NotifyMe Database Description

The NotifyMe option is to send emails that summarize search results against exact keywords, whenever available. NotifyMe stores all requests in an SQLite database, which can be further analyzed by the SPARC team to understand the search pattern and get more insights on the demand. For example, the SPARC team can find out the most common keywords searched with no existing matches and decide on actions to fulfill such needs.

Admins can set the database name in the properties.ini file, The current default name is "notifyme.db". The database will be automatically created during the first call of the NotifyMe module. If for any reason, admins need to drop and recreate the database you can call the create_notifyme_db function from Notifyme_utils and set the clean option to True.

The database consists of 4 main tables:

1. **NEW_REGISTER**

Column Name	Description
entry_id	A unique identifier for email requests (*autoincrement integer*)
email	The user entered email *(validated at the front end)*
register_date	The system date and time corresponding to the record creation, which is the time of the request initialization
keywords	The user-entered search keywords
status	All records in this table should show a 'New' status
last_modified	In case the record get modified for any reason, this record is representing the last modification date and time for the corresponding record

2. **WAITING_LIST**

Column Name	Description
entry_id	A unique identifier for email requests (*automatically created in the new_register table*)
email	The user-entered email *(validated at the front end)*
register_date	date and time of the request
keywords	The user-entered search keywords
status	Request current status. Can be either 'New' if no hits still matching, or 'Failed' if the last attempt to send an email failed, the detailed error will be stored in Failed_emails
last_modified	The date and time for the last modification of this record
hits	The current number of matching hits exists against the search keywords. This should be '0' for records remaining in 'New' status, and an actual number for 'Failed' records
failed_reqid	This is the reference for the latest corresponding Failed_emails record, showing the exact error that explains why this request failed

3. ARCHIEVED_LIST

Column Name	Description
entry_id	A unique identifier for email requests (*automatically created in the new_register table*)
email	The user-entered email *(validated at the front end)*
register_date	date and time of the request
keywords	The user-entered search keywords
status	Request current status. Can be either: - 'Sent': for successfully sent emails - 'Duplicate': case the request identified earlier to be a duplicate request/entry - 'Failed': case the email request raises an error consistently for more than one month.
last_modified	The date and time for the last modification of this record. Should be corresponding to the time the email is sent if the status is 'Sent'.
hits	The number of matching hits sent against the search keywords. In case of failed requests, it will be the number of hits that exists at the time the record moved from the waiting_list table to here
failed_reqid	This is the reference for the latest corresponding Failed_emails record, showing the exact error that explains why this request failed

4. FAILED_EMAILS

Column Name	Description
failed_reqid	A unique identifier for an error recorded against an email request (*autoincrement integer*)
corresponding_entry_id	The corresponding entry_id of the email request will normally find either in the waiting_list table or the archieved_list table if we fail to send an email for more than a month
register_date	date and time of the request
error_message	The detailed error message leading for email posting failure
error_date	The system date and time when the error was triggered