Xapian::QueryParser Syntax

This document describes the query syntax supported by the Xapian Search Plugin in ZORA. The syntax is designed to be similar to other web based search engines, so that users familiar with them don't have to learn a whole new syntax.

Operators

AND

expression AND expression matches documents which are matched by both of the subexpressions.

OR

expression OR expression matches documents which are matched by either of the subexpressions.

NOT

expression NOT expression matches documents which are matched by only the first subexpression. This can also be written as expression AND NOT expression. If FLAG_PURE_NOT is enabled, then

NOT expression will match documents which don't match the subexpression.

XOR

expression XOR expression matches documents which are matched by one or other of the subexpressions, but not both. XOR is probably a bit esoteric.

Bracketed expressions

You can control the precedence of the boolean operators using brackets. In the query one OR two AND three the AND takes precedence, so this is the same as one OR (two AND three). You can override the precedence using (one OR two) AND three.

The default precedence from highest to lowest is:

- +, - (equal)
- AND, NOT (equal)
- XOR
- OR
'+' and '-'

A group of terms with some marked with + and - will match documents containing all of the +
terms, but none of the - terms. Terms not marked with + or - contribute towards the document
rankings. You can also use + and - on phrases and on bracketed expressions.

**NEAR**

one NEAR two NEAR three matches documents containing those words within 10 words of
each other. You can set the threshold to $n$ by using NEAR/$n$ like so: one NEAR/6 two.

**ADJ**

ADJ is like NEAR but only matches if the words appear in the same order as in the query. So
one ADJ two ADJ three matches documents containing those three words in that order and
within 10 words of each other. You can set the threshold to $n$ by using ADJ/$n$ like so: one
ADJ/6 two.

**Phrase searches**

A phrase surrounded with double quotes (""") matches documents containing that exact phrase.
Hyphenated words are also treated as phrases, as are cases such as filenames and email
addresses (e.g. /etc/passwd or president@whitehouse.gov).

**Searching within a probabilistic field**

If the database has been indexed with prefixes on probabilistic terms from certain fields, you
can set up a prefix map so that the user can search within those fields. For example
author:dickens title:shop might find documents by dickens with shop in the title. You
can also specify a prefix on a quoted phrase (e.g. author:"charles dickens") or on a
bracketed subexpression (e.g. title:(mice men)).

**Searching for proper names**

If a query term is entered with a capitalised first letter, then it will be searched for unstemmed.

**Synonyms**

The QueryParser can be configured to support synonyms, which can either be used when
explicitly specified (using the syntax ~term) or implicitly (synonyms will be used for all
terms or groups of terms for which they have been specified).

**Wildcards**

The QueryParser supports using a trailing '*' wildcard, which matches any number of trailing
characters, so wildc* would match wildcard, wildcarded, wildcards, wildcat, wildcats, etc.
This feature is disabled by default - pass Xapian::QueryParser::FLAG_WILDCARD in the
flags argument of Xapian::QueryParser::parse_query(query_string, flags) to enable it, and tell the QueryParser which database to expand wildcards from using the QueryParser::set_database(database) method.

You can limit the number of terms a wildcard will expand to by calling Xapian::QueryParser::set_max_wildcard_expansion(). If a wildcard expands to more terms than that number, an exception will be thrown. The exception may be thrown by the QueryParser, or later when Enquire handles the query. The default is not to limit the expansion.