

MySQL FULLTEXT Searching

Searching for Dummies

The “Old” Way

- Reverse indexes
 - Difficult to maintain the integrity of the index
 - I’m not an expert in the area of searching algorithms
- You didn’t actually use LIKE or REGEXP did you?
 - Does not scale
 - Does not work well with large amounts of text

The new way

- What is a FULLTEXT index?
 - Not REGEXP or LIKE
 - Provides easy way to search text fields (ie. char, varchar, text, etc.)
- What does FULLTEXT provide?
 - Stopwords
 - Relevancy ranking
 - Boolean search terms (ie. “+dog -cat”)

Pros & Cons

- Pros

- Easy to implement
- Powerful search capabilities
- Cleaner solution when compared to rolling your own

- Cons

- Black box
- May not fit highly customized searching environments

FULLTEXT Tips

- Things to remember
 - Your search term must be a constant, cannot be derived from query fields
 - Search results are automatically ordered by MySQL's relevancy rating
 - Search results from search terms longer than 20 characters will not be sorted
 - Fields in the MATCH() statement must match the fields in the FULLTEXT() definition
 - Search terms that appear in more than 50% of the records will return 0 results (aka 50% threshold)
 - Search terms are limited by the ft_min_word_len

FULLTEXT Tips

- Things to remember
 - FULLTEXT indexes are NOT supported in InnoDB tables
 - By default, your search query must be at least four characters long and may not exceed 254 characters
 - MySQL has a default stopwords file that has a list of common words (i.e., the, that, has) which are ignored in search terms
 - MySQL's relevancy rating does not appear to be a useful number

Creating a FULLTEXT index

```
create table blog (  
  blogID int(9) unsigned not null default '0',  
  categoryID int(9) unsigned not null default '0',  
  title char(255) not null,  
  entry text not null,  
  PRIMARY KEY (blogID),  
  KEY (categoryID),  
  FULLTEXT (title,entry)  
);
```

```
create table blog_categories (  
  categoryID int(9) unsigned not null default '0',  
  name char(100) not null default '',  
  PRIMARY KEY (categoryID)  
);
```

Querying FULLTEXT Indexes

```
SELECT *  
FROM blog  
WHERE MATCH(title,entry) AGAINST ('lauren');
```

```
SELECT B.*,C.name  
FROM blog AS B, blog_categories AS C  
WHERE B.categoryID=C.categoryID AND  
       B.categoryID=1 AND  
       MATCH(B.title,B.entry) AGAINST ('michigan');
```

IN BOOLEAN MODE

```
SELECT B.*,C.name
FROM blog AS B, blog_categories AS C
WHERE B.categoryID=C.categoryID AND
      B.categoryID=1 AND
      MATCH(B.title,B.entry)
      AGAINST ('+michigan -ohio' IN BOOLEAN MODE);
```

IN BOOLEAN MODE Tips

- IN BOOLEAN MODE ignores the 50% threshold
- Results are not automatically sorted as they are in non-IN BOOLEAN MODE
- A boolean full-text search can also work even without a FULLTEXT index, although it would be **slow**.
- IN BOOLEAN MODE supports +-<>{}~*"

Altering FULLTEXT Behavior

- Changing `ft_min_word_len`
 - Must be changed at startup

```
./bin/safe_mysqld --user=mysql -O ft_min_word_len=3 &
```
 - Must repair tables using FULLTEXT indexes after restart

```
REPAIR TABLE tbl_name QUICK
```
- A recompile is required to change the 50% threshold and is not recommended by the MySQL team

Scalability?

- A search for 'Dell' on a products table with 13,000 records returns 716 rows in 0.09 seconds.
- Upgrade to 4.0 (if you have not already) if you want to use FULLTEXT in production.
- Reports of searches returning in under a second with over 2 million records.
- Specifically fine tuned for large collections of data

Adding PHP to the Mix

- Add search to anything with text
- Cache text into a single table to provide a central search engine for your site
- Use PHP's ispell/aspell module to spell check search terms and offer alternatives
- Cross reference data by allowing authors to add keywords to articles, products and other entries

Looking to the Future

- Proximity operators
- Making stopwords support i18n
- Adding support for stemming
- Increased performance

Conclusion

- FULLTEXT indexes are easy to use, easy to implement and offer a powerful searching solution
- No reason not to add this functionality to your MySQL database

Questions?

