Volltextsuche

Wie kriegen wir eine effiziente und schnell Volltextsuche hin?

Beispiel: Jemand sucht nach allen Einträge mit dem Ausdruck

Folgende Möglichkeiten der Suche gibt es:

1. LIKE-Suche: LIKE mit führendem Wildcard, also z.B. LIKE '%%'

wohl die einfachste Variante aber doch sehr prozessor- und speicherlastig.

2. Teilstring-Index: Vorherige Indizierung bestimmter bzw. möglichst vieler (sinnvoller) Teilstrings.

Damit meine ich bereits die Zerlegung eines Eintrages in alle (oder alle sinnvollen) Bestandteile:

> ...

oder so ähnlich (z.B. mit Mecab-basiertem Tokenizing), jedenfalls wird es dann einen sehr umfangreichen Index geben. Die Frage ist dann nur noch, wie man auf dieser Menge am schnellsten Suchen kann. Entweder mit einer normalen DB oder mit einer SearchEngine wie Lucene.

2A: Datanbankbasierte Teilstring-Index Suche

LIKE ohne führendes Wildcard aber mit der Suche auf allen automatisch berechneten Tokens also z.B. LIKE '%'

> *das wäre wie bisher, oder? auf alle Fälle sollte noch die Suche am Ende möglich sein also LIKE '%'*
> *nein, den Teilstring-Index gibt es ja noch nicht, der Wilcard am Ende ist ja wegen dem Teilstring-Index nicht mehr nötig*

2B: Lucene-basierte Teilstring-Index Suche

> *eine interessante Alternative, ist nur die Frage, wie das mit japanisch klarkommt.*
> *Für deutsch könnte man auch problemlos die Volltextsuche von mysql benutzen, die nicht mit japanisch klarkommt.*

Das Tokenizing würden wir ja dann nicht über Lucene machen, sondern selbst die Teilstrings raussuchen, nach unseren eigenen Logik. Der CJK-Tokenizer bei Lucene greift (wie fast alle Tokenizer) nur richtig gut bei großen Texten. Die Zerlegung einzelner Wörter in Bestandteile macht der nicht (afaik).
De Facto würde man in diesem Fall Lucene aber nur als DB verwenden, und wäre nicht wirklich schneller wie eine richtige DB, vorallem wenn sie im RAM ist.

3. cgrep-Suche auf Index-Datei

Per cgrep auf einer index-datei suchen und die UIDs zurückliefern.
Beispielhaft auf dem Enwicklungsserver implementiert

Vor- und Nachteile?

- LIKE ist möglichst zu vermeiden, da die Suche ineffizient ist
  LIKE ohne führendes Wildcard ist kein Problem, mit führendem Wildcard ist es in der Tat ineffizient
- besser wäre ein Volltextindex, welcher aber auch japanischtauglich sein muss
  genau, dass habe ich jetzt unter 2 präzisiert und dein Beispiel aufgenommen, thx fürs Feedback.
- Momentan überwiegen scheinbar die Vorteile für 2A, oder?

altes Thema im Forum dazu

Volltextsuchmaschinen mit Unterstützung für die japanische Sprache

- **Senna** http://qwik.jp/senna/

  > *patcht die Volltextsuche von MySQL*
  > *benutzt MeCab als Tokenizer*
  > *Senna installation*

- **sen** https://sen.dev.java.net/

  > *Java-Implementation von MeCab*
  > *Volltextsuche basierend auf Lucene*

- **gosen** http://itadaki.org/wiki/index.php/GoSen

  > *Java-Implementation von MeCab, Aktualisierung von **sen***