```
       All other components are mandatory.

       Expressions can also be used to denote calculations that should be
       performed relative to "NOW" to determine the value, ie...

             NOW/HOUR
                ... Round to the start of the current hour
             NOW-1DAY
                ... Exactly 1 day prior to now
             NOW/DAY+6MONTHS+3DAYS
                ... 6 months and 3 days in the future from the start of
                    the current day

       Consult the DateField javadocs for more information.
    -->
  <fieldType name="date" class="solr.DateField" sortMissingLast="true" omitNorms="true"/>
  <!-- solr.TextField allows the specification of custom text analyzers
       specified as a tokenizer and a list of token filters. Different
       analyzers may be specified for indexing and querying.

       The optional positionIncrementGap puts space between multiple fields of
       this type on the same document, with the purpose of preventing false phrase
       matching across fields.

       For more info on customizing your analyzer chain, please see
       http://wiki.apache.org/solr/AnalyzersTokenizersTokenFilters
    -->
  <!-- One can also specify an existing Analyzer class that has a
       default constructor via the class attribute on the analyzer element
  <fieldType name="text_greek" class="solr.TextField">
    <analyzer class="org.apache.lucene.analysis.el.GreekAnalyzer"/>
  </fieldType>
  -->
  <!-- A text field that only splits on whitespace for exact matching of words -->
  <fieldType name="text_fgs" class="solr.TextField" positionIncrementGap="100">
    <analyzer>
      <tokenizer class="solr.StandardTokenizerFactory"/>
      <filter class="solr.LowerCaseFilterFactory"/>
    </analyzer>
  </fieldType>
  <!-- A text field that only splits on whitespace for exact matching of words -->
  <fieldType name="text_ws" class="solr.TextField" positionIncrementGap="100">
    <analyzer>
      <tokenizer class="solr.WhitespaceTokenizerFactory"/>
    </analyzer>
  </fieldType>
  <!-- A text field that uses WordDelimiterFilter to enable splitting and matching of
       words on case-change, alpha numeric boundaries, and non-alphanumeric chars,
       so that a query of "wifi" or "wi fi" could match a document containing "Wi-Fi".
       Synonyms and stopwords are customized by external files, and stemming is enabled.
       Duplicate tokens at the same position (which may result from Stemmed Synonyms or
       WordDelim parts) are removed.
       -->
  <fieldType name="text" class="solr.TextField" positionIncrementGap="100">
    <analyzer type="index">
      <tokenizer class="solr.WhitespaceTokenizerFactory"/>
      <!-- in this example, we will only use synonyms at query time
      <filter class="solr.SynonymFilterFactory" synonyms="index_synonyms.txt" ignoreCase="true"
expand="false"/>
      -->
      <filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt"/>
      <filter class="solr.WordDelimiterFilterFactory" generateWordParts="1" generateNumberParts="1"
catenateWords="1" catenateNumbers="1" catenateAll="0"/>
      <filter class="solr.LowerCaseFilterFactory"/>
      <filter class="solr.EnglishPorterFilterFactory" protected="protwords.txt"/>
      <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
    </analyzer>
    <analyzer type="query">
      <tokenizer class="solr.WhitespaceTokenizerFactory"/>
      <filter class="solr.SynonymFilterFactory" synonyms="synonyms.txt" ignoreCase="true" expand="true"/>
      <filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt"/>
      <filter class="solr.WordDelimiterFilterFactory" generateWordParts="1" generateNumberParts="1"
catenateWords="0" catenateNumbers="0" catenateAll="0"/>
      <filter class="solr.LowerCaseFilterFactory"/>
      <filter class="solr.EnglishPorterFilterFactory" protected="protwords.txt"/>
      <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
    </analyzer>
  </fieldType>
  <!-- Less flexible matching, but less false matches.  Probably not ideal for product names,
       but may be good for SKUs.  Can insert dashes in the wrong place and still match. -->
  <fieldType name="textTight" class="solr.TextField" positionIncrementGap="100">
    <analyzer>
      <tokenizer class="solr.WhitespaceTokenizerFactory"/>
      <filter class="solr.SynonymFilterFactory" synonyms="synonyms.txt" ignoreCase="true" expand="false"/>
      <filter class="solr.StopFilterFactory" ignoreCase="true" words="stopwords.txt"/>
      <filter class="solr.WordDelimiterFilterFactory" generateWordParts="0" generateNumberParts="0"
catenateWords="1" catenateNumbers="1" catenateAll="0"/>
      <filter class="solr.LowerCaseFilterFactory"/>
      <filter class="solr.EnglishPorterFilterFactory" protected="protwords.txt"/>
      <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
    </analyzer>
  </fieldType>
  <!-- This is an example of using the KeywordTokenizer along
       With various TokenFilterFactories to produce a sortable field
       that does not include some properties of the source text
    -->
```