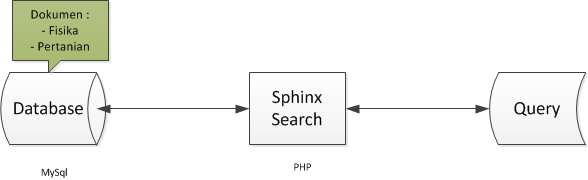
**Sphinx Search**

****

**Cara Kerja Sphinx Search**

Beberapa faktor yang mempengaruhi pembobotan pada sphinx search 1) classic statistical BM25 dan 2) Sphinx specific phrase proximity.

## BM25

BM25 merupakan sebuah metode perangkingan yang digunakan oleh search engine untuk merankingkan dokumen sesuai dengan kerelevanannya dengan query yang diterima. BM25 merupakan teknik perankingan yang didesign oleh Stephen E. Robertson, BM merupakan singkatan dari Best Match (Kecocokan terbaik).

Proses perangkingan BM25 berdasarkan jumlah query yang muncul di dokumen. Sehingga term-frequency dan juga inverse dokumen frequency digunakan dalam penghitungan BM25. Term Frequency (tf) adalah penentuan bobot term pada suatu dokumen berdasarkan jumlah kemunculannya dalam dokumen tersebut. sehingga semakin besar jumlah kemunculan suatu term (tf tinggi) dalam dokumen, semakin besar pula bobotnya dalam dokumen atau akan memberikan nilai kesesuian yang semakin besar. Inverse Document Frequency (idf) adalah pengurangan dominansi term yang sering muncul di berbagai dokumen. Hal ini diperlukan karena term yang banyak muncul di berbagai dokumen, dapat dianggap sebagai term umum (common term) sehingga tidak penting nilainya. Sebaliknya faktor kejarangmunculan kata (term scarcity) dalam koleksi dokumen harus diperhatikan dalam pemberian bobot.



Persamaan tersebut menjelaskan bahwa merupakan term frequency pada sebuah dokumen, IDFmerupakan inversedari jumlah dokumen yang memuat kata kueri, k1 merupakan parameter bebas dengan nilai yang digunakan yaitu k1=1.2. Rumus untuk menghitung IDF ditunjukkan pada Persamaan sebagai berikut:



N merupakan banyaknya koleksi dokumen, dan t merupakan jumlah dokumen yang memuat kata t. Perhitungan yang digunakan untuk peringkat dokumen terhadap kueri menggunakan pembobotan BM25 yang sudah disediakan Sphinxdengan algoritme:

Contoh pengaplikasian algoritma BM25

BM25 = 0

foreach ( keyword in matching\_keywords )

{

n = total\_matching\_documents ( keyword )

N = total\_documents\_in\_collection

k1 = 1.2

TF = current\_document\_occurrence\_count ( keyword )

IDF = log((N-n+1)/n) / log(1+N)

BM25 = BM25 + TF\*IDF/(TF+k1)

}

*// normalize to 0..1 range*

BM25 = 0.5 + BM25 / ( 2\*num\_keywords ( query ) )

Keterangan:

N : Total dokumen dalam korpus

n : Total dokumen yang memiliki

kata kueri

TF : Frekuensi termt pada dokumen

IDF : Invertedindeks dokumen

K1 : Parameter positif (1.2).

**Phrase proximity**

Frase faktor kedekatan, sebaliknya, tidak peduli tentang frekuensi kata kunci sama sekali dan menyumbang disposisi saling kata kunci permintaan dalam dokumen. Alih-alih frekuensi kata kunci yang digunakan untuk BM25, Sphinx menganalisis posisi kunci di segala bidang dan menghitung nilai kedekatan frase sebagai terpanjang umum sub-urutan (SKB) panjang antara permintaan dan dokumen. Pada dasarnya, per-bidang frase kedekatan adalah jumlah kata kunci yang terjadi dalam dokumen dalam urutan yang sama persis seperti yang mereka lakukan dalam query. Berikut pergi beberapa contoh:

1) query = satu dua tiga, field = satu and dua tiga

field\_phrase\_weight = 2 (karena 2-panjang keyword "two three" yang cocok)

2) query = satu dua tiga, field = satu and dua and tiga

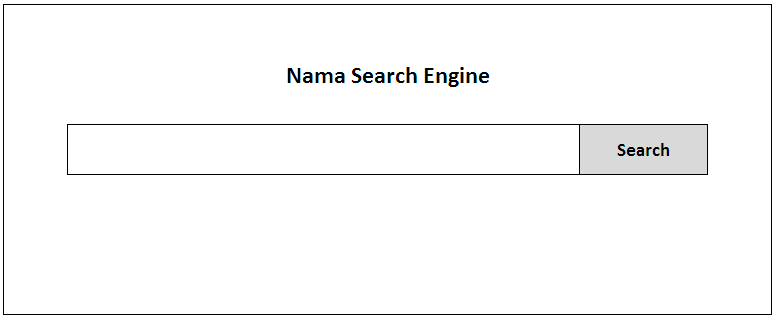
field\_phrase\_weight = 1 (karena hanya satu kata yang cocok)

3) query = satu dua tiga, field = tida ada yang cocok

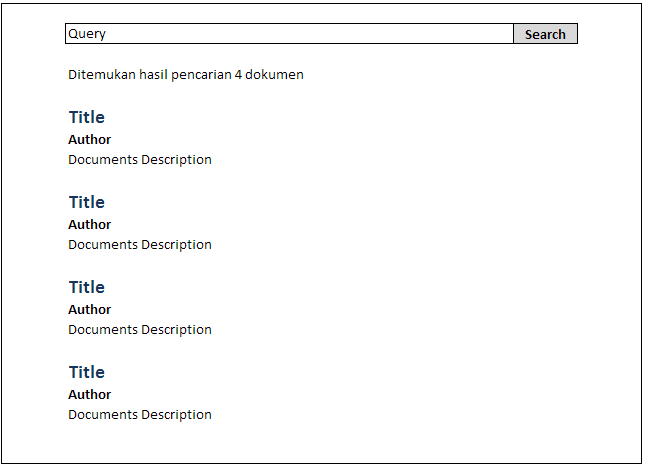
field\_phrase\_weight = 0

**Cara Membuat Aplikasi Search dengan PHP dan MySql**

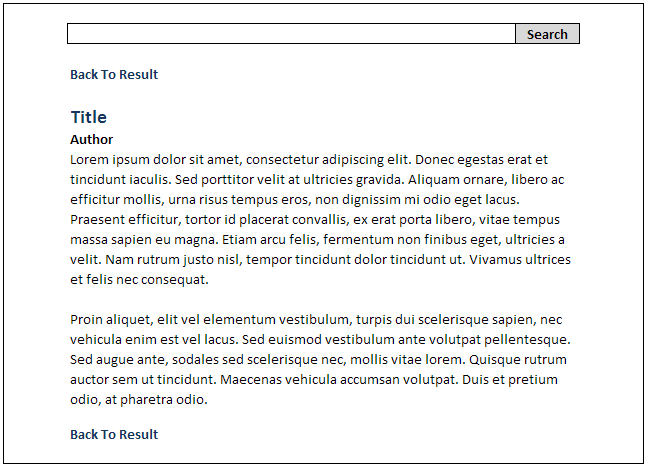
**User Interface**



Halaman Utama Pencarian



Halaman Hasil Pencarian



Halaman Detail