

NAME

process_feed – load feed to Oracle database.

SYNOPSIS**process_feed.ksh**

```
myname:=caller_full_pathname
[owner=schema_owner]
[table=table_name]
[login=loader_login]
[control:=control_file]
[file:=feed_file]
[header=Y|N] | [trailer=Y|N]
[discard=max_discards]
[error=max_error]
[skip=num_records]
[move=Y|N]
[log_table=Y|N]
[help=Y]
```

process_feed.ksh

P1: Name of incoming data file to be loaded
 P2: Y/N flag to determine if header/trailer check should be performed
 P3: Relative path to jobs directory, eg trade/for_trade
 P4: Name of SQL*Loader control file
 P5: Name of Oracle user to perform load
 P6: Oracle schema to be loaded
 P7: Oracle table name to be loaded
 P8: Maximum allowable discards by SQL*Loader, exclusive of header/trailer
 P9: Maximum allowable errors by SQL*Loader

DESCRIPTION

process_feed.ksh drives the loading of data into Oracle. This script checks the accuracy of the incoming ascii file (compare the trailer count and actual count); runs SQL*Loader; checks the SQL*Loader log to ensure that the file loaded successfully; updates status table in the database with the job's completion status and relevant load information.

The new key=val argument style requires only one mandatory argument, myname, which is the program name that calls **process_feed.ksh** (the rest can be either derived or have default values). However, the caller name is available as "\$0" in the caller's program, this is not user replaceable part. If we can rely on the default, and derivable values, then the caller program is maintenance free. For different jobs, the content of the caller program is identical, only the names differ.

And if you do need to supply additional parameters, it can be in any order. ":= " is used for case sensitive options, myname and file, and "= " is used for other options, which are case insensitive.

The old argument style requires exactly 9 arguments, in the order shown above.

The new **process_feed.ksh** is backward compatible, ie, it accepts both new and old argument style. I hope you find the new style is easier to use and maintain.

COMMAND LINE OPTIONS

The feed file name is optional in new argument style. If not specified, it is derived from control file. A line containing "infile '*incoming_file*'", not preceded with "--" in the control file is assumed, and the word "infile" can be in mixed case and there can be any number of white spaces between the two words. The control file can be either full path or basename. In the latter case, the directory \$INCOMING_DIR defined in job configuration file is assumed.

The “header trailer flag” is optional in new argument style. The default is Y. Either header=Y or trailer=Y turn on the “header trailer flag”.

Relative path to jobs directory is deprecated, which is only required when using the old argument style.

control file name is optional in new argument style. If not specified, it is assumed to be the same as calling program name with .ctl extension, in the same directory as the program itself. For example, if the program is bc_load_pending_trades.ksh, then the control file is assumed to be bc_load_pending_trades.ctl.

The loader login name is optional in new argument style. If not specified, it is assumed to be the same as the schema owner.

The schema owner is optional in new argument style. If not specified, it is assumed to be the same as immediate directory containing the caller program, as the caller programs are organized by schemas.

The table name is optional in new argument style. If not specified, it is derived from control file. A line containing "into table *table_name*" not preceded with "—" in the control file is assumed. Mixed case and any number of white spaces are accepted.

Maximum allowable discards (besides header and trailer if present), and maximum allowable errors are optional. They are assumed to be 0 by default.

“move=y|n” option specifies whether the feed file should be moved (archived). The default value Y works for most cases. However, if the feed is used by multiple load programs, it should not be moved until all the load programs finish.

"skip=*num_records*" skips the first *num_records*. It is useful for reload a job which is failed in the middle. Supposed *n* records are already loaded, then you want to skip *n* records if there is no header, and *n+1* records if there is header.

“log_table=y|n” specifies whether to load the log table. The info is collected regardless. The use of log table allows to generate a nice looking report. The precedence is the command line option, job configuration file, and the default which is Y.

If you do use log_table, the creation script is

```
create table jobs_log (
  PROGRAM_NAME      VARCHAR2(200) NOT NULL,
  FEED_NAME         VARCHAR2(200),
  HDR_LABEL         VARCHAR2(200),
  TRL_LABEL         VARCHAR2(200),
  TRLR_RECS_RCVD   NUMBER,
  FILE_RECS_COUNT  NUMBER,
  FILE_RCVD_DATE   DATE,
  MODULE_NAME      VARCHAR2(200),
  LOAD_STATUS      NUMBER,
  RECS_LOADED      NUMBER,
  RECS_REJ         NUMBER,
  RECS_DSC         NUMBER,
  LOGFILE          VARCHAR2(200),
  PROCESS_DATE     DATE          NOT NULL
)
```

“help=y” prints out the man page you are reading.

CONFIGURATION FILE

A sample configuration file is shown below.

```
# BASIC ENVIRONMENT VARIABLES FOR ORACLE
```

```

typeset -x ORACLE_SID=BCCS1
typeset -x ORACLE_HOME=/bcc-bccs1-s/ora01/app/oracle/product/8.1.7
typeset -x PATH=$(/bin/getconf PATH):/usr/local/bin:$ORACLE_HOME/bin
typeset -x TNS_ADMIN=/var/opt/oracle
typeset -x LD_LIBRARY_PATH=

# USERS AND PASSWORDS

idlog=<login>          # user name and
pwdlog=<passwd>        # passwd for "INSERT INTO JOBS_LOG"
<SCHEMA>PASS=<passwd> # <SCHEMA> passwd for process feed and sqlfile
pwd=<passwd>           # common passwd in the absense of <SCHEMA>PASS

# EXIT STATUS CODES

(( SUCCESS                = 0 ))
(( FEED_NOT_EXISTS        = 1 ))
(( FEED_HDR_NOT_EXISTS    = 2 ))
(( FEED_TRL_NOT_EXISTS    = 3 ))
(( FEED_COUNT_MISMATCH    = 4 ))
(( LOAD_COUNT_ZERO        = 5 ))
(( LOAD_COUNT_MISMATCH    = 6 ))
#
(( SQL_PROCESSING_ERROR    = 9 ))
(( ARG_MISMATCH            = 10 ))
(( DIRECTORY_NOT_EXISTS    = 11 ))
(( SQLLOAD_EXECUTION_ERROR = 12 ))
(( EXCEED_MAX_ERRORS       = 13 ))
(( EXCEED_MAX_DISCARDS     = 14 ))

# FEED

INCOMING_DIR=~ftpstars
BACKUP_DIR=~ftpstars/backup
(( DATA_KEEP = -1 )) # days to keep data file (negative number = forever)
(( LOG_KEEP = -1 )) # days to keep log file (negative number = forever)

# NOTIFICATION

JOB_EMAIL="foo@my.com,bar@my.com"
JOB_PAGER="1234567@pager.com,7654321@pager.com"

```

DIRECTORY STRUCTURE

I described below a preferred directory structure. No absolute path is hard coded in this setup, which makes the program very portable.

First, you setup the top level job directory, *whatever_dir*/jobs, which we will refer as *TOP_DIR*.

Secondly, you setup bin directory under *TOP_DIR*, that is where this program **process_feed.ksh** should reside. And its configuration file should be in *TOP_DIR*/etc and should be named “.jobs_env”. It should be read/write by owner only.

Thirdly, the caller programs should be organized by schema. For example,

```
bc_load_pending_tradesctl
bc_load_pending_tradesksh
```

are placed under *TOP_DIR*/trade. **process_feed.ksh** will put the log files under

```
I<TOP_DIR>/trade/bc_load_pending_trades:

log/bc_load_pending_trades/bc_load_pending_trades.02-03_00:48.out
log/bc_load_pending_trades/bc_load_pending_trades.02-03_00:48.log
log/bc_load_pending_trades/bc_load_pending_trades.02-03_00:48.dsc
log/bc_load_pending_trades/bc_load_pending_trades.02-03_00:48.bad
```

The *.out is the unix stdout and stderr, and the *.log is the Sql*Loader log file, and *.dsc, *.bad are Sql*Loader discard and bad files respectively if we have them.

The autosys command line would be:

```
command: I<TOP_DIR>/trade/bc_load_pending_trades.ksh
```

SECURITY

The user program does not handle passwd, *process_feed.ksh* retrieve the passwd for the loader login. The passwd file should be read only by the program, and the program disables trace, and is only modifiable by the owner.

The passwd is not part of sqlldr command arguments and therefore does not show up in ps output.

CVS REPOSITORY

By design, all programs, control files, sql files are under the same top level directory, it is easier to put them under version control system CVS. This avoids the clutter we had on the production system.

Here is an example to commit a change to CVS:

```
$ cvs ci -m "ver 3.0 tested" process_feed.ksh
Checking in process_feed.ksh;
/bcc-bccs1-s/ora01/home/oracle/jobs-repo/jobs/bin/process_feed.ksh,v
+ <-- process_feed.ksh
new revision: 1.6; previous revision: 1.5
done
```

EXAMPLES

New style:

```
$ cat bc_load_pending_trades.ksh

#!/bin/ksh
[[ $0 == /* ]] || exec $(pwd -P)/$0 "$@"
${0%%/jobs*}/jobs/bin/process_feed.ksh myname:=$0 move=n skip=3
```

Old style:

```
$ cat bc_load_pending_trades1.ksh
```

```
#!/bin/ksh
[[ $0 == /* ]] || exec $(pwd -P)/$0 "$@"
${0%*/jobs*}/jobs/bin/process_feed.ksh \
  TR0152TC.FTPTRADE      \
  Y                       \
  trade                  \
  bc_load_pending_trades \
  trade                  \
  trade                  \
  pending_trades        \
  0                       \
  0
```

Please note is there is no hard coded path and other info in new argument style.

SEE ALSO

process_sqlfile.ksh help=y.

AUTHORS

Michael Wang, 2001–2002, <xw73@columbia.edu>.

Chetan Sheth, 2000–11–17.

Carol Sheng, 1999–06–06 – 1999–05–19, 1999–05–24.

Sheck Cho, 05/05/1999.