

Эффективный полнотекстовый поиск по базам данных



Андрей Аксенов, Петр Зайцев
Percona Ltd.
shodan (at) shodan.ru

Поиск по базам?

- **Базы постоянно растут**
 - 1М записей “есть у всех”
 - 10-100М записей не редкость
 - Существуют базы с 1В+ записей, в которых надо искать по тексту (ярчайший пример – Google)
- **Широко используются open-source СУБД**
 - Мы будем говорить о MySQL
 - “Люди говорят”, в мирах других СУБД схожие проблемы
- **Встроенные решения с полнотекстовым поиском справляются недостаточно хорошо**
 - Особенно, если кроме “просто” поиска нужны...

Виды специальных задач

- **“Просто” поиск – ключевая задача, но...**
 - В чистом виде бывает на удивление редко
 - Задача, скорее, для Web поисковиков
- **Часто нужна дополнительная сортировка**
 - По отличному от релевантности ключу – например, по цене товара
- **Часто нужна дополнительная фильтрация**
 - Например, по категории товара или по автору поста
- **Часто нужна группировка найденных записей**
 - Например, по дате или по источнику данных
- **Что предлагают встроенные решения?**

Встроенный MySQL FTS

- **Плюс – встроен, обновляется “сразу”**
- **Минус – Только MyISAM**
- **Минус – плохо масштабируется**
- **Минус – не учитывает позиции слов в индексе**
 - Проблемы с релевантностью
 - Медленный поиск по фразам
- **Минус – только 1 FT индекс на запрос (поля...)**
- **Минус – не работает с другими индексами**
 - те. обработка WHERE, ORDER/GROUP BY, LIMIT будет производиться отдельно и “вручную”
- **Вывод – зачастую не годится**

Shootout внешних решений

- Тестировались известные (нам) open-source решения
 - Коммерческие решения пусть рекламируют сами производители ☺
- MySQL FTS
- mnoGoSearch, <http://mnogosearch.org/>
 - Предназначен для Web-поиска, но адаптируется (htdb)
- Lucene, <http://lucene.apache.org/>
 - Популярная Java-библиотека для поиска
- Sphinx, <http://sphinxsearch.com/>
 - Изначально спроектирован для поиска по БД

Результаты тестирования

- ~3.5М записей, ~5 GB текста (из Wikipedia)
- mпоGoSearch сошел с дистанции
- подробнее в презентации Петра Зайцева на EuroOscop'2006

	MySQL	Lucene	Sphinx
Индексация, min	1627	176	84
Индекс, MB	3011	6328	2850
Match all, ms/q	286	30	22
Match phrase, ms/q	3692	29	21
Match bool top-20, ms/q	24	29	13

Имеющиеся решения

- **mnoGoSearch**
 - Минус – проблемы со скоростью индексации и поиска
 - FATAL – 5 GB за 24 часа проиндексировать не успел
- **Lucene**
 - Плюс – обновляющийся “на лету” индекс
 - Плюс – wildcard, fuzzy поиск
 - Минус – цена интеграции (это Java библиотека)
 - Минус – реализация фильтров (скорость поиска)
 - Минус – отсутствие группировки
- **Sphinx**
 - Минус – “монолитные” индексы
 - Плюс – все остальное 😊

Sphinx – обзор

- **Внешнее решение для поиска по СУБД**
- **Две основные программы**
 - Indexer, для переиндексации FT индексов
 - Searchd, поисковой демон
- **Легкая интеграция**
 - Встроенная поддержка MySQL, PostgreSQL
 - Наличие APIs для PHP, Python, Perl, Ruby, и т.д.
 - Наличие MySQL Storage Engine
- **Высокая скорость**
 - Скорость индексации – 4-10 MB/sec
 - Скорость поиска – avg 20-30 ms/q @ 5 GB, 3.5M docs

Sphinx – идеология

- Индексация локально доступных баз данных
- Изначальная поддержка структурированных “как в SQL” документов
 - До 256 полнотекстовых полей
 - Любое количество атрибутов (integer/timestamp/etc)
- “Быстрая переиндексация вместо медленного поиска”
 - Необновляемый формат индекса – изначально был выбран в расчете на максимально быстрый поиск
 - Оказалось – что переиндексация тоже очень быстрая
 - Для частичных обновлений – переиндексация “частичных” (delta) индексов раз в N минут

Sphinx – поиск

- **Качество**
 - Учитываются позиции слов, не только их частоты
- **Масштабируемость**
 - До 50-100 GB текста на 1 CPU
 - Возможен распределенный поиск
 - Распределенные индексы полностью прозрачны для клиентских приложений
- **Примеры**
 - Boardreader.com – около 1Bil записей, более 1TB GB текста, в кластере 16 CPU (4*2 Dual Core Servers)
 - Mininova.org – мало (менее 1M) записей, но 2-3M запросов в сутки, поиск подстрок

Sphinx – спецвозможности

- **Сортировка**
 - По любой комбинации атрибутов, SQL синтаксис
- **Фильтрация записей по условию**
 - Учитывается при поиске на максимально ранней стадии – для скорости
 - Атрибуты всегда либо загружены в память, либо скопированы в нужном порядке в индексе – для скорости
 - Fun fact – иногда полный перебор всех записей и фильтрация при помощи Sphinx оказываются быстрее соответствующей выборки из MySQL – и используются вместо нее в production...

Sphinx – спецвозможности

- **Группировка**
 - Возможна по какому-либо атрибуту
 - Выполняется в фиксированной памяти
 - Выполняется неточно (!)
 - Выполняется быстро (по сравнению с MySQL итп)
- **Подсветка найденных слов**
 - Спец-сервис, которому передаются тексты документов и запроса
- **MySQL Storage Engine**
 - Для еще более сложной обработки на стороне MySQL, чем возможна на стороне Sphinx
 - Для упрощения интеграции

Выводы

- Для полнотекстового поиска в средних и больших БД необходимы внешние решения
- Есть ряд требований к таким решениям, помимо “просто” поиска (фильтрация, группировка, и т.п)
- Есть ряд open-source решений разной степени полезности – и соответствия требованиям
- Для большей части задач, try Sphinx,
- <http://sphinxsearch.com/>