

Fields to be selected in the tables (**SELECT** clause of the MySQL query).

For a **work** identified by its ID (**\$ID**): we will display the value of the field

access_point_language.concatenation with the following jonction:

ON access_point_language.access_point_id = access_point.id

ON access_point.work_id = \$ID

For a **manifestation** identified by its ID (**\$ID**): we will display the concatenation of two fields:

CONCAT_WS(' . - ', manifestation_concatenation.zone1,

manifestation_concatenation.zone4)

ON manifestation_concatenation.manifestation_id = \$ID

Sort of a work: **BY access_point_language.sorting**

Sort of a manifestation: **BY manifestation_proper_title.content_sort**

The jonction between the table **manifestation_concatenation** and the table **manifestation_proper_title** is a little tricky, because we have to use an intermediate table (**manifestation_title**) according to this jonctions:

ON manifestation_concatenation.manifestation_id =

manifestation_title.manifestation_id

ON manifestation_title.id =

manifestation_proper_title.manifestation_title_id

Fields to be searched in the tables (**WHERE** clause of the MySQL query).

According to the type of search (combo-box *Where?*), the requested string should be search into:

<i>Where?</i>	<i>Name of the field</i>	
Titles	<code>title.item</code>	(in any of those 5fields)
	<code>manifestation_proper_title.content</code>	
	<code>manifestation_paralle_title.content</code>	
	<code>manifestation_variant_title.content</code>	
	<code>manifestation_variant_parallel_title.content</code>	
Authorities	<code>authority_form.item</code>	
ISBN-ISSN	<code>manifestation_identifier.item</code>	
Call Number	<code>item_identifier.content</code>	
Classification		
Anywhere	<code>access_point.item</code>	(in any of those 2 fields)
	<code>manifestation_concatenation.item</code>	

Rules for searching a requested string.

Let name the requested string **\$query**.

Remove spaces at the beginning and at the end of **\$query**:

\$query=trim(\$query)

\$query should count at least 3 characters. A message is returned to the user if not.

Do not use **strlen(\$query)** which counts the **bytes**,

use instead **mb_strlen(\$query, "UTF-8")** which counts the **characters**.

Parts of the query between double-quotes (" inserted by the user in his query) refer to parts not to be cut. The part between " should be kept unchanged. Replace (temporarily) the spaces of those unbreakable parts by sign **\$** and remove the double-quotes (").

Then escape the characters in order to apply the queries to MySQL by using function:

real_escape_string(\$query).

Split the query into different « words » but without cutting these parts between double-quotes. Just split into separate words according to the spaces and replace again **\$** by spaces in each of those words.

For instance, if `$query='book "Islamic heritage" new'`, then `$query` will be split at the end of this process into 3 words:

```
$query_words[0]='book'
$query_words[1]='Islamic heritage'
$query_words[3]='new'
```

For each word (each element of `$query_words`), adjust some typographical issues.

1. For Latin languages.

If the user asks for 'oe', the search should find 'oe' and 'œ' (U+0153).

If the user asks for 'ae', the search should find 'ae' and 'æ' (U+00E6).

2. For the Arabic.

If the user asks for 'ي', the search should find 'ي' and 'ى'.

For the management of the definite article (ال), some complex rules apply.

If the search is done by words (“some of these words” or “all these words”), the article is removed before searching the query in the database.

For instance, if the user asks for السلام, he will find سلام and السلام (you have to search سلام, by removing the article ال from the query string).

But it's not true for this unique word: الله! If the user asks for this word, he should find الله and اللهم but of course not لهم (therefore you have to search لله and not له).

For the *alif* (ا) (but not for the *alif* of the definite article!), if the user asks for 'ا', the search should find 'ا' and 'إ' and 'آ' and 'أ'.

It means that if the word begins with an *alif* (ا) followed by something *else than lam* (ل), the search should find the words beginning with 'ا' or 'إ' or 'آ' or 'أ' and the rest of the query.

If the word begins with the article followed by an *alif* (it begins with 'الا'), the search should find the words beginning with 'ا' or 'إ' or 'آ' or 'أ' and the rest of the query (if the search is done by word), and the words beginning with 'الا' or 'الإ' or 'الآ' or 'الأ' and the rest of the query.

3. For the Persian.

If the user asks for 'ک', the search should find 'ک' and 'گی' (U+06AF).

If the user asks for 'گ', the search should find 'ک' and 'گی' (U+06AF).

If the user asks for 'ء', the search should find 'ء' and 'ة' (U+06C0).

If the user asks for 'ة', the search should find 'ء' and 'ة' (U+06C0).

If the user asks for 'ب', the search should find 'ب' and 'پ' (U+067E).

If the user asks for 'پ', the search should find 'ب' and 'پ' (U+067E).

If the user asks for 'ج', the search should find 'ج' and 'چ' (U+0686).

If the user asks for 'چ', the search should find 'ج' and 'چ' (U+0686).

The combo-box “How?” will precise the way the search is made in the tables.

How?	SQL search clause
contains these words	(for each word w:) LIKE '%w%'
exactly like	= 'q'
begins with	LIKE 'q%'

The Annex gives the previous versions of those manipulations in Al-Kindi OPAC v. 3.