'Funds affected' column lists the funds that were linked to the order items that were affected by running the script. The adjustment will be the difference in the base currency before and after running the script.
- Where the report is not run in update mode, you are advised with the message "Funds would be affected" rather than "Funds affected".
- After the reporting of orders/open orders, a breakdown of the funds affected is displayed. The 'Adjustment' column contains the difference in the value of the fund commitments linked to the fund as a result of running this script. This could be a positive or negative value. For open orders, the FUND_DISTRIBUTION.SPENT value will need to be taken into consideration with this.
- Using this script to apply the default values will not zero the other fields of the price per copy and sub cost pa forms, i.e. Service VAT, VAT, Other and Other VAT.
- The user will need to run fun_totals.pl and sup_totals.pl after running the script for the amendments to be reflected in fund and supplier commitment totals.

orr_unverified.pl

The **orr_unverified.pl** script allows staff to produce a list of Order requests which are currently unverified. The list can be restricted by Date range and the type of unverified Orders. This report may be used in two ways:

- Frequently, to select unverified Order requests created recently (for example, daily).
- Periodically, to identify older Orders which are still unverified.

Usage

Log on as **talis** and enter the following command:

orr_unverified.pl -b<start days> -d<database> -e<end days> -h -o<output filename> -r<report directory> -t<type of processing> -z<dept>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the minimum number of days since an unverified Order request was created for inclusion in the report. If the argument is omitted, the default is "0" i.e. today.
-e	This optional argument specifies the maximum number of days since an unverified Order request was created for inclusion in the report. If the argument is omitted, the date is translated as on or before argument "-b".
-t	This mandatory argument specifies the type of processing performed by the report, based on the type of requests to be reported. The permissible values and the corresponding reports are: **UNV**: All unverified Order requests, whether arising from public purchase requests or Library staff not enabled to verify Orders. **PUNV**: Unverified Orders from public requests. **PUB**: All Orders from public purchase requests. If "unv", "punv", or"pub" is not specified as one of the values for "-t", then the script will terminate with an error message explaining how the setting for the -t argument is invalid.
-z	This optional argument specifies that the output file should be sorted by requester's Department.

pay_inv_exp.pl

The **pay_inv_exp.pl** script enables flat files of Invoice data to be produced from Alto for use in other databases. All payment types, namely main & supplementary payments and credit notes, are handled. The script caters for Invoice-level charges in addition to payments relating to individual Items or subscriptions. Since Invoice-level charges may be either charges or allowances, null values are permitted where payments relate to an Invoice level charge.

The **pay_inv_exp.pl** script produces four data files, which may be concatenated if required. The records in the files are variable length, delimited by the newline character (HEX "0A"). They contain a fixed number of fields, each field delimited by the pipe character (HEX "7C").

The four output files are:

- **i_pay_inv_exp.out**: This file contains one record for each Invoice selected. The record will be a summary of all the payments relating to the Invoice (i.e. all payments containing the same Invoice number and Supplier). The Payment Type is included in each Invoice record. If the invoice contains valid "mixed" Payment Types, the output file indicates an "X" for Payment Type.
- **f_pay_inv_exp.out**: This file contains one record for each Fund updated by the payments relating to a selected Invoice. This will include the amount spent against the Fund on the Invoice. The Spent total in any fund records relating to credit note will be a negative value. The output file now also includes elements for Invoice-level charges and Invoice-level allowances.
- **o_pay_inv_exp.out**: This file contains one record for each Order relating to a selected Invoice.
- **t_pay_inv_exp.out**: This contains file totals.

Usage

Log on as **talis** and enter the following command:

pay_inv_exp.pl -b<begin date> -e<end date> -h -d<database> -n<financial year> -r<report directory> -s<save directory> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	"-b" is an optional argument, which may be used to specify the start date for report selection, in the format "dd/mm/yy". Invoices with payments created or edited on or since this date will be selected.
-e	"-e" is an optional argument, which may be used to specify the final date for report selection, in the format "dd/mm/yy". Invoices with payments created or edited on or before this date will be selected.
-n	This argument is optional. It allows you to specify a particular financial year for selection of Invoice data. The year must be specified in four figure format, i.e. 2004 for the year "2004/05". If not specified, all years will be processed by default. It is not possible to use this argument in conjunction with the "-b" and/or "-e" options.
-t	The argument "-t" specifies the type of processing/outputs you require; the options are "1" (concatenate) or "2" (do not concatenate). This argument defaults to "1" (concatenate).

Notes

- The file "i_pay_inv_exp.out" contains a line for each invoice processed.
  More info

i_pay_inv_exp.out

```
record_type | payment_type | invoice_number |
invoice_date | financial_year | supplier_code |
account_number | number_of_invoice_level_charges |
number_of_invoice_level_allowances |
number_of_orders | number_of_items |
base_currency_net_value |
base_currency_net_discount |
base_currency_VAT | base_currency_service |
base_currency_service_VAT |
base_currency_other_charges |
base_currency_other_charges_VAT |
base_currency_invoice_level_charge |
base_currency_invoice_level_charge_VAT |
base_currency_invoice_level_allowance |
base_currency_invoice_level_allowance_VAT |
base_currency_total_value |
base_currency_total_VAT | currency_code |
exchange_rate | currency_net_value |
currency_discount | currency_VAT |
currency_service_value | currency_service_VAT |
currency_other_charges |
currency_other_charges_VAT |
currency_invoice_level_charge |
currency_invoice_level_charge_VAT |
currency_invoice_level_allowance |
currency_invoice_level_allowance_VAT |
currency_total_value | currency_total_VAT
```

An Invoice may contain both main and supplementary payments and be valid, but it will be rejected if there is a mixture of credits and main or supplementary payments. It is possible for an Invoice to contain Invoice level allowances in combination with any Payment Type relating to Items or Subscriptions. If the invoice contains valid "mixed" Payment Types, the output file indicates an "X" for Payment Type.

An invoice will be rejected if all its payment records do not contain the same Financial Year or Currency Code and Exchange Rate. The Invoice Number, Supplier Code and an appropriate error message will be output to "pay_inv_exp.rep".

- The file "f_pay_inv_exp.out" contains a line for each Fund processed.
  More info

  Format

  ```
  record_type | invoice_number | supplier_code |
  fund_code | expenditure_code | spent_total |
  number_of_items | number_of_invoice_level_charges|
  number_of_invoice_level_allowances
  ```

  Example:

  ```
  f|034583|DIRECT|SEIT||12.00|2||
  f|0916871|DAWSON|HSWP||22720.11|165||
  f|0916871|DAWSON|EGGP||3894.76|4||
  f|0916871|DAWSON|SKEP||481.65|3||
  f|0916871|DAWSON|HKP||12454.81|53||
  f|0916871|DAWSON|HPSYP||60148.21|225||
  ```

- The file "o_pay_inv_exp.out" contains a line for each Order processed.
  More info

  Format

  ```
  record_type| invoice_no| supplier_code|
  order_no| order_type| qty| sterling_price_paid
  ```

  Example:

  ```
  o|0916871|DAWSON|DL94002277|1|1|53.01
  o|0916871|DAWSON|DL94002370|1|1|44.71
  o|0916871|DAWSON|DL91003575|1|1|260.35
  o|0916871|DAWSON|DL94003231|1|1|64.00
  o|0916871|DAWSON|DL94004101|1|1|154.48
  o|0916871|DAWSON|DL91006640|1|1|355.61
  ```

- The file "t_pay_inv_exp.out" contains a line of file totals.
  More info

  ```
  record_type| total_no_of_invoices|
  total_value_of_invoices| total_VAT
  ```

  Example:

  ```
  t|169|91769.93|814.09
  ```

pay_prev_run

The Fund roll-over process links all Items, Interloan charges and Subscriptions to the new financial year's Funds. The script **pay_prev_run** allows Libraries to make a payment from a financial year other than the current (new) year. Item payments may be re-linked only if they are the main payments in each case (as opposed to supplementary payments or credit notes). This script will not handle Interloan charges, but these can be handled online (by removing the old charge and adding a new one in the new financial year).

This script performs the individual Item, Subscription and Payment re-linkages. If you still wish to re-link an Item to Funds in a previous financial year, you must delete all but the Main Payment (via the online Acquisitions function "Unpay") and re-link back before re-adding additional payments. This script performs the individual Item, Subscription and Payment re-linkages required to pay out of Funds from an alternative financial year. Individual invoices have to be specified. If the Invoice Number specified relates to Item payments the script will check that the payments are main payments.

Usage

You must run the fun_totals and sup_totals scripts after running **pay_prev_run** in order to sum up the Total Committed and Total Spent values for each Base Fund and each Supplier respectively.

Log on as **talis** and enter the following command:

pay_prev_run

The script will prompt for the financial year to pay from. Type in the display value of the financial year i.e. in the same format as shown on the Fund Prompt Bar and Payment screens (for example "2003/2004").

The script will prompt for the invoice number. After the first Invoice has been processed, the system will ask whether you have any more invoices to process in the same way. You may continue in this way until you have completed all of the invoices which require processing. (The **pay_prev_run** facility may be run again, whenever required).

res_add_itms

The **res_add_itms** script is used for adding Items to existing reservations. The script examines the ITEM table in order to locate Items which may be added to existing reservations.

The primary use of res_add_itms is to ensure newly acquired Items may be used to satisfy reservations. New Items added to stock (i.e. with an "In Stock/Loanable" status) will be eligible for use in satisfying reservations.

There is a broader related use for the script, in that **res_add_itms** may also be used to add Items to reservations where these may have been missed for inclusion previously. For example, Items which were "Missing", "At binding" or with any other non-In Stock/Loanable status may subsequently be employed in satisfying reservations when they re-appear in stock.

Usage

Log on as **talis** and enter the following command:

res_add_itms -d<database> -r<report directory> -D<start date> -b<begin id> -e<end id> -m<max works> -t<type of processing> -h

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-D	The argument "-D" is optional and may be used to specify a start date from which to calculate the start_id when examining the ITEM table. (This argument must be given in UPPER case). Warning: This is the slowest way of running "res_add_itms.rep". Running with this option will cause the whole ITEM table to be trawled since Items are not indexed by date. This option cannot be used with "-b" or "-m" options (see below).
-b	The argument "-b" is used to specify the ID of the Item from which to commence processing the ITEM table. If this argument is not given and the Start Date argument is not given then processing will commence from the first Item on the database (i.e. where ITEM_ID=1). Note: Use of Begin ID and/or End ID arguments involves first investigating the ITEM ID numbers suitable for the job in hand. You may be interested in running "res_add_itms" against the Begin ID which equates with a given date, for example to run the script against all Items received since January 1st 1996.
-e	The argument "-e" may be used to specify the ID of the Item with which to finish processing. If this argument is not given then the process will finish after processing the Item with the highest ITEM_ID.
-m	The argument "-m" may optionally be used to specify the maximum number of Items to process in this run. For example, if your Library typically acquires 1,000 Items per week you may wish to run the script against the newest 1,000 Items on the database each week. If this option is used, the process will commence processing from the next ITEM_ID from where the previous run finished, and process the specified number of Items.

	Note: This option cannot be used with the "-D", "-b" or "-e" options. For this option to function correctly, it is essential that the report file from the previous run is left intact.
-t	This argument may be set to "ALL" or "INTRAN" (case insensitive). The default is "ALL". • tALL indicates that eligible Items will be added to all reservations. • tINTRAN indicates that Items will not be added to a reservation which has an Item in transit to satisfy it unless they are at the reservation's collection site. **Note**: It is possible to configure Alto to remove items from a reservation once an item has been put in transit to satisfy it unless that item is at the collection site. To prevent "res_add_itms" re-adding these unreserved Items, it should be run using the -tINTRAN argument when Alto is being run with the environment variable "TAL_INTRAN_UNRES" set to "YES".

Some care is needed if the script is consecutively run with different switches. For example **-m** starts processing from the last finishing point, so if **-e** was used previously it will define the next run's start point. The best practice is to stick to a given argument strategy to achieve a task, for example: use the practise described in running the script retrospectively (below). To Remove Backlog to sort out the backlog; then after the backlog is completed, run as it as described in adding new items (below) to achieve the on-going addition of newly acquired Items.

Adding new items

As most Libraries have large Item tables (i.e. greater than 500,000 Items in the ITEM table) it is suggested that runtimes should be minimised by running the script regularly using the **-b** argument. For example, a Library with an ITEM table of 1,000,000 items, choosing:

```
res_add_itms -b900000 <Enter>
```

would mean only the latest 100,000 Items are processed, reducing the run time considerably. The value to use with **-b** depends on several factors, chiefly:

- How often the script is being run and the time available. For example, if it is to be run weekends and several hours are available then more Items can be processed. Conversely, nightly runs will probably mean less time is available.
- As the ITEM_IDs are chronological, the **-b** argument effectively processes all Items created since a given date. This means the **-b** argument should be chosen such that any Items created before that date are unlikely to be recently receipted. You may wish to process all ITEMS since migration, or in the last 12 months or for the length of a reservation's lifetime.

Once a date is chosen, it needs to be associated to an ITEM_ID. The following SQL example shows one way to achieve this: select MIN(ITEM_ID) from ITEM where CREATE_DATE like 'Jan%1995%'

This will find the lowest ITEM_ID created in January 1995.

This SQL does not use indexed attributes.

Running the script retrospectively to remove backlog

When first running the script you may wish to add previously receipted Items into reservations retrospectively. This is best achieved using the **-m** argument.

As the script starts processing from its latest finishing point you may wish to invoke the first run with **-b** and **-e** if you do not wish to begin at the start of the ITEM table. For example, if you wish to process the backlog of Items beginning at ITEM_ID 50000 then run it initially as follows:

```
res_add_itms -b499999 -e500000
```

The "-e" argument then sets the starting point for the first run with "-m".

Running the script to capture "Older" Items

The script will add in existing Items, not just new ones. For example, Items may have been excluded from the original reservation because they were a non-reservable Item type (e.g. for reference only) or at an inappropriate site. These are best added as described in Adding New Items but a lower **-b**

value should be considered (suitable Items may appear anywhere in the ITEM table). This may mean running it in this way at irregularly intervals when there is "spare capacity" in the "cron".

res_item_rotate.pl

The Item Request functionality generates requests for sites to check their shelves for not on loan items that are required to satisfy reservations. A request is circulated around the sites that have an item on the shelves until an item is supplied or all possible sites have been tried. The script r**es_item_rotate.pl** circulates the requests.

The script will activate any Pending requests created since it was last run. It will attempt to allocate the request to the first site in the rotation pattern associated with the reservation filter that has a not on loan item.

The order of sites that the script will use will be the order specified in the rotation pattern if the home site of the request (that is, the collection site of the reservation) is not in the pattern. If the home site is in the pattern this will be taken as the first site in the pattern, the site following this will be the second site and so on.

Usage

Log on as **talis** and enter the following command:

res_item_rotate.pl -d<database> -h -r<directory>

The script uses standard script arguments, as described here.

Notes

- If the site is open the status of the request will be updated to Active and it will be allocated to this site. If the site is closed a record will be added to the Request log table (RES_REQUEST_LOG) to show this and the script will attempt to allocate the request to the next site in the pattern with a not on loan item. If there is no other site that the request can go to its status will be updated to Exhausted and it will be allocated to its home site.
- The script will attempt to move any Active request on to the next site in the pattern that has a not on loan copy. A record will be added to the Request log table to show that the previous site did not respond. If there is no other site that the request can go to its status will be updated to Exhausted and it will be allocated to its home site.
- The script will also attempt to move on any request flagged as Not found by the current site . If there is no other site that the request can go to its status will be updated to Exhausted and it will be allocated to its home site.

resupdate.pl

The **resupdate.pl** script should be run regularly to update the status of reservations. It can be used to:

- update the status of outstanding and uncollected reservations when their last useful date has passed.
- activate Not yet effective reservations when their effective date is reached
- update the status of collected, uncollected and deleted reservations, so that they are no longer visible online.

Usage

Log on as **talis** and enter the following command:

resupdate.pl -d<database> -e<no of days> -h -r<report directory> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-e	This is an optional argument used only with the -tLOGDEL argument (see below). It specifies the number of days grace (between 1 and 999) to allow for a temporary historical set of collected/ uncollected/cancelled reservations to remain visible on the system, e.g. A grace period of 30 days would keep the previous month's reservation data visible. The value must be a number between 1 and 999.

-t	This is a mandatory argument which describes the type of processing to carry out. There are five options: **UNCOL** : If the argument is set to UNCOL, reservations of status Waiting collection that have passed their last useful date will be set to Uncollected. **DEL** : If the argument is set to DEL, reservations of status Active or Not yet effective which have passed their last useful date will be set to Deleted. In addition, it will delete any current item request associated with a deleted reservation and add a row to the Request log table to indicate that the request was cancelled. **ACT** : If the argument is set to ACT, the status of Not yet effective reservations which have reached their effective date will be updated to Active. In addition, a pending status item request is created for any reservation it activates if the reservation filter is linked to a rotation pattern and at least one of the reserved items is not on loan. **ALL** : If the argument is set to ALL, all the processing functions carried out by the ACT, UNCOL and DEL options are performed in one run. **LOGDEL** : If the argument is set to LOGDEL, reservations will be updated as follows: ▪ Collected will be set to Deleted-Collected ▪ Uncollected will be set to Deleted-Uncollected ▪ Cancelled will be set to Deleted-Cancelled ▪ Deleted will be set to Deleted-Deleted (from Alto 5.0 onwards) Logically deleted reservations will no longer appear in the list of Borrower Reservations when this option is selected from the Borrower Information Menu online. Reservations having these new deleted statuses can still be retrieved using MIS scripts for audit/management information purposes. If the **-e** argument is used, reservations updated to Collected, Uncollected, Cancelled or Deleted within the number of days specified will not be deleted.

rlb_non_isbn.pl

The **rlb_non_isbn.pl** script allows libraries to create files of holdings records for export to UnityWeb or another database. Unlike the wrk_rbn_exp.pl script it handles records with any type of control number.

The script can report on all items, or it can produce notifications of stock changes where the first copy has been added or the last copy has been deleted since the script was last run. It is possible to limit reporting to given material types (for example non-reference stock). An option to report serial records, monograph records or both is available. If required, the script will produce output based solely on a given list of Works.

The output file format can be either a simple list of control numbers, MARC exchange records, Unity-style 'Notify' records or any combination of these.

Usage

When running a full dump or other job that results in a large number of records being output, the amount of disk space used can be considerable. It is therefore important that sufficient disk space exists before the run, and that output files are removed when no longer required.

Log on as **talis** and enter the following command:

rlb_non_isbn.pl –d<database> -h –o<output file> -p<parameter file> -r<report directory> -s<output directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This mandatory switch specifies the parameter file. The parameter file (described in the following section) must be located in the data directory.

Parameter file

The parameter file must be located in the data directory. The default parameter file, **rlb_non_isbn.param.default**, is located in **/usr/opt/blcmp/data/expdir**, and can be copied to create your file.

Parameter	Description
LIST=	This parameter controls how the LIST_VALUES= option will work. If this parameter is set to N or is not set, the script will ignore the LIST_VALUES= parameter. ▪ Set it to FI if output is to be created from a file of WORK_IDS. ▪ Set it to FC if output is to be created from a file of control numbers. ▪ Set it to I if output is to be created from a list of WORK_IDs specified in the LIST_VALUES= parameter. ▪ Set it to C if output is to be created from a list of control numbers specified in the LIST_VALUES= parameter. If any of these values is set the script will only process works in the file or list and will ignore all other parameters.
LIST_VALUES=	This parameter is used in conjunction with the LIST parameter above. ▪ If the value I is specified in the LIST parameter, then a comma-separated list of WORK_IDs should be specified here. ▪ If the value C is specified in the LIST parameter, then a comma-separated list of control numbers should be specified here. ▪ If the FI value is given in the LIST parameter then the name of a file that contains a list of WORK_IDs or control numbers should be given here. This file should be located in the data directory. ▪ If the FC value is given in the LIST parameter then the name of a file that contains a list of control numbers should be given here. This file should be located in the data directory.
BIB_LEVEL=	This mandatory parameter determines whether the script will select monographs (using the ITEM table), serials (using the SITE_SERIAL_HOLDINGS) or both. ▪ Specify M for monographs. Specifying M will select everything that isn't a serial, including non-book material. ▪ Specify S for serials. ▪ Specify B for both.
MODE=	This optional parameter determines the content of the output file to be created. There are five possible options: ▪ FDA produces a full dump of all bibliographic records for all Items ▪ FLT produces a file of bibliographic records for all Items that match the Status, Type and Location specified ▪ ADD produces a file of bibliographic records for which the first item has

	been added since the script was last run (a Changes run) ▪ DEL produces a file of bibliographic records for which the last item has been either withdrawn or changed to a not in stock status since the script was last run (a Changes run). ▪ ADL produces a file of both of the above Changes. When MODE=ADD, DEL or ADL, the statuses that represent 'in stock' or 'out of stock' should be specified using the TAL_IN_STOCK or TAL_NOT_IN_STOCK environment variable. The variable should be set in the .profile of the talis user. If neither is set, IS is assumed to be the only 'in stock' status. The default is FLT. A full bibliographic database dump can be produced by specifying MODE=FDA in the parameter file. This will use the WORKS table.
LOCATION=	This optional parameter can be used to restrict the selection to Items that belong to a specific site. A list of comma-separated site codes can be entered. If no sites are specified then all sites will be selected.
ITEM_STATUS=	This optional parameter can be used to restrict selection to particular item statuses. A list of comma-separated status codes may be entered (e.g. REC, IS). If no statuses are specified then all statuses will be selected.
ITEM_TYPE=	This optional parameter can be used to restrict selection to particular item types. A list of comma-separated type codes may be entered (e.g. AF,AN, JF, JNF). If no types are specified then all types will be selected. If the ITEM_TYPE parameter is used to limit the items types processed by the script then the REFERENCE_TYPES parameter values must also be listed in the ITEM_TYPE parameter.
MAX_LENGTH	This optional parameter specifies the maximum length of control numbers that will be output in a Notify Format file only. The default is 10. If a Work selected has a control number longer than this then the record is not given in the standard output file, but is written to the file rlb_non_isbn.skipped in the data directory (along with other excluded records). This file can be used for reference. It is overwritten each time the script is run but you may wish to delete it in the meantime if it is a big file.
REFERENCE_TYPES	This optional parameter may be used specify which of the Item Types specified in the ITEM_TYPE= parameter should be treated as reference stock. Item Type codes should be specified, separated by a comma. If none is specified then all Item Types are assumed to be lending. If the ITEM_TYPE parameter is used to limit the items types processed by the script then the REFERENCE_TYPES parameter values must also be listed in the ITEM_TYPE parameter.
LOC_CODE	This mandatory parameter specifies your four-digit library code. This is used in the creation of Unity-style Notify export data and in the naming of the output files and in each record line. This number can be found by clicking on Directory on the UnityWeb home page and searching for your institution. Click View. The number shows under Library Code.
FILE_FORMATS	This mandatory parameter specifies the types of output file to be created. The possible values are LIST (a simple number list of WORK_IDs), MARC (a file of MARC records) and NOTIFY (a file of Unity-style Notify records). All three options can be given, comma-separated.

Unity contribution guidelines

Please note that the following parameters/output format combinations are recommended for contributions to UnityWeb:

- FDA (Full dump) can be in MARC, LIST or NOTIFY format.
- ADL (Additions and Deletions) should only be in NOTIFY format.
- ADD (Additions to holdings) should be in MARC format.
- DEL (Deletions) should be in LIST format.

UnityWeb will reject files created in FLT mode. This is to prevent you contributing a subset of records by accident when you intended to provide a complete catalogue dump. To submit such a file you should rename it by changing FLT in the filename to FDA, thus indicating that you have checked the file and it is a true representation of the records you wish to display in UnityWeb. Please notify Capita Support if you do inadvertently submit an FLT file.

Notes

The script will automatically exclude certain types of work:

- Analytical 'child' records
- Serial volume 'child' records
- Multipart monograph 'parent' records
- Series 'parent' records
- ILL request records (i.e. Works where the WORK_ID exists in the ILL_REQUEST table)

roll_aggfunds_run.pl

The **roll_aggfunds_run.pl** script forms part of the Financial Year Rollover (FYR) suite. The script is used to create aggregate funds in the new financial year. For more information, refer to the appropriate Financial Year Rollover documentation located at the Capita Documentation web pages.

roll_basefunds_run.pl

The **roll_basefunds_run.pl** script forms part of the Financial Year Rollover (FYR) suite. The script is used to create base funds for the new financial year, and link unpaid items, subscriptions and ILL charges to the New Year. For more information, refer to the appropriate Financial Year Rollover documentation located at the Capita Documentation web pages.

roll_fyr_backup

The **roll_fyr_backup** script forms part of the Financial Year Rollover (FYR) suite. The script secures important Fund and Supplier related tables independently of **full_dbdump** so that they can be restored quickly in the event of a problem with just the rollover parameters. For more information, refer to the appropriate Financial Year Rollover documentation located at the Capita Documentation web pages.

roll_fyr_drop

The **roll_fyr_drop** script forms part of the Financial Year Rollover (FYR) suite. The script drops the backup tables created by the backup run from any previous years' rollover or in the 'dummy' run. For more information, refer to the appropriate Financial Year Rollover documentation located at the Capita Documentation web pages.

roll_fyr_recover

The **roll_fyr_recover** script forms part of the Financial Year Rollover (FYR) suite. The script restores back all fund and supplier related tables, using the saved tables secured by roll_fyr_backup. Anticipate that this will take at least twice as long as the initial backup. For more information, refer to the appropriate Financial Year Rollover documentation located at the Capita Documentation web pages.

sel_works

'sel_works' is a utility for selecting the WORK_ID of items added after a given date, from the ITEM table on the database.

Usage

Log on as **talis** and enter the following command:

sel_works -h -D -d<database> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-D	This is the only mandatory argument. WORK_IDs will be extracted from all items created after the date specified by the _D argument.

ser_qty

The Serials quantity denormaliser (**ser_qty**) recalculates the quantity of issues/items expected and received in ISSUE_CHECK_IN, in case corruption may have occurred online or through a software defect. It calculates the quantity receipted and expected from the Item check-in records. If no Item check-in rows exist then the quantity expected is derived from the number of subscriptions for each Work at each Site, excluding closed subscriptions.

In addition, **ser_qty** deletes ISSUE_CHECK_IN rows where the quantity expected and the quantity received are both zero.

The **ser_qty** script may be run against all Sites or at one specified Site. A large number of rows may have to be processed, so the script is capable of being run on a number of consecutive occasions in order to process the whole database.

Usage

Log on as **talis** and enter the following command:

ser_qty -h -d<database> -s<site ID> -c -r<directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-s	Names which site to process, if no site argument is given the program will run for ALL sites. This argument cannot be used with the -c argument.
-c	This argument continues from the last run and will use the last site and checkin id from the previous run. In order for this feature to work the file ser_qty.cont must not be deleted. This argument cannot be used with the -s argument.
-t	This argument specifies how long the program will run in hours. If not specified the default time is 24 hours.

Notes

- The script will generate a report (ser_qty.report), sorted by ISSN within Site, listing Check-in rows updated and deleted.

site_parameter_transfer

When creating a new site profile it is often likely that many of the parameters set for the new site profile match an existing site profile. The **site_parameter_transfer** script allows certain parameters to be copied from one site profile to overwrite those in a new site profile.

For more information on this script, log into the Developer Network.

soc_seq_reset

The **soc_seq_reset** report may have to be run as a result of multiple insertions in the standing order check-in list. It should be run if the following warning message is displayed:

```
"No more space to insert rows for (standing_order_control_number). Please see your system manager."
```

In such instances, make a note of the control number and contact your System Manager.

Usage

Log on as **ops** and enter the following command:

soc_seq_reset -h -d<database> -l<delivery site> -n<control number>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-l	This mandatory argument specifies the delivery site of the items for which rows on the standing order check-in form are to be reset.
-n	This mandatory argument specifies the control number of the parent work for which rows on the standing order check-in form are to be reset.

Notes

- When run, the report displays the number of rows and standing order check-in rows that are updated.

std_prnt_cleanup

The print cleanup script **std_prnt_cleanup** has been provided to remove files (created each time a user prints information) at regular intervals and report how many have been removed. A file is created under /**scratch/tal_output** (or any other default directory specified), each time a user prints information.

Usage

Log on as **talis** or **ops** and enter the following command:

std_prnt_cleanup -s<directory> -b<age of file> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-s	The "-s" argument specifies the scratch directory. This defaults to /scratch/tal_output and does not need to be specified unless the directory specified using the environment variable TAL_OUTPUT_DIR differs from this.
-b	The "-b" argument specifies the age of the output files deemed eligible for deletion (in hours). Files older than this time period will be removed. This argument defaults to 24 hours, if not given.

sup_totals.pl

The **sup_totals.pl** script ensures that the committed and spent monies and items for each supplier record are correct in the current database before financial year rollover. For more information, refer to the Financial Year Rollover documentation located at the Documentation web pages.

Usage

Log on as **talis** and enter the following command:

sup_totals.pl -d<database> <-h> -p<filename> -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Argument	Description
-p	This mandatory argument specifies the name of the parameter file to be used.

Parameter file

A default parameter file **sup_totals.param.default** is found in the **$BLCMP_HOME/data/utils** directory. This should be copied to **sup_totals.param**, for example, which should then be edited to requirements.

There is one optional parameter within the parameter file, SUPPLIER_CODE, which can be used to specify Supplier code(s) to be included in processing. If not specified, it will default to all Suppliers. Multiple parameters should be separated by a comma, for example:

```
SUPPLIER_CODE=BLAZEBKS,BARONS,COV,DAW
```

unlock

To avoid problems arising from different persons or processes attempting to update the same record(s) at the same time, each user - or process - is given undivided access to relevant record(s) they are using for the duration of particular transaction(s). Those records are said to be locked. It may occasionally happen that records are left in a locked state unintentionally; for example in the event of a system crash. There are two utilities which allow the System Manager to look for locked records on the database (findlock), and to unlock those records (unlock) either individually, in multiples or altogether.

Usage

To unlock locked records, log on as **talis** and enter the following command:

unlock <control number>

The script returns the control number(s) of any locked records.

Notes

- More than one Control Number may be specified at the same time, separated by a space.
- When the control number entered after the **unlock** command is for an audio-visual Work and derived from the manufacturer's number, the "r" prefix normally added to Control Numbers for audio-visual materials will need to be entered as an upper case "R". For example, the number "rvhr2830" should be specified as "Rvhr2830".
- Provided that all users are logged out of the system, output from the findlock command can be piped directly into the unlock command, in order to find and unlock all records at once. To do this, enter the following command:

  findlock | unlock.

unlocker

Capita support will set up a daily **unlocker** script, initiated automatically using the UNIX cron facility, in order to unlock any records left locked at the end of the day. This routine will be run late at night after everyone has logged off, thereby eliminating the possibility of interfering with Alto users' work in progress. The "unlocker" script works on the principles explained in the unlock script.

upd_ser_cns

The script **upd_ser_cns** is used to add Control Numbers to Serial Volume Works which do not have Control Numbers already.

Usage

Log on as **talis** and enter the following command:

upd_ser_cns -d<database> -b<begin control number> -e<end control number> -h

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument is the first number in a series of Local Control Numbers to be used when assigning Control Numbers to the Serial Volume Works. (If Beginning and End Numbers are not specified, the script simply counts the Serial Volume Works without Control Numbers).
-e	This is the last number in a series of Local Control Numbers to be used when assigning Control Numbers to the Serial Volume Works. If the Beginning Number is specified, the End Number must also be present.

update_daily_access_points

The **update_daily_access_points** script updates the old-style OPAC indexes to reflect any changes made to the database, since the last time it ran.

This script calls two daemon processes. The first is **mcoll_dae**, which works out which collections works should be part of, the second is access_points_dae, which assigns works to OPAC indexes.

update_daily_access_points usually runs every night to update OPAC indexes. This is normally scheduled using the cron, but it can be started manually. It is not essential that all users are logged off before running this, although they may experience a slight degradation in performance whilst it is running.

Usage

To start update_daily_access_points manually, Log on as **ops** and enter the following command:

update_daily_access_points

The time this will take to complete will vary, depending on the size of the database, the number of records to update and the processing capacity of your machine.

wel_update

The wel_update script updates the talis_aggregates database with the combined number of issues and renewals for a particular week of a year. This data is taken from the prod_talis database on MIS server.

The talis_aggregates database should be backed up to tape for security. However as the data is derived from the LOAN table the data can be rebuilt from scratch if necessary although this will take time as it needs to be done by running the wel_update script for each individual week of each year required.

Those using loan compression should ensure that the wel_update script should be run for any weeks required before those weeks are compressed.

Usage

The script must be run on the MIS server as the talis user using the following syntax:

wel_update -n<week> -q<year> -r<directory> -h

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	A mandatory switch which specifies the week to be used in the processing. The value must be a number between 1 and 54. It is possible for there to be 54 weeks in a year as a week is defined from Sunday to Saturday. It is therefore possible for the beginning or end of the year to fall with a partial week. These partial weeks are counted separately.

-q	A mandatory switch which specifies the year to be used in the processing. The format is YYYY.
-r	Names the report directory. If the option is not given, the default is defined by the TAL_REP_DIR environment variable, if TAL_REP_DIR is not set then it will default to /scratch. The report generated is called wel_update.rep.
-h	Displays the syntax and possible switches.

Notes

If a previous version of prod_talis is restored to the MIS server that does not contain the source data for a week that has already been built in the talis_aggregates database it is possible if the wel_update script is run for this week that the rows will be removed from the talis_aggregates universe.

For more information, refer to the Decisions Manual.

wku_compress.pl

The **wku_compress.pl** script is a database compressor for the WORK_UPDATE table, enabling specific rows to be deleted from this table according to their QUEUED_DATE. Only rows of statuses which indicate that the Work update has been processed or unsuccessful will be deleted.

This script should be run regularly, by including it in a daily or weekly cron, to keep the WORK_UPDATE table to a manageable size. Ensure users are logged off before running this script.

Usage

Log on as **talis** and enter the following command:

wku_compress.pl -d<database> -e<number of days> -r<report directory) -h

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-e	This mandatory argument is used to specify the age of Work rows (expressed as a number of days before this script is run). Any WORK_UPDATE rows are deleted if their QUEUED_DATEs in WORK_UPDATE exceed this age. For example, "-e30" deletes all WORK_UPDATE rows which have a queued date which is 30 days or more earlier than the current date and satisfy the status criteria in Select Criteria. The maximum value allowed is 999 days. If an invalid "-e" value is entered, the script will abort processing and report:

wku_update.pl

In Alto, works are imported into the local database as a separate step from opening those works for editing. As the import process inserts a row into **WORK_UPDATE**, the user would be prevented from editing the work until it has been processed by the **work_exp_dae** and authorisor_dae daemons.

To get round this problem, the Alto import process uses a new set of statuses for the rows it inserts. If a row with one of these new statuses is encountered either by Alto, it will be ignored for the purposes of preventing edits.

A script has been supplied that can be run regularly from the cron, to change the status of WORK_UPDATE rows inserted by Alto, so that they will be processed by the daemons as required. The new WORK_UPDATE statuses will follow existing logic, but use statuses above 200. The script resets the rows to the usual values, and removes any duplicates.

The script should be run at regular intervals (at least daily), and timed so that the **work_exp_dae** and **authorisor_dae** daemons have time to process the works before **update_daily_access_points** starts.

Usage

Log on as **talis** and enter the following command:

wku_update.pl -d<database> -g<minutes> -h -r<report directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table. Note that if the report directory is not given, then by default the report will be written to the **$BLCMP_HOME/data/utils** directory.

Argument	Description
-g	Specifies the grace period (in minutes) that will elapse before the record's status is modified by the script. Values 1-30 are allowed. If the -g argument is not used, the grace period will default to 5 minutes.

work_logdelete.pl

The **work_logdelete.pl** script either logically deletes Works which satisfy certain criteria, or suppresses them from OPAC. The script operates in such a way that Works may be logically deleted or suppressed from OPAC depending on whether they have Items of particular statuses attached.

If there are no Items attached to the Work (i.e. no rows in ITEM for that WORK_ID), the script checks whether the Work record needs to be retained for any other reason. If the Work is used in any Order, Open Order, Interloan request or active or not yet effective reservation, it is not deleted but it is suppressed from OPAC. If the WORK_ID is not used, it is deleted.

If any Items are found (i.e. rows in ITEM for that WORK_ID), the script checks their status against the statuses listed in the parameter **NOT_SUPP_STATUS** or **SUPP_STATUS**. If Items defined as "In Stock" are found, no further processing is required for that WORK_ID. If no Items defined as "In Stock" are found, the Work is suppressed from OPAC.

A Work identified as requiring deletion is logically deleted from the database. A row is added to WORK_UPDATE with STATUS = 130, so that **update_daily_access_points** will process the deletion. Any related Work links are deleted. If all the child Works of a Parent Work are deleted, the Parent Work is also deleted. If the Work deleted is the parent of an Analytical record, the child Works are also deleted.

Usage

Log on as **talis** and enter the following command:

work_logdelete.pl -b<begin id> -d<database> -e<end id> -h -m<max rows to process> -p<parameter file> -r<report directory> -s<data directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the WORK_ID in the WORKS table from which to begin processing. If this option is not given then processing commences from the lowest WORK_ID. It is usually used in conjunction with the -e argument.
-e	This optional argument specifies the WORK_ID in the WORKS table at which to end processing. If this option is not given, processing continues to the highest WORK_ID in the WORKS table. It is usually used in conjunction with the -b argument.
-m	This optional argument states the maximum number of rows to process from the WORKS table. When specified, the script only attempts to process the number of rows specified, continuing from the last run (using the LAST_PROCESSED from the UTILITY_LOG table). If the "-m" argument is not

	specified then the script attempts to process all rows from the relevant table, continuing from the last run.

-p	This optional argument gives the pathname of the parameter file. If this argument is not given, the script will use the default parameter file work_logdelete.param in the directory $BLCMP_HOME/data/utils.
	
	This file is not shipped. It has to be created by the user, and needs to contain just a single line which will either be:
	
	`NOT_SUPP_STATUS=[item statuses which are "In stock"]`
	
	or
	
	`SUPP_STATUS=[item statuses which are "Not in stock"]`
	
	The statuses should be specified as Type Status Codes entered in upper or lower case and separated by a comma.
	
	If neither or both of `NOT_SUPP_STATUS` and `SUPP_STATUS` are given, the script terminates.

-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.

Notes

- The script will not process any rows which are already set to "Delete" status, or which are already suppressed from OPAC by the inclusion of a 150*x field.
- The script will not process rows which legitimately have no Items attached, and which are required in OPAC because they are either:
  - Serial Works (see the note below)
  - Multi-Volume Parent Works
  - Series Parent Works
  - Analytical Children: "Child" Works of analyticals
- All Serial Control Numbers are ignored during processing. This includes all ISSNs and old-style Talis Serial Numbers.
- The script does not process Serial Control Numbers.

wrk_class_reset

The **wrk_class_reset** script is a database repair script to address the following problems:

- Missing or incorrect single line OPAC class number display.
- Inconsistencies in the CLASS_WORK_LINK table where the "Main" class number may be flagged incorrectly.

The script restores the CLASS_DISPLAY attribute in the WORK table and corrects the SELECTED flag in the CLASS_WORK_LINK table where this is incorrect.

Usage

Log on as **talis** or **ops** and enter the following command:

wrk_class_reset -h -u -v -d<database> -r<report directory> -b<begin id> -e<end id> -m<max works> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	The begin_id argument ("-b") may be used to specify the WORK_ID from

	which to commence processing. If this argument is not given then all Works up to that identified by the end_id argument ("-e") will be processed i.e. the process will commence from the Work where WORK_ID=1.
-e	The end_id argument ("-e") may be used to specify the WORK_ID at which to terminate processing. If this argument is not given then processing will end at the Work with the highest WORK_ID.
-m	As an alternative to using the "begin_id" and "end_id" pair of arguments, the max_works ("-m") argument can be used to specify the maximum number of Works to be processed in the current run. If this option is used, the process will commence processing from the next WORK_ID from where the previous run finished, and process the specified number of Works. For this option to function correctly it is essential that the report file from the previous run is left intact.
-t	The "-t" argument is used to specify the type of processing required for updating the WORK and CLASS_WORK_LINK tables. There are four options: Using **-t1**, the lowest CLASS_ID for a given WORK_ID, and update only where the WORK.CLASS_DISPLAY is missing. Using **-t2** causes the script to use the most frequently occurring CLASS_ID associated with Items for a given WORK_ID in the ITEM table, and update only where the WORK.CLASS_DISPLAY is missing. Using **-t3** causes the script to use the lowest CLASS_ID for a given WORK_ID to correct the existing WORK.CLASS_DISPLAY. Using **-t4** causes the script to use the most frequently occurring CLASS_ID associated with Items for a given WORK_ID in the ITEM table to correct the existing WORK.CLASS_DISPLAY.
-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.

wrk_counts

The **wrk_counts** script updates the count of "In Stock" and "On Order" Items in the WORK table, so that the correct counts will be displayed online within Acquisitions. This script may be run from anywhere, but is held under **/usr/opt/blcmp/talis/utils/bin**. It will normally only be necessary to run **wrk_counts** infrequently; twice annually is recommended as a safety precaution.

It is essential that a full_dbdump is taken of the database before and after running wrk_counts. The script will ask whether the database has been dumped before running.

Usage

Log on as **talis** and enter the following command:

wrk_counts <database> <time in hours>

If run without arguments, the script will prompt you for the name of the target database from which a full dump is to be taken. Similarly, if no running time is specified, the script will prompt for one.

Notes

- After running "wrk_counts", the script reports out either that all works have now been processed, or works remain outstanding. In this case processing will continue the next time wrk_counts is run.
- Messages will appear on-screen to indicate when it started running, the estimated completion time and the actual completion time.

wrk_disp_reset

The daemon **wrk_disp_dae** was written specifically to fix database problems resulting from the migration of Works from BLS into Talis, whereby the OPAC display attributes of certain WORK rows may be incorrectly generated. The daemon will overcome the problem by generating the OPAC display attributes of WORK rows from the WORK_SUBFIELD table. Normally this daemon would be run as a result of identifying Work records online which have undergone an unsatisfactory migration, or simply as a precautionary measure after having running migration of Works.

The daemon may be started in two different ways. The preferred option is to run the "wrk_disp_reset" script, plus arguments, at the command line. The alternative option is to run **wrk_disp_dae** as a daemon.

Usage

Log on as **talis** or **ops** and enter the following command:

wrk_disp_reset -h -d<database> -b<begin id> -e<end id> -m<max works>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	The begin_id argument ("-b") may be used to specify the WORK_ID from which to commence processing. If this argument is not given then all Works up to that identified by the end_id argument ("-e") will be processed i.e. the process will commence from the Work where WORK_ID=1.
-e	The end_id argument ("-e") may be used to specify the WORK_ID at which to finish processing. If this argument is not given then processing will end at the Work with the highest WORK_ID. If a begin_id and end_id range is not specified then all Works on the system will be processed.
-m	As an alternative to using the begin_id and end_id pair of arguments, the max_works ("-m") argument can be used to specify the maximum number of Works to be processed in the current run. If this option is used, the process will commence processing from the next WORK_ID from where the previous run finished, and process the specified number of Works. For this option to function correctly it is essential that the configuration (.cfg) file from the previous run is left intact. This option cannot be used either with the "-b" and "-e" options, or if the previous run used a different database.

Managing the work display daemon

The alternative to running the script from the command line is to run **wrk_disp_dae** as a daemon. This will involve "manually" editing the **wrk_disp_dae.cfg** file to specify the required begin and end WORK_IDs, and executing the **wrk_disp_dae** daemon with the **dae_start** command.

The configuration file "wrk_disp_dae.cfg" is held in the /usr/opt/blcmp/talis/control directory. The parameter settings held in this file are read and interpreted by the daemon when run. Parameters are as follows:

DBNAME

This identifies the name the database to be processed. The default is "prod_talis".

START

This setting identifies the start WORK_ID of a range of WORKs to be processed. If the START_ID is not specified in the configuration file, it will default to "1".

END

This setting identifies the highest WORK_ID in the range of WORKs to be processed. If END_ID is not specified in the configuration file, it will default to the highest WORK_ID on the database. If an

END_ID is specified but this is higher than the highest WORK_ID on the database then it will default to the latter value.

Example:

```
DBNAME prod_talis
START 100000
END 199999
```

Starting and stopping the daemon

The Work Display daemon may be started with the command:

dae_start wrk_disp_dae

The Work Display daemon may be terminated in a controlled manner with the command:

dae_term wrk_disp_dae

To stop the daemon in an uncontrolled manner, you may type in:

dae_stop wrk_disp_dae

The "dae_stop" command should only be used where "dae_term" has failed. This will terminate the process running the daemon.

wrk_rbn_exp.pl

The Regional Library Bureaux Notifications script **wrk_rbn_exp.pl** generates notifications of Library holdings from Alto to be sent to Regional Library Bureaux such as LASER for loading into the Unity™ and Viscount™ systems.

The **wrk_rbn_exp.pl** script produces notifications of stock changes, where the first copy has been added or the last copy has been deleted since the last run. It is possible to limit reporting to given material types (for example, to exclude reference stock). An option to report Serial records, in addition to Monograph records, is available.

Control Numbers and Library holdings are normally reported. Only those records having a standard Control Number (ISBN, BNB, Library of Congress, CILLA) are reported. Output files may be generated in either Unity™ or Viscount™ (LASER) formats.

The script is run initially scheduled using the cron to report the whole of the Library's holdings. It should then be run on a periodic basis to report changes to the Library's holdings over the interim period.

The **wrk_rbn_exp** script should be run only when there is no activity in Alto regarding the addition and/or editing of Works and/or Items. This includes the running of any batch scripts such as **import_work** which update the database, so care should be taken when scheduling this or similar jobs in "cron".

Usage

Log on as **talis** and enter the following command:

wrk_rbn_exp.pl -d<database> -h -m<max rows> -o<output file> -p<filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-m	This optional argument the maximum number of rows to process from the ITEM_UPDATE or WORKS table. When specified, the script only attempts to process the number of rows specified, continuing from the last run (using the LAST_PROCESSED from the UTILITY_LOG table). If the "-m" argument is not specified then the script attempts to process all rows from the relevant table, continuing from the last run.
-p	This mandatory argument specifies the parameter file. The "wrk_rbn_exp.pa" parameter file must be located in the data directory.

Parameter file

The parameter file **wrk_rbn_exp.pa** must be located in the data directory. The default data directory is **/usr/opt/blcmp/data/expdir**. The parameters are case-insensitive. The following parameters are valid. The parameter file must be created by the customer before the script is run for the first time.

The parameters are described in the following table:

Argument	Description
PROCESS_TABLE=	This mandatory parameter specifies update or bulk processing mode. It may be set to ITEM_UPDATE for processing updates, or WORKS for processing the whole database. No other values are valid. .
FORMAT=	This mandatory parameter specifies the format in which the data should be output. It may be set to the values UNITY or VISCOUNT (in upper or lower case). No other values are valid.
REGION_CODE=	This mandatory parameter specifies the ILL Region Code. It takes a single alpha-numeric character.
LIBRARY_CODE=	This mandatory parameter specifies the Library Code. It takes a three digit code in the range 001-255. This is not to be confused with the BLCMP Library Code. Check with LASER etc. if you do not know the correct code.
INSTOCK_STATUSES=	This optional parameter may be used to specify the Item Statuses which should be included in the report. Status Codes should be given. If multiple Statuses are specified these should be separated by a comma. If this parameter is not set, the environment variables "TAL_IN_STOCK" and "TAL_NOT_IN_STOCK" are used, and if these variables are not set then Items of all Statuses are included.
LENDING_TYPES=	This optional parameter may be used to specify the Item Types which should be included as lending stock. Item Type Codes should be given. When multiple Item Types are required, these should be specified separated by a comma. Items of all Types are included if this parameter is not set.
REFERENCE_TYPES=	This optional parameter may be used specify which Item Types should be included as reference stock. Item Type Codes should be specified, separated by a comma. If none is specified then all Item Types are assumed to be loanable. If both REFERENCE_TYPES= and LENDING_TYPES= are specified then just the Types specified are used. For example, if LENDING_TYPES=AAA, BBB and REFERENCE_TYPES=XXX,ZZZ then only those Types will be used; i.e. all other Types will be ignored, whereas, if the Library had simply defined REFERENCE_TYPES=XXX, ZZZ without defining any LENDING_TYPES= then all other Types would be treated as loanable.
WORK_TYPES=	This optional parameter may be used for specifying whether to process Monographs, Serials, or both. It can take the values "M" for Monographs, "S" for Serials, or both (separated by a comma). If none is specified, the monographs only are processed by default.
BNB =	This optional parameter may be set to NO to exclude works with BNB control numbers from the processing. If it is not set, or is set to YES, works with BNB control numbers will be included.

CILLA =	This optional parameter may be set to NO to exclude works with CILLA control numbers form the processing. If it is not set, or is set to YES, works with CILLA control numbers will be included.
ISBN =	This optional parameter may be set to NO to exclude works with ISBN control numbers form the processing. If it is not set, or is set to YES, works with ISBN control numbers will be included.
LC =	This optional parameter may be set to NO to exclude works with LC control numbers form the processing. If it is not set, or is set to YES, works with LC control numbers will be included.

TAL_IN_STOCK/TAL_NOT_IN_STOCK

The Library can use either INSTOCK_STATUSES= in the parameter file (as explained in Section 64.3.2: The "wrk_rbn_exp.pa" Parameter File) or default to the environment variables "TAL_IN_STOCK" or "TAL_NOT_IN_STOCK" to determine which Item Statuses will be reported. If any of these settings are changed it will be necessary to do a full bulk run of the stock to accurately reflect the changes. The setting of INSTOCK_STATUSES= in the parameter file will override the environment variables.

For more information about TAL_IN_STOCK/TAL_NOT_IN_STOCK, refer to the Alto Configuration Online Help.

Notes

- Use of the "TAL_IN_STOCK"or "TAL_NOT_IN_STOCK" environment variables is optional, but if used it is mandatory that one only of these two mutually exclusive environment variables is specified.
- If the Library changes the reporting criteria (in terms of the types and/or statuses of stock to include) it is necessary to report the whole database afresh as the script does not maintain a record of notifications reported previously.
- A Library planning to run this script must know their ILL Region Code and the Library Code assigned by the region (this is not to be confused with the BLCMP Library Code). Both ILL Region Code and Library Code must be entered in the parameter file.
- The script uses the ITEM_UPDATE table to identify the Works which have had Items added, edited or deleted.

wrk_unsuppress_M21.pl

The **wrk_unsuppress_M21.pl** script, available in Alto 5.2 and above, can be used to unsuppress Works that now have Items of particular statuses attached.

If any Items are found (i.e. rows in ITEM for that WORK_ID), the script checks their status against the statuses listed in the parameter **NOT_SUPP_STATUS** or **SUPP_STATUS**. If Items defined as "In Stock" are found, the Work is unsuppressed from OPAC. If no Items defined as "In Stock" are found, the Work remains suppressed.

A row is added to WORK_UPDATE with STATUS = 30, so that **update_daily_access_points** and **marcgrabber** will process the deletion.

Usage

Log on as **talis** and enter the following command:

wrk_unsuppress_M21.pl -p<parameter_file> -d<database_name>-b<begin_id> -e<end_id> -h -m<maximum rows to process> -r<directory> -s<directory> -u -v

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the WORK_ID in the WORKS table from which to begin processing. If this option is not given then processing commences from the lowest WORK_ID. It is usually used in conjunction with the -e

	argument.
-e	This optional argument specifies the WORK_ID in the WORKS table at which to end processing. If this option is not given, processing continues to the highest WORK_ID in the WORKS table. It is usually used in conjunction with the -b argument.
-m	This optional argument states the maximum number of rows to process from the WORKS table. When specified, the script only attempts to process the number of rows specified, continuing from the last run (using the LAST_PROCESSED from the UTILITY_LOG table). If the "-m" argument is not specified then the script attempts to process all rows from the relevant table.

-p	This optional argument gives the pathname of the parameter file. If this argument is not given, the script will use the default parameter file wrk_unsuppress_M21.param in the directory $BLCMP_HOME/data/utils.
	
	The file needs to contain just a single line which will either be:
	
	`NOT_SUPP_STATUS=[item statuses which are "In stock"]`
	
	or
	
	`SUPP_STATUS=[item statuses which are "Not in stock"]`
	
	The statuses should be specified as Type Status Codes entered in upper or lower case and separated by a comma.
	
	If neither or both of `NOT_SUPP_STATUS` and `SUPP_STATUS` are given, the script terminates.

-r	The -r argument may be used to specify the report directory where the process will write its report file. If a report file of the same name already exists in the same directory, it will be renamed with a date/time extension. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable.

Perl MIS reports

About MIS reports

Introduction to Perl MIS reports

Many standard reports have been written in Perl (Practical Extraction and Report Language). This is a third party programming language, which allows scope for local customisation of formats and select criteria.

Perl scripts are run from the UNIX command prompt, providing management information and business functionality reporting. Significantly, Perl reports will often make changes within the database rather than simply extracting information. For this reason the scripts that update the database must be run against the live database rather than a copy on the MIS machine.

A series of Perl reports are shipped as standard (such as overdue letters and printed orders) with the LMS. Perl also allows scope for local customisation of formats and select criteria. Scripts written in Perl are recognisable by their **.pl** file extension.

- Capita can provide standard or tailored training in Perl. For further information on, please contact enquiries@capita.co.uk
- For more information about Perl, refer to the O'Reilly Official Perl Home Page.

About the MIS server

Many management information queries consume large amounts of processing and take considerable time to run. Running them can have a significant impact effect on the performance of the LMS, and for this reason they are usually scheduled to run at off-peak times when the Library is closed.

An alternative to running these utilities and reports against the main LMS server is to purchase a MIS server, thereby permitting "passive" queries to be run whenever necessary without slowing-down performance of the LMS. A further advantage of this approach is that it affords greater freedom and flexibility in scheduling those "active" queries and utilities which must be run using the main database ("prod_talis").

The MIS server is a networked machine which holds a duplicate copy of the main database for processing independently from the main server.

How to Run the Queries

Queries are run in the same way as described in this document.

Caution

Do not attempt to run active queries which are designed to update the database, unless this is done deliberately for testing purposes.

Potential Secondary Uses of the MIS Server

MIS servers can be used for a number of incidental purposes, which include:

- Testing local versions of tailored reports.
- Running the Library's own locally developed unix shell scripts, isql or MIS Perl reports against the MIS server (assuming these do not need to update the main database).
- Providing an alternative to the "tutor_talis" database for training purposes. This copy of the "prod_talis" database may contain more meaningful data.
- Testing the run-times/performance of new MIS scripts. The MIS server affords the ideal environment for experimentation, as it holds a temporary, expendable, copy of the main "prod_talis" database.

Perl MIS Report Structure

Each standard Perl MIS report is made up of a parameter file and five files of Perl code.

The parameter file for each report is located in the **/usr/opt/blcmp/talis/mis/param** directory and bears the name of the report (for example **orr_order_letter**). One of the functions of the parameter file is to indicate the location of the five component files. Creating local versions of theses scripts is described in the topic Tailoring Reports.

The Perl code files are located in a directory, which also bears the name of the report, under **/usr/opt/blcmp/talis/mis**.

Every new report will have its own directory and will have a parameter file within the directory **/usr/opt/blcmp/talis/mis/param**. For example, Overdues will have a directory called:

/usr/opt/blcmp/talis/mis/loa_odue_letter

In the param directory there will be a file called **loa_odue letter**:

/usr/opt/blcmp/talis/mis/param/loa_odue_letter

The individual reference directories for each new MIS script contain five source code files. The following table describes the function of each file, with an example from the loa_odue_letter script.

File	Description	Example from loa_odue_letter
main.pl	Contains the main body of the program which controls the overall processing of the report, including the default preselect.	The default preselect selects all LOAN rows that are overdue and matches the parameters given on the command line (for example, limits set by **-b** and **-e**).

preselect.pl	This file is provided as a stub. The user inserts their variant preselect in this file.	This could be tailored to limit the overdue preselect to a given Borrower Type.
select.pl	Further refinements to the preselect to just produce the rows needed in the output.	An Overdue Letter is re-sent if the Item has been renewed and has gone overdue again.
retrieve.pl	Retrieval of all the data elements required in the final output.	All the Work and Item details associated with the overdue loan.
format.pl	Template file for the output.	

Running MIS reports

A separate UNIX login called **report** is provided for running Perl reports. You must be logged as the user **report** or **talis** to run MIS reports. The basic syntax for running a Perl report is:

report -p[report_name]

The **-p** argument is mandatory for every report and identifies the report parameter file.

There are a number of other optional arguments, which vary according to which report is being run. The command

report -p[report_name] -h

will display a summary of all the switches which can be applied to a given report. Some common switches are shown below:

Argument	Description
-a	Appends the data to the file name stated, rather than overwriting it.
-b	Defines the beginning of a range. Usually this will be a date or a number of days in the past.
-e	Defines the end of a range. Usually this will be a date or a number of days in the past.
-h	Will display a summary of all the switches which can be applied to a given report.
-o	Renames the output file to the name given. The default name is [script_name].out
-s	Determines where the output file will reside. The default directory should be **/usr/opt/blcmp/data/mis**.
-d	Determines which database is used. The default is **prod_talis**.
-n	This argument varies in meaning/usage from script to script, but is commonly used. Use the **-h** command to determine the exact use.
-q	This argument varies in meaning/usage from script to script. Use the **-h** command to determine the exact use.
-r	Names the report directory where the process will create its report file. If the option is not given, then the default is defined by the TAL_REP_DIR environment variable.
-t	This argument varies in meaning/usage from script to script. It typically defines a type of processing. For example, it might be used to specify an Overdue Letter Number (-t1,2) or Order Format (-t8x4).

Output files

By default, both the output file and the report file are file are written to the directory **/usr/opt/blcmp/data/mis**. Subsequent runs of the report overwrite neither the output file nor the report file. They are preserved by the addition of a date/time suffix to the last output and report file.

Example

To produce a file of all orders created since 1st September 2004, you would enter the command:

report -porr_order_letter -b01/09/04

This would create a file of orders in the directory **/usr/opt/blcmp/data/mis** called **orr_order_letter.out** with a corresponding report file **orr_order_letter.rep**.

Tailoring reports

Prior to tailoring any reports, please note the following:

- Do not amend any source code or parameter files shipped by Capita; make copies instead.
- Do not keep local copies under **/usr/opt/blcmp/talis** as they will be lost when the next release of Altois installed.

This means that any changes to the shipped parameter file, including changes to a parameter (for example, **pagelength**) will demand a local version.

Setting up a local MIS area

The best practice is to mimic the MIS directory structure below a directory of your choice. Local versions of MIS would have a natural home under **users/report** in a new directory called **mis**. Using this example, below **/users/report/mis** would be a directory called **param** and a further sub-directory for each script which requires local tailoring.

The following instructions describe how to set-up a local MIS area for a local **format.pl** for Overdue Letters (i.e. the script **loa_odue_letter**).

To set up a local MIS area

1. Log on as the user **report**.
2. Change directory:

   cd /users/report

3. Create a directory called **mis**. This is a once-only operation when first setting up your permanent local MIS area:

   mkdir mis

4. Change directory to **/users/report/mis**

   cd mis

5. Create a directory called **param**. This is a once-only operation the first time the local area is created.

   mkdir param

6. In the directory **/users/report/mis** create a directory for the script in question, for example:

   mkdir loa_odue_letter

7. Change directory to **/users/report/mis/param**:

   cd param

8. Copy the master parameter file to your local area. For example:

   cp $TALIS_HOME/mis/param/loa_odue_letter .

   Note that the final full stop is part of the command. It stipulates that the target files should be copied to the current directory with the same names.

9. Change directory to the loa_odue_letter directory:

   cd /users/report/mis/loa_odue_letter

10. Copy the source code files that require local amendment. For example:

    cp $TALIS_HOME/mis/loa_odue_letter/*.pl .

11. Amend the parameter file so that it picks up the local version(s) of the source code. For example:

    vi /users/report/mis/param/loa_odue_letter

    In particular, set the new location of your local files. For example:

    format = /users/report/mis/loa_odue_letter/format.pl

    Repeat this for all files (i.e. preselect.pl, main.pl, retrieve.pl and select.pl).

12. Set the environment variable $LOCAL_MIS_HOME to point at the local area. For example:

    LOCAL_MIS_HOME=/users/report/mis; export LOCAL_MIS_HOME

    If this is a permanent area designated for local MIS then the above two commands are best added to the **.profiles** for the **talis** and **report** users.

13. Amend your format.pl to cater for local requirements. For example:

    cd /users/report/mis/loa_odue_letter
    vi format.pl

Documenting and maintaining local amendments

Before making any amendments it is important to consider how you will document and maintain your local versions. When Capita ships amended software for existing reports (for example fixes to known problems) any local amendments will, in most instances, have to be re-introduced to new copies of the **shipped.pls**.

The best method of documentation is to add lines of comments in the top of the file after the existing comments:

```
# Local changes are as follows:
-------------------------------------------------------------------------------
# Local changes are as follows:
#
#   Amended text in the footer and removed the fourth line of
#   borrower's address
#   Pete Smith  12/09/03
#
#   Commented out borrower's address and replaced with Dept.
#    Jane Smith   12/05/04
```

For changes to **retrieve.pl** (and similar files containing source code) you are advised to comment in the header as described and to comment in the body of the file too. For example:

```
# Get the borrower's department.   Local change  added 10/10/03
# KDG.
($Department) = &sql($d,"
select DEPARTMENT_ID
from BORROWER
where BORROWER_ID = $borrowerID
");
```

Tailoring output formats

This section is best read while viewing a **format.pl** file on screen. A good example is:

/usr/opt/blcmp/talis/mis/SK_loa_school_letter

Local formats are achieved primarily by editing the local version of **format.pl**. The **format.pl** contains two types of printed data:

- Text "as seen"
- Data retrieved from the database

Literal text ("as seen")

To amend text "as seen", similarly overtype, remove or move it to an alternative position within **format.pl**.

Variable data

Variable data appears in **format.pl** as two lines:

- The first line shows the position and length on the output
- The second line assigns it to a variable

For example:

```
@>>>>>>>>>>>>
$BorrowerBarcode
```

- If you wish to change its position on the screen, move both lines to a new position.
- To increase or decrease its length simply add or remove right angle brackets (each one, together with the leading character represents the number of characters that will be printed).
- Similarly, if a given data element is not required on the output then remove the two lines from **format.pl**.
- Do not include literal text on the same line as variables (for example, $BorrowerBarcode) as it will not be displayed.

Formatting blocks

Data printed to the output appears in formatting blocks within **format.pl**. A block begins with the format command and ends with a full stop in column one of a given line. For example:

```
FORMAT bottom
This is the last paragraph of a letter
.
```

The following two rules apply to formatting blocks:

1. All blocks in a **format.pl** must be kept, even if all data between the start and end is deleted.
2. Everything required in the output (i.e. both text as seen and data from the database) must appear in one of the formatted blocks.

Warning:

Text outside a block will cause problems.

Tailoring selection criteria

Each MIS report will have some default preselect criteria. For example, the Overdues preselect selects loans that are past their Due Date and are still on loan. It is likely variant preselects will be required for local differences. These may be defined on a permanent or one-off basis. An example is a special Overdue run for a given group of individual Borrowers. This may demand a preselect that selects on specific Borrower barcodes.

To create local preselects

1. Identify the default preselect in **main.pl**. It appears as shown in the following format:

```
#
# Do preselect if not done by the user
#
if (&preselect == 0)
```

```
{

(@preselect_results) = &sql($d,"
select ILL_ID from ILL_REQUEST
where ILL_REQUEST.ILL_STATUS = 3
order by ILL_ID
");

}
```

The preselect is everything between the curly brackets which are aligned below the if statement.

2. In the local version of **preselect.pl**, insert a copy of the default preselect after the first left-hand curly bracket. For example:

```
#
# Main preselect subroutine
#
sub preselect {

   (@preselect_results) = &sql($d,"
        select ILL_ID from ILL_REQUEST
        where ILL_REQUEST.ILL_STATUS = 3
        order by ILL_ID
         ");

return(0);
}
1;
```

3. Amend the preselect so it reads "return(1)" .

4. Amend the SQL as required.

Caution

Problems will occur if the attributes selected or the order by are changed. Amendments should be limited to changes to the where clause.

bor_charge_history

The **bor_charge_history** report produces a list of the charges incurred by a particular Borrower. The list may be limited to the charges incurred in a specified period. The credits (if any) relating to each charge selected are reported.

Usage

Log on as **talis** and enter the following command:

report -pbor_charge_history -a<filename> -b<begin date> -d<database name> -e<end date> -h
-n<borrower number> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument may be used to specify the beginning of a date range of charges to be selected. The format required is "DD/MM/YY" or "DD/MM/YYYY". Only charges incurred on or since the given date are reported.
-e	This optional argument may be used to specify the end of a date range of charges to be selected. The format is "DD/MM/YY" or "DD/MM/YYYY". Only charges incurred on or before the given date will be reported.
-n	This mandatory argument must be used to specify the Borrower Number for the user whose charges are to be reported. Any lower case alphabetic characters in the number will be converted to upper case. The number should be input without a prefix. If the Borrower Number specified does not match a Borrower barcode on the database the report terminates with the following

	message: `ERROR: borrower [number] does not exist number`

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
borrower_sites_in=[site],[site]	[site] is a Location Code (case insensitive) for example "BU". If this optional parameter is used and the Home Site of the Borrower does not match a specified Site, this causes the report to terminate with the message: `ERROR: borrower is registered at an excluded site`
borrower_sites_out=[site],[site]	[site] is a Location Code (case insensitive). If this optional parameter is used and the Home Site of the Borrower matches a specified Site this causes the report to terminate with the message: `ERROR: borrower is registered at an excluded site`

bor_charge_stats

bor_charge_stats produces a report listing the amount of money paid and waived in a given period, analysed by the Charge Type within each Library Site. It is possible to report on one, several or all Library Sites. The report produces totals of the amounts paid and waived for the Charge Types:

- Fines
- Hire
- Miscellaneous charges
- Reservations.

There is a grand total for the amounts waived and paid.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/bor_charge_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -pbor_charge_stats -a<filename> -b<start date> -d<database> -e<end date> -h -n<site,site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies start date from which to include charges. The format is "DD/MM/YY" must be used.
-e	This mandatory argument specifies the end date of the date range from which to include charges. The format is "DD/MM/YY" is required. The start and end dates should not be greater than 1 year apart.
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameter is included:

```
pagination = linefeed
```

bor_loan_history

The **bor_loan_history** report produces a list of the loans transactions relating to a particular Borrower. The list may be limited to Items issued in specified period. All the transactions relating to each loan are reported.

Usage

Log on as **talis** and enter the following command:

report -pbor_loan_history -a<filename> -b<begin date> -d<database name> -e<end date> -h -n<borrower number> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument may be used to specify the beginning of a date range of loans to be selected. The format required is "DD/MM/YY" or "DD/MM/YYYY". Only Items issued on or since the date given are reported.
-e	This optional argument may be used to specify the end of a date range of loans to be selected. The format is "DD/MM/YY" or "DD/MM/YYYY". Only Items issued on or before the given date are reported.
-n	This mandatory argument must be used to specify the Borrower Number for the user whose loans are to be reported. Any lower case alphabetic characters in the number will be converted to upper case. This should be specified without a prefix. If the Borrower Number specified does not match a Borrower barcode on the database, the report terminates with the following message: `ERROR: borrower [number] does not exist`

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
borrower_sites_in=[site],[site]	[site] is a Location Code (case insensitive) for example "BU". If this optional parameter is used and the Home Site of the Borrower does not match a specified Site, this causes the report to terminate with the message: `ERROR: borrower is registered at an excluded site`
borrower_sites_out=[site],[site]	[site] is a Location Code (case insensitive). If this optional parameter is used and the Home Site of the Borrower matches a specified Site this causes the report to terminate with the message: `ERROR: borrower is registered at an excluded site`

bor_ite_charge_stats

bor_ite_charge_stats produces a report listing the amount of money paid and waived in a given period, analysed by the Charge Type within each Library Site. It is possible to report on one, several or all Library Sites. The report produces totals of the amounts paid and waived for the Charge Types:

- Fines
- Hire
- Miscellaneous charges
- Reservations.

For the Charge Types "Hire" and "Fines", a further breakdown by Item Type is given. It is possible to report on one, several or "ALL" Item Types. There is a grand total for the amounts waived and paid.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/bor_ite_charge_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -pbor_ite_charge_stats -a<filename> -b<start date> -d<database> -e<end date> -h -n<site,site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies start date from which to include charges. The format "DD/MM/YY" must be used.
-e	This mandatory argument specifies the end date of the date range from which to include charges. The format "DD/MM/YY" is required. The start and end dates should not be greater than 1 year apart.
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
Pagination	The report page breaks after each Site total and the grand total appears on a separate page. `pagination = linefeed` Pagination is controlled by the printer using the "pagination = linefeed" technique. Standard A4 paper size is assumed.
item_types_in	For Fine and Hire Charges only, the charge will only be included in the totals if the transaction to which the charge relates is for an Item of a Type specified. The format is: `item_types_in = [value],[value]` where [value] must be a valid Item Type Code. This is optional and defaults to all Item Types.

bor_loc_stats

bor_loc_stats generates a report on the total number of Borrowers, analysed by Borrower Type within Site. This report excludes Borrowers whose Expiry Date has passed.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/bor_loc_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -pbor_loc_stats -a<filename> -d<database> -h -o<output filename> -r<report directory> -s<data directory>

The Standard script arguments are described here.

Parameter file

The parameter file **bor_loc_stats** is found under the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved. The script uses form feeds, with A4 as the expected paper size.

bor_mes_finedays_ins

bor_mes_finedays_ins sets a standard message against Borrowers with "old" fines still unpaid. Messages are set against a Borrower when that Borrower has outstanding fines older than the value specified by the number of days parameter at the command line. The actual message placed against a Borrower is determined by a Message ID input at the command line. A listing of the Borrowers affected during the run of this script is produced in the output file.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/bor_mes_finedays_ins**. To run in the script log on as **report** or **talis** and enter the following command:

report -pbor_mes_finedays_ins -a<filename> -b<days> -d<database> - h -n<charge> -o<output filename> -q<message id> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Specifies the number of days used to decide whether a trap message should be set against a Borrower. Only Borrowers with a charge incurred create_date older than (today's date minus the number of days specified) will have a trap message set against them. This argument is mandatory.
-n	This specifies the CHARGE_INCURRED_ID from which processing will begin. Only rows with an ID greater than or equal to this will be processed. If "LAST" is specified instead of an ID, then the process will continue from where it stopped in the last run. The default is all CHARGE_INCURRED_IDs with a charge incurred create_date older than (today's date minus the number of days specified). This argument is optional.
-q	Specifies the Message ID of the trap message to be set against Borrowers. The value of -q will only be accepted if the Message ID exists; otherwise the report will produce an error. This argument is mandatory.

Parameter file

The parameter file **bor_loc_stats** is found under the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

Notes

- After the script has been run once, the charges in that range do not need to be processed again (either they were paid or unpaid, and if unpaid the message will have been set already). Avoiding unnecessary processing may become particularly important when the whole CHARGE table has been processed and the script is being run regularly for newly added charges (for example,

running the script every month with -b30 to obtain charges over a month old). There are two ways to run the script to reduce processing time:

-nlast

This will begin where the script finished last. Once set, this argument can be left unchanged.

-nID

Where ID is a CHARGE_INCRRED_ID from the CHARGE_INCURRED table. This could prove useful if you wish to re-run the script over a given range but want to avoid "touching" all charges.

col_shelf_list

The **col_shelf_list** script produces a list of Works which are not fully "satisfied": i.e. where the number of Items required for the Collection Maintenance request exceeds the number of Items assigned to it. It produces a listing by Site, organised in Classmark order by Site. This listing may be used to check the shelves for suitable Items. Details are displayed in the output to aid decisions whether to purchase extra copies, obtain them from another Site or place reservation requests for the Work(s).

Usage

The script's source code is located in **usr/opt/blcmp/talis/mis/col_shelf_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pcol_shelf_list -d<database> -r<report directory> -h -s<data directory> -o<output filename> -a<filename> -n<site>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the Locations for inclusion, separated by a comma if more than one. Site Codes should be used as the arguments: for example, "-nCL,MH,KG". Omitting "-n" will default to all Sites actively using Collection Maintenance. Note: The script only reports on Sites which actively use Collection Maintenance.

Parameter file

The parameter file **bor_loc_stats** is found under the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

cop_dates_upd

The **cop_dates_upd** script updates the Start and End Dates of Borrowers' Addresses for given Borrower Types and, optionally, given Address Names. The updated Start and End Dates are stored in the CONTACT_POINT database table. The script allows multiple Address names to be updated in the same run, as multiple Address names may be specified in the parameter file.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/cop_dates_upd**. To run in the script log on as **report** or **talis** and enter the following command:

report -pcop_dates_upd -a<filename> -b<start date> -d<database> -e<end date> -h -n<borrower type> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument determines the new Start Date to be assigned to the Addresses. The date format may be either "DD/MM/YY" or "DD/MM/YYYY".

-e	This mandatory argument determines the new End Date to be assigned to the Addresses. The date format may be either "DD/MM/YY" or "DD/MM/YYYY".
-n	This mandatory argument specifies the Borrower Types to have their Address Start and End Dates updated. These must be specified in Borrower Type Code format (case insensitive). If required, multiple Borrower Type Codes may be specified separated by a comma; for example "-nSTAF,FULL". Borrower Type Codes must be valid for the database being used.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameter is included:

Parameter	Description
address_name	This parameter allows the user to specify the Address names to be updated with the new dates. If more than one Address name is specified, these must be comma separated. for example: TERM,ADDRESS1 Note: If no Address names are specified, the default is taken to mean all Address names. Address names specified must be an exact match with those used in the database, and are case sensitive. For example "Home Address" will not match "Home address". If an Address name is made up of two or more words, blank spaces can be used but these must be written as in the database.

Notes

- Output is on continuous stationery without page breaks.
- The output is printed in Borrower Type order, within Address Name order. Each Address Name shows the total number of Borrowers with each Borrower Type and the total number of Borrowers that have been given new Address Start and End Dates within the Address name.

conversionMopUp.pl

The **conversionMopUp.pl** script identifies TalisMARC records that require converting to MARC 21 following the completion of the conversion of the bibliographic records. 'It identifies records that have been updated by the Cataloguing Service since the full conversion was run, and records which don't have a MARC 21 equivalent. It reports out the WORK_IDs for manual conversion, or (customers on Alto 4.0 only) prepares the records for automatic conversion (if you have scheduled this to manage one or all of imported EDI quotes, Public Purchase Requests and Self Service ILL requests).

The script is expected to have a limited life and to be used by:

- Libraries who are on Alto 3.0 once they have converted all their Talis MARC records.
- Libraries who are on Alto 4.0 until they upgrade to Alto 4.1

It is not expected to remain in use after Alto 4.1 has been installed.

Usage

You can download the file from the Lyra Programme: MARC Conversion Manager section of http://alto.talis.com which you will need a user name and password to access.

You will need to transfer the file **conversionMopUp.pl** into the **/usr/opt/blcmp/talis/utils/bin** directory on the LMS server once you have downloaded it onto your PC.

When transferring the file from your PC to your LMS Server using FTP do not use the 'bin' command. Doing this may cause the file to be corrupted.

conversionMopUp.pl –b<begin id> -d<database> -e<end id> -h -j<start date> -k -r<report directory> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies which WORK_ID to start processing the WORKS table from. If not given then the processing will start from the lowest WORK_ID in the database.
-e	This optional argument specifies which WORK_ID to end processing at. If not supplied then the processing will continue to the highest WORK_ID in the database.
-j	The –j argument is mandatory and takes the form of a date (dd/mm/yy or dd/mm/yyyy). It is used to identify records which have a WORKS.EDIT_DATE which is equal to or greater than this value. The first time the script is run this should be set to the date the full conversion was started. For subsequent runs it should be set to the date the script was last run.
-k	If this optional argument is set then bibliographic records that need conversion will be fed into the automatic conversion process. (i.e. a dummy row is entered into the WORKS_META table and the TOCONVERT attribute is set to 'T'). Note: you must have loaded Alto 4.0 to set this switch. For details of how to set up the automatic conversion process see section 9. 4 ("Schedule the Conversion software to run automatically") in the Talis Alto 4.0 release notice.
-t	This mandatory argument specifies the type of processing required. If set to CAT_SERV the script looks for records that have been updated by the Cataloguing Service, where WORKS.EDIT_OPERATOR = "BLCMP tape". If set to NO_MARC21 it looks for records where there is no MARC 21 record i.e. there is no WORKS_META row. If set to BOTH it performs both types of processing.

Notes

Following the complete conversion of all the TalisMARC records to MARC 21, some unconverted records may be reintroduced to the system for the following reasons:

For libraries running Alto 3.0

- TalisMARC records created in Talis Alto 3.0 since the conversion was completed. For example, records imported from Talis Base 1, or created via Quick Manage Work.
- Records whose MARC 21 version has been deleted. (This could occur after the conversion has been completed and before the upgrade to Alto 4.)
- TalisMARC records that have been updated since they were converted to MARC 21 (by Cataloguing Services record updates)

For libraries running Alto 4.0:

- Defect 6240 that means that when a multivolume order is created, the 'child' works are only created as TalisMARC records, not as MARC 21. It is therefore not possible to edit the child records, to add extra details etc.

This script can be used to identify any records edited since the full conversion has been completed (using WORKS.EDIT_DATE) and there is no corresponding row in the WORKS_META table (where the MARC21 records are stored).

The script can also be used to identify records that have been updated by Cataloguing service imports. To do this it checks the WORKS.EDIT_DATE and WORKS.EDIT_OPERATOR is 'BLCMP tape'. WORK_IDs can either be reported out for manual conversion or, if you have loaded Alto 4.0, fed into the automatic conversion process (when the script sets the TOCONVERT flag to 'T').

edi_orders_list

The **edi_orders_list** script lists:

- "All Orders waiting to be sent to Suppliers with an "Awaiting transmission" status, and/or
- "All unsuccessful EDI Order transmissions with a "Transmission failed" or "Conversion failed" status.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/edi_orders_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pedi_orders_list -a<filename> -b<start date> -d<database> -e<end date> -h -o<output filename> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies the start of the date range within which Orders are processed. The format required is "DD/MM/YY" or "DD/MM/YYYY".
-e	This optional argument may be used to specify the end of the date range within which Orders are to be processed. The format required is "DD/MM/YY" or "DD/MM/YYYY". If omitted, the end of the date range is taken to be the current date.
-t	This optional argument specifies the type of processing, and hence the type of report to be produced. Valid options are "UNSENT", "UNSUC" or "BOTH". Only one processing type may be specified: ▪ "UNSENT" lists Orders whose status is "Awaiting transmission". ▪ "UNSUC" lists unsuccessful EDI Order transmissions. ▪ "BOTH" produces both lists. If omitted, this argument defaults to "UNSENT".

fun_ill_list

The fun_ill_list script produces a printed report of Interloan Fund statistics, for one, several or all Funds. The report is based on the existing **fun_orders_list** Perl report which reports Fund statistics for Order requests. It lists all Interloan requests with a charge for the specified Fund/s, in a specified Financial year, and where the Interloan was added between the given dates. The report presents the charges in alphabetical order by Fund, and within each Fund by the Interloan create date.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/fun_ill_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pfun_ill_list -a<filename> -b<start date> -d<database name> -e<end date> -h -n<fund code(s)> -o<output filename> -q<financial year> -r<report directory> -s<directory> -t<processing_type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This argument specifies the start date of the date range within which Interloans will be printed by Fund. The Interloan's create date is used, and the format required is "DD/MM/YY". This argument is mandatory.
-e	This argument specifies the end date of the date range within which Interloans will be printed by Fund. The Interloan's create date is used, and the format required is "DD/MM/YY". This argument is mandatory.

-n	This optional argument may be used to specify the Fund Code(s), separated by a comma, for inclusion. If no Fund Codes are specified, the default is all Fund Codes in the current financial year.
-q	This optional argument may be used to specify the Financial year to report on. If used, the display value for the required year should be entered. If no Financial year is specified, the default is the current Financial year.
-t	This optional argument may be used to specify the type of processing, where there are three options: ▪ **UNPAID** means report only unpaid charges for the Fund/s ▪ **PAID** means report only paid charges for the Fund/s ▪ **ALL** means report on all non-deleted charges for the Fund/s. This is the default which applies if no processing type is specified.

Parameter file

The parameter file ("fun_ill_list") is found under the **/usr/opt/blcmp/talis /mis/param** directory. Pagination will ensure that reports on individual Interloans, summary Interloan statistics and summary Fund statistics are not broken up.

fun_order_audit

The **fun_order_audit script** script makes it possible to:

- Monitor any changes to Funds and Orders (i.e. normal Orders and Open Orders).
- Review the Funds assigned to new Orders since the script was last run.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/fun_order_audit**. To run in the script log on as **report** or **talis** and enter the following command:

report -pfun_order_audit -a<filename> -d<database> -h -o<output filename> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-t	This optional argument specifies the type of processing, and hence the type of report to be produced. Valid options are "FULL" or "EXCEPT" (case insensitive). Only one processing type may be specified. If omitted, the default is "EXCEPT". The "FULL" option produces a list of all Orders and Funds which have been edited, and Funds assigned to ordered Items or subscriptions, since the last run. The "EXCEPT" option produces a list of edited Orders and Funds edited or used by Sites other than the parent Site of the Fund or Order, since the last run.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameter is included:

Parameter	Description
institutions	It is possible to specify multi-site Institutions in the parameter file. The names of the Institutions are specified following the "institutions=" parameter, comma separated if there is more than one Institution.

The Sites relating to each Institution are specified in the form:

```
[inst]=[site],[site]
```

where [inst] is an Institution named in the "institutions=" parameter and [site] is a Site Code.

Note: Funds and Orders created and edited at different Sites linked to the same Institution do not generate reports when the script is run in "exception only" mode.

For example:

```
institutions=SOUTH,NORTH
SOUTH=WH,HG,NR
NORTH=KT,MG,KH,DH
```

Notes

- When the **fun_order_audit** script is run for the first time the following considerations apply:

  The script run in "full report" mode picks up all edited Orders and Funds, and all Funds assigned. The script run in "exception only" mode picks up all Orders or Funds which have been used or modified at another location.

- In "exception only" mode, an Order which has used Funds belonging to another institution is reported in successive reports until it has been dealt with.

fun_user_links

The **fun_user_links** script reports on which fund user profiles are linked to which funds, and which users are linked to which fund user profiles. This is because some systems need to be able to limit the use of individual funds to specific users, and therefore create links between operators and funds via the fund user profile.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/fun_user_links**. To run in the script log on as **report** or **talis** and enter the following command:

report –pfun_user_links -d<database> –h -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-t	Determines the processing type. The valid options are 'FUND' and 'USER' (case-insensitive). If the processing type is not given with the –t argument , processing will terminate with the standard error for a mandatory argument not being specified. If the value provided with the –t argument is any value other than FUND or USER, processing will terminate with the error message Invalid processing type <processing_type> specified with –t argument .

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
FUND_CODE	The FUND_CODE parameter will enable libraries to only produce a report for specified fund codes and will be used in conjunction with the processing type 'FUND'. If the processing type is not 'FUND', any value provided with FUND_CODE will be ignored.

OPERATOR	The OPERATOR parameter will enable libraries to only produce a report for specified operators and will be used in conjunction with the processing type 'USER'. If the processing type is not 'USER', any value provided with OPERATOR will be ignored.
FINANCIAL_YEAR	The FINANCIAL_YEAR parameter will default to the current financial year if not specified.

fun_orders_list

The **fun_orders_list script** lists all normal Orders by Fund. Orders with an Order Date between given dates are listable, by one, several or all Funds. Orders are grouped by Status within each Fund. Reports are generated for Funds in the current Financial Year only. The date range search is based on Order Date rather than Order Generation Date.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/fun_orders_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pfun_orders_list -a<filename> -b<start date> -d<database> -e<end date> -h -n<fund code(s)> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This argument specifies the start date of the date range within which Orders will be printed by Fund. The format is "DD/MM/YY". This argument is mandatory.
-e	This argument specifies the end date of the date range within which Orders will be printed by Fund. The format required is "DD/MM/YY". This argument is mandatory.
-n	This optional argument may be used to specify the Fund Code(s), separated by a comma, for inclusion. If no Fund Codes are specified, the default is all Fund Codes in the current financial year.

Parameter file

The parameter file **fun_orders_list** is found under the **/usr/opt/blcmp/talis /mis/param** directory. Pagination will ensure that reports on individual Orders, summary Order statistics and summary Fund statistics are not broken up.

ill_art_cancel

This script is replaced in Alto 5.4 by **ill_art_cancel_v2**.

The **ill_art_cancel** script generates ARTEmail transmissions to the BLDSC to cancel ill requests. Criteria on which this script will select are:

- Status = Cancelled
- Request Method = ARTEmail
- One of the following Report codes is associated with the request: "USE", "USE O/D", "O/O WL", "HW".

Output will be in the format of standard ARTEmail ASCII files within the pending mailbox, and the TX line will contain the word "CANCEL". The line following the TX line will contain the text "Please remove from waiting list". The ill_art_cancel.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_cancel**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_cancel -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_cancel script now uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line (previously it used the TYPE_STATUS.CODE).
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_cancel_v2

The **ill_art_cancel_v2** script generates ARTEmail transmissions to the BLDSC to cancel inter-library loan requests.

It is a replacement, released with Alto 5.4, for the original **ill_art_cancel** script.

Criteria on which this script will select are:

- Status = Cancelled
- Request Method = ARTEmail
- The request has received a report with one of the Report codes listed in the **required_codes** parameter. This parameter is set by default to the codes: "USE", "USE O/D", "O/O WL", "HW".

Output will be in the format of standard ARTEmail ASCII files within the pending mailbox, and the TX line will contain the word "CANCEL". The line following the TX line will contain the text "Please remove from waiting list". The ill_art_cancel_v2.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_cancel_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_cancel_v2 -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

Notes

- The ill_art_cancel_v2 script uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line.
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_chase

This script is replaced in Alto 5.4 by **ill_art_chase_v2**.

The **ill_art_chase** script generates chaser ARTEmail transmission to the BLDSC for ill requests. Criteria on which this script will select are:

- The supplier must have the code **BLDSC** or **DSC** or **XY/N-1**.
- Status = Requested
- Request Method = ARTEmail
- Where Search level is "X", request was requested 14 days ago or more.
- There are no reports from BLDSC associated with the request report unless the codes **BUP**, **LONDON**, **ABROAD**, **O/O WL**, **SUPPL** or **USE** have been received and added.
- The request has not been chased before.

Output will be a standard ARTEmail ASCII file (written to the pending mailbox), and the TX line will contain the word "CHASER". The ill_art_chase.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_chase**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_chase -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may

	decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_chase script now uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line (previously it used the TYPE_STATUS.CODE).
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_chase_v2

The **ill_art_chase_v2** script generates chaser ARTEmail transmissions to the BLDSC for ill requests.

It is a replacement, released with Alto 5.4, for the original **ill_art_chase** script.

Criteria on which this script will select are:

- The supplier must have the code **BLDSC** or **DSC** or **XY/N-1**.
- Status = Requested
- Request Method = ARTEmail

- The request was requested 14 days ago or more.
- No reports from BLDSC have been added to the request.
- The request has not been chased before.

Output will be a standard ARTEmail ASCII file (written to the pending mailbox), and the TX line will contain the word "CHASER". The ill_art_chase_v2.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_chase_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_chase_v2 -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_chase_v2 script uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line.
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author

- Thesis details
- Series details

- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_new

This script is replaced in Alto 5.4 by **ill_art_new_v2**.

The **ill_art_new** script generates an ARTEmail transmission to the BLDSC for new ill requests. Criteria on which this script will select are:

- Status = Pending
- Request Method = ARTEmail
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

The status of requests processed by this script will be updated to "Requested". The ill_art_new.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_new**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_new -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).

-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_new script now uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line (previously it used the TYPE_STATUS.CODE).
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_new_v2

The **ill_art_new_v2** script generates an ARTEmail transmission to the BLDSC for new ill requests.

It is a replacement, released with Alto 5.4, for the original **ill_art_new** script.

Criteria on which this script will select are:

- Status = Pending
- Request Method = ARTEmail
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

The status of requests processed by this script will be updated to "Requested". The ill_art_new_v2.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_new_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_new_v2 -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific

	Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_new_v2 script uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line.
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_reapply

This script is replaced in Alto 5.4 by **ill_art_reapply_v2**.

The **ill_art_reapply** script generates an ARTEmail transmission to the BLDSC for reapplications. Criteria on which this script will select are:

- Status = Reapply
- Request Method = ARTEmail
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

The output will be in the form of a standard ARTEmail ASCII file in the pending mailbox, but the TX line will contain the word "REAPP". The status of requests processed by this script will be updated to "Requested". The ill_art_reapply.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_reapply**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_reapply -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_reapply script now uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line (previously it used the TYPE_STATUS.CODE).
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.

- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_reapply_v2

The **ill_art_reapply** script generates an ARTEmail transmission to the BLDSC for reapplications.

It is a replacement, released with Alto 5.4, for the original **ill_art_reapply** script.

Criteria on which this script will select are:

- Status = Reapply
- Request Method = ARTEmail
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

The output will be in the form of a standard ARTEmail ASCII file in the pending mailbox, but the TX line will contain the word "REAPP". The status of requests processed by this script will be updated to "Requested". The ill_art_reapply_v2.rep file will be written to **/usr/opt/blcmp/data/mis**.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_reapply_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_reapply_v2 -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_reapply_v2 script uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line.
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_renew

This script is replaced in Alto 5.4 by **ill_art_renew_v2**.

The **ill_art_renew** script generates an ARTEmail to the BLDSC transmission for renewals. Criteria on which this script will select are:

- Criteria on which this script will select are:
- Status = Renew Pend
- Request Method = ARTEmail

The output will be standard ARTEmail ASCII files file in the pending mailbox, the TX line will contain Request Numbers pertaining to the renewal and the word "RENEW". The first line after the TX line will contain the text "Please renew" followed by the original Request Number. The status of requests processed by this script will be updated/set-back to "On Loan". The ill_art_renew.rep file will be written to **/usr/opt/blcmp/data/mis**.

While Alto will assign a new request number when a request is renewed online, the ill_art_renew script will send the original request number to the BLDSC. You will be able to search for the original request number within Alto.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_renew**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_renew -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run.

	If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_renew script now uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line (previously it used the TYPE_STATUS.CODE).
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.
- Details of rows that are truncated or that are not sent are included in the report file.

ill_art_renew_v2

The **ill_art_renew_v2** script generates an ARTEmail to the BLDSC transmission for renewals.

It is a replacement, released with Alto 5.4, for the original **ill_art_renew** script.

Criteria on which this script will select are:

- Status = Renew Pend
- Request Method = ARTEmail

The output will be standard ARTEmail ASCII files file in the pending mailbox, the TX line will contain Request Numbers pertaining to the renewal and the word "RENEW". The first line after the TX line will contain the text "Please renew" followed by the original Request Number. The status of requests processed by this script will be updated/set-back to "On Loan". The ill_art_renew_v2.rep file will be written to **/usr/opt/blcmp/data/mis**.

While Alto will assign a new request number when a request is renewed online, the ill_art_renew_v2 script will send the original request number to the BLDSC. You will be able to search for the original request number within Alto.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_art_renew_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_art_renew_v2 -n<start processing> -b<begin date> -e<end date>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

Notes

- The ill_art_renew_v2 script uses the TYPE_STATUS.NAME attribute when finding out the search level details to include in the TX line.
- Only request numbers with up to a three character prefix and up to a five digit number will generate a request.
- Certain bibliographical data will be split over two lines (e.g. 80 characters). These are:
  - Journal/Book Title
  - Volume/Title
  - Author
  - Thesis details
  - Series details
- The source of reference details will be split over three lines (e.g. 120 characters) if necessary. Any data that exceeds these limits will be truncated when the request is sent.
- It is possible that a request may exceed the total number of rows that can be sent to the BLDSC in a single request. In this situation the request will not be sent to the BLDSC as the request would fail.

- Details of rows that are truncated or that are not sent are included in the report file.

ill_letter_chase

This script is replaced in Alto 5.4 by **ill_letter_chase_v2**.

The **ill_letter_chase** script generates ill chase letters for suppliers. Criteria on which this script will select are:

- Status = Requested
- Request Method = Print
- No report codes have been received.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_chase**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_chase -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan reqiests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument . If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_chase_v2

The **ill_letter_chase_v2** script generates ill chase letters for suppliers.

It is a replacement, released with Alto 5.4, for the original **ill_letter_chase** script.

Criteria on which this script will select are:

- Status = Requested
- Request Method = Print
- No report codes have been received.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_chase_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_chase_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan reqiests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument . If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_new

This script is replaced in Alto 5.4 by **ill_letter_new_v2**

A single general command line is used to request generation of all types of letters and e-mail transmission files; the different types of letter/transmission files being identified by the parameter file specified on the command line. The different types of Supplier letters and ART e-mail transmissions available are identified by the ill_letter or ill_art parameter file which has to be specified at the command line when running the scripts. Each Supplier Letter script ("ill_letter") produces an output file (".out") and a report (".rep") file. The scripts for generating BLDSC ART files ("ill_art") produce a separate file for each request.

The **ill_letter_new** script generates new ill request letters for suppliers. Criteria on which this script will select are:

- Status = Pending
- Request Method = Print
- Effective Date <= Today's date unless -b and-e have been specified

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_new**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_new -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_new_v2

A single general command line is used to request generation of all types of letters and e-mail transmission files; the different types of letter/transmission files being identified by the parameter file specified on the command line. The different types of Supplier letters and ART e-mail transmissions available are identified by the ill_letter or ill_art parameter file which has to be specified at the command line when running the scripts. Each Supplier Letter script ("ill_letter") produces an output file (".out") and a report (".rep") file. The scripts for generating BLDSC ART files ("ill_art") produce a separate file for each request.

The **ill_letter_new_v2** script generates new ill request letters for suppliers.

It is a replacement, released with Alto 5.4, for the original **ill_letter_new_v2** script.

Criteria on which this script will select are:

- Status = Pending
- Request Method = Print
- Effective Date <= Today's date unless -b and-e have been specified

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_new_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_new_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_reapply

This script is replaced in Alto 5.4 by **ill_letter_reapply_v2**.

The **ill_letter_reapply** script generates ill re-application letters for suppliers. Criteria on which this script will select are:

- Status = Reapply
- Request Method = Print
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_reapply**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_reapply -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument . If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_reapply_v2

The **ill_letter_reapply_v2** script generates ill re-application letters for suppliers.

It is a replacement, released with Alto 5.4, for the original **ill_letter_reapply** script.

Criteria on which this script will select are:

- Status = Reapply
- Request Method = Print
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_reapply_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_reapply_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument . If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_renew

This script is replaced in Alto 5.4 by ill_letter_renew_v2.

The **ill_letter_renew** script generates ill request renewal letters for suppliers. Criteria on which this script will select are:

- Status = Renew Pend
- Request Method = Print

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_renew**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_renew -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table

	becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new") and new ART requests ("ill_art_new"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument. If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_letter_reapply_v2

The **ill_letter_reapply_v2** script generates ill re-application letters for suppliers.

It is a replacement, released with Alto 5.4, for the original **ill_letter_reapply** script.

Criteria on which this script will select are:

- Status = Reapply
- Request Method = Print
- Effective Date <= Today's date unless "-b" and "-e" have been specified.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_letter_reapply_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_letter_reapply_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways:

	▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run. If the last run cannot be determined then the whole database will be processed. The use of "-nLAST" should be adopted with caution. As it only processes newer rows since the last run it is not really suitable for Cancellations, Chasers or Renewals. It is best suited to new Letters to Suppliers ("ill_letter_new_v2") and new ART requests ("ill_art_new_v2"). Note, however, that "-nLAST" will not pick up deferred requests (i.e. Interloan requests created some time ago with Effective Dates in the future).
-b	The argument "-b" is used to specify the start date of an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date of an inclusive date range. This is used with the "-b" argument . If not specified, processing will terminate when all Interloan requests have been processed up until (and including) the current date.

ill_memo_arrival

This script is replaced in Alto 5.4 by **ill_memo_arrival_v2**.

The **ill_memo_arrival** script generates arrival notification letters for requesters. Criteria on which this script will select are:

- Status = "Received"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_arrival**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_arrival -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of

	Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_arrival_v2

The **ill_memo_arrival_v2** script generates arrival notification letters for requesters.

It is a replacement, released with Alto 5.4, for the original **ill_memo_arrival** script.

Criteria on which this script will select are:

- Status = "Received"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_arrival_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_arrival_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_cancel

This script is replaced in Alto 5.4 by **ill_memo_cancel_v2**.

The **ill_memo_cancel** script generates cancellation letters for requesters. Criteria on which this script will select are:

- Status = "Cancelled"
- Request has a Report other than "RQCANC" or "WEBCANC"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_cancel**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_cancel -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_cancel_v2

The **ill_memo_cancel_v2** script generates cancellation letters for requesters. Criteria on which this script will select are:

- Status = "Cancelled"
- Request has a Report other than "RQCANC" or "WEBCANC"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_cancel_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_cancel_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_confirm

This script is replaced in Alto 5.4 by **ill_memo_confirm_v2**.

The **ill_memo_confirm** script confirms renewal requests to requesters. Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- Request has a Report "RENEW CONF".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_confirm**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_confirm -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data.

	Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_confirm_v2

The **ill_memo_confirm_v2** script confirms renewal requests to requesters.

It is a replacement, released with Alto 5.4, for the original **ill_memo_confirm** script.

Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- The request has received a report with one of the Report codes listed in the **required_codes** parameter. This parameter is set by default to the codes:"RENEW CONF", "RENEWED".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_confirm_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_confirm_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes

-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_delay

This script is replaced in Alto 5.4 by **ill_memo_delay_v2**.

The **ill_memo_delay** script generates letters indicating that items have been delayed. Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- One of the following Reports is associated with the request: "ABS", "BDG", "BUP", "CAT", "CONS", "LOC", "ENDX", "HW", "LOC", "LOC SEARCH", "LOST", "MISS", "NCOP", "NFICT", "NLOAN", "NOE", "NOP", "NOS", "NOT", "NPUR/BR", "NPUR/EX", "NPUR/NA", "NPUR/NR", "NPUR/OP", "NQ", "NRL", "NTRANS", "NUKL", "NYP", "OLWL", "O/O" "O/O NLOAN", "O/O WL", "OOS", "OOS/OR", "OWN", "RETRY", "SELPUR", "TRY", "USE", "USE O/D", "WD", "WLNR", "WLR", "WLS".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_delay**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_delay -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of

	Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_delay_v2

The **ill_memo_delay_v2** script generates letters indicating that items have been delayed.

It is a replacement, released with Alto 5.4, for the original **ill_memo_delay** script.

Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the codes: "ABROAD", "BDG", "BUP", "CAT", "CAT REAPPL", "CONS LOC", "DIRECT", "ENDSEARCH", "ENDX", "HW", "LOC", "LOC SEARCH", "LONDON", "LOST", "MISS", "NCOP", "NFICT", "NLOAN", "NO BACKUPS", "NOE", "NOP", "NOS", "NOT", "NOT NUKL", "NPUR/BR", "NPUR/EX", "NPUR/NA", "NPUR/NR", "NPUR/OP", "NQ", "NRL", "NTRANS", "NUKL", "NYP", "OLWL", "O/O", "O/O NLOAN", "O/O WL", "OOS", "OOS/OR", "OWN", "RCOP", "RETRY", "SELPUR", "TRY", "USE", "USE O/D", "WD", "WLNR", "WLR", "WLS".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_delay_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_delay_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument. If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).

-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_form

This script is replaced in Alto 5.4 by **ill_memo_form_v2**.

The **ill_memo_form** script generates letters indicating that items are available in alternate formats. Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- One of the following Reports is associated with the request: "ABS", "FICHE", "FILM"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_form**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_form -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the

	current date.

ill_memo_form_v2

The **ill_memo_form_v2** script generates letters indicating that items are available in alternate formats.

It is a replacement, released with Alto 5.4, for the original **ill_memo_form** script.

Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the codes: "ABS", "FICHE", "FILM"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_form_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_form_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes

	must exist in the ILL_REPORT_CODES table.

ill_memo_overdue

This script is replaced in Alto 5.4 by **ill_memo_overdue_v2**.

The **ill_memo_overdue** script generates overdue loan letters for requesters. Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- One of the following Reports is associated with the request: "DUE", "DUE BILL", "OVERDUE".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_overdue**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_overdue -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_overdue_v2

The **ill_memo_overdue_v2** script generates overdue loan letters for requesters.

It is a replacement, released with Alto 5.4, for the original ill_memo_overdue script.

Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the codes:: "DUE", "FINAL DEMA", "OVERDUE".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_overdue_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_overdue_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument. If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_recall

This script is replaced in Alto 5.4 by ill_memo_recall_v2.

The **ill_memo_recall** script generates recall letters to requesters, for items on loan. Criteria on which this script will select are:

- Status = "Received", "On Loan" or "Renew Pend"
- The request has a "DUE WAIT" report.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_recall**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_recall -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_recall_v2

The **ill_memo_recall_v2** script generates recall letters to requesters, for items on loan.

It is a replacement, released with Alto 5.4, for the original ill_memo_recall script.

Criteria on which this script will select are:

- Status = "Received", "On Loan" or "Renew Pend"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the code: "DUE WAIT".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_recall_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_recall_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end

	of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_refuse

This script is replaced in Alto 5.4 by **ill_memo_refuse_v2**.

The **ill_memo_refuse** script generates renewal refusal letters to requesters. Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- Request has a Report "NRENEW" or "NO_RENEW"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_refuse**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_refuse -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways:

	▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_refuse_v2

The **ill_memo_refuse_v2** script generates renewal refusal letters to requesters.

It is a replacement, released with Alto 5.4, for the original **ill_memo_refuse** script.

Criteria on which this script will select are:

- Status = "On Loan" or "Renew Pend"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the codes: "NO_RENEW", "NRENEW"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_refuse_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_refuse_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on

	the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_source

This script is replaced in Alto 5.4 by **ill_memo_source_v2**.

The **ill_memo_source** script generates letters to requesters asking for more information about the item, including source of reference. Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- One of the following Reports is associated with the request: "CRF/ANI", "CRF/ART", "CRF/CONF", "ERR", "NCPAP", "NPUR", "NPUR/NT", "NPUR/OP", "SOR", "QOR", "CRF/DNF", "CFR/INF", "WEBSOR", "CONF", "CRF"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_source**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_source -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on

	the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_source_v2

The **ill_memo_source_v2** script generates letters to requesters asking for more information about the item, including source of reference. Criteria on which this script will select are:

- Status = "Pending", "Reapply" or "Requested"
- The request has received a report with one of the Report codes listed in the required_codes parameter. This parameter is set by default to the codes: "CONF", "CRF CANNOT", "CRF/ANI", "CRF/ART", "CRF/CONF", "CRF/DNF", "CRF/INF", "ERR", "NCPAP", "NPUR", "NPUR/NT", "NPUR/OP", "QOR", "SOR", "WEBSOR"

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_source_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_source_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: ▪ An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. ▪ The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
required_codes	This parameter lists report codes which must have been added to the request in order for the script to select the request for processing. Report codes should be entered in uppercase and separated by commas with no spaces. The codes must exist in the ILL_REPORT_CODES table.

ill_memo_uncollected

This script is replaced in Alto 5.4 by **ill_memo_uncollected_v2**.

The **ill_memo_uncollected** script generates uncollected request letters to requesters. Criteria on which this script will select are:

- Status = "Received"
- Collect By date is earlier than tomorrow's date.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_cancel**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_uncollected -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_memo_uncollected_v2

The **ill_memo_uncollected_v2** script generates uncollected request letters to requesters. Criteria on which this script will select are:

- Status = "Received"
- Collect By date is earlier than tomorrow's date.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_memo_uncollected_v2**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_memo_uncollected_v2 -n<start processing> -b<begin date> -e<end date> -o<output file> -r<report directory> -d<database> -a<filename> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This argument states the start point of the run in terms of an ILL_ID. Its purpose is to prevent unnecessary processing of ILL_IDs which are "lower" in the table. It could become an important argument if the main Interloan table becomes large. For example, after running Interloans for 3 years you may decide to always run using -n100000 where ILL_ID 10,000 represents the end of the first year. That is, all reports should bypass the first year of data. Specifying a starting point for processing may be done in one of two ways: An integer, indicating that the script should start processing at a specific Interloan request ID or Report ID. The word "LAST", indicating that the script will effectively process only the records added since the last time it was run The use of "-nLAST" should be adopted with caution. Its use is particularly suited to memos which produce outputs based on the chronological receipt of Report Codes
-b	The argument "-b" is used to specify the start date to be used for an inclusive date range. This is used with the "-e" argument . If not given, processing will start at the beginning (i.e. at the Interloan request with the earliest date on the database).
-e	The argument "-e" specifies the end date in an inclusive date range. This is used with the "-b" argument . If not given, processing will end when all Interloan requests have been processed up until (and including) those on the current date.

ill_unverified

The **ill_unverified** script produces a list of Interloan requests which are currently unverified and have a status of "Pending". The list can be limited by Date range and Site. This report may be used in two ways:

- Frequently, to select Interloan requests created recently (for example, daily).
- Periodically, to identify older Interloan requests which are still unverified. Display of Report Codes assist Staff in ascertaining why the Interloan requests are still unverified (for example, awaiting for the requestor's source of reference).

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ill_unverified**. To run in the script log on as **report** or **talis** and enter the following command:

report -pill_unverified -a<appended filename> -b<days> -d<database> -e<days> -h -n<site,site(s)> -o<output filename> -r<report directory> -s<data directory> -v

Standard script arguments are described here. The arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the minimum number of days since an unverified request was created for inclusion in the report. If the argument is omitted, the report does not discriminate by minimum request age, i.e. even new unverified requests are listed.
-e	This optional argument specifies the maximum number of days since an unverified request was created for inclusion in the report. If the argument is omitted, the report does not discriminate by maximum request age, i.e. long-standing unverified requests are listed.
-n	This optional argument may be used to specify the Delivery Site Code(s), separated by a comma, for inclusion. If no Delivery Site Codes are specified, the default is all Site Codes.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

IS_loa_odue_letter

The **IS_loa_odue_letter** report differs from the standard Overdues Letters script (**loa_odue_letter**) in one way. It has been designed for Islington Council Leisure and Library Services (IS) to work with Economailer datamailers.

Libraries using Economailers can run this version of overdues. It is not in itself bespoke to Islington Libraries. As this script differs in only the one way, all other functionality can be found in **loa_odue_letter**.

ite_miss_del

The **ite_miss_del** script produces a list of all Items which are either deleted and/or missing. The Items listed are ordered by Site, Status, "Last copy condition", Classmark, Author and then by Title. The definition of missing is specified using a parameter which accepts user-defined Item Status Codes. This effectively allows the listing of all Items of any user-defined status (for example, binding).

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ite_miss_del**. To run in the script log on as **report** or **talis** and enter the following command:

report -pite_miss_del -a<appended filename> -b<days> -d<database> -e<days> -h<help> -n<site,site> -o<output filename> -q<status_code,status_code> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the minimum number of days at Item must have been at the missing or deleted status given by -q. The value is an integer which defaults to zero.
-e	This optional argument specifies the maximum number of days an Item must have been at the missing or deleted status given by -q. The value is an integer, which if omitted defaults 9999 days. The highest value allowed is also 9999 days.
-n	This optional argument specifies which Site codes are to be included in the processing of the report. If omitted, it will default to all Sites. The format required is the location code(s), separated by commas, for example "-nBU,WV".

-q	This mandatory argument specifies the Item Status Codes to be included and reported on. The format required is the Item Status Codes, separated by commas; for example, "-qMISS,WITH,DEL". Only user-defined Item Status Codes or that for deleted (Items) are acceptable.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
last_copy_check	This is a mandatory parameter, the value of which must be "yes" or "no". If "last_copy_check" is set to "yes" (any case) then a text message is included at the end of an item record if there are no Items of status "in stock/loanable" for the associated Work. By default the last copy warning message should read: `"*** There are now no "In stock" copies associated with this work ***".` The user is able to amend the text of this message within "format.pl". If "last_copy_check" is set to "no" then the check is not made.

standard_item_message	This is an optional parameter, where <value> must be a single integer which is a valid standard Item Message id. The parameter is optional.
	The message ids associated with standard Item Messages can be found in the Online Utilities parameter interface for messages (i.e. under Parameters Names Messages Item). Alternatively the following SQL will produce a list of standard item messages with their associated Ids:
	`select MESSAGE_ID, convert(char(60),TEXT)`
	`from MESSAGE`
	`where MESSAGE_ID >=1`
	`and TYPE=1`
	`and STANDARD='T'`

ite_wrk_upd

The **ite_wrk_upd** script produces a listing, by Supplier and Official Order Number, in Control Number order, of Works and Items which have been updated since a given date. The Bibliographic data is output in ISBD format, along with the main classmark of the Work. The Item data includes total number of copies added The Site, Barcode, Status, Type and Shelfmark are indicated for each Item. The script may be run either interactively or from the "cron".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/ite_wrk_upd**. To run in the script log on as **report** or **talis** and enter the following command:

report -pite_wrk_upd -a<appended filename> -b<begin date> -d<database> -e<end date> -h -n<supplier, supplier> -o<output filename> -r<report directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This specifies the beginning of a date range of ITEMs to be processed where "ite_wrk_update" has updated the database. Only ITEMs updated on or since the specified date will be processed. This argument is mandatory. The date should be entered in the format "DD/MM/YY".

-e	This specifies the end date for processing where "ite_wrk_update" has updated the database. This argument is mandatory. The date should be entered in the format "DD/MM/YY".
-n	This argument specifies the Supplier codes, comma separated, which are to be processed. If this argument is absent, all suppliers will be processed.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

iti_transit

The **iti_transit** script produces a listing of Items currently in transit. The list is in two logical sections, showing first a list of Items by sending location and then a list by destination site. The list can be limited by date and Site. This list shows what each Site is sending and expecting to receive. The output can be compared with that from iti_transit_del; the latter list is useful as a shelf-check at the Item's Home Site.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/iti_transit**. To run in the script log on as **report** or **talis** and enter the following command:

report -piti_transit -a<filename> -b<start date> -d<database> -e<end date> -h<help> -n<site,site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument that specifies the minimum number days an Item must have been in transit for it to be selected. The argument is a number, which if omitted is taken to be zero days.
-e	This optional argument that specifies the maximum number days for an Item must have been in transit for it no longer to be selected. The argument is a number, and if omitted there is no maximum age.
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV". Note: A given location will be reported for both sending and destination locations.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

iti_transit_del

The **iti_transit_del** script removes Items from in transit based on a removal date parameter. Optionally, the Items removed can be given a user-defined status, for example "Missing"; but only if such an amendment is allowed for a given Item within Alto. The report produces a list of Items effected, sorted by home site of the Item. The output list allows a shelfcheck to be made. By default the report runs in "report mode" only, i.e. without updating the database; but an update parameter allows it to be run in "update mode". Both the report and the output indicate what is or is not being updated.

Usage

The script's source code is located in **usr/opt/blcmp/talis/mis/iti_transit_del**. To run in the script log on as **report** or **talis** and enter the following command:

report -piti_transit_del -a<appended filename> -b<start date> -d<database name> -e<end date> -h<help> -n<site,site> -o<output filename> -q<status> -r<report directory> -s<data directory> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies start date from which to include charges. The format "DD/MM/YY" must be used.
-e	This mandatory argument specifies the end date of the date range from which to include charges. The format "DD/MM/YY" is required. The start and end dates should not be greater than 1 year apart.
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".
-q	This optional argument specifies the new Item Status when updating the Items. This only takes effect if the report is run with the -u "update mode" option, and if certain integrity checks permit. If the -q argument is omitted then the Item Status is not updated. The format required is a single Item Status Code; for example "- qWITH". Only user-defined statuses are acceptable.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

Notes

- The list is sorted by the Home Site of the Item and each in transit Item record is ordered by Classmark, then by Author and Title within Classmark.
- A page break is inserted after each Item Site change. Paper size is controlled by the printer allowing "standard" paper sizes, such as A4.

iti_transit_stats

The **iti_transit_stats** script produces statistics on the number of Items currently in transit, by Site. Statistics are produced for both the sending and destination locations.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/iti_transit_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -piti_transit_stats -a<appended filename> -b<minimum days> -d<database> -e<maximum days> -h<help> -n<site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument that specifies the minimum number days an Item must have been in transit for it to be selected. The argument is an integer, and if omitted is taken to be zero days.

-e	This optional argument that specifies the maximum number days for an Item to have been in transit for it no longer to be selected. The argument is an integer, and if omitted there is taken to be no maximum age.
-n	This optional argument specifies which location is being reported. If not specified this defaults to all Sites. The format required is a single location code. If a single Site Code is specified then the resulting statistics show the traffic to and from that location. The "Sending Locations" sections gives a count of the number of transits being sent to the Site. The "Destination Locations" gives statistics for transits received from the nominated Site.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

itm_onloan_stats

The **itm_onloan_stats** script generates a report to show the number of Items currently on loan, broken down by Item Type.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/itm_onloan_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -pitm_onloan_stats -a<appended filename> -d<database> -h -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument that specifies the minimum number days an Item must have been in transit for it to be selected. The argument is an integer, and if omitted is taken to be zero days.
-e	This optional argument that specifies the maximum number days for an Item to have been in transit for it no longer to be selected. The argument is an integer, and if omitted there is taken to be no maximum age.
-n	This optional argument specifies which location is being reported. If not specified this defaults to all Sites. The format required is a single location code. If a single Site Code is specified then the resulting statistics show the traffic to and from that location. The "Sending Locations" sections gives a count of the number of transits being sent to the Site. The "Destination Locations" gives statistics for transits received from the nominated Site.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

itm_rotate

If an item is not on loan and becomes due for rotation then it is added to the shelf-check list that is generated by an off-line MIS report. The **itm_rotate** report enables library staff to locate these items and ensure that they are correctly rotated to the next site.

This report should he run at regular intervals to identify items on the shelves which are due to he rotated. After running the report, the output file should he used to pick the stock off the shelf. Each item then needs to be checked-out of the current site to place it in transit to the next site in the plan.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/itm_rotate**. To run in the script log on as **report** or **talis** and enter the following command:

report -pitm_rotate -d<database> -h -n<location> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	Specifies the location codes for sites to be included in the processing. ▪ Separate multiple locations with a comma. ▪ If not set then all sites will be included.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are involved.

Notes

- Each location that participates within the rotation scheme will have a site-specific output file generated. The output file will identify the each item and include the following information.
  - Sequence code
  - Size Code
  - Class number
  - Suffix
  - Rotation plan
  - Next site
  - Rotation date
  - Author
  - Title
  - Barcode
- The output will be in the order of sequence code, size code, class number and suffix.

loa_borr_loan_stats

The **loa_borr_loan_stats** script produces a count of loan transactions. The column headings are locations and the row headings are Borrower Types. The create location can be a logical grouping, giving the ability to report on a user defined Site Location Name which may encompass one or more Site IDs. These logical locations are user definable within the parameter file. Output is in the form of a table and can produce results for a specified date range for one of four types of statistics:

- **iss**: Issue statistics only
- **issren**: Issue and renew statistics
- **ren**: Renew statistics only
- **dis**: Discharge statistics only

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_borr_loan_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_borr_loan_stats -a<filename> -b<begin date> -d<database name> -e<end date> -h -o<output filename> -r<report directory> -s<data directory> -t<statistics>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-b	This specifies the beginning of a date range or rows to be processed. Only LOAN rows with a CREATE_DATE on or since the specified date will be processed. It is a mandatory argument . Note: The maximum value for this argument is one year ago.
-e	This optional argument specifies the end of a date range of rows to be processed. Only LOAN rows with a CREATE_DATE on or before the specified date will be processed. If not given, it defaults to today's date. Note: The maximum range between -b and -e is one year.
-t	This specifies the type of statistics to be reported. Values must be either: **iss**, **issren**, **ren** or **dis**. It is a mandatory argument .

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
Locations	A list of locations can be entered in the parameter file. These will be the column headings under which the issue statistics are reported. Each location listed in the parameter file may point to more than one CREATE_LOCATION. If so, another variable of the same name as the location will need to be entered by the user into the parameter file, with value(s) set to the various CREATE_LOCATIONs that comprise the logical location name. Otherwise, the location name is taken to be the CREATE_LOCATION to be reported on. In the parameter file example shown above, the report column headings will be "PL", "SOUTH", "WEST", "N" where: ▪ "PL" represents the CREATE_LOCATION of "PL" ▪ "SOUTH" represents the CREATE_LOCATIONs of "W" & "GL" ▪ "WEST" represents the CREATE_LOCATIONs of "BSTL" & "SUSX" ▪ "N" represents the CREATE_LOCATION of "N". Note: The order of location headings is the same as that input into the parameter file and hence is user definable. If a location does not exist (in the database) it will be included in the column headings and within the Site Totals. It will always report counts of "0". It is effectively treated as a logical Site which is not assigned any true locations.
Borrower_types	The user can define which Borrower Types are to be included or excluded in the report by using one of the variables in the parameter file: ▪ borr_types_in or ▪ borr_types_out These cannot be used together.
num_loan_batches	The number of loans to process could be many millions, so processing has to be done in batches. This optional parameter determines the number of LOAN rows to be processed in each batch. It sets the maximum number of LOANs that can be retrieved at any one time from the database. This is not related to the number of statistics that are included in the output. This value should not require any alteration.
Page dimensions	By using the variables pagewidth and pagelength specified in the parameter file, the user is able to determine, to an extent, how many locations headings

appear on a line.

Example

```
#
# Parameter file for loa_borr_loan_stats
#
# Author: ahg/ogb
#
#
main = loa_borr_loan_stats/main.pl
preselect = loa_borr_loan_stats/preselect.pl
select = loa_borr_loan_stats/select.pl
format = loa_borr_loan_stats/format.pl
retrieve = loa_borr_loan_stats/retrieve.pl
locations = PL,SOUTH,WEST,N

SOUTH=W,GL
WEST=BSTL,SUSX
# borr_types_in =
# borr_types_out =
num_loan_batches = 500000
pagewidth = 72
pagelength = 62
```

loa_ite_fmt_loan_stat

The **loa_ite_fmt_loan_stat** script produces Loan Statistics for nominated Item Formats for each given Location within a specified date range. Statistics can be limited to type four types of statistics:

- **iss**: Issue statistics only
- **issren**: Issue and renew statistics
- **ren**: Renew statistics only
- **dis**: Discharge statistics only

The results are reported in a table, showing Sites in the columns and Item Formats in the rows. Summary totals are given by Item Format and Site.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_ite_fmt_loan_stat**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_ite_fmt_loan_stat -a<filename> -b<begin date> -d<database> -e<end date> -h -o<output filename> -s<data directory> -r<report directory) -t<type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This specifies the beginning of a date range for rows to be processed. Only LOAN rows created on or since the specified date will be processed. It is a mandatory argument . The format required is "DD/MM/YYYY". Note: The maximum range between -b and -e is one year.
-e	This optional argument specifies the end of a date range of rows to be processed. Only LOAN rows with a CREATE_DATE on or before the specified date will be processed. If not given, it will default to today's date. Note: The maximum range between -b and -e is one year.
-t	This mandatory argument specifies the type of statistics to be reported. Values must be either:

- "iss" Issue statistics only.
- "issren" Issue and renew statistics.
- "ren" Renew statistics only.
- "dis" Discharge statistics only.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
num_loan_batches	The number of loans to process could be many millions, so processing often has to be done in batches. This optional parameter determines the number of LOAN rows to be processed in each batch. It sets the maximum number of LOANS that can retrieved at any one time from the database. This is not related to the number of statistics that are included in the output. This value should not require any alteration.

locations	This optional parameter specifies the Sites to be included in the statistics. Each location will be a column heading under which the Loan Statistics are reported.
	
	Each argument to locations may be either an actual Site or a logical Site. An actual site will be found in the database. A logical Site is defined elsewhere in the parameter file as one or more actual Sites. When a logical Site appears as a column heading the data in the column is an accumulated total for the actual Sites it represents.
	
	For example, the parameter file shown above would report Loan Statistics on the actual Sites "CL" and "HW", in addition to statistics for the logical Site "SOUTH" as an accumulation of loans from actual Sites "S", "SE" and "SW":
	
	`locations=CL, SOUTH, HW`
	
	`SOUTH=S, SE, SW`
	
	Thus each location listed in the parameter file may point to more than one CREATE_LOCATION, providing another variable of the same name is entered in the parameter file with value(s) set to the various CREATE_LOCATIONs that comprise the logical Site.
	
	Note: The order of location headings is the same as the input into the parameter file and is hence user defined. If the "locations" parameter is omitted altogether, all Sites in the database are included (excluding logical Sites). A standard report generator error is given for any invalid parameter.

item_formats	The user can define which Item Formats are to be included or excluded in the report by using one of the variables in the parameter file, item_formats_in or item_formats_out. If neither item_formats_out nor item_formats_in is used, the default is to report on all Item Formats. These cannot both be used together.
	
	item_formats_in is an optional parameter which accepts existing Item Format Codes as arguments. When used, loan statistics are limited to loans whose associated Item Format belongs to the set given. Example:
	
	`item_formats_in=VID, MAP`
	
	item_formats_out is the complementary optional argument which accepts valid Item Format Codes as arguments. When used, loan statistics are limited to loans whose associated Item Formats are not specified by this parameter. Example:
	
	`item_formats_out =CD,3D`

Page dimensions	By using the variables pagewidth and pagelength specified in the parameter file, it is possible to determine, to an extent, how many locations headings appear on a line. The pagewidth parameter determines how wide the output format will be and effectively specifies how many Site columns are reported across each page. The pagelength parameter determines the length of the page and effectively specifies how many Item Formats are reported on each page.

loa_fine_rate_upd

The **loa_fine_rate_upd** script changes the Fine Rate for all current loans that have been recalled (i.e. those for which Recall Letters have been sent) so long as they are of a given Loan Type. Fine Rates are linked to the Loan Type, so the script changes the Fine Rate by changing the Loan Type.

Loans that have their Fine Rate changed are restricted to those where a recall letter has been sent, so this report should be run after the Recall Letters report in order for all recalled loans to be included in the run of this script.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_fine_rate_upd**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_fine_rate_upd -a<filename> -d<database> -h -o<output filename> -r<report directory> -s<data directory>

The Standard script arguments are described here.

Parameter file

The report uses a parameter file in which the Loan Types to be changed (**loantypes_old**) and the new Loan Types to which they will be changed (**loantypes_new**) are specified.

Loan Type Codes are the required for both Loan Type parameters. If multiple Loan Types are specified, they must be separated by a comma. At least one valid Loan Type Code needs to be specified for both the **loantypes_old** parameter and the **loantypes_old** parameter. If multiple Loan Types are specified, the script checks to ensure that the same number of Loan Types have been specified for each parameter. Multiple occurrences of the same Loan Type cannot be specified for the **loantypes_old** parameter, although they are permitted for the **loantypes_old** parameter.

The first Loan Type specified in the **loantypes_old** parameter is changed to the first Loan Type specified in the **loantypes_new** parameter. The second Loan Type specified in the **loantypes_old** parameter gets changed to the second Loan Type specified in the the **loantypes_new** parameter, and so on. If this script is run using the parameters illustrated in the example given below, Loan Type "STA" changes to "SHL" and Loan Type "OWL" changes to "SHL".

```
#
# The variables below specify the Loan Types to be changed and the
# type they will be changed to.
loantypes_old=STA,OWL,SHL
loantypes_new=SHL,SHL,REF
```

Note: These changes occur simultaneously. In the above example, Loan Type "OWL" is converted to "SHL" but is not then converted further to "REF".

loa_ite_loan_stats

The **loa_ite_loan_stats** script produces a count of issues for nominated Item Types grouped by their create location. The column headings are locations and the row headings are Item Types. The create location can be a logical grouping, giving the ability to report on a user defined Site Location Name that may encompass one or more Site IDs. These logical locations are user definable within the parameter file.

Output is in the form of a table and can produce results for a specified date range for one of four types of statistics:

- **iss**: Issue statistics only
- **issren**: Issue and renew statistics
- **ren**: Renew statistics only

- **dis**: Discharge statistics only

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_ite_loan_stats**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_ite_loan_stats -a<filename> -b<begin date> -d<database> -e<end date> -h -o<output filename> -r<report directory> -s<data directory> -t<statistics>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This specifies the beginning of a date range or rows to be processed. Only LOAN rows with a CREATE_DATE on or since the specified date will be processed. It is a mandatory argument . Note: The maximum range between -b and -e is one year.
-e	This optional argument specifies the end of a date range of rows to be processed. Only LOAN rows with a CREATE_DATE on or before the specified date will be processed. If not given, it defaults to today's date. Note: The maximum range between -b and -e is one year.
-t	This specifies the type of statistics to be reported. Values must be either: "iss", "issren", "ren", "dis". It is a mandatory argument .

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
locations	A list of locations can be entered in the parameter file. These will be the column headings under which the issue statistics are reported. Each location listed in the parameter file may point to more than one CREATE_LOCATION. If this is the case then another variable of the same name as the location will need to be entered by the user into the parameter file, with value(s) set to the various CREATE_LOCATIONs that comprise the logical location name. Otherwise, the location name is taken to be the CREATE_LOCATION to be reported on. In the parameter file example shown above, the report column headings will be "PL", "SOUTH", "WEST", "N" where: ▪ "PL" will be taken the mean the CREATE_LOCATION of "PL" ▪ "SOUTH" will be taken to mean CREATE_LOCATIONs of "W" & "GL" ▪ "WEST" will be taken to mean CREATE_LOCATIONs of "BSTL" & "SUSX" ▪ "N" will be taken to mean CREATE_LOCATION of "N". Note: The order of location headings is the same as the input into the parameter file and is hence user defined. The use of Location Sites (such as "SOUTH") is to report loan statistics for a combination of Sites, for example to cumulate the issue statistics at a junior and adult library, or for an area. If a location does not exist (in the database) it will be included in the column headings and within the Site Totals. It will always report counts of "0". It is effectively treated as a logical Site which is not assigned any true locations

item_types	The user can define which Item Types are to be included or excluded in the report by using one of the variables in the parameter file: ▪ ite_types_in ▪ ite_types_out These cannot be used together.
num_loan_batches	The number of loans to process could be many millions, so processing has to be done in batches. This optional parameter determines the number of LOAN rows to be processed in each batch. It sets the maximum number of LOANs that can be retrieved at any one time from the database. This is not related to the number of statistics that are included in the output. This value should not require any alteration.
Pagination	By using the variables pagewidth and pagelength specified in the parameter file, the user is able to determine, to an extent, how many locations headings appear on a line.

Example

```
#
# Parameter file for loa_ite_loan_stats
#
# Author: ahg/ogb
#
#

main = loa_ite_loan_stats/main.pl
preselect = loa_ite_loan_stats/preselect.pl
select = loa_ite_loan_stats/select.pl
format = loa_ite_loan_stats/format.pl
retrieve = loa_ite_loan_stats/retrieve.pl
locations = PL,SOUTH,WEST,N

SOUTH=W,GL
WEST=BSTL,SUSX
# ite_types_in =
# ite_types_out =
num_loan_batches = 500000
pagewidth = 72
pagelength = 62
```

loa_long_odue

The **loa_long_odue** script produces a listing of all Items that are long overdue. Long overdue may be defined by a date range and, optionally, the sending of a specific Overdue Letter. The output can be either a listing by Borrower or a listing by Location and Item. The latter has been designed to be suitable for a shelf-check list. Optionally the report can place standard messages against Borrowers and Items which appear in the output. It can also change the Item's status. The ability to make these updates supports Borrower blacklisting and the management of long overdue Items.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_long_odue**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_long_odue -a<appended filename> -b<minimum days> -d<database> -e<maximum days> -h -n<site,site> -o<output filename> -q<overdue letter number> -r<report directory> -s<data directory> -t<output format> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-b	This mandatory argument specifies the minimum number of days a loan must have been overdue for it to be considered long overdue and be selected. The value is a positive number, which must be less than or equal to 9999.
-e	This optional argument specifies the maximum number of days a loan must have been overdue for it to be still considered long overdue. The value is an integer which must be less than or equal to 9999. If not specified, this defaults to no maximum age (i.e. open-ended).
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".
-q	This optional argument specifies the minimum number of Overdue Letters that must have been sent already for the loan to be considered long overdue and selected. The argument must be an integer in the range 1-4.
-t	This mandatory argument names the type of output to be produced: the only valid options are -tborrower and -titem.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
loan_types_in	The parameter is optional. Use it to list the loan types to be included in the report. If no codes entered then all loan types will be included. `loan_types_in=<value>,<value>,<value>`
borrower_types_in	The parameter is optional. Use it to list the borrower types to be included in the report. If no codes entered then all borrower types will be included. `borrower_types_in=<value>,<value>,<value>`
item_types_in	The parameter is optional. Use it to list the item types to be included in the report. If no codes entered then all item types will be included. `item_types_in=<value>,<value>,<value>`
standard_borrower_message	Enter the ID of a borrower standard message that you wish to add to a borrower who has long overdues items identified by the script. If no message ID is entered then no message will be added.
standard_item_message	Enter the ID of a item standard message that you wish to add to an item that has long overdues items identified by the script. If no message ID is entered then no message will be added.
new_item_status	Enter a single item status code. Any items identified by the script as long overdue will be updated to the specified item status. If no value is entered the items remain at their current status.
pagebreak	The parameter is optional and is only meaningful when running with -tborrower. If it is set to "borrower" and -t is "borrower" then the output should pagebreak after each change of Borrower and after each change of location. This means that a double page

	break occurs after the last Borrower for a location; thus a blank page is printed. This allows a visual division of the output for those Libraries that initially wish to divide the print-out by location for sending sections out to each Site. Each Site will then receive a list pagebreaking by Borrower. If the parameter is omitted, pagebreaks occur after a change of location only. If the parameter is set to "borrower" and -titem is specified then the former parameter is ignored. `pagebreak_after=<value>`

pagination	Pagination is controlled by the printer using the "pagination = linefeed" technique. Standard A4 paper size is assumed.
	
	`pagination=linefeed`
	
	`pagelength=<number>`

Example

```
#
# Parameter module for loa_long_odue
#
#

main = loa_long_odue/main.pl
preselect = loa_long_odue/preselect.pl
select = loa_long_odue/select.pl
format = loa_long_odue/format.pl
retrieve = loa_long_odue/retrieve.pl

pagelength=66
pagination=linefeed

loan_types_in=ADFT
borrower_types_in=
item_types_in=CA1
pagebreak_after=
standard_borrower_message=
standard_item_message=
new_item_status=
```

loa_odue_charges

the **loa_odue_charges** script produces overdue notices for all Borrowers that have outstanding loans within a date range. The default format is A4 stationery. The script produces Overdue Notices for Items (based on the borrower type) either:

- Between x and y days overdue or
- Between x and y days since the last Overdue Letter.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_odue_charges**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_odue_charges -a<appended filename> -b<minimum days> -d<database> -e<max days> -h -n<locations> -o<output filename> -q<loan types> -r<report directory> -s<data directory> -t<overdue levels>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-b	Specifies the minimum number of days overdue (0-999). This argument is optional.
-e	Specifies the maximum number of days overdue (0-999). This argument should only be used with the '-b' argument .
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".
-q	Specifies the loan type(s) to be used in processing, separated by commas. If not given, defaults to all loan types. This argument is optional.
-t	Specifies which overdue notices are to be processed, separated by commas. If not given, the default is all overdue notices.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
processing_type	This mandatory parameter affects how -b and -e switches are applied to 2nd & subsequent overdues, and can be 'Overdue' or 'SinceLastLetter'.

output_by	The only two valid arguments are the strings "location" and "level" and they may appear in either order, but there is implied preference. Their use determines the use of multiple output files for the product of the overdue run: for example, if the overdue run is selecting Sites X and Y then, with:
	`output_by = location`
	set, two separate output files are produced, prefixed "X_" and "Y_", are produced
	Similarly, the use of the setting:
	`output_by = level`
	produces output files prefixed by "1st_", "2nd_", "3rd_" and "4th_".
	The use of one or other of these "order_by" options means that the equivalent argument specifying values must be used in the command line.
	The use of both "order_by" options involves creation of permutations of output files, for example, "X_1st_[output]" .. "X_2nd_[output]"

maximum_loans_printed	Specifies maximum number of loans to appear on a page. Can be used for datamailer stationary and with borrower address footers. If the number of items is greater than maximum_loans_printed, then a MAX_MESSAGE is displayed.

minimum and maximum days	To allow all four Overdue Letters to be run together, start and end dates for each Overdue Type are available in the parameter file. If the parameter file values are used then all eight have to be specified, regardless of the -t values nominated. i.e.
	`odue1_min=7`
	`odue1_max=99`
	`odue2_min=14`
	`odue2_max=99`

	odue3_min=21 odue3_max=99 odue4_min=30 odue4_max=99 If the –b and –e switches are specified in the command line then the parameter values are overridden by the command line values.
borrower_position	Positioning the Borrower's address in a fixed position at the footer of the Overdue Letter, regardless of the variable amount of bibliographic/overdue data which precedes it, is achieved by specifying the "borrower_position" parameter for each of the four letters. There are 4 parameters, one for each overdue level, with the number specifying the row number for output of the Borrower's address. For example: borrower position1=54 borrower_position2=54 borrower_position3=54 borrower_position4=54
borrower charges	Use charge_bor_types to list borrower types to charge. Use charge_amounts to list the amounts to charge (in the same order as borrower types). Use charge_reason to specify a borrower message id or a text to use for the charge reason.
pagination	The value linefeed will cause the script to output blank lines to fill up a page. The default is formfeed.
pagelength	If pagination is used, this specifies the length of the page. The default is 66.
sort_order	You can sort the letters within the output file(s) in the default order or post code order.

loa_odue_loclev

loa_odue_loclev is a variant of the standard loa_odue_letter (Overdue Letters) script, which produces overdue notices for all Borrowers that have outstanding loans within a date range. **loa_odue_loclev** introduces an **output_by** parameter to determine the use of multiple output files for the product of the overdue run.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_odue_loclev**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_odue_loclev -a<filename> -b<minimum days> -d<database> -e<maximum days> -h -n<locations> -o<output filename> -q<loan types> -r<report directory> -s<data directory> -t<overdue levels>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Specifies the minimum number of days overdue (0-999). This argument is optional.
-e	Specifies the maximum number of days overdue (0-999). This argument should only be used with the '-b' argument .
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for

	example, "BU, WV".
-q	Specifies the loan type(s) to be used in processing, separated by commas. If not given, defaults to all loan types. This argument is optional.
-t	Specifies which overdue notices are to be processed, separated by commas. If not given, the default is all overdue notices.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
processing_type	This mandatory parameter affects how -b and -e switches are applied to 2nd & subsequent overdues, and can be 'Overdue' or 'SinceLastLetter'.

output_by	The only two valid arguments are the strings "location" and "level" and they may appear in either order, but there is implied preference. Their use determines the use of multiple output files for the product of the overdue run: for example, if the overdue run is selecting Sites X and Y then, with:
	`output_by = location`
	set, two separate output files are produced, prefixed "X_" and "Y_", are produced
	Similarly, the use of the setting:
	`output_by = level`
	produces output files prefixed by "1st_", "2nd_", "3rd_" and "4th_".
	The use of one or other of these "order_by" options means that the equivalent argument specifying values must be used in the command line.
	The use of both "order_by" options involves creation of permutations of output files, for example, "X_1st_[output]" .. "X_2nd_[output]"

maximum_loans_printed	Specifies maximum number of loans to appear on a page. Can be used for datamailer stationary and with borrower address footers. If the number of items is greater than maximum_loans_printed, then a MAX_MESSAGE is displayed.

minimum and maximum days	To allow all four Overdue Letters to be run together, start and end dates for each Overdue Type are available in the parameter file. If the parameter file values are used then all eight have to be specified, regardless of the -t values nominated. i.e.
	`odue1 min=7`
	`odue1_max=99`
	`odue2_min= 14`
	`odue2_max= 99`
	`odue3 min= 21`
	`odue3_max= 99`
	`odue4_min= 30`
	`odue4_max=99`
	If the –b and –e switches are specified in the command line then the parameter values are overridden by the command line values.

borrower_position	Positioning the Borrower's address in a fixed position at the footer of the Overdue Letter, regardless of the variable amount of

	bibliographic/overdue data which precedes it, is achieved by specifying the "borrower_position" parameter for each of the four letters. There are 4 parameters, one for each overdue level, with the number specifying the row number for output of the Borrower's address. For example: borrower_position1=54 borrower_position2=54 borrower_position3=54 borrower_position4=54
pagination	The value linefeed will cause the script to output blank lines to fill up a page. The default is formfeed.
pagelength	If pagination is used, this specifies the length of the page. The default is 66.

loa_odue_letter

The **loa_odue_letter** script produces overdue notices for all Borrowers that have outstanding loans within a date range. The default format is A4 stationery. The script produces Overdue Notices for Items, either:

- Between x and y days overdue or
- Between x and y days since the last Overdue Letter .

The report updates the database to record that the relevant Overdue Letter has been sent to the Borrower for a specific Item.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/loa_odue_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -ploa_odue_letter -h -a<filename> -b<minimum days> -e<max days> -d<database> -n<locations> -s<data directory> -r<report directory> -q<loan type> -o<output filename> -t<letter type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Specifies the minimum number of days overdue (0-999). This argument is optional.
-e	Specifies the maximum number of days overdue (0-999). This argument should only be used with the '-b' argument.
-n	This optional argument specifies which locations are to be included in the statistics. If not specified this defaults to all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".
-q	This is an optional argument which specifies the Loan Types to be used in processing. Loan Types to be reported on should be separated by commas. If this argument is not used, the default is all Loan Types.
-t	This is an optional argument which specifies the Overdue Notices to be processed. If more than one is specified, these should be separated by commas. The possible values are: 1, 2, 3 and 4. If not specified, the default will be taken to mean all Overdue Notices: this has the advantage of avoiding the need for a separate run for each Overdue Letter under certain (parameter) conditions.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
processing_type	The type of processing to be performed may be set to one of two alternatives; i.e. either the number of days overdue (a date range) or the number of days since the last Overdue Letter was sent. The type of processing must be specified in the parameter file - either **overdue** or **sincelastletter**. When using the processing type of "sincelastletter", there is no previous letter, so the first Overdue is sent for loans between X and Y days overdue because a previous letter cannot have been sent
max_loans_printed	The datamailer page is of limited size (the precise number of loans which will fit on a single page varies from Library to Library). The idea of setting the "max loans printed" parameter in the "loa_odue_letter" parameter file is to specify the maximum number of overdues that can appear on one page. When this number is exceeded for a Borrower, a standard line of text will appear at the bottom of the letter. This setting only has relevance to Libraries using a datamailer. The value of the "max loans printed" parameter is something which each Library will need to determine locally by experimentation.
Pagination	Pagination can be achieved in two different ways. The default (i.e. no parameter is specified) is the insertion of a form feed character at the end of each letter. This effectively gives control of the page length to the printer, which will feed the paper based on the page length value to which it has been set. The alternative method is to set the parameter: `pagination=linefeed` This uses the pagination parameter defined in the parameter file and achieves a new page by outputting blank lines to fill a page (as defined by "pagelength="). This method is controlled by the MIS report software.

Notes

- A letter is produced for each Borrower's loans at a given Site. For example, if a Borrower has overdue loans at two Sites then a letter is produced for each Site. The report produces both a count of the number of letters sent and the total number of loans associated with those letters.
- Any loans for Borrower Types who have their Borrower Rule set to "over supp = Yes" will be ignored. This functionality is similar to that of the Report Writer reports which this script replaces. The Borrower Rule is related to the Site Profile attribute. In this case it is the Site Profile associated with the create location of each loan.
- The Classmark associated with the Item is used for display in the letter. If this has not been assigned for a given Item then the Classmark associated with the Work will be displayed.
- Running -t with more than one value to specify several Overdue Notices to be processed (for example, -t2,3,4) is only worthwhile if you process as "sincelastletter" and each overdue is sent the same number of days since the previous letter.

loa_odue_letter_ftr

loa_odue_letter_ftr is a variant of the standard loa_odue_letter (Overdue Letters) script. It contains a formatting block, with Borrower details, that can be begin on a fixed line number on each letter. It is practical to use this overdue report when the Borrower's name and address has to be printed at the foot of the page. Apart from the above feature it is functionally the same as loa_odue_letter.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameter is included:

Parameter	Description
processing_type	The type of processing to be performed may be set to one of two alternatives; i.e. either the number of days overdue (a date range) or the number of days since the last Overdue Letter was sent. The type of processing must be specified in the parameter file - either **overdue** or **sincelastletter**. When using the processing type of "sincelastletter", there is no previous letter, so the first Overdue is sent for loans between X and Y days overdue because a previous letter cannot have been sent
max_loans_printed	The datamailer page is of limited size (the precise number of loans which will fit on a single page varies from Library to Library). The idea of setting the "max loans printed" parameter in the "loa_odue_letter" parameter file is to specify the maximum number of overdues that can appear on one page. When this number is exceeded for a Borrower, a standard line of text will appear at the bottom of the letter. This setting only has relevance to Libraries using a datamailer. The value of the "max loans printed" parameter is something which each Library will need to determine locally by experimentation.
Pagination	Pagination can be achieved in two different ways. The default (i.e. no parameter is specified) is the insertion of a form feed character at the end of each letter. This effectively gives control of the page length to the printer, which will feed the paper based on the page length value to which it has been set. The alternative method is to set the parameter: `pagination=linefeed` This uses the pagination parameter defined in the parameter file and achieves a new page by outputting blank lines to fill a page (as defined by "pagelength="). This method is controlled by the MIS report software.
borrower_position	Positioning the Borrower's address in a fixed position at the footer of the Overdue Letter, regardless of the variable amount of bibliographic/overdue data which precedes it, .is achieved by specifying the "borrower_position" parameter for each of the four letters. There are 4 parameters, one for each overdue level. The parameter file contains four "borrower_positionN=[value]" parameters, where "N" is an integer between 1 and 4, and "x" is a number specifying the row number for output of the Borrower's address. For example: ▪ borrower_position1=54 ▪ borrower_position2=54 ▪ borrower_position3=54 ▪ borrower_position4=54 The presence of 4 parameters allows the Borrower details to be printed at different places on different Overdue Letter types. Each of the four "borrower_position" parameters is optional (but non-use implies that "loa_odue_letter" could have been run instead). Note: If specified, the number value must be less than the pagelength parameter.

Notes

- The template file for the output (**format.pl**) has a formatting block called BOR_ADDRESS. This block is output after the footer of the letter. If the line number of the current letter prior to printing this block is less than the **borrower_position** variable then an appropriate number of

blank lines are output to force BOR_ADDRESS to appear on the specified line number. If the current line number is equal to or exceeds the nominated value, then BOR_ADDRESS is output immediately (i.e. no blank lines are inserted).

- Note: The above logic means consideration should be given to using the parameter **maximum_loans_printed**. Without this parameter a large number of Overdue loans on the letter will force BOR_ADDRESS to print lower than expected.

oor_claim_letter

The **oor_claim_letter** script can be used to generate the First, second, third and cancellation Claims Letters may be generated, optionally limited by Claim Number, Delivery Site and/or Supplier Code. The Claim Number ("1", "2", "3" or "C") determines the letter format to be produced at intervals as defined in parameters set in Online Utilities Parameters Rules Acquisitions Rules Claim Sequences; i.e. 1st, 2nd, 3rd Claims Letters and a Cancellation Letter.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/oor_claim_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -poor_claim_letter -a<appended filename> -d<database> -h -n<delivery site> -o<output filename> -q<supplier code> -r<report directory> -s<report directory> -t<claim number>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies which delivery sites are to be included in the report. If not specified this defaults to all Sites. The format required is the Location Code, separated by commas if more than one Site is specified; for example, "-nBU,WV". Sites entered must be valid Site Codes within the database being used (case insensitive).
-q	This specifies the parameter file "oor_claim_letter". This argument is mandatory.
-t	An optional switch which specifies the letter formats to be produced. Valid options are 1,2,3 and C. If multiple claim numbers are specified, they must be separated by a comma. If omitted, it will default to all claim numbers.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

Parameter	Description
loan_types_in	**loan_types_in** will only select current loans which have any of the Loan Type codes that appear after the "=" sign.
loan_types_out	loan_types_out is similar to loan_types_in but selects all loans with Loan Type codes other than those specified. The loan_type parameters are optional, but if used only one can be specified.
locations_in	Specifies the location codes to be included in the report. Note that locations_in and locations_out are mutually exclusive.
locations_ out	Specifies the location codes to be excluded from the report. Note that locations_in and locations_out are mutually exclusive.

Notes

- Messages to be output in each letter are held in "format.pl". The messages are as follows:
- First Claims Letter Message
  "The publication listed below has not been received by us. Please supply it as soon as possible or send a report."
- Second Claims Letter Message
  "The publication listed below has not been received by us in spite of a previous claim. Please supply it as soon as possible or send a report."
- Third Claims Letter Message
  "The publication listed below has not been received by us in spite of two previous claims. Please supply it urgently or send a report."
- Cancellation Letter message
  "The publication below is no longer required by us. Please cancel this issue."

oor_order_letter

The **oor_order_letter** script produces Purchase Orders for Open Orders. It can process:

- Individual Open Order(s) (of any status)
- Outstanding Open Orders from a start date (optional) and allow Supplier Codes to be included.

The Open Orders are printed in Order Number sequence if Open Order Numbers are specified. If no arguments, or dates or Supplier Code(s) are specified, the sequence is ordered by Supplier Code / Order Number.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/oor_order_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -poor_order_letter -a<appended filename> -b<start date> -d<database> -h -n<open order numbers> -o<output filename> -q<supplier code> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument may be used to specify the start date of the period from which Open Orders will be printed. The required format is "dd/mm/yy" or "dd/mm/yyyy". The end date is the current date. If not specified, all Open Orders will be processed regardless of date. This argument cannot be used with the "-n" argument .
-n	This optional argument specifies the Open Order Number(s) to be processed. These will be separated by commas, for example "order_number,order_number, order_number". The Open Order Numbers must be valid numbers within the database being used. This argument cannot be used with the "-b" and "-q" switches.
-q	This optional argument specifies the Suppliers to be included in the processing. The format is Supplier Code. If multiple Supplier Codes are specified, they must be separated by commas; for example "-qWHICH,BLAPER". If omitted, this defaults to all Suppliers. This argument cannot be used with the "-n" argument .
-t	This optional argument specifies the type of processing and therefore the type of letter format which will be produced. If this argument is not specified, the value of the "processing_type" parameter in the parameter file will be used. The valid options are "3PART", "8X4" and "MULTI" (i.e. multiple 8X4). For single Open Order processing using the "-n" argument , the only valid

	processing types are "8X4" and "3PART".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
processing_type	The valid options for the "processing_type" parameter are "3PART", "8X4" and "MULTI" (i.e. multiple 8X4).
pagination	This parameter dictates that the pagelength should be achieved by the insertion of linefeeds (i.e. blank lines). Omitting this parameter defaults to achieving pagination using a form feed character at the end of each letter.
pagelength	The pagelength parameter is not used by this report. The pagelength is determined by the processing type instead: **Processing type** **Page Length** MULTI 24 3PART 66 8X4 24

orr_can_ords_list

The **orr_can_ords_list** report lists Orders where a cancellation letter has been sent (i.e. where a third Chaser Letter has been sent), and the Order Status is neither "Received" nor "Cancelled".

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/orr_can_ords_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -porr_can_ords_list -a<appended filename> -d<database> -h -n<supplier code> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the Suppliers to be included in the processing. If multiple Supplier Codes are specified, they must be separated by commas; for example "-nWHICH,BLAPER". If omitted, this defaults to all Suppliers. Supplier Codes entered must be valid within the database being used (case insensitive).

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are needed.

Notes

- The output is sorted by Supplier Code and then by Order Number within Supplier.
- The output is produced on continuous stationery.

orr_imp

The **orr_imp** script provides the importing Library with management information on Potential and Proposed Orders imported from Suppliers by the orr_import utility. It can be run on a regular basis to list Orders imported using orr_import.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/orr_imp**. To run in the script log on as **report** or **talis** and enter the following command:

report -porr_imp -a -b<begin date> -d<database> -e<end date> -h -k -n<status> -o -q<supplier code> -r -s

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-e	This optional argument specifies the end date for report selection (selecting on the Order Create Date). The format required is "dd/mm/yy".
-k	This optional argument validates the parameter file.
-n	This optional argument specifies the Status of the Orders to be reported. This is specified as the Status Code, i.e. "POT" or "PROP". "POT" reports Potential Orders. "PROP" reports Proposed Orders. If not given both "POT" and "PROP" are reported.
-q	This optional argument specifies the Supplier Code. Several Suppliers may be specified, separated with a comma, for example "-qJMLS,ASK,PETERS".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

Notes

- Output is sorted by Supplier Code, then by Order Status and then Order Number. The following data is output for each Order:

  Order Number
  Order Date
  Supplier Reference
  Format
  Price
  Bibliographic Information (Control Number, Author, Title, Volume Number, Volume Title, Series, Edition, Publisher, Publication Date and Classification)
  Copy Information (Number of Copies, Locations, Item Shelfmark and Funds).

orr_chaser

The **orr_chaser** script produces 1st, 2nd, and 3rd Claims Letters for any outstanding Orders. The 3rd Claims Letter is a cancellation of the Order.

By default, a Claims Letter is sent 28 days after either the Order or the previous Claims Letter has been sent. It is possible to change this interval in the parameter file. The number of days elapsed before sending each Claims Letter is specified in the parameter file via the "chaser1", "chaser2", and "chaser3" parameters.

Claiming can be prevented if a report has been received from the Supplier within the number of days specified via the **exclude_report_days** parameter.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/orr_chaser**. To run in the script log on as **report** or **talis** and enter the following command:

report -porr_chaser -a<filename> -d<database> -h<help> -n<order number> -o<output filename> -r<report directory> -s<data directory> -t<letter type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the Order Number(s) to be processed. If multiple Order Numbers are specified, these must be separated by a comma, for example "-nQA12345678,QA23456789". This argument cannot be used on the same command line as the "-t" argument . The "chaser" and the "exclude_report_days" parameters in the parameter file are ignored when the "-n" argument is used.
-t	This optional argument specifies the Claim Letter types to be output. Valid options are 1, 2 and 3. If multiple letter types are specified, they must be separated with a comma. If "-t" is not specified, the default is taken as producing all Letter Types (i.e. 1, 2 and 3). The "-t" argument cannot be used with the "-n" argument .

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
exclude_report_days	The "exclude_report_days" parameter prevents a Claim Letter being sent if a report has been received from the Supplier within the number of days specified. It must be set between 0 and 999 days. The default value is 0 days. This parameter is ignored when the "-n" argument has been specified.
chaser 1, **chaser 2**, **chaser 3**	The default setting for each chaser parameter is 28 days. If any or all of the chaser parameters are not specified in the parameter file, they will automatically default to 28 days. Any or all of the values for "chaser1", "chaser2" or "chaser3" parameters can be changed but must be set to an integer value between 0 and 999. For example: **chaser1 = 28** would send first chaser report 28 days after the Order has been sent, if no items have been received. c**haser2 = 28** would send the second chaser 28 days after the first chaser has been sent, if no items have been received. **chaser3 = 28** would send the third chaser 28 days after the second chaser has been sent, if no items have been received.

pagination	`pagelength =`
	`pagination =`
	
	These above parameters allow pagination to be achieved in one of two ways.
	
	The default, when neither parameter is specified or pagination = formfeed, is the insertion of a form feed character at the end of each letter. This gives control of the page length to the printer. This method is good for "standard" pagelengths such as A4.
	
	The alternative method is pagination=linefeed. This achieves a new page by outputting blank lines to fill a page to the length specified in "pagelength=xx".. .

Notes

- The Library address appearing on the Chaser Letter is derived in one of two ways.

i. The address can be inserted into the "format.pl" to be printed as seen. The default "format.pl" has included the Address as "LocationAddressLine1", "LocationAddressLine2" and so on; these variables can be amended to that of the preferred Library address.

ii. The second approach allows each Chaser Letter to have a different Library address. The address is derived from the LOCATION_ADDRESS table, based on the create location of the Order. This will be the default method of producing the Address on the Chaser Letters.

- If one central delivery address is required, but the Orders are placed from different Sites, then the location address variables in the "format.pl" should be replaced by the actual name and address of the preferred Library address.

orr_ite_returns

orr_ite_returns produces a list of Items which have been returned to the Supplier and for which no replacement(s) have been provided. It can only be used for normal (i.e. not open) Orders. A returned Item is defined by specific standard Item messages. The listing can be used either as a "chaser letter" to the Supplier or as an internal listing for Library use.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/orr_ite_returns**. To run in the script log on as **report** or **talis** and enter the following command:

report -p orr_ite_returns -a<append filename> -b<start date> -d<database> -e<end date> -h<help> -n<supplier,supplier> -o<output filename> -q<format,format> -r<report directory> -s<data directory> -t<output>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the minimum number days since an Item was returned to the Supplier. The argument is an integer and defaults to zero days. Values between 0 and 9999 are valid.
-e	This optional argument specifies the maximum number days since an Item was returned to the Supplier. The argument is an integer and defaults to no maximum age if not specified. Values between 0 and 9999 are valid.
-n	This optional argument specifies which Suppliers to select, thereby limiting the Items to be reported by nominated Supplier(s). It defaults to all Suppliers. Valid Supplier Codes must be used. If multiple codes are specified, these should be separated by commas.
-q	This optional argument specifies which Item Formats are to be selected. It defaults to all Item Formats. The format required is one or more Item Format Codes. If multiple codes are specified, these should be separated by commas; for example, "-qCD,AV,MAP".
-t	This optional argument determines the type of output. Two values are allowed: either "chaser" or "internal". If the argument is omitted, the default is "internal". Output for "chaser" includes the Supplier's name and address and the list pagebreaks and sorts on Supplier. Output for "internal" does not include the Supplier address and is sorted by Author/Title."

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
supplier_return_message_id	This parameter takes the form:

	supplier_return_message_id=<value>,<value> where <value> is an integer. This is mandatory, as Items are selected if they have an active message for one or more of the MESSAGE_IDs given. The -b and -e date arguments are compared to the date the messages were set. Library staff may wish to nominate several alternative messages as goods returned. When an Item has more than one of the specified standard messages active the Item is output once only, (but each assigned message associated with the Item will be included in the output). The Message ids associated with standard Item Messages can be found in the Online Utilities parameter interface for messages (i.e. under Parameters Names Messages Item). Alternatively the following SQL (run within dwb or isql) will produce a list of standard item messages with their associated Ids: select MESSAGE_ID, convert(char(60),TEXT) from MESSAGE where MESSAGE_ID >=1 and TYPE=1 and STANDARD='T'

orr_order_letter

The **orr_order_letter** script produces Purchase Orders. It can process:

- Individual Order(s) (of any status)
- Outstanding Orders from a start date (which is optional) and allow individual Supplier Codes to be included .

The output will be determined from the processing_type parameter in the parameter file. It will be set to A4 as a default, but can be overridden by a command line argument to allow output of other formats to be produced, for example 3PART, 8X4 and LOCN.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/orr_order_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -porr_order_letter -a<appended filename> -b<start date> -d<database> -h -n<order number(s)> -o<ouput filename> -q<supplier codes> -r<report directory> -s<data directory> -t<processing_type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Specifies the start date of the period from which outstanding Orders will be printed. The format required is "dd/mm/yy". This argument is optional. Note: This argument cannot be used with -n argument.
-n	This specifies the Order Number(s), input by the user, to be processed. These will be separated by commas, for example "order_number, order_number, order_number". This argument is optional. This argument cannot be used with the -b argument.
-q	Specifies the Supplier Code(s), input by the user, to be processed. These will be separated by commas, for example "supplier_code, supplier_code, supplier_code". This argument is optional. Note: This argument cannot be used with the -n argument.

-t	This specifies the processing type for this Order run. The type of output is determined by this argument. The valid options are "A4", "3PART", "8X4" and "LOCN". The default, if not specified, is "A4". For single Order processing via the -n argument, the only valid processing types are "3PART" and "8X4". This argument is optional. -t takes precedence over any processing type parameter specified in the parameter file. If -t is omitted here then the parameter value in the parameter file is used.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
pagination=linefeed	This parameter dictates that the pagelength should be achieved by the insertion of linefeeds (i.e. blank lines). Omitting this parameter defaults to achieving pagination using a form feed character at the end of each letter
processing_type=	lThis parameter dictates which output will be produced from the run of the script. The processing_type can be set to one of "A4", "3PART", "8X4" or "LOCN". Each one produces a different output. This parameter can be overridden at the command line with the -t argument, for example -tLOCN.
pagelength=	The pagelength parameter is not used by this report. The pagelength is determined by the processing type instead: A4 66 LOCN 24 3PART 66 8X4 24

Notes

- The Library Address appearing on the Order Letter is derived in one of two ways depending on which format (or -t parameter) is used.
- "A4" and "LOCN" (-tA4 and -tLOCN) have an Address which must be inserted into "format.pl" to be printed as seen. The default "format.pl" has included the Address as "Library name" and "Library address line 1". This should be amended to that of the real sending location.
- The processing types -t3part and -t8x4 derive the Library name and address from the LOCATION_ADDRESS table based on the create location of the Order.
- The later approach means each letter can have a different delivery address. If one central delivery address is required, and the Orders are placed from different sites, then the location address variables in "format.pl" should be replaced by the actual name and address of the sending Library site.
- Note: Attempting the converse approach (i.e. replacing the "hard-coded" Library address with the variables that pick up the create location from each order for formats "A4" and "LOCN") is not recommended,. This is not recommended because the formats "A4" and "LOCN" print out multiple Orders per Supplier Letter; the Library Address would be from the create location of the first Order on the Letter.
- Unlike some other Letters, there is no pagelength parameter in the parameter file. The page length is set dynamically, based on the -t values used.

pay_inv_list

The **pay_inv_list** script lists Invoiced Orders, with Invoice details. All Orders are listed on one, several or all Invoice Numbers. The report indicates the Number of Copies, the Price and Item details. It has been updated to reflect credit notes, and the Payment Type will be displayed. This script is useful to identify inconsistencies when the online View Invoice Prompt is unable to display a particular Invoice Number.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/fun_orders_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -ppay_inv_list -a<appended filename> -d<database> -h -n<invoice number> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the Invoice Number(s) separated by a comma. If no numbers are given, the default will be all. Invoice Numbers - if given - will be validated.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included:

pay_sup_charges

The **pay_sup_charges** script generates an analysis of service charges and discounts by Supplier. This may be for one, several or all Suppliers, over a specified date range. The payment details can be for all Financial Years, or a specific Financial Year.

The report totals the following values, for each Supplier:

- Item Net Cost
- Discount
- VAT (on Item Net Cost less Discount)
- Service Charge
- Service VAT
- Total Item Cost

These totals are also broken down into percentage and average values.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/pay_sup_charges**. To run in the script log on as **report** or **talis** and enter the following command:

report -ppay_sup_charges -a<filename> -b<start date> -d<database> -e<end date> -h -n<supplier code(s)> -o<output filename> -q<financial year(s)> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This argument specifies the start date of the date range within which payments are selected. The format required is "DD/MM/YY". This argument is mandatory.
-e	This argument specifies the end date of the date range within which payments are selected. The format required is "DD/MM/YY". This argument is mandatory.
-n	This optional argument specifies the Supplier Code(s), separated by a comma if more than one. If no Supplier Codes are given, the default will be all Suppliers. Supplier Codes - if given - will be validated.

-q	This is an optional argument, used for specifying the Financial Year(s) to be included, separated by a comma if there is more than one. If no Financial Year is specified, the default is taken to mean all Financial Years.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. Pagination is set as follows:

```
Pagelength=66
```

rec_loa_letter

The **rec_loa_letter** script produces Recall Notices for Item's that have already been recalled by Alto. It updates the database to record the sending of the relevant Recall Letter.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/rec_loa_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -prec_loa_letter -a<appended filename> -b<minimum days> -d<database> -e<max days> -h -n<locations(s)> -o<output filename> -q<loan type(s)> -r<report directory> -s<data directory> -t<letter type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This specifies the minimum number of days since the previous Recall Letter was sent. This argument is optional. It overrides the values rec2_min, rec3_min and rec4_min if the latter are set in the parameter file.
-e	This specifies the maximum number of days since the previous Recall Letter was sent. This argument is optional. It overrides the values rec2_max, rec3_max and rec4_max if the latter are set in the parameter file.
-n	This specifies the Site(s) to be used in processing. If not give will default to all sites. This argument is optional.
-q	This specifies the Loan Type(s) to be used in processing. If not given, it defaults to all Loan Types. This argument is optional.
-t	This specifies the Recall Notice(s) to be processed. If not given, this defaults to all Recall Notices. This argument is optional. Allowed values are 1, 2, 3 and 4.

The switches -b and -e only work as "days since last letter" and override any values set by rec2_min, rec2_max, rec3_min, rec3_max, rec4_min and rec4_max. This means -b and -e do not apply to first recalls (-t1) as the number of days since the create date of the loan is only controlled by rec1_days (the age of loans before they have a first Recall Letter generated).

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
borrower_types_in,borrower_types_out	Only the Borrower codes following the "=" sign for borrower_types_in will be have Recall Letters sent. Recall Letters will be sent for all Borrower Types except those codes following the "=" for borrower_types_out. Note that these fields are

	mutually exclusive. If a value given is not a known Borrower Type code then the report will stop running and give an error.
item_types_in, item_types_out	Item_types_in and Item_types_out behave in a similar manner to Borrower Types (Borrower_types_in and Borrower_types_out above) except that the codes are known Item Types. Note that these fields are mutually exclusive.
rec1_days	This parameter specifies the age of a loan (i.e. the number of days old) before it has a first Recall Letter generated. For example, rec1_days=10 means loans that are 10 days or older will have a first Recall Letter generated. Note: This parameter is the only way of specifying the date of the first Recall Letter. -b and -e are not used for first recalls. Omitting rec1_days will result in all loans, irrespective of their dates of creation, being included in the pre-select stage of processing.
rec2_min, rec2_max	rec2_min is the minimum number of days that must have elapsed since the first Recall Letter was sent. Similarly rec2_max is the maximum number of days since the first Recall Letter was sent. For example, if today is 25th December 1996 and rec2_min=5 and rec2_max=24 then only loans that had a first recall letter between the 1st and 20th December will be candidates for a Recall Letter. If -b and/or -e are specified on the command line, these values take precedence over rec2_min and rec2_max.
rec3_min and rec3_max	These work as described for rec2_min and rec2_max, but the arguments refer to the days since the second Recall Letter was sent.
rec4_min and rec4_max	These work as described for rec2_min and rec2_max, except that the arguments refer to the days since the third Recall Letter was sent.
pagination = linefeed	This parameter dictates that the pagelength should be achieved by the insertion of linefeeds (i.e. blank lines). Omitting this parameter defaults to achieving pagination using a form feed character at the end of each letter.

Notes

- The numeric arguments following all the recall days in the parameter file (for example rec1_days, will cause an error if they have one or more trailing spaces.

rec_loa_letter_dmail

The **rec_loa_letter_dmail** script produces Recall Notices for Items that have already been recalled by Alto. It updates the database to record the sending of the relevant Recall Letter.

The **rec_loa_letter_dmail** MIS Perl script is a variant of the standard Recall Letters script rec_loa_letter. **rec_loa_letter_dmail** has been designed for use with data-mailer stationery which has the Borrower Address printed on a fixed line on every page. It contains a formatting block, with Borrower details, that begins on a fixed line number on each letter thereby ensuring the Borrower Address is printed correctly.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/rec_loa_letter_dmail**. To run in the script log on as **report** or **talis** and enter the following command:

report -p rec_loa_letter_dmail -a<filename> -b<min days> -d<database> -e<max days> -h -n<locations(s)> -o<output filename> -q<loan type(s)> -p<filename> -r<report directory> -s<data directory> -t<letter type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This specifies the minimum number of days since the previous Recall Letter was sent. This argument is optional. It overrides the values rec2_min, rec3_min and rec4_min if the latter are set in the parameter file.
-e	This specifies the maximum number of days since the previous Recall Letter was sent. This argument is optional. It overrides the values rec2_max, rec3_max and rec4_max if the latter are set in the parameter file.
-n	This specifies the Site(s) to be used in processing. If not give will default to all sites. This argument is optional.
-q	This specifies the Loan Type(s) to be used in processing. If not given, it defaults to all Loan Types. This argument is optional.
-t	This specifies the Recall Notice(s) to be processed. If not given, this defaults to all Recall Notices. This argument is optional. Allowed values are 1, 2, 3 and 4.

The switches **-b** and **-e** only work as "days since last letter" and override any values set by rec2_min, rec2_max, rec3_min, rec3_max, rec4_min and rec4_max. This means **-b** and **-e** do not apply to first recalls (-t1) as the number of days since the create date of the loan is only controlled by rec1_days (the age of loans before they have a first Recall Letter generated).

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
borrower_position	The parameter file contains a "borrower_position=[value]" parameter, where "x" is a number specifying the row number for output of the Borrower's address. For example: `borrower_position1=54` Note: If specified, the number value must be less than the pagelength parameter.
borrower_types_in, borrower_types_out	Only the Borrower codes following the "=" sign for borrower_types_in will be have Recall Letters sent. Recall Letters will be sent for all Borrower Types except those codes following the "=" for borrower_types_out. Note that these fields are mutually exclusive. If a value given is not a known Borrower Type code then the report will stop running and give an error.
item_types_in, item_types_out	Item_types_in and Item_types_out behave in a similar manner to Borrower Types (Borrower_types_in and Borrower_types_out above) except that the codes are known Item Types. Note that these fields are mutually exclusive.
rec1_days	This parameter specifies the age of a loan (i.e. the number of days old) before it has a first Recall Letter generated. For example, rec1_days=10 means loans that are 10 days or older will have a first Recall Letter

	generated. Note: This parameter is the only way of specifying the date of the first Recall Letter. -b and -e are not used for first recalls. Omitting rec1_days will result in all loans, irrespective of their dates of creation, being included in the pre-select stage of processing.
rec2_min, rec2_max	rec2_min is the minimum number of days that must have elapsed since the first Recall Letter was sent. Similarly rec2_max is the maximum number of days since the first Recall Letter was sent. For example, if today is 25th December 1996 and rec2_min=5 and rec2_max=24 then only loans that had a first recall letter between the 1st and 20th December will be candidates for a Recall Letter. If -b and/or -e are specified on the command line, these values take precedence over rec2_min and rec2_max.
rec3_min and rec3_max	These work as described for rec2_min and rec2_max, but the arguments refer to the days since the second Recall Letter was sent.
rec4_min and rec4_max	These work as described for rec2_min and rec2_max, except that the arguments refer to the days since the third Recall Letter was sent.
pagination = linefeed	This parameter dictates that the pagelength should be achieved by the insertion of linefeeds (i.e. blank lines). Omitting this parameter defaults to achieving pagination using a form feed character at the end of each letter.
max_loans_printed	The datamailer page is of limited size (the precise number of loans which will fit on a single page varies from Library to Library). The purpose of setting the "max loans printed" parameter in the "rec_loa_letter_dmail" parameter file is to specify the maximum number of loans that can appear on one page. When this number is exceeded for a Borrower, a standard line of text will appear at the bottom of the letter, indicating: `"You may have other loans not shown"`

Notes

- The numeric arguments following all the recall days in the parameter file (for example rec1_days, will cause an error if they have one or more trailing spaces.

rec_long_soon_letter

The **rec_long_soon_letter** script produces first Recall Notices for reserved Items. Recalls are generated for the Items either:

- On loan the longest, or
- Due for return the soonest.

The Recall Notices to be generated are determined by an argument in the command line. Recall Notices are not generated for Items which have had the recall flag set against them in Alto.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/rec_long_soon_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -prec_long_soon_letter -a<filename> -d<database name> -h<help> -n<location(s)> -o<output filename> -q<loan type(s)> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-n	This optional argument specifies the Locations(s) to be included in processing. The format required is the Location Code, for example "ML, ED". If multiple codes are specified, they must be separated by a comma. If not given it defaults to all Sites.
-q	This optional argument specifies the Loan Type(s) to be included in processing. The format required is the Loan Type Code, for example "TERM,OWL". If multiple codes are specified, they must be separated by a comma. If not given, it defaults to all Loan Types.
-t	This mandatory argument specifies the type of processing. Only the two values, "SOON" and "LONG" are valid. "SOON" selects Items due soonest to be recalled. "LONG" selects those Items on loan longest to be recalled.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
max_days_overdue	This parameter is used to specify the maximum number of days an Item must be overdue to be recalled. It is only used in conjunction with "-tLONG" on the command line. If no value is entered, the default value is 60.
min_days_onloan	This parameter is used to specify the minimum number of days an Item must be on loan before it can be recalled. It is only used in conjunction with "-tLONG" on the command line. If no value is entered, the default value is 8.
due_days	This parameter is used to specify the number of days hence that the Due Date of the loan should be set. It is only used with "-tSOON" on the command line. If no value is entered, the default value is 7.
due_time	This parameter is used to specify the time that the Item will be due for return. It must be set using the 24 hour clock format. Four digits must be entered, for example 09:30 not 9:30. It is only used with "-tSOON" on the command line. If no value is entered, the default value is 23:59. If the "-t" argument is set to "SOON", then the "max_days_overdue" and "min_days_onloan" parameters are ignored. Likewise, if the "-t" argument is set to "LONG", the "due_days" and "due_time" parameters are ignored.
borrower_types_in	This parameter is used to specify which Borrower Types may have Recall Letters sent. Multiple Borrower Types (case insensitive) may be entered, separated by a comma. The format required is Borrower Code(s). If no Borrower Types are specified, the default is to include all Borrower Types.
borrower_types_out	This parameter is used to specify which Borrower Types should not have Recall Letters sent. Multiple Borrower Types (case insensitive) may be entered, separated by a comma. The format required is Borrower Code(s). Note: The "borrower_types_in" and "borrower_types_out" parameters are mutually exclusive and cannot both be set. If neither is set, the default is to include all Borrower Types.
item_types_in	This parameter is used to specify which Item Types may have Recall Letters sent. Multiple Item Types (case insensitive) may be entered, separated by a comma. The format required is Item Type Code(s). If no Item Types are specified, the default is to include all Item Types.

item_types_out	This parameter is used to specify which Item Types should not have Recall Letters sent. Multiple Item Types (case insensitive) may be entered, separated by a comma. The format required is Item Type Code(s). Note: The "item_types_in" and "item_types_out" parameters are mutually exclusive and cannot both be set. If neither are set, the default is to include all Item Types.

Notes

- The numeric arguments following all the recall days in the parameter file (for example rec1_days, will cause an error if they have one or more trailing spaces.

res_dem_od_list

The **res_dem_od_list** script produces lists of Reserved Works, either:

- A list of Works for which there are more active reservations than loanable copies, or
- A list of overdue Items with active reservations.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_dem_od_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_dem_od_list -a<appended filename> -d<database> -h -n<days overdue> -o<output filename> -q<site> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the number of days an Item needs to be overdue before it is included in the processing. If omitted, it defaults to 60 days. The value specified must be a number between 1 and 999. Note: This switch cannot be used if the "–t" switch is set to "DEM".
-q	This optional argument specifies the Sites to be included in the processing. The format required is one or more Site Code(s). If multiple Site Codes are specified, they must be separated by a comma; for example "BU, CL". If omitted, it defaults to all Sites. This argument cannot be used if the "–t" argument is set to "DEM".
-t	This optional argument specifies the type of processing, and hence the type of report to be produced. Valid options are "OD" or "DEM". Only one processing type may be specified. The "OD" option produces a list of overdue loans with reservations against them. The "DEM" option lists Works for which reservations exceed the number of copies available. If omitted, it defaults to "DEM".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No extra parameters are included.

res_itm_noloan

The **res_itm_noloan** script produces a list of Items which are reserved but not on loan. The list enables a shelfcheck to be carried out. The list has three possible sort orders, to cater for different Libraries requirements.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_itm_noloan**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_itm_noloan -a<appended filename> -d<database> -h -n<site,site> -o<output filename> -r<report directory> -s<data directory> -t<sort order>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies which locations are to be included in the report, i.e. which Site Codes are to have a listing of reserved Items. The location refers to the Home Site of the Item; it determines which Sites will appear on the list.. The format required is the location codes separated by commas; for example "BU, WV". If omitted, this will default to all Sites."
-t	This optional argument specifies which sort order to use. ▪ -tITEMSEQ sorts by location, item types, sequence, classmark and then author. ▪ -tCLASS sorts by location, classmark, author and then barcode. ▪ -tITEMCLASS sorts by location, item type, classmark, author and then title. ▪ -tSEQ sorts by location, sequence, classmark, author, title and then item type. If this argument is omitted then the sort order -tCLASS is used.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
in_transit	The in_transit parameter can be set to "YES" or "NO". If "YES" then any reservations which are in intransit are included in the listing. If "NO" then these reservations are excluded. If the parameter is omitted, it defaults to "NO".
old_reservations	The old_reservations parameter can be an integer between 1 and 9999. Any reservations with a create date older than the number of specified days should have a text message appended to the reservation details. By default the message should be worded: " *** This reservation is more than X days old ***" where X is the integer value in the parameter file. If this parameter is omitted then no message is displayed.

res_outstanding

The **res_outstanding** script produces a listing of Works which have associated reservations with an Active or In transit status. The listing for each location is divided into two distinct parts:

- The main listing: a list of each Site's current reservations.
- An optional secondary listing, showing Works reserved by other Sites, where the reserving Site(s) have no Items associated with the reservation but the current Site's copies are included in the reservation.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_outstanding**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_outstanding -a<appended filename> -b<start date> -d<database> -e<end date> -h -n<site,site> -o<output filename> -q<format,format> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument selects outstanding reservations by a specified minimum number days since the reservations were created. The argument is an integer and defaults to zero days.
-e	This optional argument selects outstanding reservations by a specified maximum number of days since the reservations were created. The argument is an integer and defaults to no maximum.
-n	This optional argument is used to specify which reserving Sites are to be included in the output. If not given, the default is all Sites. The format required is the location code, separated by commas if more than one Site is specified; for example, "BU, WV".
-q	This optional argument may be used to specify the selection of reservations where one or more Items have the Item Formats Codes given. If omitted, it defaults to all Item Formats. If one or more Items have one of the Format Codes specified then that reservation is selected. Multiple Format Codes must be separated by commas; for example, "-qCD, AV, MAP".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
borrower_types_in	This optional parameter specifies valid Borrower Type codes. Only reservations associated with Borrowers of the nominated types are selected.
item_types_in	This optional parameter specifies valid Item Type codes. Only reservations associated with Items of the nominated types are selected.
other_res_sites_in	This parameter defaults to "Yes". If set to "No" then the secondary listing, which shows reservations at other Sites, is not included in the output.

Notes

- This list - entitled "Reservations at Other Sites" - shows how different Sites reserve each other's Stock. For each "record" in the list is a Work and it is followed by useful information to make decisions on progressing the reservations. For example, are there enough Items to be able to satisfy the reservations listed? Where is my Site in the Reservation Queue? The output pagebreaks after each Site, so the printout can be sent out to individual reserving Sites, allowing local decisions on reservation management.
- This list - entitled "Reservations at Other Sites" - shows how different Sites reserve each other's Stock. For example, in the listing for Site A, it will show all the Works and reservations where other Sites have reserved Site A's Items. A reservation is only included if the other Site(s) have none of their own Items reserved (i.e. these Sites have no Items of their own so they are attempting to utilise other Sites' copies).The secondary listing for a given Site, follows immediately after the main listing. This allows a combined list of main and secondary lists to be delivered together (the list pagebreaks after each secondary listing).

- Running with -n[sites] will only produce a secondary listing for the Sites nominated. For example, -nA, B will only put Site B's reservation in A's secondary listing. So -n and with single argument will fail to produce a secondary listing. In practice, the secondary listing is best run for all Sites (omitting -n).

res_query_letter

The **res_query_letter** script is used to produce notifications that inform borrowers that the item they have reserved is missing (that is, the item that has been reserved is subject to an item query).

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_query_letter**. To run the script log on as **report** or **talis** and enter the following command:

report -pres_query_letter -a<filename> -b<begin_date> -d<database_name> -e<end_date> -h -n<LAST> -o<filename> -r<directory> -s<directory> -u

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	Specifies the beginning of a date range of item query rows to be processed. Only item queries created on or since the specified date will be processed.
-e	Specifies the end of a date range of item query rows to be processed. Only item queries created on or before the specified date will be processed. If no value is used then the date will default to the date the script runs.
-n	Specifies if the script is to run from the last query processed. The only value allowed is -nLAST. The -n switch cannot be used with the -b and -e switches.
-u	Specifies that the script is to update the selected reservations to Cancelled status. A message will be added to each letter to inform the user of the reservation status change.

Parameter file

The parameter file **res_query_letter** is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
ITEM_TYPE_IN= **ITEM_TYPE_OUT=**	Allows you to specify item types to be included or excluded. It is not possible to run the script with both parameters set. If no values are set then all item types are included. Enter item type codes separated by commas.
LOCATION_IN= **LOCATION_OUT=**	Allows you to specify reservation collection sites to be included or excluded. It is not possible to run the script with both parameters set. If no values are set then all reservation collection sites are included. Enter reservation collection site codes separated by commas.
BORROWER_TYPE_IN= **BORROWER_TYPE_OUT=**	Allows you to specify borrower types to be included or excluded. It is not possible to run the script with both parameters set. If no values are set then all borrower types are included. Enter borrower type codes separated by commas.

borrower_posistion=	If you wish to display the borrower address details in the footer of the output then this parameter allows a user to state which line the address will start to be displayed from. By default the **format.pl** file contains the field holders and variables for displaying the borrower details in both the header and the footer. The rows need to be deleted from either the header or footer as required.

As well as producing output in a letter file (by default **res_query_letter.out**), the script also produces a report file that contains a shelf checking list sorted by item home site.

Example report file

```
res_query_letter_loca      Reservation query letters           19/05/08 15:21:09

                           ~~~~~~~~~~~~~~~~~~~~~~~~

Command line: res_query_letter_local -pres_query_letter_local -b01/01/08

Processing records on database 'prod_talis'
Running for dates between 01/01/08 00:00 and 19/05/08 11:59PM

Pagelength: 66
Pagination: linefeed
Item statuses: QUER
Reservations will NOT be cancelled

Total reservations processed  : 5
Total letters output          : 5


Item Home Site: Central
==============================

Bib ID:         371842
Control Number: 0786402571
Item Barcode:   05046114
Bib Details:    Rhodes, Gary Don, 1972-. - Lugosi : his life in films, on stage,
Shelfmark:      791.43028092LUG,RHO
Borrower:       22222222

Item Home Site: Main
==============================

Bib ID:         176885
Control Number: 0904383113
Item Barcode:   5602424185
Bib Details:    Greenwood, Victoria. - In demand / (by) Victoria Greenw
Shelfmark:      343.62(41),GRE
Borrower:       97099198

Bib ID:         83721
Control Number: 68019538
Item Barcode:   5600480711
Bib Details:    Simonian, Charles. - Fencing : fundamentals. [CD ROM]
Shelfmark:      796.86,SIM
Borrower:       12300136

Item Home Site: Site A
==============================

Bib ID:         328076
Control Number: y8743290
Item Barcode:   0380769X
Bib Details:    Goldsworthy, Andy. - Andy Goldsworthy : mountain and coast, Autu
Shelfmark:      709.2GOL,GOL
Borrower:       11111111

Item Home Site: West
==============================
Bib ID:         74300
Control Number: y0122335
Item Barcode:   5600625876
Bib Details:    Singer, Charles, 1876-1960. - From magic to science : essays on
```

```
Shelfmark:      0,
Borrower:       12300136

res_query_letter_loca              Completed                  19/05/08 15:21:10

                                   ~~~~~~~~~
```

res_shelf_upd

The **res_shelf_upd** script checks for Items on loan to a Reservation Shelf which have become overdue. This facility "pushes" any overdue Items on the Reservation Shelf onto their next logical status, thereby saving staff time. It reports any actions taken on behalf of staff. Items on the Reservation Shelf which are overdue may be discharged automatically and then assigned to the next Borrower who reserved the book and/or placed in transit, or returned to the shelves if no longer required.

- If the script finds an overdue Item, it discharges it and updates the reservation to which it relates.
- The script checks if there is another active reservation for the Item. If there is not, it optionally places the Item in transit if it belongs to a Site which is not associated with the Reservation Shelf being processed.
- If there is another active reservation, the report optionally satisfies the reservation or places the Item in transit to satisfy it depending on the Reservation Collection Site; otherwise it simply reports that the Item is reserved. If there is more than one reservation the first in the queue will be updated.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_shelf_upd**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_shelf_upd -a<appended filename> -d<database> -h -n<site,site> -o<output filename> -r<report directory> -s<data directory> -t<sort order>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the location(s) of the Reservation Shelf (or shelves) to be processed. One or more Site Profile Code(s) (case insensitive) may be input, separated by a comma. If omitted, all Reservation Shelves i.e. shelves at all Sites are processed.
-t	This optional argument specifies the order in which the output should be sorted. An option which suits Reservation Shelf practices at your Library should be selected. Four values are valid: ▪ -tAUTH: This sorts by Author, Title and Reserver's name. ▪ -tBORR: This sorts by Reserver's name, Author and Title. ▪ -tCLASS: This sorts by Classmark and Filing Suffix, Author, Title and Reserver's name. ▪ -tDATE: This sorts by Item Due Date (oldest first), Reserver's name, Author and Title. If the "-t" argument is not specified, it defaults to "-tBORR".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
intransit_home	Can be set to "YES" or "NO" (case insensitive). If set to "YES", any Item being processed belonging to a Site different to the Site Profile of the Reservation Shelf being processed and not required to satisfy another reservation is placed in transit for return to its Home Site. If set to "NO", such an Item will not be placed in transit. If this parameter is omitted, the default is "NO".
intransit_res	Can be set to "YES" or "NO" (case insensitive). If set to "YES", any Item being processed required for a reservation whose Collection Site is different to the Site Profile of the Reservation Shelf being processed is placed in transit to satisfy the reservation. If set to "NO", such an Item will not be placed in transit. If this parameter is omitted, the default is "NO".
intransit_unres	Can be set to YES" or "NO" (case insensitive). If set to "YES", then if the Item being processed is placed in transit to satisfy a reservation (i.e. intransit_res=YES) a check is made for other Items attached to the reservation. Any Items found whose Home Site differs from the Collection Site of the reservation are unreserved. The in transit Item, plus any Item whose Home Site matches the Collection Site remain attached to the reservation. If set to "NO", this check is not made. If this parameter is omitted, the default is "NO".
satisfy_res	Can be set to "YES" or "NO" (case insensitive). If set to "YES", then any Item being processed required for a reservation at the Site of the Reservation Shelf being processed is used to satisfy the reservation, and is issued to the Reservation Shelf. If set to "NO", the reservation is not satisfied. If this parameter is omitted, the default is "NO".

res_wait_list

The **res_wait_list** script lists brief bibliographic details of all reserved items which have been waiting collection for longer than a given period. The output is ordered by classmark and may be produced for one, several or all Library Sites.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_wait_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_wait_list -a<appended filename> -b<days> -d<database> -h -n<site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies the number of days to be used in processing. Only reservations which have been waiting for collection longer than the specified number of days are reported. A number between 1 and 999 must be specified.
-n	This optional argument specifies the Collection Sites to be included in the report. The format required is Site Code(s). If multiple Site Codes are specified, they must be separated by a comma (for example "BU, WV"). If omitted, this defaults to all Sites.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

res_waiting_letter

The **res_waiting_letter** script produces reservation notices to Borrowers to inform them that their reserved Items are available for collection. It updates the database to record the fact that the letter has been sent out (so this script should not be run from the MIS server). The script generates a letter for each new reservation waiting collection. "New" is defined as reserved Items which have had no previous letter sent. The output can be limited by Collection Site of the reservation. Reservations past their last useful date do not produce letters.

Usage

The script's source code is located in **usr/opt/blcmp/talis/mis/res_waiting_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_waiting_letter -a<appended filename> -d<database> -h -n<site,site> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This specifies which Collection Sites are to be included in the output. The format required is the Location Code; if more than one is specified the codes should be separated by commas, example "BU,WV". The Site Location Codes are compared with the Collection Site of the reservation. If not give this default to all Sites. This argument is optional.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
pagelength	Pagination can be achieved in two different ways. The default (i.e. when no "pagelength" parameter is specified) is the insertion of a form feed character at the end of each letter. This effectively gives control of the page length to the printer, which will feed the paper based on the page length value to which it has been set. The alternative method is to set the parameter: `pagelength=linefeed`

Notes

- The letters are sorted by Collection Site and then by Borrower ID. Effectively, when a Borrower has several Items awaiting collection all the notices will appear in the same letter.
- Each Borrower's letter starts on a new page. A change of Collection Site also starts a new page.

res_work_list

The **res_work_list** script lists reserved Works, where the number of reservations equals or is greater than a number specified by the user. The report shows the total number of reservations placed between a given date range for each Work within a Classmark range. script produced a list for all Works where the number of reservations exceeded a given number. The **res_work_list** script outputs a list for all Works where the number of reservations is equal to or greater than the number specified on the command line.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/res_work_list**. To run in the script log on as **report** or **talis** and enter the following command:

report -pres_work_list -a<appended filename> -b<start date> -d<database> -e<end date> -h -n<reservations> -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This mandatory argument specifies the start of the date range within which reservations are to be processed. The format must be "DD/MM/YY" or "DD/MM/YYYY". Only reservations created on or after this date are processed.
-e	This mandatory argument specifies the end of date range within which reservations are to be processed. Only reservations created on or before this date are processed. The required format is "DD/MM/YY" or "DD/MM/YYYY".

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
start_classmark	This is a mandatory parameter which specifies the start of the Classmark range from which reservations are to be processed. Only reservations with the Classmark specified or above are processed.
end_classmark=	This is a mandatory parameter which specifies the end of the Classmark range within which reservations are to be processed. Only reservations with the Classmark specified or below are processed. Note: If the values specified in the "start_classmark" and the "end_classmark" parameters are equal, then processing is restricted to just the one Classmark specified.

Notes

- Output is on continuous stationery without pagebreaks (i.e. under the control of the printer).
- The output is sorted by Author within Classification Number.

rlt_works_list

The **rlt_works_list** script produces lists of reading lists. Each report lists the Works associated with the Course, the Owner or the Reading List Code respectively, as specified by the selection criterion and processing type given on the command line.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/SK_loa_school_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -prlt_works_list -a<appended filename> -d<database> -h -n<code> -o<output filename> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument specifies the selection criteria with which to limit processing. The type of data specified here is dependent upon what is specified in the "–t" argument. ▪ If "–t" is "COURSE" then Course Codes are specified. ▪ If "–t" is "OWNER" then Borrower barcodes are specified.

	▪ If "–t" is "CODE" then Reading List Codes are specified. If multiple barcodes or codes are specified, they must be separated by a comma. If "-n" is omitted then all Reading Lists are selected by default.
-t	This mandatory argument specifies the type of processing, and hence the type of report to be produced. Valid options are "COURSE", "OWNER" or "CODE". Only one processing type may be specified, either using upper or lower case. ▪ If "–t" is set to "COURSE", "-n" specifies Course Code(s), i.e. Reading Lists associated with particular Course(s) are listed. ▪ If "–t" is set to "OWNER", "-n" specifies one or more Owner's Borrower barcode(s), i.e. Reading Lists owned by specific Borrower(s) are listed. ▪ If "–t" is set to "CODE", "-n" specifies Reading List Codes, i.e. Works in Reading List(s) having the specified code(s) are listed.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. No additional parameters are included.

Example

```
 # Parameter file for SK_loa_school_letter
#
# Author: axk
#
#
#
main = SK_loa_school_letter/main.pl
preselect = SK_loa_school_letter/preselect.pl
select = SK_loa_school_letter/select.pl
format = SK_loa_school_letter/format.pl
retrieve = SK_loa_school_letter/retrieve.pl
pagelength = 60
initialpage = no
loan_types_in =  ADFT
# loan_types_out =
locations_in =   CL,ML
# locations_out =
```

Notes

- Both **loan_types_out** and **locations_out** can cause excessive run times. You are advised to specify inclusive Loan Type and Location parameters using **loan_types_in** and **locations_in**.

SK_loa_school_letter

SK_loa_school_letter produces letters which list all of the Borrowers' current loans. As the definition of a "school letter" is defined by either Loan Types or Locations, this script could be useful for any Library wishing to print letters detailing all current loans. The output uses form feeds to achieve pagination. This limits the output to paper sizes acceptable to the printer. In practice the script is designed to print on A4 paper.

Usage

The script's source code is located in **/usr/opt/blcmp/talis/mis/SK_loa_school_letter**. To run in the script log on as **report** or **talis** and enter the following command:

report -pSK_loa_school_letter -a<filename> -b<start date> -d<database> -h -e<end date> -n<site> -o<output filename> -p<filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description

-b	This mandatory argument specifies start date from which to include charges. The format "DD/MM/YY" must be used.
-e	This mandatory argument specifies the end date of the date range from which to include charges. The format "DD/MM/YY" is required. The start and end dates should not be greater than 1 year apart.
-n	This optional argument indicates that processing should commence with the next LOAN_ID greater than the last processed on the previous run.

Parameter file

The parameter file is held in the **/usr/opt/blcmp/talis/mis/param** directory. The following additional parameters are included:

Parameter	Description
loan_types_in	loan_types_in will only select current loans which have any of the Loan Type codes that appear after the "=" sign. Multiple loan types need to be separated by commas. Note that loan_types_in and loan_types_out are mutually exclusive.
loan_types_out	loan_types_out is similar to loan_types_in but selects all loans with Loan Type codes other than those specified. The loan_type parameters are optional, but if used only one can be specified. Multiple loan types need to be separated by commas. Note that loan_types_in and loan_types_out are mutually exclusive.
locations_in	Specifies the location codes to be included in the report. Note that locations_in and locations_out are mutually exclusive.
locations_ out	Specifies the location codes to be excluded from the report. Note that locations_in and locations_out are mutually exclusive.

Example

```
# Parameter file for SK_loa_school_letter
#
# Author: axk
#
#
#
main = SK_loa_school_letter/main.pl
preselect = SK_loa_school_letter/preselect.pl
select = SK_loa_school_letter/select.pl
format = SK_loa_school_letter/format.pl
retrieve = SK_loa_school_letter/retrieve.pl
pagelength = 60
initialpage = no
loan_types_in =  ADFT
# loan_types_out =
locations_in =   CL,ML
# locations_out =
```

Notes

- Both loan_types_out and locations_out can cause excessive run times. You are advised to specify inclusive Loan Type and Location parameters using loan_types_in and locations_in.

soc_claim_letter

The **soc_claim_letter** report produces letters for standing order items identified for claiming.

Usage

Log on as **talis** (or the report user) and enter the following command:

report -psoc_claim_letter -a<appended filename> -d<database> -h -n<order number> -o<output filename> -r<report directory> -s<data directory> -t<claim type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	This optional argument allows you to print claims for specific order numbers. You can separate multiple claim numbers with a comma. If this argument is not specified then all order numbers will be printed.
-t	This optional argument determines the letter format to be produced. Valid claim numbers are 1,2,3 and C (as set in your claim parameters rules). If this argument is not specified the default is all claims letters.

Parameter file

The **soc_claim_letter** script must be run with a parameter file. The report will only print claims that match the selection criteria defined by the parameters. Accepted parameters are defined in the following table.

Parameter	Description
DELIVERY_SITE	This parameter accepts delivery site codes. Only standing orders for the specified delivery site(s) will be included in the report. You can separate multiple site codes with a comma (for example, DELIVERY_SITE=LOC1,LOC2). If this parameter is not set, standing orders for all site codes will be included.
SUPPLIER_CODE	This parameter accepts supplier codes. Only standing orders from the specified supplier(s) will be included in the report. You can separate multiple supplier codes with a comma (for example, SUPPLIER_CODES=BBC,BFI). If this parameter is not set, standing orders from all suppliers will be included.
CLAIM2	The CLAIM2, CLAIM3 and CANCEL parameters determine the duration between claim letters. By default these are set to 28 days. For example, if you specify CLAIM2=30, then a second claim would be generated 30 days after the first claim was generated.
CLAIM3	This parameter determines the duration between the second and third claim letters. For example, if you specify CLAIM3=30, then a third claim would be generated 30 days after the second claim was generated.
CANCEL	This parameter determines the duration between the third claim and cancellation letters. For example, if you specify CANCEL=30, then a cancellation would be generated 30 days after the third claim was generated.

Notes

- The **soc_claim_letter** derives the library address from the delivery site for the order.
- Claims for the same delivery site, order, open order number and item control number produce one letter (that is, if an order has multiple subscriptions with multiple delivery sites, then a separate delivery letter is produced for each site).
- If multiple subscriptions on the same order exist with the same delivery site, one letter will be produced.
- The claim date is entered automatically into the standing orders receipt form when a claims letter is produced and is displayed in the claim date field of the check-in form.

soc_no_receipt

The **soc_no_receipt** report produces a list of standing order potential claims (that is, standing orders for which no items have been receipted within a given number of days). Libraries need to place a

manual claim from the standing order check-in form if a claim is required for any of the orders reported.

Usage

Log on as **ops** and enter the following command:

report -psoc_no_receipt -a<appended filename> -d<database> -h -o<output filename> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-n	Specifies the rotation plan that the selected items will be assigned to. This argument is compulsory and will only accept a single rotation plan code.

Parameter file

The **soc_no_receipt** script must be run with a parameter file. The script will only select standing orders that match the selection criteria defined by the parameters. Accepted parameters are defined in the following table.

Parameter	Description
DELIVERY_SITE	This parameter determines the number of days since an item was last received for a standing order. This parameter is mandatory; you must supply an integer between 0-999 (for example, DAYS_SINCE_RECEIPT=30).
SUPPLIER_CODE	This parameter accepts supplier codes, Only standing orders from the specified supplier(s) will be processed. You can separate multiple supplier codes with a comma (for example, SUPPLIER_CODES=BBC,BFI). If this parameter is not set, standing orders from all suppliers processed.
DAYS_SINCE_RECEIPT	This parameter determines the number of days since an item was last received for a standing order. This parameter is mandatory; you must supply an integer between 0-999 (for example, DAYS_SINCE_RECEIPT=30).

SelfServ

SIP2 scripts

For information on the following scripts, refer to the Talis SelfServ documentation located at the Capita Documentation web pages.

- kill_sip2_server
- start_sip2_server
- stop_sip2_server

ILL Manager

ill_caretaker

The **ill_caretaker** script checks to see if the current user has read write permissions on all directories and the mail file. The script should be run prior to running the ILL Manager Utility.

For more information, refer to the ILL Manager Utility documentation.

ill_mail

The **ill_mail** utility runs under control of the UNIX cron and checks for replies from BLDSC on a scheduled basis. This utility monitors a temporary mailbox for replies from BLDSC (and/or any other sources sending "trash" mail) on a scheduled basis.

For more information, refer to the ILL Manager Utility documentation.

ill_manager

The **ill_manager** script executes the ILL Manager Utility, which is used for sending and receipt of Interloan requests to/from the BLDSC via E-Mail using the BLDSC's ARTEmail© facility.

For more information, refer to the ILL Manager Utility documentation.

ill_send

Type **ill_send** script runs under control of the UNIX "cron" and transmits all pending requests to the BLDSC on a scheduled basis.

For more information, refer to the ILL Manager Utility documentation.

ill_stub

Type **ill_stub** script runs under control of the UNIX "cron" and transmits all pending requests to the BLDSC on a scheduled basis.

For more information, refer to the ILL Manager Utility documentation.

Import work

item_imp

The **item_imp** script (as called by the import script) adds to or updates the relevant tables in Alto with Item data. Where an update occurs, Item related information is updated, with the exception of the Item's Home Site and Loan Type.

Usage

Log on as **ops** and enter the following commands:

item_imp -z -ni -g -b<row set> -e<row set> <libcode> <database> <data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument allows a beginning row set to be nominated. It may be used if large files are being processed, or if there is a need to re-start either work_imp or item_imp.
-e	This optional argument allows an end row set to be nominated. It may be used if large files are being processed, or if there is a need to re-start either work_imp or item_imp.
-g	This optional argument disables the import of Local bibliographic data, where the weekly tapes contain both General and Local records.
-ni	This optional argument adds Items which are new to the database, but does not update existing Items.
libcode	This mandatory argument is required to check the incoming data is for the correct Library. The Libcode should be expressed in upper case and must follow any optional arguments.
data directory	The data directory to be used (for example **/scratch**)

build

The **build** script (as called by the import script) reads the input file "xxdata" and converts the content of each record into a set of rows. However, it can also be ran discreetly.

Usage

Log on as **ops** and enter the following commands:

build -fCURL <**libcode> <database> <data directory>**

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-fCURL	This optional -fCURL argument allows the process to accept CURL format records.
libcode	This mandatory argument is required to check the incoming data is for the correct Library. The Libcode should be expressed in upper case and must follow any optional arguments.
data directory	The data directory to be used (for example **/scratch**)

convert

The **convert** script (as called by the import script) re-formats the rows as necessary and outputs sets of Work and Item rows for each record. However, it can also be ran discreetly.

Usage

Log on as **ops** and enter the following commands:

convert -a<libcode> -g -c <libcode> <database> <data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-a	This optional argument allows a new Library to run with an alias where its Item-related conditions match those which have been coded for another Library. For more advice, contact Capita Support.
-g	This optional argument disables the import of Local bibliographic data, where the weekly tapes contain both General and Local records.
-c	This optional argument suppresses the update of WORK.CLASS_DISPLAY when other Work-related local data is imported.
libcode	This mandatory argument is required to check the incoming data is for the correct Library. The Libcode should be expressed in upper case and must follow any optional arguments.
data directory	The data directory to be used (for example **/scratch**)

ite_wrk_update

The **ite_wrk_update** script adds additional data to Items in the database with data imported from Book Suppliers. An Order must already exist for the Item, along with a minimum of Item-related data as follows:

- ITEM_ID

- TYPE_ID
- WORK_ID
- ACTIVE_SITE_ID

The Book Supplier sends enhanced Item data (Order fulfilments) to the Capita Gateway, where data from all participating Suppliers is collated into a single file and forwarded to each Library regularly. The script is run to update selected Item attributes in the LMS database; it also adds a Class Display value to a Work if this does not already exist and inserts a row into the WORK_UPDATE table for OPAC **access_points** processing.

The Library can control which of the following Item attributes are not to be updated by **ite_wrk_update** by entering the following values into the IMPORT_PARAMETER table by using the imp_modify utility.

The script may be run either interactively or from the cron.

Usage

Log on as **ops** and enter the following command:

ite_wrk_update -h -b<row set> -e<row set> <libcode> <database>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This is an optional argument to commence processing at a given item_row_set number. If this is not given, the script will commence processing from the first "item_row_set" in the file.
-e	This is an optional argument to finish processing at a given "item_row_set" number. If not given, the script will process to the end of the file.
libcode	This mandatory argument is required to check the incoming data is for the correct Library. The Libcode should be expressed in upper case and must follow any optional arguments.
-w	This mandatory argument must be included when importing cataloguing service files. It populates the SEARCH_WORKS table.

Notes

- Where there is more than one **itm_csv_imp.trans** file awaiting processing, the files are not concatenated. They are, instead, selected for processing one by one in ascending date order; only one **itm_csv_imp.trans** file is processed by each run of "ite_wrk_update".
- The input filename to **ite_wrk_update** does not have a ".in" suffix. The file **itm_csv_imp.r_trans[datetime]** is renamed to xxdata_itm,: i.e. to the standard input file expected by "item_imp".
- Two reports, **ite_trans_rep** and **ite_wrk_upd_rep** are generated by the **ite_wrk_update** script. If the recommended ZZ_import script (where "ZZ" is your libcode) is used, the report will be in **/usr/ops/blcmp/data/impdir/report**.

import_MARC21

The **import_MARC21** script allows MARC21 records to be converted and imported, adding URL data and creating an item for each record. It is intended specifically to import e-book monograph records, and does not allow the import of e-serial records.

For each record in the input file, the software performs a conversion from MARC21 to Talis MARC, imports the record into the local database, adding HOTLINK data from the 856 field, and creates an item for each work. The item data is created using values specified in a parameter file called import_marc21.param.

- Before running the script, you should ensure that there is a file named lib_code in the data directory, which contains your two-letter library code. If you have ever run work_imp before, you will probably already have such a file. If not, you should create one.
- A file called tag_mapping is shipped with the import_marc21 software. You should not need to edit this file, but it is important that it exists, and is not deleted. If you wish to use a directory other than /scratch as your data directory, you will need to copy this file into your new data directory.

Usage

Log on as **talis** and enter the following command:

import_MARC21 –i<input file> -tEBOOK -b<begin record number> -e<end record number> -m<maximum records to process> -r<report directory> -s<data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-t	This argument specifies the type of processing to be used. Currently –tEBOOK must be used for importing e-book records. The script should not be used for importing any other type of records. When –tEBOOK is used, an item will be created for each MARC record, and data from the 856 field will be written to the HOTLINK table.
-b	This optional argument specifies the record number at which to start processing. If not used, processing will begin with the first record in the file.
-e	This optional argument specifies the record number at which to end processing. If not used, processing will continue until all records have been processed. It may not be used if the –m argument is used.
-m	This optional argument specifies the maximum number of records to process. It may not be used if the –e argument is used.

Parameter file

Before running the software, you must edit the import_marc21.param file in the data directory. The default parameter file is loaded to **/scratch** (the default data directory). If you wish to use a different directory as the data directory you will need to copy the parameter file to that directory. This parameter file allows you to specify values for the items that will be created, Hotlinks, and record source.

CLASSMARK_SOURCE= ITEM_TYPE= SITE= #SIZE= #SEQUENCE= #CLASSMARK= #SUFFIX= #HOTLINK_TEXT=#HOTLINK_IMAGE= #RECORD_SOURCE=

Accepted parameters are described in the

Parameter	Description
CLASSMARK_SOURCE	This is a mandatory parameter. It may contain one of four values: NONE – no classmark will be associated with the items PARAMETER – the value specified in the CLASSMARK parameter will be assigned to each item 050 – the classmark given in field 050 of the MARC record (if present) will be assigned to the item created for that record 082 - the classmark given in field 082 of the MARC record (if present) will be assigned to the item created for that record
ITEM_TYPE	This is a mandatory parameter. It should contain an existing item type

	code, which will be assigned to the items that are created.
SITE	This is a mandatory parameter. It should contain an existing site code, which will be assigned to the items that are created.
SIZE	This is an optional parameter. If used, it should contain an existing size code, which will be assigned to the items that are created.
SEQUENCE	This is an optional parameter. If used, it should contain an existing sequence code, which will be assigned to the items that are created.
CLASSMARK	This is an optional parameter. If used, it should contain a classmark, which will be assigned to the items that are created.
SUFFIX	This is an optional parameter. If used, it should contain a suffix, which will be assigned to the items that are created.
HOTLINK_TEXT	This is an optional parameter. If used, it should contain the words you wish to appear in OPAC for the hotlink. For example, 'Click here to access the e-book'. If the parameter is not used, the contents of 856 subfield z (if present) will be used.
HOTLINK_IMAGE	This is an optional parameter. If used, it should contain the filename for the image file you wish to appear in OPAC for the hotlink.
RECORD_SOURCE	This is an optional parameter. If used, it should contain the source of the e-book records. e.g. 'Taylor & Francis' or 'NetLibrary'. This information will be used in cases where the incoming MARC record has a system control number which is not valid in Talis. That number will be copied to the 035 field, and, if there is no 003 field in the record, the source given in this parameter will be used to precede the control number. For example, if the parameter gives 'Taylor & Francis', the record has '2397123' in 001, and there is no 003 field to show the source of the control number, the imported record will contain an 035 field with '*a(Taylor & Francis) 2397123'. A Talis local monograph control number will be assigned to the 001 field.

Notes

- As part of the conversion process, the control number in 001 may be changed. If the existing 001 is not recognised as a valid control number for Talis, an ISBN from elsewhere in the record may be used instead. If there are multiple ISBNs, one where 'ebook', 'electronic bk' or 'electronic book' is given will be preferred. If there is no ISBN, a local Talis monograph control number will be assigned to the record.
- Although a work would not be added if there was already a work with the same control number, if the script is run more than once on the same input record, more than one item would be created for the same work. Therefore care should be taken to ensure that each input file, or section of records from that file, is only processed once.

import_oclc

import_oclc is a utility which enables bibliographic and item data from OCLC to be imported into Alto. OCLC supply tapes of records to Talis. Talis then FTP each file to the Library, where it is processed by "import_oclc". This utility:

- Performs several conversion routines to produce records in the required format for each stage of the process (for example UKMARC, WORK_SUBFIELD).
- Changes the Control Number of each record to that which is most appropriate for Talis. If an ISBN, BNB or LC Number (in order of preference) is not found in a Monograph record, or an ISSN in a Serial record, a Local Control Number is allocated.

- Splits each record into General Work, Local Work and Item information. (A parameter file called "tag_mapping" is used to identify fields containing Local bibliographic and Item information. This file should not normally require editing. It may be edited using "vi" if necessary, following advice from Talis Support).
- Uses Library-specific software to manipulate the Item information.
- Trims any records which are too large and reports the data dropped.
- Generates work_imp and item_imp input files.

Usage

Log on as **ops** and enter the following command:

import_oclc -b<begin record> -e<end record> -h -i<input filename> -m<maximum to process> -r<report directory> -s<data directory> -t<type of processing>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-b	This optional argument specifies the number of the record in the input file with which to begin processing. For example, if –b100 is specified, processing begins at the 100th record in the file. If this argument is not given, processing begins with the first record in the input file..
-e	This optional argument specifies the number of the record in the input file at which to end processing. For example, if –e999 is specified, processing ends at the 999th record in the file. If this argument is not given, the script attempts to process to the end of the input file.
-m	This optional argument specifies the number of records to process from the record at which the processing begins. This argument is not compatible with use of the "–e" argument .
-t	This optional argument names the report directory where the process will create the report file. When not given, the report will be written to the directory specified by the $TAL_REP_DIR environment variable. This variable will default to the $TALIS_HOME/reports directory if not already set. The report file takes the name "oclc_import_rep.[datetime]".

IMPORT_PARAMETER

In generating "item_imp" data the script checks whether the Item Type, Site and Sequence values in an incoming record match the VALUE_1 attributes in the IMPORT_PARAMETER table. If a match is found, the corresponding VALUE_2 in the IMPORT_PARAMETER row will be used in the "item_imp" data. If a match is not found, the value in the incoming record will be used and a message will be output to the report file.

Before running the script you should check that the required values are in the IMPORT_PARAMETER table to avoid large numbers of messages being generated. You will need to use ISQL or DWB to do this (please contact Talis Support for assistance if necessary). If you need to add extra rows you should use the imp_modify script.

Notes

- A report file called "oclc_import_rep.[datetime]" is produced in the report directory. This contains reports of large records which have been trimmed and messages generated in processing Item data. This report also contains the reports generated by the runs of "work_imp" and "item_imp".
- If the script terminates abnormally, the report gives an indication of the reason. In addition a file called "oclc_import_log.[datetime]" will contain the number within the file of the last record successfully processed.

svol_imp

svol_imp is a utility which reads a file of pseudo-Talis rows each of which include attributes relating to a Serial Volume. Attributes are used to insert new rows into the Talis SERIAL_VOLUME and WORK tables.

Usage

Log on as **ops** and enter the following command:

svol_imp -d<database> -i<source file> -b<row no1> -e<row no2>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This is a mandatory argument, specifying the full pathname of the file containing the input rows.
-b	This is an optional argument, used to specify the first row in the input file from which point onwards the data will be processed. The default is the first row.
-e	This is an optional argument, used for specifying the last row in the input file to be processed. The default is the last row.

Notes

- The "svol_imp" utility writes errors and counts to "talis_rep.[date.time]". This report includes any errors encountered and input row counts.

work_imp

The **work_imp** script (as called by the import script) adds to or updates the relevant table in Alto with work data. However, it can also be ran discreetly.

Usage

Log on as **ops** and enter the following command:

work_imp -nw -ra -j -l -b<row set> -e<row set> <libcode> <database name> <data directory>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-nw	This optional argument adds Works which are new to the database, but does not update existing Works.
-ra	This optional argument creates a flat file of the Control Numbers of all new Works added; each Control Number is new-line delimited. This output file is called "work_adds_out.[date.time]" and will be written to the /users/ops/report directory
-j	This optional argument causes the CONTROL_NUMBER table to be checked to see if the incoming Control Number is an Added Number, when it has not been found as a Main Number in the WORK table
-l	This optional argument, "-l", should be used when importing Works from a Supplier which does not allow the Library to pass on records, i.e. where records are for your Library only. It sets the contribution status of all Works added or updated to "Never Contribute" (CONTRIBUTION_TYPE set to 2). This argument may be applied to the "import" or "process_data" command lines, or to the "work_imp" process itself. If this argument is not given, all Works

	added or updated are given the contribution status "Not contribute".
-b	This optional argument allows a beginning row set to be nominated. It may be used if large files are being processed, or if there is a need to re-start either work_imp or item_imp.
-e	This optional argument allows an end row set to be nominated. It may be used if large files are being processed, or if there is a need to re-start either work_imp or item_imp.
libcode	This mandatory argument is required to check the incoming data is for the correct Library. The Libcode should be expressed in upper case and must follow any optional arguments.
data directory	The data directory to be used (for example **/scratch**)

wrk_upd_imp

The **wrk_upd_imp** script calls the standard import_work scripts and is used to process Bibliographical data which has been electronically transmitted to a customer machine. It has two functions - it updates skeleton Bibliographical data (created for an Order or imported from a Supplier) with a full record from the Base database, and processes any "weekly" tape.

In order for **wrk_upd_imp** to identify the type of file it is processing, a parameter file has to be specified on the command line. The script can be run either interactively or from the cron.

Usage

Log on as **ops** and enter the following command:

wrk_upd_imp -h -d<database name> -p<parameter file> -r<report directory> -s<data directory> -t<processing type>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-p	This is a mandatory argument which specifies the parameter file; the parameter file must be located in the data directory.
-t	This specifies the type of processing to be performed; only "prep" is valid for use with this argument. If present, the script will validate and format the input file, but will not perform the import processing. If "-tprep" is not given, the file will be completely processed through all import stages.

Parameter file

The **wrk_csv.param** parameter file should be created in the data directory (which defaults to /usr/opt/blcmp/data/impdir). The **wrk_csv.param** parameter file has 2 functions:

- To identify the optional switches which are to be passed to the "import" script "process_data".
- To identify that the transmission file type is "csv".

The parameter file must contain the following 2 valid labels only:

Parameter	Description

IMPORT_OPTIONS=	This defines which switches are to be passed to the "import" command line; any of the switches which can be applied to the "import" command line are valid. You should ensure that you maintain consistency between the weekly and interim Work import updating. The majority of Libraries will not require this parameter, but it should still be included in the parameter file with no switches specified. Note: The label IMPORT_OPTIONS= accepts the local "-l". The optional argument, "-l", if specified, sets the contribution status of all Works added or updated to "Never Contribute". If this argument is not given, all Works added or updated are given the contribution status "Not contribute".
TRANS_FILE_TYPE=	This specifies the type of input file to be processed; only "csv" and "upd" are valid types. If both "csv" and "upd" files are to be processed, it will be necessary to set up 2 parameter files: i. For "csv" processing, set up "wrk_csv.param" to contain, for example: IMPORT_OPTIONS= -g -c TRANS_FILE_TYPE=csv Note: The "import" argument "-nw" should not be used when processing "csv" files, as this will prevent the upgrade of existing skeleton "Bibliographic" records. ii. For "upd" processing, set up "wrk_upd.param" to contain, for example: IMPORT_OPTIONS= -g -j TRANS_FILE_TYPE=upd

Notes

- The **wrk_upd_imp** script generates a single report for each run; this report is written to the /usr/opt/blcmp/data/impdir directory, unless an alternative report directory has been specified.
- The **wrk_upd_imp.rep** has two identifiable sections:
  - the report of the parameter file validation and file handling procedure up to and including the message "Pre-processing completed".
  - the standard Work import report output which has been generated during the standard import processes.

wwl_imp

wwl_imp is a utility which reads a file of pseudo-Talis rows, each of which include a pair of Control Numbers (master and related), a value representing the Type of relationship between the two Control Numbers and a fixed value representing Level. The Control Numbers in each row are replaced by the WORK_ID of associated Works already present in the database, and the resulting rows are added to the WORK_WORK_LINK table.

Usage

Log on as **ops** and enter the following command:

wwl_imp -d<database> -i<source file> -b<row no1> -e<row no2>

Standard script arguments are described here. The remaining arguments for this script are described in the following table.

Argument	Description
-i	This is a mandatory argument, specifying the full pathname of the file containing the input rows.
-b	This is an optional argument, used to specify the first row in the input file from which point onwards the data will be processed. The default is the first

	row.
-e	This is an optional argument, used for specifying the last row in the input file to be processed. The default is the last row.

Notes

- The "wwl_imp" utility writes errors and counts to "talis_rep.[date.time]". This report includes any errors encountered and input row counts.