

# Sphinx Full-text Indexing

What is full-text indexing?

- Hash of keyword → [ doc\_id, doc\_id ]

# Do you need it?

- Probably not.
- You use it when you can't predict what a user will search for, and results must be returned quickly.
- Sound app design can usually preclude the necessity.

# Symptoms you might need it

- Lots of entries in the MySQL slow query log
- Searches on unindexed text in excess of 100k rows
- Your data is mostly text

# Other Options

- MySQL built-in FULLTEXT engine
  - Works, but old and clunky
- Lucene
  - Very well could work for you
  - Java bohemeth, slow indexer
- Tokyo Dystopia
  - Unfriendly, sparse management, slow
- MongoDB
  - Not really fulltext, but can be shoehorned

# Why Sphinx?

- Simple, powerful
- Superb API
- Great community
- Great documentation
- Fastest indexer known to (this) man
- Advanced functionality (groupby, etc.)
- Indexes MS-SQL, ODBC, arbitrary data via XML

# How does it work?

1. Indexes built that *refer* to the actual data
2. App queries Sphinx
3. Sphinx returns id references to actual data, along with attributes
4. App queries data by id to get actual content

# How does it work?

- Sounds like a lot of work, but
  - Sphinx queries are very, very fast (1 ms/million rows)
  - Follow-up DB call can be avoided with attributes
  - SphinxSE (storage engine) is available to MySQL

# Indexing

- Several types of indexes
  - MySQL (using native libs)
  - MS-SQL (on Windows)
  - ODBC
  - XML pipe (arbitrary data)

# Example index definition

```
source forum_posts {  
    type = mysql  
    sql_db = ...  
    sql_host = ...  
    sql_user = ...  
    sql_query = SELECT id, user_id, post FROM forum_posts  
    sql_attr_uint = user_id  
}  
  
index form_posts {  
    source = forum_posts  
    path = /data/sphinx/forum_posts  
}
```

# Dynamic indexes

- What if you want to change your indexed data on the fly?

```
sql_query_pre = SELECT MIN(id) INTO @first,  
MAX(id) INTO @last FROM forum_posts  
WHERE timestamp > DATE_SUB(NOW(),  
INTERVAL 1 WEEK)
```

```
sql_query = SELECT id, user_id, post FROM  
forum_posts WHERE id BETWEEN @first AND  
@last
```

# Dynamic indexes

- Then just re-index whenever you want to update (`indexer -rotate forum_posts`)
- Sphinx can seamlessly rotate for zero downtime
- Allows for base + delta scheme
  - Jan-June in base index
  - June-Aug in delta 1
  - Aug-Dec in delta 2, etc.

# That's a lot of work

- Not really, but it's definitely not simple.
- Enter real-time indexes

# Real-time indexes

- Brand new (introduced this spring, Alpha code)
- Means no more re-indexing
- Adding to an index is an INSERT
  - INSERT into MySQL
    - Then, INSERT into Sphinx using MySQL protocol
- Sphinx does the indexing and MySQL holds the content