Testament: Batch Utilities Version 1

CML00072-01

Code Magus Limited (England reg. no. 4024745)
Number 6, 69 Woodstock Road
Oxford, OX2 6EY, United Kingdom
www.codemagus.com

August 16, 2016

Contents

1 Introduction **2**

2 `cmlrsload` **3**
2.1 Utility Overview 3
2.2 Utility Synopsis 3
2.3 Making Reservations 7
2.4 Making linked Reservations 15

3 `cmlrslinkd` **18**
3.1 Utility Overview 18
3.2 Utility Synopsis 18
3.3 Loading Record data 22
3.4 Pruning Record data 31

4 `cmlrsloadpool` **37**
4.1 Utility Overview 37
4.2 Utility Synopsis 37
4.3 Loading Pool data 40

5 Using Application Parameters **43**

A Examples **44**
A.1 Example Input File 44
A.2 Example Object types definition 45
A.3 Example Copy Book 46
A.4 Example Application Parameter Definition File 47

1 Introduction

The Code Magus Limited Testament batch utilities are productivity enhancements of the Code Magus Limited Testament [5] web based system, allowing batch reservations to be made, link data entries to be specified and a pool of candidate entities to be maintained.

They consist of the following utilities:

- 'Testament Batch Loader'

  The 'Testament Batch Loader' utility tool, `cmlrsload`, allows a user to reserve entities in bulk for their own purpose. It processes configuration information and a sub set of data from a recio [1] input stream, using object types [2] and expression evaluation [3] for selecting records and fields, to produce reservations in the Reservation's data base. In other words by processing data output by a system it can quickly reserve entities specific to that system. It also allows for entities related to the main entity to be reserved. For example with in a test system using a join of customer account details and credit cards along with attributes that indicate which are valid, the Testament Batch Loader tool can automatically reserve the accounts required along with the credit card numbers that are linked to each account.

- 'Testament Batch Link Data Loader'

  The 'Testament Batch Link Data Loader' utility tool, `cmlrslinkd`, allows a user to register the data record that was used for a particular reservation in order to render a print out of the data in a formatted layout within the on-line system. This allows a user viewing a reservation to view the data that the reservation was created for.

- 'Testament Batch Pool Loader'

  The 'Testament Batch Pool Loader' utility tool, `cmlrsloadpool`, allows a user to load a pool of entities and related data records for the use by web based users who may search for and reserve entities or view and use the data for test scenarios.

The Testament system must be installed as a prerequisite to running these tools They then also require the database connection information in order to interface to it. This connection information can be stored in an application parameter definition file (APD) [4] so that it does not always have to be specified anew when the tool is invoked.

2 cmlrsload

2.1 Utility Overview

cmlrsload can automatically reserve entities (and linked entities) from a single input source (for example a master file) by leveraging the Code Magus Object Types[2] and expression evaluation[3] in order to identify the entity within the data. The entity is added into the Testament System held within the database. On completion cmlrsload will report on the number of input records read and the number of reservations made.

2.2 Utility Synopsis

cmlrsload can be invoked from the command line in Unix, Linux or Windows and if invoked with the --help parameter will show all the valid parameters that may be supplied as follows:

```
Code Magus Limited Reservation System Loader V1.0: build 2011-02-08-17.48.19
[./cmlrsload] $Id: cmlrsload.c,v 1.6 2011/01/28 22:32:14 hayward Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
[stephen@codemagus.com].
Usage: cmlrsload [OPTION...] <access>(<object>[,<options>]) ...
  -t, --objtypes=<objtypes-name>                              Name of object
                                                             types definition
  -s, --select="from <type-name> [where <expression>]"       Expression for
                                                             selecting input
                                                             records
  -e, --entity=<expression>                                  Expression
                                                             specifying entity
                                                             value
  -g, --reserve-group=<group>                                Group name for
                                                             reservation
  -y, --reserve-type=<type>                                  Type name for
                                                             reservation
  -p, --reserve-purpose=<purpose>                            Purpose of the
                                                             reservation
  -E, --link-entity={|<expression>]}                         Expression
                                                             specifying linked
                                                             entity value
  -G, --link-group={|<group>]}                               Group name for
                                                             linked reservation
  -Y, --link-type={|<type>]}                                 Type name for
                                                             linked reservation
  -P, --link-purpose={--reserve-purpose|<purpose>]}          Purpose of the
                                                             linked reservation
  -a, --applparms={|<APD_file_name>}                         Application
                                                             Parameters for
                                                             Database values
  -u, --rsuser={APD.Reservation_User|<userID>}               Reservation
                                                             System User ID
  -w, --rspassword={APD.Reservation_Password|<password>}     Reservation
                                                             System password
  -I, --dbinstance={APD.Database_Instance|<name>}            Database Instance
                                                             the reservation
                                                             system resides in
  -D, --database={APD.Database_Name|<database_name>}         Database Name
  -U, --dbuser={APD.Database_User|<userID>}                  Database User ID
```

```
  -W, --dbpassword={APD.Database_Password|<password>}        Database password
  -S, --dbschema={APD.Database_Schema|<schema>}              Database schema
  -v, --verbose                                              Verbose
                                                             processing mode

Help options:
  -?, --help                                                 Show this help
                                                             message
  --usage                                                    Display brief
                                                             usage message
```

The parameters are explained below in more detail:

- `<access>(<object>[,<options>])`
  This is the input file that holds records that will be used to resolve the entity name to reserve. This file can be any input file that can be read with a Code Magus recio open string specification.

- `-t, --objtypes=<objtypes-name>`
  This is a required parameter and identifies the object types definition file that specifies the meta-data of the input file. This file with the associated copy book file or files maps the data in the input data stream.

- `-s, --select="from <type-name> [where <expression>]"`
  This is a required parameter and defines the set of records to process from the input recio stream. At a minimum it must select all records from a record type identified in the object types definition (in other words no `where` clause).

  `--select="from <type>"`

  where `<type>` is the object type defined in the object types definition file with the `type` keyword.

- `-e, --entity=<expression>`
  This is a required parameter and identifies an expression over the current record whose value is to be the entity to reserve. This must be a valid expression within the context of the object types loaded.

- `-g, --reserve-group=<group>`
  This is a required parameter and identifies the group within the reservation system to which all the new reserved entities will belong. It must already exist within the reservation system.

- `-c, --reserve-type=<type>`
  This is a required parameter and identifies the type that is used to constrain all the entity names to be reserved within the reservation system. It must already exist within the reservation system. If a derived entity name fails to match this constraint then the entity will not be reserved. In this case either change the type's constraint or use a different type.

- `-p, --reserve-purpose=<purpose>`
  This is a required parameter and identifies the purpose of all the entities that will

be reserved in this batch.

- `-E, --link-entity={|<expression>]}`
  This is an optional parameter and if specified is an expression on the current object types that resolves the value of the entity linked to the one defined in `--entity`.
- `-G, --link-group={|<group>]}`
  This parameter must be specified if `--link-entity` is specified. It identifies the group within the reservation system to which all the new reserved link entities will belong. It must already exist within the reservation system.
- `-C, --link-type={|<type>]}`
  This parameter must be specified if `--link-entity` is specified. It identifies the type that is used to constrain all the link entity names to be reserved within the reservation system. It must already exist within the reservation system. If a derived link entity name fails to match this constraint then the entity will not be reserved. In this case either change the type's constraint or use a different type.
- `-P, --link-purpose={--reserve-purpose|<purpose>]}`
  This parameter may only be specified if `--link-entity` is specified, but if it is not then it defaults to the same value as `--purpose`. It identifies the purpose of all the linked entities that will be reserved in this batch.
- `-a, --applparms={|<APD_file_name>}`
  This is an optional parameter, and if specified will provide defaults for the following five database parameters. As the database connection parameters remain constant they can be stored in the APD file for each run. Furthermore the password field can be stored encrypted or typed in each time making it harder for the security information to be compromised. For more information see section 5 on page 43.
- `-u, --rsuser={APD.Reservation_User|<userID>}`
  This parameter names the reservation system user that will own the reservations made in this batch. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.
- `-w, --rspassword={APD.Reservation_Password|<password>}`
  This parameter names the reservation system user password of the user specified with `--rsuser`. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.
- `-I, --dbinstance={APD.Database_Instance|<name>}`
  This names the DB2 instance within which the reservation system database re-

sides. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-D, --database={APD.Database_Name|<database_name>}`
  This names the DB2 database within which the reservation system database resides. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-U, --dbuser={APD.Database_User|<userID>}`
  This names the DB2 user name that will be used to connect to DB2. This user must have the appropriate authorisation on the objects within the reservation system database. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-W, --dbpassword={APD.Database_Password|<password>}`
  This names the password associated with the DB2 user name. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-S, --dbschema={APD.Database_Schema|<schema>}`
  This names the schema under which all objects (tables and table spaces) are created within the database the holds the reservation system. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-v, --verbose`
  This optional parameter when specified causes the tool to output detailed information about processing and especially any configurations that are used to direct the processing.

The tool will report on how many input records were read, how many reservations were made and, if asked to make linked reservations, how many of them were made. If `--verbose` is specified or the environment variable `CODEMAGUS_MSGLEVEL` is set to a value of `VERBOSE` all configuration settings are reported on. If `CODEMAGUS_MSGLEVEL` is set to a value of `TRACE` then information on what reservations were made is also output.

2.3 Making Reservations

In order to make batch reservations the input data file, the object types definition and all required command line (and/or application) parameters must be specified.

An example of making reservations using the example data and configuration files is as follows:

```
#
# File: unit_test.sh
# This script tests adding reservations records to the data base.
#
# $Author: hayward $
# $Date: 2012/07/19 12:33:51 $
# $Id: unit_test.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $
# $Name:  $
# $Revision: 1.1 $
# $State: Exp $
#
# $Log: unit_test.sh,v $
# Revision 1.1  2012/07/19 12:33:51  hayward
# Add Unit test scripts to CVS.
#
[[ ${DEBUG} == Y ]] && set -x
typeset -r CVS=\
"\$Id: unit_test.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $"
printf "%s\n" "${CVS}"

NAME=$(basename $0)
B=$(dirname $0)
cd ${B}
BASE=${PWD}
cd -

export CODEMAGUS_OUTPUT_FLUSH_ALL=1
export CODEMAGUS_MSGLEVEL=TRACE
export CMLRSLOAD_TEST_FORMATS=${BASE}
DATAFILE=${BASE}/testfile.txt
DATAATTRIB="mode=r"

CMD="${BASE}/../cmlrsload"
PARMS="--verbose \
   --entity=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
   --reserve-type=TCCARDS \
   --reserve-group=TESTING \
   --reserve-purpose=TESTING \
   --objtypes=${CMLRSLOAD_TEST_FORMATS}/testtype.objtypes \
   --select=\"from ACCOUNT_CREDIT_CARD ; \" \
   --rows-in-commit=500 \
   --applparms=${CMLRSLOAD_TEST_FORMATS}/cmlrsload_parameters.apd \
   \"text(${DATAFILE},${DATAATTRIB})\""
```

```
if [[ $1 != [dD] ]]
then
   eval ${CMD} ${PARMS}
   exit $?
fi
#
# Otherwise use GDB to run the program.
cat >${BASE}/${NAME}.gdb <<-@@END@@
# Generated GDB file by ${NAME} on $(date)
file ${CMD}
set break pending on
set env CODEMAGUS_MSGLEVEL=TRACE
set env CMLRSLOAD_TEST_FORMATS=${BASE}
set args ${PARMS}
break main
r
show args
@@END@@
gdb -x ${BASE}/unit_test.gdb
```

and is explained in more detail below:

- Input File

  The example text file is shown in appendix A.1 on page 44. Looking at the copy book for this file it can be seen that the first 20 bytes is the account number used as the main entity for reservations.

- Meta Data

  An example of the object types definition is shown in appendix A.2 on page 45 and of the copy book in appendix A.3 on page 46. These files supply the meta data for the following parameters.

- Selection of Input Records

  The selection of input records as specified via the `--select` parameter selects all records where the `ACCOUNT_NUMBER` with in the record type `ACCOUNT_CC_JOIN` is greater than or equal to the value '`900000000000004`'. From the example file it can be seen that this would exclude the first three records.

- Entity Name

  In the example the entity is specified as:

  ```
  --entity="account_cc_join.account_number"
  ```

  which, by referring to the copy book, is the first 20 bytes of each input record.

- Specification of the Reservation Attributes

  In the example the reservation attributes are specified by

```
--reserve-type=TACCOUNT \
--reserve-group=TESTING \
--reserve-purpose=TESTING \
```

and are applied to each reservation that is made.

- Application Parameters

  In this example the database connection information is stored in APD file. It is shown in appendix A.4 on page 47.

If this command is run in a fully functional system with a message level of `TRACE` the output looks like the following:

```
Code Magus Limited Reservation System Loader V1.0: build 2010-12-21-10.25.00
[./cmlrsload] $Id: cmlrsload.c,v 1.5 2010/12/21 09:18:17 hayward Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
[stephen@codemagus.com].
---Start of Processing APPLICATION PARAMETERS Config testing/cmlrsload_parameters.apd---
OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/cmlvars.env
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/software"

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/";
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/software/build/bin/"

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/software]"/CodeMagus//Cat
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagus/software/CodeMagus/

export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/lib/";
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/software/build/lib/"

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/%s.amd";
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/software/build/bin/%s.

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"

#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY"

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C9F611A816053B"

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
```

```
# End
OSMODS: No user environ file set
application cmlrsload_parameters;
-- This APD file defines the parameters used by the CML reservation batch load
-- system. It defines the database connection parameters so that they do not
-- have to be supplied on the command line and parameters such as the password
-- can be stored in an encrypted format.

-- $Author: hayward $
-- $Date: 2010/10/22 06:34:53 $
-- $Id: cmlrsload_parameters.apd,v 1.2 2010/10/22 06:34:53 hayward Exp $
-- $Name:  $
-- $Revision: 1.2 $
-- $State: Exp $
--
-- $Log: cmlrsload_parameters.apd,v $
-- Revision 1.2  2010/10/22 06:34:53  hayward
-- Make test files into a correct example.
--
-- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
-- Add CML Reservation System Loader to CVS.
--

   title "Reservation System Load parameters";
   description "This file defines the database connection parameters "
               " for the Reservation System Batch Loader."
     ;

   set LOGHOME = ${HOME}[=/home/hayward] "/logs";
   set TODAY = ${DATE_YYYYMMDD}[=20101221];

   store ${LOGHOME}[=/home/hayward/logs];

   interface "apcmdui.so";
   entry "apcmdui_entry";

   parameter Database_Instance
      title "Database Instance name";
      default ${DB2INSTANCE}[=db2i82];
      options ;
      description "The database instance to connect to.";
      constraint "^.*$";
   end

   parameter Database_Name
      title "Database name";
      default ${DB2DBDFT}[=sample];
      options ;
      description "The database name to connect to.";
      constraint "^.*$";
   end

   parameter Database_User
      title "Database User ID";
      options ;
      description "The database User Id to use when connecting.";
      constraint "^.*$";
   end

   parameter Database_Schema
      title "Database Schema";
      options ;
      description "The database schema that qualifies the tables of the "
                  "reservation system.";
      constraint "^.*$";
   end
```

```
   parameter Database_Password
      title "Password for the Databse user";
      default NULL;
      options secret;
      description
         "This is the password associated with the database user."
         ;
      constraint "^[^ ]\+$"; --- at least one character, no spaces.
   end
end.

OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/cmlvars.env
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/software"

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/";
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/software/build/bin/"

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/software]"/CodeMagus//Cat
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagus/software/CodeMagus/

export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/lib/";
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/software/build/lib/"

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/%s.amd";
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/software/build/bin/%s.

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"

#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY"

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C9F611A816053B"

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
# End
OSMODS: No user environ file set
---End of APPLICATION PARAMETERS Config---
  -- File: testtype.objtypes

Application Parameter Command Interface.
show header
 Name:        cmlrsload_parameters
 Title:       Reservation System Load parameters
```

```
 Description: This file defines the database connection parameters for the
              Reservation System Batch Loader.

Use the Application Parameters Command interface to
  update and review parameter values or apply values
  from the log file. Use "help" for help on commands.
apui>exit  --
  -- Test object type specification
  --
  -- Copyright (c) 2010 Code Magus Limited. All rights reserved.
  --
  -- $Author: hayward $
  -- $Date: 2010/12/21 09:18:17 $
  -- $Id: testtype.objtypes,v 1.3 2010/12/21 09:18:17 hayward Exp $
  -- $Name:  $
  -- $Revision: 1.3 $
  -- $State: Exp $
  --
  -- $Log: testtype.objtypes,v $
  -- Revision 1.3  2010/12/21 09:18:17  hayward
  -- Add program to load linkdata records.
  --
  -- Revision 1.2  2010/10/22 06:34:53  hayward
  -- Make test files into a correct example.
  --
  -- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
  -- Add CML Reservation System Loader to CVS.
  --
path ${CODEMAGUS_FORMATS}[=/home/hayward/mystuff/codemagus/software/cmlrsload]"/testing/%s.cpy";

type ACCOUNT_CREDIT_CARD
   title "Example of an account and credit card join file"
   book testbook
   map account_cc_join
   include account_cc_join;

SYMBOLS: /usr/local/CodeMagus/etc/CMLSWKEY: No such file or directory
SYMBOLS: Support: www.codemagus.com
SYMBOLS: License key not found in license file /usr/local/CodeMagus/etc/CMLSWKEY.
SYMBOLS: Support: www.codemagus.com

      * File TESTBOOK.cpy                                          ->000001
      * This COBOL copy book is an example of using OBJ types to   ->000002
      * map the input of a file used for making reservations.      ->000003
      * It simulates a join between the customer account data and  ->000004
      * the credit card data so that entities of customer number   ->000005
      * and liked entities of credit card numbers can be reserved  ->000006
      * in the reservation system.                                 ->000007
      *                                                            ->000008
      * $Author: hayward $                                         ->000009
      * $Date: 2010/10/22 11:34:13 $                               ->000010
      * $Id: TESTBOOK.cpy,v 1.4 2010/10/22 11:34:13 hayward Exp $  ->000011
      * $Name:  $                                                  ->000012
      * $Revision: 1.4 $                                          ->000013
      * $State: Exp $                                             ->000014
      *                                                            ->000015
      * $Log: TESTBOOK.cpy,v $                                     ->000016
      * Revision 1.4  2010/10/22 11:34:13  hayward                 ->000017
      * Add filler at end of record so that                        ->000018
      * the complete record can be seen in the                     ->000019
      * documentation (there are no trailing blanks)               ->000020
      *                                                            ->000021
      * Revision 1.3  2010/10/22 06:34:53  hayward                 ->000022
      * Make test files into a correct example.                    ->000023
      *                                                            ->000024
       01  account-cc-join.                                        ->000025
```

```
           03 account-number          pic x(20).                          ->000026
           03 other-data1             pic x(10).                          ->000027
           03 other-data2             pic x(10).                          ->000028
           03 credit-card-number      pic x(20).                          ->000029
           03 filler                  pic x.                              ->000030
Start of Open Spec processing: text(testing/testfile.txt,mode=r)
Access Method: TEXT
Object Name:   testing/testfile.txt
Option String: mode=r
Start of Option String processing: mode=r

Start of Option String Scanning:
mode=r
access text(mode,texttype="LOCAL");

  -- File: TEXT.amd
  --
  -- This file contains an access method definition which is used to read
  -- and write local files of as a text stream. Support is provided for
  -- various different text types including DOS, UNIX and MVS (or USS).
  -- Any text type file can be read or written on any of the platforms
  -- and if not specified it is defaulted to LOCAL (ie the platform the
  -- Access Method is running on).
  --
  --
  -- Author: Stephen R. Donaldson [stephen@codemagus.com].
  --
  -- Copyright (c) 2008 Code Magus Limited. All rights reserved.

  -- $Author: hayward $
  -- $Date: 2009/11/10 11:15:56 $
  -- $Id: TEXT.amd,v 1.8 2009/11/10 11:15:56 hayward Exp $
  -- $Name:  $
  -- $Revision: 1.8 $
  -- $State: Exp $
  --
  -- $Log: TEXT.amd,v $
  -- Revision 1.8  2009/11/10 11:15:56  hayward
  -- Allow text to use SKIP_INPUT and point method.
  --
  -- Revision 1.7  2009/05/27 08:52:39  hayward
  -- Correct documentation.
  --
  -- Revision 1.6  2009/03/12 11:50:55  hayward
  -- After testing on MVS and USS and reading the C runtime manual for
  -- fwrite() and fread() it transpires that MVS and USS can be handled
  -- in the same way. This requires using 0x15 as the line delimiter and
  -- opening the file without "b,type=record". On USS the 0x15 is written
  -- to the file (in the same manner as 0x0a on Unix) and on MVS it is
  -- stripped at write time and a record boundary is inserted; whereas
  -- on read a record boundary is replaced with a trailing 0x15.
  -- This is the best outcome because now we can generate any platform
  -- based file on any platform (not including ASCII/EBCDIC issues though).
  --
  -- Revision 1.5  2009/03/10 08:42:36  hayward
  -- Major change to add texttype parameter to the AMD and how we read/write
  -- text files.
  --
  -- Revision 1.4  2008/04/22 17:32:15  hayward
  -- Add ability for callers to append to files by
  -- adding 'a' to the mode constraint.
  --
  -- Revision 1.3  2008/04/09 13:57:00  hayward
  -- Fix path statement.
  --
  -- Revision 1.2  2008/03/31 22:30:45  stephen
```

```
  -- Add usage of environment variables to AMD file
  --
  -- Revision 1.1  2008/03/20 14:56:25  stephen
  -- Add new contents of recio text access method
  --

   modes seq_input, seq_output, skip_input;

   implements open;
   implements close;
   implements read;
   implements write;
   implements tell;
   implements point;

   describe mode as
      "The mode is the open mode string which will be passed to the C Standard "
      "I/O Library.";

   describe texttype as
      "The texttype parameter determines the line end delimiter for the text "
      "being written.";

   constrain mode as "^[rwa]\(,type=record\)\{0,1\}$";
   constrain texttype as "^\(LOCAL\|DOS\|UNIX\|MVS\)$";

   path = ${CODEMAGUS_AMDLIBS}[=/home/hayward/mystuff/codemagus/software/build/lib/] "%s";
   module = "textam" ${CODEMAGUS_AMDSUFDL}[=.so];
   entry = textam_init;

end.
$Id: textam.c,v 1.22 2010/10/25 08:09:54 hayward Exp $
Text File format UNIX
Connected to DB2 instance=sample, db2_user=hayward
DB2 Schema set to CMLRT00
Increased internal read buffer from 65536 to 131072
Entity "900000000000004" reserved
Entity "900000000000005" reserved
Entity "900000000000006" reserved
Entity "900000000000007" reserved
Entity "900000000000008" reserved
Entity "900000000000009" reserved
Entity "900000000000010" reserved

text(testing/testfile.txt,mode=r): Input Records Read=17
Reservations Made=7
```

2.4 Making linked Reservations

Making linked entity reservation occurs at the same time as making main entity reservations. The invocation is the same as for reserving the main entity, but the optional parameters `--link-entity`, `--link-type`, `--link-group` and `--link-purpose` should be specified. An example of a command to do this that uses the same input file and configuration as the previous example is:

```
#
# File: unit_test_link.sh
# This script tests adding link reservations records to the data base.
#
# $Author: hayward $
# $Date: 2012/07/19 12:33:51 $
# $Id: unit_test_link.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $
# $Name:  $
# $Revision: 1.1 $
# $State: Exp $
#
# $Log: unit_test_link.sh,v $
# Revision 1.1  2012/07/19 12:33:51  hayward
# Add Unit test scripts to CVS.
#
[[ ${DEBUG} == Y ]] && set -x
typeset -r CVS=\
"\$Id: unit_test_link.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $"
printf "%s\n" "${CVS}"

NAME=$(basename $0)
B=$(dirname $0)
cd ${B}
BASE=${PWD}
cd -

export CODEMAGUS_OUTPUT_FLUSH_ALL=1
export CODEMAGUS_MSGLEVEL=TRACE
export CMLRSLOAD_TEST_FORMATS=${BASE}
DATAFILE=${BASE}/testfile.txt
DATAATTRIB="mode=r"

CMD="${BASE}/../cmlrsload"
PARMS="--verbose \
   --entity=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
   --reserve-type=TACCOUNT \
   --reserve-group=TESTING \
   --reserve-purpose=TESTING \
   --objtypes=${CMLRSLOAD_TEST_FORMATS}/testtype.objtypes \
   --select=\"from ACCOUNT_CREDIT_CARD ; \" \
   --rows-in-commit=500 \
   --applparms=${CMLRSLOAD_TEST_FORMATS}/cmlrsload_parameters.apd \
   --link-to-entity=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
   --link-to-type=TCCARDS \
   \"text(${DATAFILE},${DATAATTRIB})\""

if [[ $1 != [dD] ]]
then
   eval ${CMD} ${PARMS}
   exit $?
fi
#
# Otherwise use GDB to run the program.
cat >${BASE}/${NAME}.gdb <<-@@END@@
# Generated GDB file by ${NAME} on $(date)
```

```
file ${CMD}
set break pending on
set env CODEMAGUS_MSGLEVEL=TRACE
set env CMLRSLOAD_TEST_FORMATS=${BASE}
set args ${PARMS}
break main
r
show args
@@END@@
gdb -x ${BASE}/unit_test_link.gdb
```

and the differences to the previous example are explained in more detail below:

- Link Entity Name

  In the example the link entity is specified as:

  ```
  --link-entity="account_cc_join.credit_card_number"
  ```

  which, by referring to the copy book, is the last 20 bytes (but 1 byte of filler) of each input record.

- Specification of the Linked Reservation Attributes

  In the example the linked reservation attributes are specified by

  ```
  --link-type=TCCARDS \
  --link-group=TESTINGL \
  --link-purpose=LINKED \
  ```

  and are applied to each linked reservation that is made.

If this command is run in a fully functional system with a message level of `TRACE` the output looks like the following (only the final lines that are different to the previous example are shown):

```
Connected to DB2 instance=sample, db2_user=hayward
DB2 Schema set to CMLRT00
Increased internal read buffer from 65536 to 131072
Entity "900000000000004" reserved
Linked Entity "5445999900040001" reserved
Entity "900000000000005" reserved
Linked Entity "5445999900050001" reserved
Linked Entity "5445999900050002" reserved
Linked Entity "5445999900050003" reserved
Linked Entity "5445999900050004" reserved
Entity "900000000000006" reserved
Linked Entity "5445999900060001" reserved
Entity "900000000000007" reserved
Linked Entity "5445999900070001" reserved
Entity "900000000000008" reserved
Linked Entity "5445999900080001" reserved
Linked Entity "5445999900080002" reserved
Linked Entity "5445999900080003" reserved
Entity "900000000000009" reserved
Linked Entity "5445999900090001" reserved
```

```
Entity "900000000000010" reserved
Linked Entity "5445999900100001" reserved
Linked Entity "5445999900100002" reserved
Linked Entity "5445999900100003" reserved

text(testing/testfile.txt,mode=r): Input Records Read=17
Reservations Made=7
Link Reservations Made=14
```

It can be seen from this output that each main entity is reserved once followed by its linked entities. This ordered output depends on the input file order, but will still work the same way if the data is not ordered according to the main entity as the tool keeps track internally of the main entities that have been reserved within the run unit and does not attempt to reserve them a second time.

If the main entity fails to be reserved then no attempt is made to reserve the associated link entity.

3 cmlrslinkd

3.1 Utility Overview

cmlrslinkd can automatically load the physical record of a reserved entity into the database ready for use by web based users to view and use. It does this by reading a from a single input source (for example a master file) and by leveraging the Code Magus Object Types[2] and expression evaluation[3] in order to identify the entity within the data. If the entity is already reserved then cmlrslinkd loads the complete record into the system.

3.2 Utility Synopsis

cmlrslinkd can be invoked from the command line in Unix, Linux or Windows and if invoked with the --help parameter will show all the valid parameters that may be supplied as follows:

```
Code Magus Limited Reservation System Loader V1.0: build 2011-02-08-17.48.19
[./cmlrslinkd] $Id: cmlrslinkd.c,v 1.2 2011/01/28 22:32:14 hayward Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
[stephen@codemagus.com].
Usage: cmlrslinkd [OPTION...] <access>(<object>[,<options>]) ...
  -t, --objtypes=<objtypes-name>                              Name of object
                                                             types definition
  -s, --select="from <type-name> [where <expression>]"       Expression for
                                                             selecting input
                                                             records
  -n, --name=<expression>                                    Expression for
                                                             resolving the
                                                             link name
  -e, --entity=<expression>                                  Expression
                                                             specifying entity
                                                             value
  -y, --type=<type>                                          Type name for
                                                             reservation
  -c, --comment=<comment>                                    Additional Comment
  -d, --days={10|<integer>}                                  Number of days
                                                             until entry
                                                             expires
  -p, --prune                                                Prune/remove
                                                             expired entries
  -a, --applparms={|<APD_file_name>}                         Application
                                                             Parameters for
                                                             Database values
  -u, --rsuser={APD.Reservation_User|<userID>}               Reservation
                                                             System User ID
  -w, --rspassword={APD.Reservation_Password|<password>}     Reservation
                                                             System password
  -I, --dbinstance={APD.Database_Instance|<name>}            Database Instance
                                                             the reservation
                                                             system resides in
  -D, --database={APD.Database_Name|<database_name>}         Database Name
  -U, --dbuser={APD.Database_User|<userID>}                  Database User ID
  -W, --dbpassword={APD.Database_Password|<password>}        Database password
  -S, --dbschema={APD.Database_Schema|<schema>}              Database schema
  -v, --verbose                                              Verbose
```

```
                                                        processing mode

Help options:
  -?, --help                                            Show this help
                                                        message
  --usage                                               Display brief
                                                        usage message
```

The parameters are explained below in more detail:

- `<access>(<object>[,<options>])`
  This is the input file that holds records that will be used to resolve the entity name to reserve. This file can be any input file that can be read with a Code Magus recio open string specification.

- `-t, --objtypes=<objtypes-name>`
  This is a required parameter and identifies the object types definition file that specifies the meta-data of the input file. This file with the associated copy book file or files maps the data in the input data stream.

- `-s, --select="from <type-name> [where <expression>]"`
  This is a required parameter and defines the set of records to process from the input recio stream. At a minimum it must select all records from a record type identified in the object types definition (in other words no `where` clause).

  `--select="from <type>"`

  where `<type>` is the object type defined in the object types definition file with the `type` keyword.

- `-n, --name=<expression>`
  This is a required parameter and identifies an expression over the current record whose value is to be the name of the reserved entity. This must resolve to a unique value as it is the primary key of the table that holds the physical records. For daily transaction data it is often a good idea to concatenate any value with the date and time of the load in order to ensure that the value is unique, but for a master record the choice may be different as there would not be multiple copies of the data. This must be a valid expression within the context of the object types loaded.

- `-e, --entity=<expression>`
  This is a required parameter and identifies an expression over the current record whose value is to be the entity to reserve. This must be a valid expression within the context of the object types loaded.

- `-y, --type=<type>`
  This is a required parameter and identifies the type that is used to constrain all the entity names associated with the link data record being loaded. It must already exist within the reservation system. If a derived entity name fails to match this constraint then the link data record will not be loaded. In this case either change the type's constraint or use a different type.

- `-c, --comment=<comment>`
  This value is stored as a comment against the link data record in the database repository.
- `-d, --days={10|<integer>}`
  This specifies the number of days until the data record link is considered to be expired and therefore ready to be pruned from the database repository. The actual entry is only deleted when a prune is performed against the link data table.
- `-p, --prune`
  This parameter overrides any that indicate loading data link records and causes the tool to perform a prune of any expired records. Records are deleted from the database repository if the current date is past that specified for the date of expiry of the record.
- `-a, --applparms={|<APD_file_name>}`
  This is an optional parameter, and if specified will provide defaults for the following five database parameters. As the database connection parameters remain constant they can be stored in the APD file for each run. Furthermore the password field can be stored encrypted or typed in each time making it harder for the security information to be compromised. For more information see section 5 on page 43.
- `-u, --rsuser={APD.Reservation_User|<userID>}`
  This parameter names the reservation system user that will own the reservations made in this batch. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.
- `-w, --rspassword={APD.Reservation_Password|<password>}`
  This parameter names the reservation system user password of the user specified with `--rsuser`. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.
- `-I, --dbinstance={APD.Database_Instance|<name>}`
  This names the DB2 instance within which the reservation system database resides. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.
- `-D, --database={APD.Database_Name|<database_name>}`
  This names the DB2 database within which the reservation system database resides. This is an optional parameter, but is a required value and must be resolved

either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-U, --dbuser={APD.Database_User|<userID>}`
  This names the DB2 user name that will be used to connect to DB2. This user must have the appropriate authorisation on the objects within the reservation system database. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-W, --dbpassword={APD.Database_Password|<password>}`
  This names the password associated with the DB2 user name. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-S, --dbschema={APD.Database_Schema|<schema>}`
  This names the schema under which all objects (tables and table spaces) are created within the database the holds the reservation system. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-v, --verbose`
  This optional parameter when specified causes the tool to output detailed information about processing and especially any configurations that are used to direct the processing.

3.3 Loading Record data

In order to load link data records the input data file, the object types definition and all required command line (and/or application) parameters must be specified.

An example of loading link data records using the example data and configuration files is as follows:

```
#
# File: unit_test_linkdata.sh
# This script tests adding link data records as BLOBs to the data base.
#
# $Author: hayward $
# $Date: 2012/07/19 12:33:51 $
# $Id: unit_test_linkdata.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $
# $Name:  $
# $Revision: 1.1 $
# $State: Exp $
#
# $Log: unit_test_linkdata.sh,v $
# Revision 1.1  2012/07/19 12:33:51  hayward
# Add Unit test scripts to CVS.
#
[[ ${DEBUG} == Y ]] && set -x
typeset -r CVS=\
"\$Id: unit_test_linkdata.sh,v 1.1 2012/07/19 12:33:51 hayward Exp $"
printf "%s\n" "${CVS}"

NAME=$(basename $0)
B=$(dirname $0)
cd ${B}
BASE=${PWD}
cd -

export CODEMAGUS_OUTPUT_FLUSH_ALL=1
export CODEMAGUS_MSGLEVEL=TRACE
export CMLRSLOAD_TEST_FORMATS=${BASE}
DATAFILE=${BASE}/testfile.txt
DATAATTRIB="mode=r"

CMD="${BASE}/../cmlrslinkd"
#
# These three lines can be swapped in in order to load linked data for plastics.
#
#   --name=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
#   --entity=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
#   --type=TACCOUNT \
#
#   --name=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
#   --entity=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
#   --type=TCCARDS \
PARMS="--verbose \
```

```
   --name=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
   --entity=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
   --type=TACCOUNT \
   --comment=TESTING \
   --days=20 \
   --objtypes=${CMLRSLOAD_TEST_FORMATS}/testtype.objtypes \
   --select=\"from ACCOUNT_CREDIT_CARD ; \" \
   --rows-in-commit=500 \
   --applparms=${CMLRSLOAD_TEST_FORMATS}/cmlrsload_parameters.apd \
   \"text(${DATAFILE},${DATAATTRIB})\""

if [[ $1 != [dD] ]]
then
   eval ${CMD} ${PARMS}
   exit $?
fi
#
# Otherwise use GDB to run the program.
cat >${BASE}/${NAME}.gdb <<-@@END@@
# Generated GDB file by ${NAME} on $(date)
file ${CMD}
set break pending on
set env CODEMAGUS_MSGLEVEL=TRACE
set env CMLRSLOAD_TEST_FORMATS=${BASE}
set args ${PARMS}
break main
r
show args
@@END@@
gdb -x ${BASE}/${NAME}.gdb
exit $?
```

and is explained in more detail below:

- Input File

  The example text file is shown in appendix A.1 on page 44. Looking at the copy book for this file it can be seen that the first 20 bytes is the account number used as the name and entity for loading link data records.

- Meta Data

  An example of the object types definition is shown in appendix A.2 on page 45 and of the copy book in appendix A.3 on page 46. These files supply the meta data for the following parameters.

- Selection of Input Records

  The selection of input records as specified via the `--select` parameter selects all records where the `ACCOUNT_NUMBER` with in the record type `ACCOUNT_CC_JOIN` is greater than or equal to the value '`900000000000004`'. From the example

file it can be seen that this would exclude the first three records.

- Entity Name

  In the example the name and entity are specified as:

  ```
  --name="account_cc_join.account_number" \
  --entity="account_cc_join.account_number" \
  ```

  which, by referring to the copy book, is the first 20 bytes of each input record.

- Specification of the data link record attributes

  In the example the data link record attributes are specified by

  ```
  --reserve-type=TACCOUNT \
  --comment=TESTING \
  --days=20 \
  ```

  and are applied to each reservation that is made.

- Application Parameters

  In this example the database connection information is stored in APD file. It is shown in appendix A.4 on page 47.

If this command is run in a fully functional system with a message level of `TRACE` the output looks like the following:

```
Code Magus Limited Reservation System Loader V1.0: build 2010-12-21-10.25.00
[./cmlrslinkd] $Id: cmlrslinkd.c,v 1.1 2010/12/21 09:18:17 hayward Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
[stephen@codemagus.com].
---Start of Processing APPLICATION PARAMETERS Config testing/cmlrsload_parameters.apd---
OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/cmlvars.env
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/software"

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/";
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/software/build/bin/"

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/software]"/CodeMagus//Cat
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagus/software/CodeMagus/

export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/lib/";
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/software/build/lib/"

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/%s.amd";
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/software/build/bin/%s.

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"
```

```
#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY"

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C9F611A816053B"

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
# End
OSMODS: No user environ file set
application cmlrsload_parameters;
-- This APD file defines the parameters used by the CML reservation batch load
-- system. It defines the database connection parameters so that they do not
-- have to be supplied on the command line and parameters such as the password
-- can be stored in an encrypted format.

-- $Author: hayward $
-- $Date: 2010/10/22 06:34:53 $
-- $Id: cmlrsload_parameters.apd,v 1.2 2010/10/22 06:34:53 hayward Exp $
-- $Name:  $
-- $Revision: 1.2 $
-- $State: Exp $
--
-- $Log: cmlrsload_parameters.apd,v $
-- Revision 1.2  2010/10/22 06:34:53  hayward
-- Make test files into a correct example.
--
-- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
-- Add CML Reservation System Loader to CVS.
--

   title "Reservation System Load parameters";
   description "This file defines the database connection parameters "
               " for the Reservation System Batch Loader."
     ;

   set LOGHOME = ${HOME}[=/home/hayward] "/logs";
   set TODAY = ${DATE_YYYYMMDD}[=20101221];

   store ${LOGHOME}[=/home/hayward/logs];

   interface "apcmdui.so";
   entry "apcmdui_entry";

   parameter Database_Instance
      title "Database Instance name";
      default ${DB2INSTANCE}[=db2i82];
      options ;
      description "The database instance to connect to.";
      constraint "^.*$";
   end

   parameter Database_Name
      title "Database name";
      default ${DB2DBDFT}[=sample];
      options ;
      description "The database name to connect to.";
```

```
      constraint "^.*$";
   end

   parameter Database_User
      title "Database User ID";
      options ;
      description "The database User Id to use when connecting.";
      constraint "^.*$";
   end

   parameter Database_Schema
      title "Database Schema";
      options ;
      description "The database schema that qualifies the tables of the "
                  "reservation system.";
      constraint "^.*$";
   end

   parameter Database_Password
      title "Password for the Databse user";
      default NULL;
      options secret;
      description
         "This is the password associated with the database user."
         ;
      constraint "^[^ ]\+$"; --- at least one character, no spaces.
   end
end.

OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/cmlvars.env
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/software"

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/";
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/software/build/bin/"

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/software]"/CodeMagus//Cat
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagus/software/CodeMagus/

export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/lib/";
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/software/build/lib/"

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/build]"/bin/%s.amd";
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/software/build/bin/%s.

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"

#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
```

```
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY"

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C9F611A816053B"

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
# End
OSMODS: No user environ file set
---End of APPLICATION PARAMETERS Config---
Connected to DB2 instance=sample, db2_user=hayward

Application Parameter Command Interface.
show header
 Name:        cmlrsload_parameters
 Title:       Reservation System Load parameters
 Description: This file defines the database connection parameters for the
              Reservation System Batch Loader.

Use the Application Parameters Command interface to
  update and review parameter values or apply values
  from the log file. Use "help" for help on commands.
apui>exitDB2 Schema set to CMLRT00
  -- File: testtype.objtypes
  --
  -- Test object type specification
  --
  -- Copyright (c) 2010 Code Magus Limited. All rights reserved.
  --
  -- $Author: hayward $
  -- $Date: 2010/12/21 09:18:17 $
  -- $Id: testtype.objtypes,v 1.3 2010/12/21 09:18:17 hayward Exp $
  -- $Name:  $
  -- $Revision: 1.3 $
  -- $State: Exp $
  --
  -- $Log: testtype.objtypes,v $
  -- Revision 1.3  2010/12/21 09:18:17  hayward
  -- Add program to load linkdata records.
  --
  -- Revision 1.2  2010/10/22 06:34:53  hayward
  -- Make test files into a correct example.
  --
  -- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
  -- Add CML Reservation System Loader to CVS.
  --
path ${CODEMAGUS_FORMATS}[=/home/hayward/mystuff/codemagus/software/cmlrsload]"/testing/%s.cpy";

type ACCOUNT_CREDIT_CARD
   title "Example of an account and credit card join file"
   book testbook
   map account_cc_join
   include account_cc_join;

SYMBOLS: /usr/local/CodeMagus/etc/CMLSWKEY: No such file or directory
SYMBOLS: Support: www.codemagus.com
SYMBOLS: License key not found in license file /usr/local/CodeMagus/etc/CMLSWKEY.
SYMBOLS: Support: www.codemagus.com

      * File TESTBOOK.cpy                                          ->000001
      * This COBOL copy book is an example of using OBJ types to   ->000002
      * map the input of a file used for making reservations.      ->000003
      * It simulates a join between the customer account data and  ->000004
      * the credit card data so that entities of customer number   ->000005
      * and liked entities of credit card numbers can be reserved  ->000006
```

```
      * in the reservation system.                                        ->000007
      *                                                                   ->000008
      * $Author: hayward $                                               ->000009
      * $Date: 2010/10/22 11:34:13 $                                     ->000010
      * $Id: TESTBOOK.cpy,v 1.4 2010/10/22 11:34:13 hayward Exp $        ->000011
      * $Name:  $                                                        ->000012
      * $Revision: 1.4 $                                                ->000013
      * $State: Exp $                                                   ->000014
      *                                                                   ->000015
      * $Log: TESTBOOK.cpy,v $                                           ->000016
      * Revision 1.4  2010/10/22 11:34:13  hayward                       ->000017
      * Add filler at end of record so that                              ->000018
      * the complete record can be seen in the                           ->000019
      * documentation (there are no trailing blanks)                     ->000020
      *                                                                   ->000021
      * Revision 1.3  2010/10/22 06:34:53  hayward                       ->000022
      * Make test files into a correct example.                          ->000023
      *                                                                   ->000024
       01  account-cc-join.                                             ->000025
           03 account-number          pic x(20).                       ->000026
           03 other-data1             pic x(10).                       ->000027
           03 other-data2             pic x(10).                       ->000028
           03 credit-card-number      pic x(20).                       ->000029
           03 filler                  pic x.                           ->000030
Start of Open Spec processing: text(testing/testfile.txt,mode=r)
Access Method: TEXT
Object Name:   testing/testfile.txt
Option String: mode=r
Start of Option String processing: mode=r

Start of Option String Scanning:
mode=r
access text(mode,texttype="LOCAL");

  -- File: TEXT.amd
  --
  -- This file contains an access method definition which is used to read
  -- and write local files of as a text stream. Support is provided for
  -- various different text types including DOS, UNIX and MVS (or USS).
  -- Any text type file can be read or written on any of the platforms
  -- and if not specified it is defaulted to LOCAL (ie the platform the
  -- Access Method is running on).
  --
  --
  -- Author: Stephen R. Donaldson [stephen@codemagus.com].
  --
  -- Copyright (c) 2008 Code Magus Limited. All rights reserved.

  -- $Author: hayward $
  -- $Date: 2009/11/10 11:15:56 $
  -- $Id: TEXT.amd,v 1.8 2009/11/10 11:15:56 hayward Exp $
  -- $Name:  $
  -- $Revision: 1.8 $
  -- $State: Exp $
  --
  -- $Log: TEXT.amd,v $
  -- Revision 1.8  2009/11/10 11:15:56  hayward
  -- Allow text to use SKIP_INPUT and point method.
  --
  -- Revision 1.7  2009/05/27 08:52:39  hayward
  -- Correct documentation.
  --
  -- Revision 1.6  2009/03/12 11:50:55  hayward
  -- After testing on MVS and USS and reading the C runtime manual for
  -- fwrite() and fread() it transpires that MVS and USS can be handled
  -- in the same way. This requires using 0x15 as the line delimiter and
```

```
  -- opening the file without "b,type=record". On USS the 0x15 is written
  -- to the file (in the same manner as 0x0a on Unix) and on MVS it is
  -- stripped at write time and a record boundary is inserted; whereas
  -- on read a record boundary is replaced with a trailing 0x15.
  -- This is the best outcome because now we can generate any platform
  -- based file on any platform (not including ASCII/EBCDIC issues though).
  --
  -- Revision 1.5  2009/03/10 08:42:36  hayward
  -- Major change to add texttype parameter to the AMD and how we read/write
  -- text files.
  --
  -- Revision 1.4  2008/04/22 17:32:15  hayward
  -- Add ability for callers to append to files by
  -- adding 'a' to the mode constraint.
  --
  -- Revision 1.3  2008/04/09 13:57:00  hayward
  -- Fix path statement.
  --
  -- Revision 1.2  2008/03/31 22:30:45  stephen
  -- Add usage of environment variables to AMD file
  --
  -- Revision 1.1  2008/03/20 14:56:25  stephen
  -- Add new contents of recio text access method
  --

   modes seq_input, seq_output, skip_input;

   implements open;
   implements close;
   implements read;
   implements write;
   implements tell;
   implements point;

   describe mode as
      "The mode is the open mode string which will be passed to the C Standard "
      "I/O Library.";

   describe texttype as
      "The texttype parameter determines the line end delimiter for the text "
      "being written.";

   constrain mode as "^[rwa]\(,type=record\)\{0,1\}$";
   constrain texttype as "^\(LOCAL\|DOS\|UNIX\|MVS\)$";

   path = ${CODEMAGUS_AMDLIBS}[=/home/hayward/mystuff/codemagus/software/build/lib/] "%s";
   module = "textam" ${CODEMAGUS_AMDSUFDL}[=.so];
   entry = textam_init;

end.
$Id: textam.c,v 1.22 2010/10/25 08:09:54 hayward Exp $
Text File format UNIX
Increased internal read buffer from 65536 to 131072
User hayward granted access F/F/F/F/F
Link data "900000000000004     " filed
Link data "900000000000005     " filed
Input 6: Name 900000000000005      is already on file
Input 7: Name 900000000000005      is already on file
Input 8: Name 900000000000005      is already on file
Link data "900000000000006     " filed
Link data "900000000000007     " filed
Link data "900000000000008     " filed
Input 12: Name 900000000000008      is already on file
Input 13: Name 900000000000008      is already on file
Link data "900000000000009     " filed
Link data "900000000000010     " filed
```

```
Input 16: Name 900000000000010      is already on file
Input 17: Name 900000000000010      is already on file

text(testing/testfile.txt,mode=r): Input Records Read=17
Link Data Records Added=7
```

3.4 Pruning Record data

In order to prune expired linkdata records only the prune parameter and the database connection parameters are required.

An example of pruning linkdata records using the example data and configuration files is as follows:

```
export CODEMAGUS_OUTPUT_FLUSH_ALL=1
export CODEMAGUS_MSGLEVEL=TRACE
export CODEMAGUS_FORMATS=${PWD}
./cmlrslinkd \
   --prune \
   --applparms=testing/cmlrsload_parameters.apd
```

and is explained in more detail below:

- Specify that pruning should occur.

  By specifying the prune parameter, as follows, the tool only performs the prune function.

  ```
  --prune
  ```

- Application Parameters

  In this example the database connection information is stored in APD file. It is shown in appendix A.4 on page 47.

If this command is run in a fully functional system with a message level of TRACE the output looks like the following:

```
Code Magus Limited Reservation System Loader V1.0: build 2010-12-21-10.25.00
[./cmlrslinkd] $Id: cmlrslinkd.c,v 1.1 2010/12/21 09:18:17 hayward Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
[stephen@codemagus.com].
---Start of Processing APPLICATION PARAMETERS Config testing/cmlrsload_parameters.apd
OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/c
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/so

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/soft
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagu
```

```
export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"

#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
# End
OSMODS: No user environ file set
application cmlrsload_parameters;
-- This APD file defines the parameters used by the CML reservation batch load
-- system. It defines the database connection parameters so that they do not
-- have to be supplied on the command line and parameters such as the password
-- can be stored in an encrypted format.

-- $Author: hayward $
-- $Date: 2010/10/22 06:34:53 $
-- $Id: cmlrsload_parameters.apd,v 1.2 2010/10/22 06:34:53 hayward Exp $
-- $Name:  $
-- $Revision: 1.2 $
-- $State: Exp $
--
-- $Log: cmlrsload_parameters.apd,v $
-- Revision 1.2  2010/10/22 06:34:53  hayward
-- Make test files into a correct example.
--
-- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
-- Add CML Reservation System Loader to CVS.
--
```

```
title "Reservation System Load parameters";
description "This file defines the database connection parameters "
            " for the Reservation System Batch Loader."
  ;

set LOGHOME = ${HOME}[=/home/hayward] "/logs";
set TODAY = ${DATE_YYYYMMDD}[=20101221];

store ${LOGHOME}[=/home/hayward/logs];

interface "apcmdui.so";
entry "apcmdui_entry";

parameter Database_Instance
   title "Database Instance name";
   default ${DB2INSTANCE}[=db2i82];
   options ;
   description "The database instance to connect to.";
   constraint "^.*$";
end

parameter Database_Name
   title "Database name";
   default ${DB2DBDFT}[=sample];
   options ;
   description "The database name to connect to.";
   constraint "^.*$";
end

parameter Database_User
   title "Database User ID";
   options ;
   description "The database User Id to use when connecting.";
   constraint "^.*$";
end

parameter Database_Schema
   title "Database Schema";
   options ;
   description "The database schema that qualifies the tables of the "
               "reservation system.";
   constraint "^.*$";
end

parameter Database_Password
   title "Password for the Databse user";
   default NULL;
   options secret;
   description
      "This is the password associated with the database user."
      ;
   constraint "^[^ ]\+$"; --- at least one character, no spaces.
```

```
   end
end.

OSMODS: Processing group DEFAULT
OSMODS: Applying environment variables from group DEFAULT: /usr/local/CodeMagus/etc/c
#
# File: cmlvars.env
# This CML environment variables configuration file is loaded by all Code Magus
# software utilities at program start up.
#
# Environment variables for Recio
export CODEMAGUS_SOURCE="/home/hayward/mystuff/codemagus/software";
OSMODS:Set environment variable: "CODEMAGUS_SOURCE=/home/hayward/mystuff/codemagus/so

export CODEMAGUS_AMDBINS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDBINS=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDCATPATH=${CODEMAGUS_SOURCE}[=/home/hayward/mystuff/codemagus/soft
OSMODS:Set environment variable: "CODEMAGUS_AMDCATPATH=/home/hayward/mystuff/codemagu

export CODEMAGUS_AMDCATNAME="MASTCAT";
OSMODS:Set environment variable: "CODEMAGUS_AMDCATNAME=MASTCAT"

export CODEMAGUS_AMDLIBS=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDLIBS=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDPATH=${CODEMAGUS_HOME}[=/home/hayward/mystuff/codemagus/software/
OSMODS:Set environment variable: "CODEMAGUS_AMDPATH=/home/hayward/mystuff/codemagus/s

export CODEMAGUS_AMDSUFDL=".so";
OSMODS:Set environment variable: "CODEMAGUS_AMDSUFDL=.so"

#
# Environment variable for PPMSORT
export PPMSORTCONF="/usr/local/CodeMagus/etc/ppmsort.cfg";
OSMODS:Set environment variable: "PPMSORTCONF=/usr/local/CodeMagus/etc/ppmsort.cfg"

#
# Enviornment variables for license processing
export CODEMAGUS_KEYFILE="/usr/local/CodeMagus/etc/CMLSWKEY";
OSMODS:Set environment variable: "CODEMAGUS_KEYFILE=/usr/local/CodeMagus/etc/CMLSWKEY

export CODEMAGUS_RUNTIME_KEY_MAGSTRIPE="49624B09B15FD66944C9F611A816053B";
OSMODS:Set environment variable: "CODEMAGUS_RUNTIME_KEY_MAGSTRIPE=49624B09B15FD66944C

#
# Enviornment variables for CML Clock synchronisation server
#export CODEMAGUS_REFCLOCK_SERVER="codemagus.it.nednet.co.za:60001";
# End
OSMODS: No user environ file set
---End of APPLICATION PARAMETERS Config---
Connected to DB2 instance=sample, db2_user=hayward
```

```
Application Parameter Command Interface.
show header
 Name:        cmlrsload_parameters
 Title:       Reservation System Load parameters
 Description: This file defines the database connection parameters for the
              Reservation System Batch Loader.

Use the Application Parameters Command interface to
  update and review parameter values or apply values
  from the log file. Use "help" for help on commands.
apui>exitDB2 Schema set to CMLRT00
rsdb2: Opening statement: select NAME,ENTITY,TYPE,OBJTNAME,OBJTFILEPATH,ADDEDBY,TIMES
rsdb2: Opening linkdata expired cursor
Name: 900000000000004
   Entity: 900000000000004
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.047278
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000005
   Entity: 900000000000005
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.182286
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000006
   Entity: 900000000000006
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.187373
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000007
   Entity: 900000000000007
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.188057
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000008
   Entity: 900000000000008
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.188786
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000009
   Entity: 900000000000009
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.191029
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
Name: 900000000000010
   Entity: 900000000000010
```

```
   TYPE:   TACCOUNT
   CREATED:2010-12-21-10.34.19.191364
   EXPIRES:2010-12-20-10.34.41.916096
   COMMENT:TESTING
User hayward granted access F/F/F/F/F
Number of expired link data items deleted = 7
```

4 cmlrsloadpool

4.1 Utility Overview

cmlrsloadpool can automatically load entities and the related physical record into the pool from an input file ready for use by web based users to view, reserve and use. It does this by reading a from a single input source (for example a master file) and by leveraging the Code Magus Object Types[2] and expression evaluation[3] in order to identify the entity within the data. cmlrsloadpool can also clear existing pool data before loading, which can be used to effectively update the pool information to the most current.

4.2 Utility Synopsis

cmlrsloadpool can be invoked from the command line in Unix, Linux or Windows and if invoked with the --help parameter will show all the valid parameters that may be supplied as follows:

The parameters are explained below in more detail:

- <access>(<object>[,<options>])
  This is the input file that holds records that will be used to resolve the entity name and physical record to load into the pool. This file can be any input file that can be read with a Code Magus recio open string specification.
- -t, --objtypes=<objtypes-name>
  This is a required parameter and identifies the object types definition file that specifies the meta-data of the input file. This file with the associated copy book file or files maps the data in the input data stream.
- -s, --select="from <type-name> [where <expression>]"
  This is a required parameter and defines the set of records to process from the input recio stream. At a minimum it must select all records from a record type identified in the object types definition (in other words no where clause).

  --select="from <type>"

  where <type> is the object type defined in the object types definition file with the type keyword.
- -n, --name=<expression>
  This is a required parameter and identifies an expression over the current record whose value is to be the name of the reserved entity. This must resolve to a unique value as it is the primary key of the table that holds the physical records. This must be a valid expression within the context of the object types loaded.

- `-e, --entity=<expression>`
  This is a required parameter and identifies an expression over the current record whose value is to be the entity to reserve. This must be a valid expression within the context of the object types loaded.

- `-y, --type=<type>`
  This is a required parameter and identifies the type that is used to constrain all the entity names associated with the link data record being loaded. It must already exist within the reservation system. If a derived entity name fails to match this constraint then the link data record will not be loaded. In this case either change the type's constraint or use a different type.

- `-p, --replace`
  This optional parameter causes all records that have the same type specified in `-y, --type` to be deleted before adding any new records. This allows for replacing the pool with a new set of data.

- `-c, --comment=<comment>`
  This value is stored as a comment against the link data record in the database repository.

- `-a, --applparms={|<APD_file_name>}`
  This is an optional parameter, and if specified will provide defaults for the following five database parameters. As the database connection parameters remain constant they can be stored in the APD file for each run. Furthermore the password field can be stored encrypted or typed in each time making it harder for the security information to be compromised. For more information see section 5 on page 43.

- `-u, --rsuser={APD.Reservation_User|<userID>}`
  This parameter names the reservation system user that will own the reservations made in this batch. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-w, --rspassword={APD.Reservation_Password|<password>}`
  This parameter names the reservation system user password of the user specified with `--rsuser`. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-I, --dbinstance={APD.Database_Instance|<name>}`
  This names the DB2 instance within which the reservation system database resides. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this com-

mand line option. The command line option takes precedence over the application parameters.

- `-D, --database={APD.Database_Name|<database_name>}`
  This names the DB2 database within which the reservation system database resides. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-U, --dbuser={APD.Database_User|<userID>}`
  This names the DB2 user name that will be used to connect to DB2. This user must have the appropriate authorisation on the objects within the reservation system database. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-W, --dbpassword={APD.Database_Password|<password>}`
  This names the password associated with the DB2 user name. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-S, --dbschema={APD.Database_Schema|<schema>}`
  This names the schema under which all objects (tables and table spaces) are created within the database the holds the reservation system. This is an optional parameter, but is a required value and must be resolved either through the application parameters (see section 5 on page 43) or this command line option. The command line option takes precedence over the application parameters.

- `-v, --verbose`
  This optional parameter when specified causes the tool to output detailed information about processing and especially any configurations that are used to direct the processing.

4.3 Loading Pool data

In order to load pool data records the input data file, the object types definition and all required command line (and/or application) parameters must be specified.

An example of loading pool data records using the example data and configuration files is as follows:

```
#
# File: unit_test_loadpool.sh
# This script tests loading a file of credit cards into the pool of candidate
# reservation entities.
#
# $Author: hayward $
# $Date: 2012/07/19 20:34:35 $
# $Id: unit_test_loadpool.sh,v 1.2 2012/07/19 20:34:35 hayward Exp $
# $Name:  $
# $Revision: 1.2 $
# $State: Exp $
#
# $Log: unit_test_loadpool.sh,v $
# Revision 1.2  2012/07/19 20:34:35  hayward
# Update docs for POOL loading and usage.
#
# Revision 1.1  2012/07/19 12:33:51  hayward
# Add Unit test scripts to CVS.
#
[[ ${DEBUG} == Y ]] && set -x
typeset -r CVS=\
"\$Id: unit_test_loadpool.sh,v 1.2 2012/07/19 20:34:35 hayward Exp $"
printf "%s\n" "${CVS}"

NAME=$(basename $0)
B=$(dirname $0)
cd ${B}
BASE=${PWD}
cd -

export CODEMAGUS_OUTPUT_FLUSH_ALL=1
export CODEMAGUS_MSGLEVEL=TRACE
export CMLRSLOAD_TEST_FORMATS=${BASE}
DATAFILE=${BASE}/testfile.txt
DATAATTRIB="mode=r"

CMD="${BASE}/../cmlrsloadpool"
#
# These three lines can be swapped in in order to load linked data for plastics.
# The ACCOUNT_NUMBER loads linked data for the accounts, whereas
# CREDIT_CARD_NUMBER is for plastics
#
#   --name=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
#   --entity=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
```

```
#   --type=TACCOUNT \
#
#   --name=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
#   --entity=\"ACCOUNT_CC_JOIN.CREDIT_CARD_NUMBER\" \
#   --type=TCCARDS \
PARMS="--verbose \
   --name=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
   --entity=\"ACCOUNT_CC_JOIN.ACCOUNT_NUMBER\" \
   --type=TACCOUNT \
   --replace \
   --comment=TESTING \
   --objtypes=${CMLRSLOAD_TEST_FORMATS}/testtype.objtypes \
   --select=\"from ACCOUNT_CREDIT_CARD ; \" \
   --rows-in-commit=500 \
   --applparms=${CMLRSLOAD_TEST_FORMATS}/cmlrsload_parameters.apd \
   \"text(${DATAFILE},${DATAATTRIB})\""

if [[ $1 != [dD] ]]
then
   eval ${CMD} ${PARMS}
   exit $?
fi
#
# Otherwise use GDB to run the program.
cat >${BASE}/${NAME}.gdb <<-@@END@@
# Generated GDB file by ${NAME} on $(date)
file ${CMD}
set break pending on
set env CODEMAGUS_MSGLEVEL=TRACE
set env CMLRSLOAD_TEST_FORMATS=${BASE}
set args ${PARMS}
break main
r
show args
@@END@@
gdb -x ${BASE}/${NAME}.gdb
exit $?
```

and is explained in more detail below:

- Input File

  The example text file is shown in appendix A.1 on page 44. Looking at the copy book for this file it can be seen that the first 20 bytes is the account number used as the name and entity for loading link data records.

- Meta Data

  An example of the object types definition is shown in appendix A.2 on page 45 and of the copy book in appendix A.3 on page 46. These files supply the meta data for the following parameters.

- Selection of Input Records

  The selection of input records as specified via the `--select` parameter selects all records where the `ACCOUNT_NUMBER` with in the record type `ACCOUNT_CC_JOIN` is greater than or equal to the value '`9000000000000004`'. From the example file it can be seen that this would exclude the first three records.

- Entity Name

  In the example the name and entity are specified as:

  ```
  --name="account_cc_join.account_number" \
  --entity="account_cc_join.account_number" \
  ```

  if loading pool data records for Account Numbers which, by referring to the copy book, is the first 20 bytes of each input record or ...

  In the example the name and entity are specified as:

  ```
  --name="account_cc_join.credit_card_number" \
  --entity="account_cc_join.credit_card_number" \
  ```

  if loading pool data records for credit card numbers which, by referring to the copy book, is the last 20 bytes of each input record.

- Specification of the data link record attributes

  In the example the data link record attributes are specified by

  ```
  --reserve-type=TACCOUNT \
  --comment=TESTING \
  ```

  if loading pool data records for Account Numbers or ...

  ```
  --reserve-type=TCCARDS \
  --comment=TESTING \
  ```

  if loading pool data records for Credit Card numbers.

- Application Parameters

  In this example the database connection information is stored in APD file. It is shown in appendix A.4 on page 47.

If this command is run in a fully functional system with a message level of `TRACE` the output looks like the following:

5 Using Application Parameters

The reservation system amd database connection parameters can be defined in an APD file and will be persistent from run to run. Furthermore parameters such as the password are stored in an encrypted form (or entered each time) making it more secure than entering it in plain text on the command line. More information about the application parameters interface can be found in ‘ applparms: Application Parameters Library User Guide and Reference Version 1’ [4].

Usually the application parameters are presented to the user via a defined user interface program. Once all the parameters have been updated to have a value the user then exits the interface so that the application (the Testament Batch Loader in this case) can continue. However the APD can also be defined to run non-interactively in which case all the parameters must have a value at start up; either by default or saved from a previous run.

An example of an application parameter definition file is shown in appendix A.4 on page 47 and the parameters it supplies defaults for are:

- Reservation System User ID. This parameter is defined by `Reservation_User` and is the default for the command line parameter `--rsuser`.
- Reservation System Password. This parameter is defined by `Reservation_Password` and is the default for the command line parameter `--rspassword`.
- Database instance. This parameter is defined by `Database_Instance` and is the default for the command line parameter `--dbinstance`.
- Database name. This parameter is defined by `Database_Name` and is the default for the command line parameter `--database`.
- Database User ID. This parameter is defined by `Database_User` and is the default for the command line parameter `--dbuser`.
- Database Password. This parameter is defined by `Database_Password` and is the default for the command line parameter `--dbpassword`.
- Database Schema. This parameter is defined by `Database_Schema` and is the default for the command line parameter `--dbschema`.

A Examples

A.1 Example Input File

```
900000000000001     ++++++++++----------5445999900010001    x
900000000000002     ++++++++++----------5445999900020001    x
900000000000003     ++++++++++----------5445999900030001    x
900000000000004     ++++++++++----------5445999900040001    x
900000000000005     ++++++++++----------5445999900050001    x
900000000000005     ++++++++++----------5445999900050002    x
900000000000005     ++++++++++----------5445999900050003    x
900000000000005     ++++++++++----------5445999900050004    x
900000000000006     ++++++++++----------5445999900060001    x
900000000000007     ++++++++++----------5445999900070001    x
900000000000008     ++++++++++----------5445999900080001    x
900000000000008     ++++++++++----------5445999900080002    x
900000000000008     ++++++++++----------5445999900080003    x
900000000000009     ++++++++++----------5445999900090001    x
900000000000010     ++++++++++----------5445999900100001    x
900000000000010     ++++++++++----------5445999900100002    x
900000000000010     ++++++++++----------5445999900100003    x
```

A.2 Example Object types definition

```
  -- File: testtype.objtypes
  --
  -- Test object type specification
  --
  -- Copyright (c) 2010 Code Magus Limited. All rights reserved.
  --
  -- $Author: hayward $
  -- $Date: 2012/07/19 12:34:49 $
  -- $Id: testtype.objtypes,v 1.5 2012/07/19 12:34:49 hayward Exp $
  -- $Name:  $
  -- $Revision: 1.5 $
  -- $State: Exp $
  --
  -- $Log: testtype.objtypes,v $
  -- Revision 1.5  2012/07/19 12:34:49  hayward
  -- Fix path errors and add load pool.
  --
  -- Revision 1.4  2011/02/08 17:03:30  hayward
  -- Change suffix of Copybooks so that
  -- use of the test objtypes still works
  -- with other formats.
  --
  -- Revision 1.3  2010/12/21 09:18:17  hayward
  -- Add program to load linkdata records.
  --
  -- Revision 1.2  2010/10/22 06:34:53  hayward
  -- Make test files into a correct example.
  --
  -- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
  -- Add CML Reservation System Loader to CVS.
  --
path ${CMLRSLOAD_TEST_FORMATS}"/%s.cpy";

type ACCOUNT_CREDIT_CARD
   title "Example of an account and credit card join file"
   book testbook
   map account_cc_join
   include account_cc_join;
```

A.3 Example Copy Book

```
* File TESTBOOK.cpy
* This COBOL copy book is an example of using OBJ types to
* map the input of a file used for making reservations.
* It simulates a join between the customer account data and
* the credit card data so that entities of customer number
* and liked entities of credit card numbers can be reserved
* in the reservation system.
*
* $Author: hayward $
* $Date: 2010/10/22 11:34:13 $
* $Id: TESTBOOK.cpy,v 1.4 2010/10/22 11:34:13 hayward Exp $
* $Name:  $
* $Revision: 1.4 $
* $State: Exp $
*
* $Log: TESTBOOK.cpy,v $
* Revision 1.4  2010/10/22 11:34:13  hayward
* Add filler at end of record so that
* the complete record can be seen in the
* documentation (there are no trailing blanks)
*
* Revision 1.3  2010/10/22 06:34:53  hayward
* Make test files into a correct example.
*
 01  account-cc-join.
     03 account-number          pic x(20).
     03 other-data1             pic x(10).
     03 other-data2             pic x(10).
     03 credit-card-number      pic x(20).
     03 filler                  pic x.
```

A.4 Example Application Parameter Definition File

```
application cmlrsload_parameters;
-- This APD file defines the parameters used by the CML reservation batch load
-- system. It defines the database connection parameters so that they do not
-- have to be supplied on the command line and parameters such as the password
-- can be stored in an encrypted format.

-- $Author: hayward $
-- $Date: 2012/07/19 12:34:49 $
-- $Id: cmlrsload_parameters.apd,v 1.5 2012/07/19 12:34:49 hayward Exp $
-- $Name:  $
-- $Revision: 1.5 $
-- $State: Exp $
--
-- $Log: cmlrsload_parameters.apd,v $
-- Revision 1.5  2012/07/19 12:34:49  hayward
-- Fix path errors and add load pool.
--
-- Revision 1.4  2011/02/25 22:56:35  hayward
-- Create a real NB and SC test based on a
-- Daily join file
--
-- Revision 1.3  2011/01/28 22:32:14  hayward
-- Changes required to cater for separate
-- authorisation credentials for DB2 and
-- the reservation system.
--
-- Revision 1.2  2010/10/22 06:34:53  hayward
-- Make test files into a correct example.
--
-- Revision 1.1.1.1  2010/10/19 11:47:08  hayward
-- Add CML Reservation System Loader to CVS.
--

   title "Reservation System Load parameters";
   description "This file defines the database connection parameters "
               " for the Reservation System Batch Loader."
     ;

   set LOGHOME = ${CMLRSLOAD_TEST_FORMATS};
   set TODAY = ${DATE_YYYYMMDD};

   store ${LOGHOME};

   interface "apcmdui.so";
   entry "apcmdui_entry";

   parameter Reservation_User
      title "Reservation User ID";
      options ;
      description "The Reservation User Id to use for logging on.";
```

```
   constraint "^.*$";
end

parameter Reservation_Password
   title "Password for the Reservation user";
   default NULL;
   options secret;
   description
      "This is the password associated with the Reservation user."
      ;
   constraint "^[^ ]\+$"; --- at least one character, no spaces.
end

parameter Database_Instance
   title "Database Instance name";
   default ${DB2INSTANCE};
   options ;
   description "The database instance to connect to.";
   constraint "^.*$";
end

parameter Database_Name
   title "Database name";
   default ${DB2DBDFT};
   options ;
   description "The database name to connect to.";
   constraint "^.*$";
end

parameter Database_Schema
   title "Database Schema";
   options ;
   description "The database schema that qualifies the tables of the "
               "reservation system.";
   constraint "^.*$";
end

parameter Database_User
   title "Database User ID";
   options ;
   description "The database User Id to use when connecting."
               "This is a Database specific user ID";
   constraint "^.*$";
end

parameter Database_Password
   title "Password for the Databse user";
   default NULL;
   options secret;
   description
      "This is the password associated with the database user."
      ;
```

```
      constraint "^[^ ]\+$"; --- at least one character, no spaces.
   end
end.
```

References

[1] recio: Record Stream I/O Library Version 1. CML Document CML00001-01, Code Magus Limited, July 2008. PDF.

[2] objtypes: Configuring for Object Recognition, Generation and Manipulation. CML Document CML00018-01, Code Magus Limited, July 2008. PDF.

[3] expeval: Expression Evaluation API Reference. CML Document CML00052-01, Code Magus Limited, November 2009. PDF.

[4] applparms: Application Parameters Library User Guide and Reference Version 1. CML Document CML00054-01, Code Magus Limited, January 2010. PDF.

[5] Testament: Test Entity Reservation and Tracking Version 1. CML Document CML00071-01, Code Magus Limited, October 2010. PDF.