Sphinx search technical overview

Vladimir Fedorkov
Open Source Search Devroom
FOSDEM’15
About me

• Performance geek
  – blog http://astellar.com
  – Twitter @vfedorkov

• Enjoy LAMP stack tuning
  – Especially database backend

• Love to speak on the conferences

• Use Sphinx in production from 2006
Meet Sphinx

• Created in early 200x as an alternative to MySQL full-text search
• Written on C++
• Working as separate daemon
• Running on various platforms *nix, win*, etc
  – Seen on iPhones and WiFi routers
• Now serving installations with billions or documents.
Architecture sample: querying

- Application
- Sphinx daemon
- Database
- Indexing
- Sphinx index

Additional data

Query
Search results
Agenda

• Loading data
• Current storage types
• Querying Sphinx
• Full text vs non-full-text
• Getting results
• Life after the search
• Grow Sphinx from node to cluster
Loading data into Sphinx

• Sphinx is talking to databases to pull data
  – MySQL, PostgreSQL, MSSQL and any ODBC source
• Loading structured data in XML format
  – Useful to load data from NoSQL storages
    • like Mongo, etc
  – Can be used for document pre-processing
• SQL-style updates
Storage types

• Real-time indexes
  – Push mode
    • Application pushes data to Sphinx
  – Ideal for frequently updated data

• On-disk (plain) indexes
  – Data pull mode
    • Sphinx handling indexing on itself
  – Ideal for static data

• Or else:
On disk vs Real-time indexes

Database

SELECT

Indexing

OnDisk index

Application

INSERT, UPDATE, DELETE

Sphinx daemon

Real-time RAM chunk

Disk chunk(s)
Querying

• SphinxQL:

```sql
mysql> SELECT * FROM sphinx_index
    -> WHERE MATCH('I love Sphinx')
    -> AND news_channel = 285
    -> LIMIT 5;
```

– Uses MySQL client lib to connect to sphinx
– Available in most programming languages

• Legacy API
  – PHP, Python, Java, Ruby, C is included in distro
  – .NET, Rails (via Thinking Sphinx) via third party libs
How does it work?

• Query pre processing
• Full-text search stage
• Non-full text filtering
• Ranking / Grouping / Ordering
• Applying limit
• Sending results back
Query & text pre-processing

- Removing stop words
- Transforming text
  - Applying morphology, blended chars, filters, replacements
- Prefix/infix indexing
- Other “magic”
Full-Text support

- **And, Or**
  - `hello | world, hello & world`
- **Not**
  - `hello -world`
- **Per-field search**
  - `@title hello @body world`
- **Field combination**
  - `@(title, body) hello world`
- **Search within first N**
  - `@body[50] hello`
- **Phrase search**
  - "hello world"
- **Per-field weights**
- **Proximity search**
  - "hello world"~10
- **Distance support**
  - `hello NEAR/10 world`
- **Quorum matching**
  - "the world is a wonderful place"/3
- **Exact form modifier**
  - "raining =cats and =dogs"
- **Strict order**
- **Sentence / Zone / Paragraph**
- **Custom documents weighting & ranking, etc**
Non text filters

• in SphinxQL terms, WHERE conditions
  – a = 5, a < 5, a > 5, a BETWEEN 3 AND 5
• Integers, floating point, strings are supported
• JSON
  – SELECT ALL(x>3 AND x<7 FOR x IN j.intarray)
  – SELECT j.users[3].address[2].streetname
Special integers: MVAs

• Built in “one–to–many” attributes
• Set of integers in a single value
• Useful for
  – Page tag IDs
  – Multi category items
GEO-Distance support

• Bumping up and/or filtering local results
  – Just add float latitude, longitude attributes, and..
• **GEODIST**(Lat, Long, Lat2, Long2) in Sphinx
• Has syntax for mi/km/m, deg/rad etc
Relevance tuning

• Weighting
  – Per field
  – Per index

• Expression based ranking
  – 15+ of text signals, N of yours non-text
    • OPTION ranker=expr(‘1000*sum(lcs)+bm25’) 
    • OPTION ranker=expr(‘700*sum(lcs)+bm25f(1.4, 0.8, 
      {title=3, content=1})’)
  – Several built-in rankers available
mysql> SELECT * FROM idx
     -> WHERE MATCH('I love Sphinx') LIMIT 5
     -> OPTION field_weights=(title=100, content=1);

+-------------------+--------+-------------+------------+
| id    | weight | channel_id | ts         |
+-------------------+--------+-------------+------------+
| 7637682 | 101652 |            | 1112905663 |
| 6598265 | 101612 |            | 1102858275 |
| 6941386 | 101612 |            | 1076253605 |
| 6913297 | 101584 |            | 1087685912 |
| 7139957 | 1667   |            | 1078242789 |
+-------------------+--------+-------------+------------+

5 rows in set (0.00 sec)
Life after search

- CALL SNIPPETS, making excerpts
- Building facets (Brands, price ranges)
- Showing related items
- Performing misspells corrections
- “Did you mean” service
Combining indexes

• On the single box
  – Main + Delta
  – Main + Delta + RT

• On the cluster
  – Local and distributed
Distributed search

- Yet static nodes configuration
- Weighted round-robin querying
- Load-based distribution
- Failover node
Sphinx search cluster architecture
News from the Lab

• New index format in Sphinx 3.0
  – Faster indexing and search
• No legacy 4/16Gb attribute limits per index
• Data replication between nodes
• HTTP/REST interface
• Even faster snippets
• Some secret projects I can’t talk about 😊
Find more about Sphinx

- Official website: http://sphinxsearch.com
- My blog http://astellar.com
  - Some information you may find useful
  - Slides will be there
- Twitter: @vfedorovk
  - Mainly Sphinx and MySQL performance
QUESTIONS!
THANK YOU!