

# Cron jobs and daemons

Koha is supported by a number of background tasks. These tasks can either be periodically executed tasks (cron jobs) or continuously running tasks called daemons.

A cron job is a Linux command for scheduling a command or script on your server to complete repetitive tasks automatically. Scripts executed as a cron job are typically used to modify files or databases; however, they can perform other tasks that do not modify data on the server, like sending out email notifications.

A daemon is a Linux command that is typically started when the system is booted and runs in the background doing some function. The database used by Koha (either MySQL or MariaDB) is a daemon as is the web server (typically Apache).

Koha has many cron jobs in place that you can enable (search engine indexing, overdue notice generation, data cleanup and more), and a few daemons. This chapter will explain those for you.

## Crontab example

An example of a Koha crontab can be found in `misc/cronjobs/crontab.example`

The example includes sample boilerplate cron job entries for the most commonly-used cron jobs.

## Cron jobs

The locations in the documentation assume a development install where files are found in `misc/` relative to the git root checkout. If you have installed using Debian packages or the standard install from source, you will want to look for files in `/usr/share/koha/bin/`.

Other locations are possible with other installation methods. You can perform a simple find search if they are not located in these directories.

### Note

For anyone with shell access using Debian packages, the following command is an easy way to find files installed by a Debian package:

```
dpkg -L koha-common
```

This provides a comprehensive listing of the files installed by the koha-common package. You can then easily find the file from there.

## Backup

### Daily backup

Script path: `misc/cronjobs/backup.sh`

Does: creates a daily backup of the Koha database.

Frequency suggestion: daily

## Search

## Sitemap

Script path: misc/cronjobs/sitemap.pl

Does: processes all biblio records from a Koha instance and generates sitemap files complying with the protocol as described on <http://sitemaps.org>. The goal of this script is to be able to provide to search engines direct access to biblio records. It avoids leaving search engines browsing Koha's OPAC which would generate a lot of site traffic and server workload.

**Note** A file named sitemapindex.xml is generated. It contains references to multiple sitemap files. Each file contains at most 50,000 URLs and is named sitemapXXXX.xml.

The files must be stored on the Koha OPAC's root directory. In the same directory a robots.txt file with the following contents is required:

```
Sitemap: sitemapindex.xml User-agent: * Disallow: /cgi-bin/
```

## Rebuild index

Script path: misc/migration\_tools/rebuild\_zebra.pl

Does: updates Zebra indexes with recently changed data.

Required by: Zebra

Frequency suggestion: every x minutes, (between 5-15 minutes) depending on performance needs

**Note** On newer Koha installations, this cron job has been replaced by the [koha-indexer daemon](#) which indexes new and modified Koha data every 30 seconds.

## Circulation

### Holds queue

Script path: misc/cronjobs/holds/build\_holds\_queue.pl

Does: updates holds queue report

Required by: [Holds queue report](#)

Frequency suggestion: every 1-4 hours

Description:

- A script that should be run periodically if your library system allows borrowers to place on-shelf holds. This script decides which library should be responsible for fulfilling a given hold request.

It's behavior is controlled by the system preferences [StaticHoldsQueueWeight](#) and [RandomizeHoldsQueueWeight](#).

If you do not want all of your libraries to participate in the on-shelf holds fulfillment process, you should list the libraries that *do* participate in the process here by inputting all the participating library's branchcodes, separated by commas ( e.g. "MPL,CPL,SPL,BML" etc. ).

By default, the holds queue will be generated such that the system will first attempt to hold fulfillment using items already at the pickup library if possible. If there are no items available at the pickup library to fill a hold, build\_holds\_queue.pl will then use the list of libraries defined in StaticHoldsQueueWeight. If RandomizeHoldsQueueWeight is disabled ( which it is by default ), the

script will assign fulfillment requests in the order the branches are placed in the StaticHoldsQueueWeight system preference.

For example, if your system has three libraries, of varying sizes ( small, medium and large ) and you want the burden of holds fulfillment to be on larger libraries before smaller libraries, you would want StaticHoldsQueueWeight to look something like “LRG,MED,SML”.

If you want the burden of holds fulfillment to be spread out equally throughout your library system, simply enable RandomizeHoldsQueueWeight. When this system preference is enabled, the order in which libraries will be requested to fulfill an on-shelf hold will be randomized each time the list is regenerated.

## Expired holds

Script path: misc/cronjobs/holds/cancel\_expired\_holds.pl

Does: cancels holds where the user has set an expiration date. If the library is using the [ExpireReservesMaxPickUpDelay](#) and [ExpireReservesMaxPickUpDelayCharge](#) preferences then this script will also cancel holds that have been sitting on the hold shelf for too long and will (if the library does) charge the patron for not picking up the hold.

It is possible to add a cancellation reason with the `--reason` parameter. Use the cancellation code from the [HOLD\\_CANCELLATION authorized value category](#).

Frequency suggestion: daily

## Unsuspend holds

Script path: misc/cronjobs/holds/auto\_unsuspend\_holds.pl

Does: checks to find holds that should no longer be suspended and removes the suspension if the [AutoResumeSuspendedHolds](#) preference is set to ‘allow’. This puts the patron back in to the queue where they were when the hold was suspended.

Frequency suggestion: daily

## Fines

Script path: misc/cronjobs/fines.pl

Does: calculates and charges (or increments) overdue fines per item to patron accounts. The fine calculation is done using the grace period, fine interval, fine amount and other parameters from the [circulation and fines rules](#).

Required by: [finesMode](#) system preference

Frequency suggestion: nightly

**Note** If the Koha system preference ‘finesMode’ is set to ‘production’, the fines are charged to the patron accounts. If set to ‘test’, the fines are calculated but not applied.

**Note** Fines will not be applied on a holiday.

**PARAMETERS** - `-h|--help`

- `get help message`
- `-l|--log`

- log the output to a file (optional if the -o parameter is given)
- -o|--out
  - output directory for logs (defaults to env or /tmp if the directory does not exist)
- -v|--verbose
  - verbose mode
- -m|--maxdays
  - how many days back of overdue to process
  - this can improve performance by simply the number of records that need to be processed. It can be safe to limit the overdue processed to those under X days overdue given that circulation policy often caps fines at a maximum after a number of days.

## Static fines

Script path: misc/cronjobs/staticfines.pl

Does: charges a single static fine for any/all overdue a patron currently has outstanding. The charge amount is either defined on the command line per borrower category or will use the circulation rules associated with the oldest overdue item the patron has currently checked out (for the first fine period only) Once charged, the fine is static: No new fines will be added until the existing fine is paid off in full.

Frequency suggestion: nightly

**Note** If the Koha system preference 'finesMode' is set to 'production', the fines are charged to the patron accounts. If set to 'test', the fines are calculated but not applied.

**Note** Fines won't be applied on a holiday.

## Batch writeoff charges

Script path: misc/cronjobs/writeoff\_debts.pl

Does: writes off outstanding charges in patron accounts.

## PARAMETERS

### Note

The options to select the debt records to writeoff are cumulative. For example, supplying both --added\_before and --type specifies that the accountline must meet *both* conditions to be selected for writeoff.

### Note

You must use at least one of the filtering options for the script to run. This is to prevent an accidental 'writeoff all' operation.

- -h | --help
  - Displays help message.
- -v | --verbose
  - Verbose output.
- --added-before

- Write off charges added before the specified date.
- Dates should be in ISO format, e.g., 2013-07-19, and can be generated with `date -d '-3 month' --iso-8601`.
- `--added-after`
  - Write off charges added after the specified date.
  - Dates should be in ISO format, e.g., 2013-07-19, and can be generated with `date -d '-3 month' --iso-8601`.

### Version

The `--added-after` parameter was added in Koha version 23.11.

- `--category-code`
  - Write off charges of patrons belonging to the specified [categories](#).
  - Repeatable.

### Version

The `--category-code` parameter was added in Koha version 23.11.

- `--type`
  - Write off charges of the specified type.
  - Accepts a list of [debit type codes](#).
- `--file`
  - Write off charges passed as one `accountlines_id` per line in this file.
  - If other criteria are defined it will only writeoff those in the file that match those criteria.
- `--confirm`
  - This parameter is needed to actually writeoff the charges.
  - Running the script without this parameter will only output which charges would have been written off.

## USAGE EXAMPLES

```
writeoff_debts.pl --added_after 2023-06-20 --confirm
```

Will write off charges added after 2023-06-20.

```
writeoff_debts.pl --added_before `date -d '-3 month' --iso-8601` --category-
```



Will write off charges older than 3 months for patrons in the 'K' category.

## Restrict patrons with fines

### Version

This script was added to Koha in version 23.11.

Script path: misc/cronjobs/debar\_patrons\_with\_fines.pl

Does: Adds a manual restriction to patrons with more than X amount in unpaid charges.

Frequency suggestion: nightly or depending on needs

## PARAMETERS

- -h | --help
  - Displays the help message.
- -a | --amount
  - Minimum amount the patron owes in order to be restricted.
  - Defaults to 0, meaning anyone who owes anything will be restricted.
- -m | --message
  - Message to be added as the restriction comment.
- -f | --messagefile
  - File that contains the message to be added as the restriction comment.
- -e | --expiration
  - Expiration date for the restriction.
- -c | --confirm
  - Use this parameter to confirm the changes.
  - Without this parameter, no patrons will be restricted.
- -v | --verbose
  - Shows which patrons are affected.

## USAGE EXAMPLES

```
debar_patrons_with_fines.pl -a 5 -m "Fines" -v
```

Will show which patrons have more than 5 in unpaid fees, but will not actually restrict them (missing --confirm parameter).

```
debar_patrons_with_fines.pl -a 5 -m "Fines" -e '2024-12-31' -v -c
```

Will restrict patrons who owe more than 5, the restriction will have the comment “Fines” and will expire on 2024-12-31. The script output will also show which patrons were restricted.

## Long overdues

Script path: misc/cronjobs/longoverdue.pl

Does: allows one to specify delays for changing items to different lost statuses, and optionally charge for them using the replacement price listed on the item record.

Frequency suggestion: nightly

### Note

Staff can control some of the parameters for the longoverdue cron job with the [DefaultLongOverdueLostValue](#), [DefaultLongOverdueSkipLostStatuses](#) and [DefaultLongOverdueChargeValue](#) preferences.

## Track total checkouts

Script path: misc/cronjobs/update\_totalissues.pl

Does: updates the biblioitems.totalissues field in the database with the latest tally of checkouts based on historical issue statistics.

Frequency suggestion: nightly

### **Warning**

If the time on your database server does not match the time on your Koha server you will need to take that into account, and probably use the `--since` argument instead of the `--interval` argument for incremental updating.

### **Note**

This cronjob can be used if there is a performance concern. Otherwise, use the UpdateTotalIssuesOnCirc System preference.

## **Generate patron file for offline circulation**

Script path: misc/cronjobs/create\_koc\_db.pl

Does: generates the borrowers.db file for use with the [Koha offline circulation](#) tool

Frequency suggestion: weekly

## **Automatic renewal**

Script path: misc/cronjobs/automatic\_renewals.pl

Does: renews items if you're allowing automatic renewal with your [circulation and fines rules](#).

Frequency suggestion: nightly

### **Important**

To run this properly, you must use the `--confirm` parameter, otherwise it will only run in test mode

### **PARAMETERS -h|--help**

- get help message
- `--send-notices`
  - sends the AUTO\_RENEWALS notice to patrons if the automatic renewal has been done
- `-v|--verbose`
  - verbose mode
- `-c|--confirm`
  - without this parameter no changes will be made, the script will run in test mode
  - without this parameter, the script will default to verbose mode as well

## **Automatic checkin**

Script path: misc/cronjobs/automatic\_checkin.pl

Does: automatically checks in items after the loan period. This is set at the [item type level](#).

Frequency suggestion: nightly

## Note

Optionally, holds can be filled automatically when items are checked in with this script. This option is enabled with the [AutomaticCheckinAutoFill](#) system preference.

## Recalls

### Expire recalls

Script path: misc/cronjobs/recalls/expire\_recalls.pl

Does: automatically marks as expired recalls that

- have been requested, but have not been fulfilled and are past their expiry date
- recalls that have been awaiting pickup longer than the pickup period in the [circulation rules](#) or the period set in the [RecallsMaxPickUpDelay](#) system preference

Frequency suggestion: nightly

### Overdue recalls

Script path: misc/cronjobs/recalls/overdue\_recalls.pl

Does: sets a recalled item as overdue if it hasn't been returned by the adjusted due date

Frequency suggestion: nightly

## Patrons

### Batch delete patrons

Script path: misc/cronjobs/delete\_patrons.pl

Does: deletes patron records in batch based on date not borrowed since, expired before, last seen, category code, or library branch.

## Note

Dates should be in ISO format, e.g., 2013-07-19, and can be generated with date -d '-3 month' "+%Y-%m-%d".

## Important

The options to select the patron records to delete are **cumulative**. For example, supplying both `–expired_before` and `–library` specifies that patron records must meet *both* conditions to be selected for deletion.

## PARAMETERS

- `–not_borrowed_since` Delete patrons who have not borrowed since this date.

## Warning

Patrons who have all their old checkouts anonymized will have an empty circulation history and be deleted if this option is used. Anonymization can happen because the



patron has borrowers.privacy = 2, through [cronjobs doing anonymization](#) or by the [patron choosing to anonymize their history in the OPAC](#).

- `--expired_before` Delete patrons with an account expired before this date.
- `--last_seen` Delete patrons who have not been connected since this date. The system preference [TrackLastPatronActivity](#) must be enabled to use this option.
- `--category_code` Delete patrons who have this category code.
- `--library` Delete patrons in this library.
- `-c|--confirm` This flag must be provided in order for the script to actually delete patron records. If it is not supplied, the script will only report on the patron records it would have deleted.
- `-v|--verbose` Verbose mode.

## Anonymize patron data

Script path: `misc/cronjobs/batch_anonymise.pl`

Does: removes borrowernumbers from circulation history so that the stats are kept, but the patron information is removed for privacy reasons.

## Update patron categories

Script path: `misc/cronjobs/update_patrons_category.pl`

Does: Updates the patron category of patrons matching the given criteria to another specified patron category. This can be used to convert child patrons from a child patron category to an adult patron category when they reach the upper age limit defined in the [patron category](#).

This script replaces the `j2a.pl` script.

Frequency suggestion: nightly

## DESCRIPTION

This script is designed to update patrons from one category to another using the criteria specified using command line arguments.

## PARAMETERS

- `--too_old` Update, if patron is over the upper age limit of their current [patron category](#).
- `--too_young` Update, if patron is below the minimum age limit of their [patron category](#).
- `--fo=X|--fineover=X` Update, if the total fine amount on the patron account is over X.
- `--fu=X|--fineunder=X` Update, if the total fine amount on the patron account is below X.
- `--rb=date|regbefore=date` Update, if the registration date of the patron is before the given date.
- `--ra=date|regafter=date` Update, if the registration date of the patron is after the given date.
- `-d --field name=value` Update, if the given condition is met. `<name>` has to be replaced by a column name of the borrowers table. The condition is met, if the the content of the field equals `<value>`.
- `--where <conditions>` Update, if the SQL `<where>` clause is met.
- `-v|--verbose` Verbose mode: Without this flag only fatal errors are reported.

- -c|--confirm Commits the changes to the database. No changes will be made unless this argument is added to the command.
- -b|--branch <branchcode> Update, if the home branch of the patron matches the <branchcode> given.
- -f|--form <categorycode> Update, if the patron currently has this patron category.
- -t|--to <categorycode> Update the patrons matching the criteria to this patron category.

## USAGE EXAMPLES

“update\_patrons\_category.pl”

“update\_patrons\_category.pl” -b=<branchcode> -f=<categorycode> -t=<categorycode> -c”  
(Processes a single branch, and updates the patron categories from category to category)

“update\_patrons\_category.pl” -f=<categorycode> -t=<categorycode> -v” (Processes all branches, shows all messages, and reports the patrons who would be affected. Takes no action on the database)

## Update patrons’ messaging preferences

Script path: misc/maintenance/borrowers-force-messaging-defaults.pl

Does: updates patrons’ messaging preferences to the default values set in the [patron categories](#).

Messaging preferences defaults are automatically set when [adding a new patron](#) or when [importing patrons with the patron import tool](#). However, if you import patrons directly in the database, these messaging preferences will not be set.

There is no suggested frequency. This is a tool to be used when needed, although if you regularly import patrons directly in the database (through a third-party, for example), you can add it to your crontab.

## DESCRIPTION

If the [EnhancedMessagingPreferences](#) system preference is enabled after borrowers have been created in the DB, those borrowers won’t have messaging transport preferences default values as defined for their borrower category. So you would have to modify each borrower one by one if you would like to send them ‘Hold Filled’ notice for example.

This script creates/overwrites messaging preferences for all borrowers and sets them to default values defined for the category they belong to (unless you use the options -not-expired or -no-overwrite to update a subset).

## PARAMETERS

- --help
  - Display help message.
- --doit
  - Update the patrons. The script will not update patrons’ messaging preferences without this option. It will only list the patrons who would have been updated.
- --not-expired
  - Only update patrons who are still active (whose files are not yet expired).
- --no-overwrite

- Only update patrons without any messaging preferences. This option will skip patrons who have already set their preferences.
- `--category`
  - Only update patrons from the specified category.

### Warning

This option **cannot** be repeated.

For example:

```
borrowers-force-messaging-defaults.pl --doit --category PT --category B
```

will only update patrons from category B (the last category specified).

- `--library`
  - Will only update patrons whose home library matches the given branchcode.

### Version

The `--library` parameter was added in Koha version 23.11.

- `--message-name`
  - Will only update preferences for the specific message.
  - The list of values can be found in `installer/data/mysql/mandatory/sample_notices_message_attributes.sql`, in `message_attributes.message_name` in the database, or in the [notices and slips tool](#).

### Version

The `--message-name` parameter was added in Koha version 23.11.

- `--since`
  - Only update patrons enrolled since the specified date.

### Note

This option can use specific or relative dates.

For example:

```
borrowers-force-messaging-defaults.pl --doit --since "2022-07-12"
```

will only update patrons enrolled since July 12, 2022.

And:

```
borrowers-force-messaging-defaults.pl --doit --since `date -d "1 day ago" '+%Y-%m-%d'
```



will only update patrons enrolled since yesterday.

## USAGE EXAMPLES

```
borrowers-force-messaging-defaults.pl --doit
```

Updates all patrons to give them the default messaging preferences values from their respective category.

```
borrowers-force-messaging-defaults.pl --doit --not-expired
```

Updates all patrons whose memberships are not expired to give them the default messaging preferences values from their respective category.


```
borrowers-force-messaging-defaults.pl --doit --category PT
```

Updates all patrons in the PT category to give them the default messaging preferences for that category.

```
borrowers-force-messaging-defaults.pl --doit --no-overwrite --since "2022-03-01"
```

Updates patrons who do not have any messaging preferences set and who are enrolled since March 1st, 2022.

```
borrowers-force-messaging-defaults.pl --doit --no-overwrite --since `date -d "1 day ago`
```



Updates patrons who do not have any messaging preferences set and who are enrolled since yesterday.

```
borrowers-force-messaging-defaults.pl --doit --library CPL
```

Updates patrons whose home library is CPL.

```
borrowers-force-messaging-defaults.pl --doit --message-name Item_due
```

Updates preferences for the “Item due” message only.

## Notices

### Message queue

Script path: misc/cronjobs/process\_message\_queue.pl

Does: processes the message queue to send outgoing emails and SMS messages to patrons. Messages are queued in the message queue by other scripts, such as [advance\\_notices.pl](#), [overdue\\_notices.pl](#), and [holds\\_reminder.pl](#).

### Note

Requires that [EnhancedMessagingPreferences](#) be set to ‘Allow’.

Frequency suggestion: 1-4 hours

## DESCRIPTION

This script processes the message queue in the message\_queue database table. It sends out the messages in that queue and marks them appropriately to indicate success or failure. It is recommended that you run this regularly from cron, especially if you are using the [advance\\_notices.pl](#) script.

## PARAMETERS

- -u | --username
  - Username of the mail account used to send the notices.
- -p | --password
  - Password of mail account used to send the notices.
- -t | --type
  - If supplied, only processes this type of message. Possible values are
    - email
    - sms
  - Repeatable
- -c | --code
  - If supplied, only processes messages with this [letter code](#).
  - Repeatable.
- -l | --limit
  - The maximum number of messages to process for this run.
- -m | --method
  - Authentication method required by SMTP server (see perldoc Sendmail.pm for supported authentication types).
- -h | --help
  - Help message.
- -v | --verbose
  - Provides verbose output to STDOUT.
- -w | --where
  - Filter messages to send with additional conditions in a where clause.

## Advanced notice

Script path: misc/cronjobs/advance\_notices.pl

Does: prepares “pre-due” notices and “item due” notices for patrons who request them prepares notices for patrons for items just due or coming due soon. requires [EnhancedMessagingPreferences](#) to be on

Frequency suggestion: nightly

### Note

This script does not actually send the notices. It queues them in the [message\\_queue](#) for later

## Overdue notice

Script path: misc/cronjobs/overdue\_notices.pl

Does: prepares messages to alert patrons of overdue messages (both via email and print)

Frequency suggestion: nightly

## DESCRIPTION

This script creates and queues the overdue notices according to the parameters set in the [overdue notice/status triggers tool](#).

## PARAMETERS

- `-n` | `-nomail`
  - Do not send any email. Overdue notices that would have been sent to the patrons or to the admin are printed to standard out. CSV data (if the `-csv` flag is set) is written to standard out or to any CSV filename given.
- `-max <days>`
  - Maximum days overdue to deal with.
  - Items overdue since longer than max days are assumed to be handled somewhere else, probably the [longoverdues](#) script. They are therefore ignored by this script. No notices are sent for them, and they are not added to any CSV files.
  - Defaults to 90 days.
- `-library <branchcode>`
  - Only deal with overdue from this library.
  - Use the value in the `branches.branchcode` table.
  - This parameter is repeatable, to process overdue for a group of libraries.
- `-csv <filename>`
  - Produces a CSV file.
  - If the `-n` (no mail) flag is set, this CSV data is sent to standard out or to a filename if provided. Otherwise, only overdue that could not be emailed are sent in CSV format to the admin.
- `-html <directory>`
  - Output html to a file in the given directory.
  - If a patron does not have an email address or if the `-n` (no mail) flag is set, an HTML file is generated in the specified directory. This can be downloaded or further processed by library staff.
  - The file will be called `notices-YYYY-MM-DD.html` and placed in the directory specified.
- `-text <directory>`
  - Output plain text to a file in the given directory.
  - If a patron does not have an email address or if the `-n` (no mail) flag is set, a text file is generated in the specified directory. This can be downloaded or further processed by library staff.
  - The file will be called `notices-YYYY-MM-DD.txt` and placed in the directory specified.
- `-itemscontent <list of fields>`
  - Item information in templates.
  - Takes a comma separated list of fields that get substituted into templates in places of the `<<items.content>>` placeholder.
  - Defaults to `due date,title,barcode,author`
  - Other possible values come from fields in the `biblio`, `items` and `issues` tables.
- `-borcat <categorycode>`
  - Prepare only overdue notices for specified patron categories.
  - This parameter is repeatable, to include several patron categories.
  - Use the value in `categories.categorycode`.
- `-borcatout <categorycode>`

- Do not prepare overdue notices for specified patron categories.
  - This parameter is repeatable, to exclude several patron categories.
  - Use the value in categories.categorycode.
- t | –triggered
  - This option causes a notice to be generated if and only if an item is overdue by the number of days defined in the [overdue notice trigger](#).
  - By default, a notice is sent each time the script runs, which is suitable for less frequent run cron script, but requires syncing notice triggers with the cron schedule to ensure proper behavior.
  - Add the –triggered option for daily cron, at the risk of no notice being generated if the cron fails to run on time.
- –test
  - This option makes the script run in test mode.
  - In test mode, the script won't make any changes on the DB. This is useful for debugging configuration.
- –list-all
  - By default, <<items.content>> lists only those items that fall in the range of the currently processing notice.
  - Choose –list-all to include all overdue items in the list (limited by the –max setting).
- –date <yyyy-mm-dd>
  - Emulate overdues run for this date.
- –email <email\_type>
  - Specify the type of email that will be used.
  - Can be 'email', 'emailpro' or 'B\_email'.
  - This parameter is repeatable.
- –frombranch
  - Organize and send overdue notices by home library (item-homebranch) or checkout library (item-issuebranch).
  - Defaults to item-issuebranch.

## Note

This option is only used if the [OverdueNoticeFrom](#) system preference is set to 'command-line option'.

## USAGE EXAMPLES

“overdue\_notices.pl”

(All libraries are processed individually, and notices are prepared for all patrons with overdue items for whom we have email addresses. Messages for those patrons for whom we have no email address are sent in a single attachment to the library administrator's email address, or to the address in the [KohaAdminEmailAddress](#) system preference.)

“overdue\_notices.pl -n –csv /tmp/overdues.csv”

(Sends no email and populates /tmp/overdues.csv with information about all overdue items.)

“overdue\_notices.pl –library MAIN max 14

(Prepares notices of overdue in the last 2 weeks for the MAIN library.)

## Note

This script does not actually send the notices. It queues them in the [message\\_queue](#) to be sent later or generates the HTML for printing.

## Note

See also:

The [misc/cronjobs/advance\\_notices.pl](#) script allows you to send messages to patrons in advance of their items becoming due, or to alert them of items that have just become due.

The [misc/cronjobs/process\\_message\\_queue.pl](#) script sends the emails.

## Holds reminder

Script path: misc/cronjobs/holds/holds\_reminder.pl

Does: prepares reminder messages to be sent to patrons with waiting holds.

[EnhancedMessagingPreferences](#) must be set to ‘Allow’, and patrons must have requested to have this notice (either through the [Messaging tab](#) in their online account in the OPAC, if [EnhancedMessagingPreferencesOPAC](#) is set to ‘Show’, or in their [messaging preferences](#) in the staff interface).

Frequency suggestion: nightly

## PARAMETERS

- -c | –confirm
  - Confirm flag, no email will be generated if this parameter is not set
- -date <YYYY-MM-DD>
  - Send notices as would have been sent on a specific date
- -days <number of days>
  - Number of days the hold has been waiting
  - If this parameter is not set, a notice will be sent to all patrons with waiting holds
  - Optional parameter
- -holidays
  - Use the calendar exclude holidays from waiting days
- -lettercode <lettercode>
  - Code of the [predefined notice](#) to use
  - Optional parameter, the default is HOLD\_REMINDER
- -library <branchcode>



- Only deal with holds from this library
- This flag is repeatable, to select notices for a group of libraries
- -mtt <message\_transport\_type>
  - Type of messages to send (email, sms, print)
    - 'email' and 'sms' will fall back to 'print' if the patron does not have an email address/phone number
  - The default is to use the patrons' [messaging preferences](#) for the 'Hold reminder' notice
  - Passing this parameter will force send even if the patron has not chosen to receive hold reminder notices
  - This can be repeated to send various notices
- -t | -triggered
  - Include only holds <days> days waiting, and not longer
  - If this is not set, the script will send messages for all holds waiting for equal to or more than <days> days
  - This option is useful if the cron is being run daily to avoid spamming the patrons
  - Optional parameter
- -v
  - Verbose
  - Without this flag set, only fatal errors are reported.
  - If verbose is set but not confirm, a list of notices that would have been sent to the patrons are printed to standard out
- -help
  - Brief help message
- -man
  - Full documentation

## EXAMPLES

The following are examples of this script:

::

```
holds_reminder.pl -library MAIN -days 14
```

prepares notices of holds waiting for 2 weeks for the MAIN library

::

```
holds_reminder.pl -lettercode LATE_HOLDS -library MAIN -days 14
```

prepares notices of holds waiting for 2 weeks for the MAIN library, using the 'LATE\_HOLDS' notice template

## Print overdues

Script path: misc/cronjobs/printoverdues.sh

Does: generates PDF files from HTML files in directories and prints them

## Print hold notices

Script path: misc/cronjobs/gather\_print\_notices.pl

Does: looks through the message queue for hold notices that didn't go through because the patron didn't have an email address and generates a print notice

Frequency suggestion: nightly

## Talking Tech

To learn more about setting up this third party product view the [Talking Tech](#) chapter.

### Sending notices file

Script path: misc/cronjobs/thirdparty/TalkingTech\_itiva\_outbound.pl

Does: generates Spec C outbound notifications file for Talking Tech i-tiva phone notification system.

Required by: [TalkingTechItivaPhoneNotification](#)

Frequency suggestion: nightly

### Receiving notices file

Script path: misc/cronjobs/thirdparty/TalkingTech\_itiva\_inbound.pl

Does: processes received results files for Talking Tech i-tiva phone notification system.

Required by: [TalkingTechItivaPhoneNotification](#)

Frequency suggestion: nightly

## Notify patrons of expiration

Script path: misc/cronjobs/membership\_expiry.pl

Does: sends messages to warn patrons of their card expiration to the [message\\_queue](#) cron. Can optionally renew patron accounts as well.

Requires: [MembershipExpiryDaysNotice](#)

Frequency: nightly

### PARAMETERS

- --man
  - Prints the manual page and exits.
- --help
  - Prints a brief help message and exits.
- -v
  - Verbose.
  - Without this flag set, only fatal errors are reported.
- -n

- Do not send any email. Membership expiry notices that would have been sent to the patrons are printed to standard out.
- -c
  - Confirm flag: Add this option. The script will only print a usage statement otherwise.
- -branch
  - Optional branchcode to restrict the cronjob to that branch.
- -before
  - Optional parameter to extend the selection with a number of days BEFORE the date set by the [MembershipExpiryDaysNotice](#) system preference.
- -after
  - Optional parameter to extend the selection with a number of days AFTER the date set by the [MembershipExpiryDaysNotice](#) system preference.
  - For example, --before 100 --after 100 will notify patrons who have accounts expiring within a date range of 100 days before and 100 days after the [MembershipExpiryDaysNotice](#) system preference.
- -where
  - Use this option to specify a condition. Add “me” (alias) followed by the column name from the borrowers table.
  - For example:
    - --where="me.categorycode != 'YA'" will notify patrons from categories other than ‘YA’
    - --where="me.categorycode = 'S'" will notify patrons from the category ‘S’ only
    - --where 'me.lastseen IS NOT NULL' will only notify patrons who have been seen.
- -letter
  - Optional parameter to use another notice than the default: MEMBERSHIP\_EXPIRY
- -letter\_renew
  - Optional parameter to use another renewal notice than the default: MEMBERSHIP\_RENEWED

### Version

The -letter\_renew parameter was added in Koha version 23.11.

- -active

- Followed by a number of months.
- Optional parameter to include active patrons only (active within passed number of months).
- This parameter needs the [TrackLastPatronActivityTriggers](#) system preference.
- Cannot be used with -inactive below, the two parameters are mutually exclusive

### **Version**

The -active parameter was added in Koha version 23.11.

- -inactive

- Followed by a number of months.
- Optional parameter to include inactive patrons only (inactive since passed number of months).
- This parameter needs the [TrackLastPatronActivityTriggers](#) system preference.
- Cannot be used with -active above, the two parameters are mutually exclusive

### **Version**

The -inactive parameter was added in Koha version 23.11.

- -renew

- Optional parameter to automatically renew patrons instead of sending them an expiry notice.
- They will be informed by a membership renewal notice (the default MEMBERSHIP\_RENEWED or a custom one specified by -letter\_renew)

### **Version**

The -renew parameter was added in Koha version 23.11.

## **USAGE EXAMPLES**

```
membership_expiry.pl -c
```

Will generate MEMBERSHIP\_EXPIRY notices for patrons whose membership expires in the number of days set in [MembershipExpiryDaysNotice](#).

```
membership_expiry.pl -c -renew
```

Will renew patrons whose membership expires in the number of days set in [MembershipExpiryDaysNotice](#), and generate MEMBERSHIP\_RENEWED notices for them.

```
membership_expiry.pl -c -renew -letter_renew PATRON_RENEWAL
```

Will renew patrons whose membership expires in the number of days set in [MembershipExpiryDaysNotice](#), and generate the custom “PATRON\_RENEWAL” notices for them. A “PATRON\_RENEWAL” notice would have to have been created in the [notices and slips tool](#) beforehand.

```
membership\_expiry.pl -c -before 30
```

Will generate MEMBERSHIP\_EXPIRY notices for patrons whose membership expires 30 days before the number of days set in [MembershipExpiryDaysNotice](#).

```
membership_expiry.pl -c -renew -active 3
```

Will renew patrons whose membership expires in the number of days set in [MembershipExpiryDaysNotice](#), and who have been active in the last three months (“activity” is determined by the [TrackLastPatronActivityTriggers](#) system preference), and generate MEMBERSHIP\_RENEWED notices for them.

```
membership_expiry.pl -c -inactive 6 -letter INACTIVE_PATRON
```

Will generate the custom “INACTIVE\_PATRON” notices for patrons whose membership expires in the number of days set in [MembershipExpiryDaysNotice](#), and who have been inactive for the last six months (“activity” is determined by the [TrackLastPatronActivityTriggers](#) system preference). An “INACTIVE\_PATRON” notice would have to have been created in the [notices and slips tool](#) beforehand.

## In processing/book cart

Script path: misc/cronjobs/cart\_to\_shelf.pl

Does: updates all items with a location of CART to the item’s permanent location.

Required by: [NewItemsDefaultLocation](#), [UpdateItemLocationOnCheckin](#), and [UpdateItemLocationOnCheckout](#) system preferences.

Frequency suggestion: hourly

## Catalog

### Import webservice batch

Script path: misc/cronjobs/import\_webservice\_batch.pl

Does: processes import batch queues of type ‘webservice’. Batches can also be processed through the UI.

#### Note

This script is used for OCLC Connexion

### Batch item deletion

Script path: misc/cronjobs/delete\_items.pl

Does: generates a query against the items database and deletes the items matching the criteria specified in the command line arguments. A lightweight batch deletion tool for items, suitable for running in a cron job.

## PARAMETERS

- --help

- Prints a brief help message.
- --man
  - Prints the manual, with examples.
- --verbose
  - Prints the “WHERE” clause generated by the collected --where arguments, as well as items affected to Standard Out.
  - The item information printed is
    - itemnumber
    - barcode
    - title
- --where
  - The following argument must be a syntactically valid SQL statement which is part of the WHERE clause querying the items table.
  - Repeatable. If there are multiple --where parameters, they will be joined by AND.
- --commit
  - No items will be deleted unless this parameter is present.

## USAGE EXAMPLES

```
delete_items.pl --where "items.withdrawn != 0" --where "items.withdrawn_on <
```

This will delete items where the withdrawn status is not zero AND where the withdrawn date is older than 13 months ago.

```
delete_items.pl --where "itemlost >= '1'" --where "itemlost <='4'" --where "
```

This will delete items where the lost status is between 1 and 4 (inclusively) AND that were lost before 2014-04-28.

## Check URL quick

Script path: misc/cronjobs/check-url-quick.pl

### Note

This script replaces the deprecated check-url.pl script

Does: checks URLs from biblio records; scans all URLs found by default in 856\$u of bibliographic records and displays if resources are available or not.

## PARAMETERS

- --host=http://default.tld Server host used when URL doesn't have one, i.e. doesn't begin with 'http:'. For example, if --host=mylib.com, then when 856\$u contains 'img/image.jpg', the url checked is: <http://www.mylib.com/image.jpg>.
- --tags Tags containing URLs in \$u subfields. If not provided, 856 tag is checked. Multiple tags can be specified, for example:  
  
check-url-quick.pl --tags 310 410 856
- --verbose|v Outputs both successful and failed URLs.

- `-html` Formats output in HTML. The result can be redirected to a file accessible by http. This way, it's possible to link directly to the bibliographic record in edit mode. With this parameter `-host-intranet` is required.
- `-host-intranet=http://koha-pro.tld` Server host used to link to bibliographic record editing page in Koha intranet interface.
- `-timeout=10` Timeout for fetching URLs. By default 10 seconds.
- `-maxconn=1000` Number of simultaneous HTTP requests. By default 200 connections.

## Delete records via leader

Script path: `misc/cronjobs/delete_records_via_leader.pl`

Does: attempts to delete any MARC records where the leader character 5 equals 'd'.

### PARAMETERS

- `-c|--confirm` Script will do nothing without this parameter
- `-v|--verbose` Verbose mode
- `-t|--test` Test mode, does not delete records. Test mode cannot determine if a record/item will be deleted successfully, it will only tell you what records and items the script will attempt to delete.
- `-i|--delete-items` Try deleting items before deleting record. Records with items cannot be deleted.

## Update authorities

Script path: `misc/cronjobs/merge_authorities.pl`

Does: updates biblio data with changes to authorities records

### Note

The name of this script is misleading. This script does not merge authorities together it instead merges authority data with linked bib records. Edits to authority records will be applied to bibliographic records that use that authority when this script is run.

Required by: [AuthorityMergeLimit](#) system preference

Frequency suggestion: nightly

## Serials update

Script path: `misc/cronjobs/serialsUpdate.pl`

Does: checks if there is a "late" issue on active subscriptions, and if there is, the script will set it as late, and add the next one as expected.

Frequency suggestion: nightly

## Automatic item update

Script path: `misc/cronjobs/automatic_item_modification_by_age.pl`

Does: updates items based on the list of rules set forth in the [Automatic item modifications by age](#) tool

Required by: [Automatic item modifications by age](#)

Frequency suggestions: nightly

## Stock rotation

Script path: misc/cronjobs/stockrotation.pl

Does: moves items from one [stock rotation](#) stage to the next, if they are available for processing.

Each library will receive a report with “items of interest” for them for today’s rota checks. Each item there will be an item that should, according to Koha, be located on the shelves of that branch, and which should be picked up and checked in.

### Note

The email sent is based on the SR\_SLIP template. It can be customized in the [Notices and slips tool](#).

The item will either:

- have been placed in transit to their new stage library;
- have been placed in transit to be returned to their current stage library;
- have just been added to a rota and will already be at the correct library;

Upon check-in,

- items that need to be transferred elsewhere will be put in transit and a message will pop up requesting the item be sent to their new branch.
- items that are already at the correct library will be checked in and no message will pop up.

Required by: [Stock rotation](#) tool

Frequency suggestion: nightly

## PARAMETERS

- -a|--admin-email
  - An address to which email reports should also be sent
  - This is an additional email address to which all email reports will be sent, in addition to sending them to branch email addresses.
- -b|--branchcode
  - Select branch to report on for ‘email’ reports (default: all)
  - If the ‘email’ report is selected, you can use the ‘branchcode’ parameter to specify which branch’s report you would like to see.
  - The default is ‘all’.
- -x|--execute
  - Actually perform stockrotation housekeeping
  - By default, this script merely reports on the current status of the stock rotation subsystem. In order to actually place items in transit, the script must be run with the ‘execute’ argument.
- -r|--report
  - Select either ‘full’ or ‘email’
  - The ‘report’ argument allows you to select the type of report that will be emitted.
  - The default is ‘full’.
  - If the ‘email’ report is selected, you can use the ‘branchcode’ parameter to specify which branch’s report you would like to see.
- -S|--Send-all



- Send email reports even if the report body is empty
  - This argument causes even reports with an empty body to be sent.
- -s|--send-email
  - Send reports by email
  - This argument causes the script to send reports by email.
- -h|--help
  - Display the help message

## OPAC

### RSS feeds

Script path: misc/cronjobs/rss/rss.pl

Does: produces an RSS XML document for any SQL query (not used for search results RSS feed). [Learn more](#).

Frequency suggestion: hourly

### Authorities browser

Script path: misc/cronjobs/build\_browser\_and\_cloud.pl

Does: generates content for authorities browse in OPAC

Required by: [OpacBrowser](#) system preference

#### Important

This preference and cron job should only be used on French systems.

### Subject/author clouds

Script path: misc/cronjobs/cloud-kw.pl

Does: generates HTML keywords clouds from Koha Zebra indexes. misc/cronjobs/cloud-sample.conf has a sample of how this script operates.

Frequency: This is the type of script you can run once a month or so, the content generated isn't going to change very much over time.

## System administration

### Services throttle

Script path: misc/cronjobs/services\_throttle.pl

Does: resets the xISBN services throttle

Frequency suggestion: nightly

### Clean up database

Script path: misc/cronjobs/cleanup\_database.pl

Does: truncates the sessions table, cleans out old zebraqueue entries, action logs and staged MARC files.

See <http://schema.koha-community.org/> for the Koha database schema.

## PARAMETERS - --del-exp-selfreg

- Delete expired self registration accounts (accounts that haven't been upgraded from the 'temporary' category) from the borrowers table.
- The number of days for expiry is set in the [PatronSelfRegistrationExpireTemporaryAccountsDelay](#) system preference.
- The temporary patron category is set in the [PatronSelfRegistrationDefaultCategory](#) system preference.
- --del-unv-selfreg DAYS
  - Delete all unverified self registrations in borrower\_modifications older than DAYS.
- --deleted-catalog DAYS
  - Purge bibliographic records deleted more than DAYS days ago from tables deletedbiblio, deletedbiblio\_metadata, deletedbiblioitems and deleteditems.
- --deleted-patrons DAYS
  - Purge patrons deleted more than DAYS days ago from the deletedborrowers table.
- --fees DAYS
  - Purge entries in the accountlines table older than DAYS days, where the amountoutstanding is 0 or NULL.
  - In the case of --fees, DAYS must be greater than or equal to 1.
- -h|--help
  - Get help message
- --import DAYS
  - Purge entries from import tables older than DAYS days.
  - This includes import\_batches, import\_biblios, import\_items, import\_record\_matches and import\_records.
  - In import\_batches, the batches that are the result of Z39.50 searches are removed with the --z3950 parameter.
  - Defaults to 60 days if no days specified.
- --list-invites DAYS
  - Purge (unaccepted) list share invites from the virtualshelfshares table older than DAYS days
  - If this parameter is passed without a value, the value in the [PurgeListShareInvitesOlderThan](#) system preference is used.
  - Defaults to 14 days if no days specified and [PurgeListShareInvitesOlderThan](#) is empty.
- --logs DAYS
  - Purge entries from the action\_logs table older than DAYS days.
  - Defaults to 180 days if no number of days specified.
- --log-modules
  - Specify which action\_log modules to purge.

- This option is repeatable.
  - See [action logs modules and actions](#) for the module names
- --preserve-logs
  - Specify which action\_log modules to exclude.
  - This option is repeatable.
  - See [action logs modules and actions](#) for the module names
- -m|--mail DAYS
  - Purge entries from the message\_queue table that are older than DAYS days.
  - Defaults to 30 days if no days specified.
- --merged
  - Purge completed entries from the need\_merge\_authorities table.
- --oauth-tokens
  - Delete expired OAuth2 tokens
- --old-issues DAYS
  - Purge checkouts returned more than DAYS days ago from the old\_issues table.
- --old-reserves DAYS
  - Purge holds more than DAYS old from the old\_reserves table.
- --restrictions DAYS
  - Purge [patrons restrictions](#) from the borrower\_debarments table expired since more than DAYS days.
  - Defaults to 30 days if no days specified.
- --return-claims
  - Purge all [resolved return claims](#) older than the number of days specified in the system preference [CleanUpDatabaseReturnClaims](#).
- --all-restrictions
  - Purge all expired patrons restrictions from the borrower\_debarments table.
- --searchhistory DAYS
  - Purge entries from the search\_history table older than DAYS days.
  - Defaults to 30 days if no days specified
- --sessions
  - Purge the sessions table.
  - If you use this while users are logged into Koha, they will have to reconnect.
- --sessdays DAYS
  - Purge only sessions older than DAYS days.
- --statistics DAYS

- Purge entries from the statistics tables that are more than DAYS days old.
- `--temp-uploads`
  - Delete temporary uploads from the `uploaded_files` table older than the number of days specified in the [UploadPurgeTemporaryFilesDays](#) system preference.
- `--temp-uploads-days DAYS`
  - Override the [UploadPurgeTemporaryFilesDays](#) system preference value.
- `--transfers DAYS`
  - Purge transfers completed more than DAYS days ago from the `branchtransfers` table.
- `--unique-holidays DAYS`
  - Delete all unique holidays from the `special_holidays` table older than DAYS
- `--uploads-missing FLAG`
  - Delete upload records for missing files when FLAG is true, count them otherwise
- `-v|--verbose`
  - verbose mode
- `--zebraqueue DAYS`
  - Purge completed zebraqueue entries older than DAYS days.
  - Defaults to 30 days if no days specified.
- `--z3950`
  - Purge records from import tables that are the result of Z39.50 searches.
  - To purge all other import information, see the `--import` parameter above.

## Share usage stats

Script path: `misc/cronjobs/share_usage_with_koha_community.pl`

Does: sends your info to the [Hea website](http://hea.koha-community.org/) [http://hea.koha-community.org/] if you're sharing information via the [UsageStats](#) feature

Frequency: monthly

## Search for data inconsistencies

Script path: `misc/maintenance/search_for_data_inconsistencies.pl`

Does: reveals problems in data, such as

- items without home or holding [library](#)
- items without [item type](#) or with invalid [item type](#)
- bibliographic records without [item type](#) or with invalid [item type](#)
- bibliographic records with invalid MARCXML
- bibliographic records without biblionumber or biblioitemnumber in MARCXML
- bibliographic records without title
- invalid values in fields where the [framework](#) limits to an [authorized value category](#).

- authority records without [authority\\_type](#) or with invalid [authority\\_type](#)
- patrons who are too old or too young for their [category](#).

Some of these issues can cause problems in circulation or catalog search, so it's important that they be corrected.

There is no suggested frequency. This is a tool to be used when needed.

## Acquisitions

### Clean up old suggestions

Script path: misc/cronjobs/purge\_suggestions.pl

Does: removes old suggestions from the suggestion management area.

#### PARAMETERS

- help|?

Show help message

- days

Define the age of suggestions to be deleted, based on 'managed on' date

#### Note

The system preference [PurgeSuggestionsOlderThan](#) can also be used to define the number of days used in the script. If using the system preference, don't use the 'days' parameter.

#### Note

The number of days is based on the 'managed on' date of the suggestion.

- confirm

This parameter is mandatory for the script to run.

### Email suggestions to process

Script path: misc/cronjobs/notice\_unprocessed\_suggestions.pl

Does: generates a notice to the fund owner that there are suggestions in need of processing

### EDI message processing

Script path: misc/cronjobs/edi\_cron.pl

Does: sends and received EDI messages

Frequency: Every 15 minutes

### Remove temporary EDI files

Script path: misc/cronjobs/remove\_temporary\_edifiles.pl

Does: removes temporary EDI files that are older than 5 days

## E-resource management (ERM)

### Harvesting cron job

Script path: /misc/cronjobs/erm\_run\_harvester.pl

Does: this script will run the SUSHI harvesting for any Active usage data providers set up in the [E-resource management module](#).

Frequency: it is recommended you set it up to run at a regular interval (e.g., every month since providers usually produce statistics data monthly).

#### PARAMETERS

- --help or -h
  - Prints a help message
- --begin-date
  - Set the start date for the harvest in yyyy-mm-dd format (e.g.: '2023-08-21')
- --end-date
  - Set the end date for the harvest in yyyy-mm-dd format (e.g.: '2023-08-21')
- --dry-run
  - Produces a run report, without actually doing anything permanent
- --debug
  - Prints additional debugging information during the run

#### USAGE EXAMPLE

```
erm_run_harvester.pl --begin-date 2023-06-21 --debug
```

Will run the SUSHI harvest for Active usage data providers for the period starting 21 June 2023 to the present date (or to the date for which data is available). Additional debugging information about the way the harvest has run will be displayed.

## Reports

### Run report

Script path: misc/cronjobs/runreport.pl

Does: runs pre-existing [saved reports](#), optionally sends the results by email.

#### PARAMETERS

- -h | --help
  - Displays help message
- -m | --man
  - Displays full documentation
  - Same as --help --verbose

- -v | --verbose
  - Verbose output
  - Without this parameter, only fatal errors are reported
- --format=s
  - Selects output format
  - Possible values:
    - text
    - html
    - csv
    - tsv
  - At the moment, 'text' and 'tsv' both produce tab-separated output
  - Defaults to 'text'
- -e | --email
  - Send the output by email (implied by --to or --from)
- --send\_empty
  - Send the email even if the report returns no results

## Version

The --send\_empty parameter was added in Koha version 23.11.

- -a | --attachment
  - Attach the report as a file
  - Cannot be used with html format
- --username
  - Username to pass to the SMTP server for authentication
- --password
  - Password to pass to the SMTP server for authentication
- --method
  - The type of authentication, i.e. LOGIN, DIGEST-MD5, etc.
- --to=s
  - E-mail address to which to send report results
  - If --email is specified, but --to is not, the address in [KohaAdminEmailAddress](#) is used
- --from=s
  - E-mail address from which to send report
  - If --email is specified, but --from is not, the address in [KohaAdminEmailAddress](#) is used
- --subject=s

- Subject for the e-mail
- --param=s
  - Pass value for the [runtime parameter](#)
  - Repeatable
  - Provide one --param per runtime parameter requested for the report. Report parameters are not combined as they are on the staff side, so you may need to repeat parameters.
- --separator=s
  - Separator character
  - Only for csv format
  - Defaults to comma
- --quote=s
  - Quote character
  - Only for csv format
  - Defaults to double quote
  - Empty string is allowed

## Version

The --quote parameter was added in Koha version 23.11.

- --store-results
  - Store the result of the report into the saved\_reports database table.
  - To access the results, go to [Reports > Guided reports > Saved report](#).
- --csv-header
  - Add column names as first line of csv output

## ARGUMENTS

- reportID
  - Report ID Number from saved\_sql.id
  - Multiple ID's may be specified
  - Mandatory

## USAGE EXAMPLES

```
runreport.pl 1
```

Will output results from report 1 in the terminal (STDOUT).

```
runreport.pl 1 5
```

Will output results from reports 1 and 5 in the terminal (STDOUT).

```
runreport.pl --format html --to admin@myDNSname.org 1
```

Will send results from report 1 to [admin@myDNSname.org](mailto:admin@myDNSname.org) in HTML format.

```
runreport.pl --format html --to admin@myDNSname.org --param CPL --param FICTION 1
```



Will send results from report 1 to [admin@myDNSname.org](mailto:admin@myDNSname.org) in HTML format. 'CPL' will be passed to the first runtime parameter, and 'FICTION' will be passed to the second runtime parameter.

```
runreport.pl --store-results 1
```

Will save the report results in the saved\_reports database table, and they will be available from the staff interface in [Reports > Guided reports > Saved report](#).

## Social data

### Get report social data

Script path: misc/cronjobs/social\_data/get\_report\_social\_data.pl

Does: downloads data from Babelthèque to add to OPAC records

Frequency suggestion: nightly

### Update social data

Script path: misc/cronjobs/social\_data/update\_social\_data.pl

Does: updates OPAC records with Babelthèque social data

## Daemons

Daemons are continuously running tasks that help support Koha operation. Your database and web-server are run as daemons.

### Automatically started daemons

Newer versions of Koha start two different daemons for most koha instances:

- zebra - this is the index server
- koha-indexer - this daemon updates the index server with new and modified data (biblios and authorities)

These daemons are started by the script /etc/init.d/koha-common.

### Zebra indexer daemon

Script path: /usr/sbin/koha-indexer (invoked from /etc/init.d/koha-common)

The koha-indexer script invokes rebuild\_zebra.pl in daemon mode. In this mode, the script will run continuously and check the database for new or modified data every 30 seconds. New or modified records are then sent to Zebra for indexing, which only takes a second or so. The advantage of this approach is a search system which is much more responsive to changes, compared to the [cron job approach](#).

### Other daemons

These are not started automatically by Koha. You could run them manually, or create your own systemd unit to keep them running.

### OCLC Connexion import daemon

Script path: misc/bin/connexion\_import\_daemon.pl

Does: Listens for requests from OCLC Connexion clients and is compliant with the OCLC Gateway specification.

See [Setting up the OCLC Connexion Daemon](#) for more details.

## Deprecated scripts

These should not be run without modification:

Script path: misc/cronjobs/update\_items.pl

Script path:misc/cronjobs/smsoverdues.pl

Script path:misc/cronjobs/notifyMailsOp.pl

Script path:misc/cronjobs/reservefix.pl

Script path:misc/cronjobs/zebraqueue\_start.pl