Front-End Query Language

Query Language Used in the Testbed of the Stanford Digital Libraries Project

The testbed of the Stanford Digital Library Project supports a Boolean query language for users to search information over the underlying services, such as Knight-Ridder's DIALOG, DEC's AltaVista, and WebCrawler. To address the problem of non-uniform query languages used by the underlying services, our approach is to allow users to compose Boolean queries in the front-end language, which is to be described in this documentation. The front-end queries are then transformed by the Query Translator according to the capabilities and syntax of the target services.

Overview

The testbed supports a Boolean query language. A query in this language specifies conditions that must be satisfied by the matching documents. Only documents that satisfy the query are returned, in no particular order.

A query consists of predicates (simple sub-queries) connected with Boolean operators., e.g., TI : color (W) printer AND PY >= 1996. A predicate (e.g., TI : color (W) printer) specifies conditions to be matched with a particular portion of documents (e.g., the attribute TI).

The language does not define any specific attributes for search. (Attributes used in the examples of this documentation are for illustration only.) The actually supported attributes and how they can be used to formulate queries depend on the specific search interface you are using. However, the language does define several common attribute types that will be referred to by search interfaces to define how their supported attributes can be used in queries.

In the following, we first describe Boolean operators and predicates. Then, we introduce attribute types, which define how attributes of certain types can be used to formulate queries.
Boolean Operators

The front-end query language is Boolean, so you can use binary Boolean operators AND, NOT, and OR to connect predicates (or simple sub-queries). The following examples illustrate the usage of Boolean operators:

"color printer" AND scanner
   The operator AND ensures that both conditions are present in the resulting documents.
"color printer" NOT scanner
   The operator NOT is used to match documents satisfying the first but not the second condition. In other words, NOT actually means AND-NOT.
"color printer" OR scanner
   The operator OR ensures that at least one condition is present in the resulting documents.

The operators AND and NOT have higher precedence than OR. That is, in a complex query, AND and NOT are evaluated before OR. However, to enforce the order of evaluation, you can use parentheses to group conditions. For example,

color and printer or scanner
(color and printer) or scanner
   The above two queries are equivalent. They return documents containing both color and printer, together with documents containing scanner.
color and (printer or scanner)
   This query uses parentheses to group the OR-ed conditions. That is, it matches documents containing color and, in addition, in the same document, either printer or scanner.

Predicates (Simple Sub-Queries)

Queries are constructed from predicates by connecting them with Boolean operators. Conceptually, documents consist of attributes (also called fields) such as title, author, and text. A predicate specifies a condition to be matched with a particular attribute. Syntactically, a predicate is of the form

attributeName comparisonOperator searchExpression

The attributeName refers to the name of an attribute, e.g., AU (author), TI (title), and PY (publication year)– they should be defined by the search interface you are using. The searchExpression specifies the search terms, e.g., "color printer". Finally, the comparisonOperator specifies how the attribute value
Both the attributeName and comparisonOperator are optional:

- If the attributeName is not supplied, it refers to searches on the default attributes as appropriate for the target service.
- If the comparisonOperator is omitted, the default is the operator colon ":" , which means to match documents whose attribute attributeName contains searchExpression.

"color printer"

Retrieve documents with "color printer" contained in (the default attributes of) the document

TI : color
TI color

Both of the above two predicates retrieve documents whose TI attribute contains color.

TI = "the latex companion"

Retrieve documents whose TI attribute equals to "the latex companion".

PY>1996

Retrieve documents with publication year greater than 1996.

The supported search attributes and how they are searched are independent of the query language; they should be defined separately by the search interface you are using. The attributes used in the examples of this documentation are for illustration only.

The search interface defines the search attributes it supports, documentation of the attributes, and their attribute types. From this attribute definition, users can choose the attributes to search on and refer to their attribute types to formulate proper queries. To illustrate, the following table is a sample attribute definition.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Documentation</th>
<th>Attribute Type</th>
<th>Query Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>TI</td>
<td>Title of the document.</td>
<td>ShortText</td>
<td>TI : color(W)printer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TI = &quot;unix in a nutshell&quot;</td>
</tr>
<tr>
<td>TX</td>
<td>Full text of the document.</td>
<td>LongText</td>
<td>TX : (color AND printer)</td>
</tr>
<tr>
<td>PY</td>
<td>Publication year.</td>
<td>Number</td>
<td>PY &gt;= 1996</td>
</tr>
</tbody>
</table>

Table 1: Sample attribute definition.

The table shows that three attributes are supported: TI, TX, and PY. Furthermore, the attribute types refer to how these attributes can be used in queries. For example, because PY is of Number type, it can
be searched with the "\( \geq \)" operator, which is not allowed for attribute TI.

---

**Attribute Types**

The attribute type of an attribute restricts how it can be used in queries. For instance, you cannot use comparison operator "\( \geq \)" for an attribute of type ShortText such as TI (title). Three query types are currently defined: LongText, ShortText, and Number.

---

**LongText**

Attributes of attribute type LongText can be searched using the operator ":" to match documents whose specified attribute contains the search expression. Note that the operator ":" is the default and thus can be omitted.

Search expressions for LongText type attributes can be words (e.g., color) or phrases (e.g., "color printer") connected with the proximity operators, or the Boolean operators. Words can be truncated using the "*" symbol to match all words of the same prefixes. For instance, cat* will match any words starting with "cat". Similarly, a phrase can also be truncated, e.g., "unix in *" will match any phrase starting with "unix in".

In addition, words can also be Stemmed with the symbol "!"-- meaning that it is supposed to match any words with the same "root". For instance, computer! will match the words computer as well as computation, computing, and so on.

There are two kinds of proximity operators: (nW) and (nN), where n is a positive integer. The expression A (nW) B specifies that the term A must precede B by no more than n words. Notice that (0W) can be written as (W). If the order of terms does not matter, operator (nN) and (N) can be used instead.

The following examples illustrate predicates on the TX attribute, assuming it is of type LongText.

**TX : scanner**
- Match documents with the word scanner contained in the attribute TX.

**TX : "color printer"**
- Match documents with the phrase "color printer" contained in the attribute TX.

**TX : scanner (5N) "color printer"**
- Match documents whose TX attribute contains both scanner and "color printer", within 5 words
distance to each other.

**TX : color (W) laser (W) printer**

Match document whose **TX** attribute contains the words **color laser printer** appearing next to each other and in this order.

**TX : (scanner AND color (10W) printer)**

Match documents whose **TX** attribute contains **scanner**, and **color** preceding **printer** by no more than 10 words. Notice that parentheses are needed because operator **AND** binds less tightly than ":". This query is equivalent to **TX : scanner AND TX : color (10W) printer**.

---

**ShortText**

For attributes of type **ShortText**, you can use the "=" (equals) operator, in addition to the ":" operator. The ":" operator can be used with a ShortText attribute in the same way as illustrated for **LongText** attributes. For instance, assuming **TI** is a ShortText attribute,

**TI : unix (2W) nutshell**

Match documents with the **TI** attribute containing the word **unix** preceding **nutshell** by no more than 2 words.

The "=" operator specifies that the attribute is to equal the search expression exactly. The search expression is a phrase, possibly truncated.

**TI = "unix in a nutshell"**

Match documents with the **TI** attribute equal to "unix in a nutshell".

**TI = "unix in *"**

Match documents with the **TI** attribute starting with "unix in", e.g., "unix in a nutshell", "unix in seven days"

---

**Number**

You can search **Number** type attributes with the typical relational operators: =, <, >, >=, <=. The search expression is simply a number.

Assuming the **PY** attribute is of type Number, the following examples illustrate the usage:

**PY = 1996**

Match documents with **PY** attribute equal to 1996.
PY >= 1997

Match documents with PY attribute equal or greater than 1997.

Last update: Jun 25, 1997