Groonga ではじめる全文検索

版　　発行

第 1 章

MySQL で作る全文検索システム

Groonga は様々なソフトウェアに組み込み、全文検索機能を提供することができます。その一つ、MySQL に組み込むための Mroonga を使って、全文検索システムを作ってみましょう。

1.1 Mroonga の概要

Mroonga は、MySQL のストレージエンジンの一つです[*1]。MySQL のストレージエンジンには MyISAM や InnoDB などがあり、聞いたことがある人も多いでしょう。それぞれにパフォーマンスや対障害性などに特徴があり、テーブルごとに使い分けることができます。

その一つとして Mroonga は、Groonga を組み込んで、高速かつ多機能な全文検索を可能にしたストレージエンジンです。これを MySQL に組み込んで使うことで、SQL を使って全文検索を行えるようになります。

Mroonga を全文検索システムの本番環境で使うには、公式マニュアルのインストールのページを参照してください。ここでは、本書で試す目的で、簡易に環境を準備する方法を紹介します。

1.2 Mroonga の環境の準備

本書では、MySQL 及びその他の環境の準備に Docker を使います。

Docker のインストール

（後で書く）

Docker コンテナの起動

本書用に必要なソフトウェアをインストールした Docker イメージを作成してあります。ターミナルで以下のコマンドを実行してください。

[*1] MySQL から派生した MariaDB というデータベースシステムでも使用できます。最新の MariaDB では始めから Mroonga が組み込まれています。

リスト 1.1　Docker イメージの取得とコンテナの起動

```
% cd path/to/project
% docker run --detach --name=pdfsearch --publish=8080:80 \
    --volume=$PWD:/var/lib/pdfsearch kitaitimakoto/grnbook-mroonga
```

`docker run` コマンドを実行すると、以下の二つのことが行われます。

1. Docker イメージの取得（もしシステム上に存在しない場合）
2. 取得した Docker イメージを元にした Docker コンテナの起動

Docker イメージの取得は一度行えば充分なので、コンテナの二度目以降の起動には別のコマンドを使用します。

1zwDocker コンテナの起動

```
% docker start grnbook
```

1zwDocker コンテナの停止

```
% docker stop grnbook
```

コンテナの操作に使用している `grnbook` という名前は、リスト 2.1 における `name` オプションで指定した物を使用します。

動作確認

Docker コンテナが実際に動作していることを確認しましょう。このイメージには PHP が含まれているので、以下のようなファイルを作成することで動作確認ができます。

リスト 1.2　info.php

```
<?php
phpinfo();
```

ブラウザーで http://localhost:8080/info.php にアクセスしてください。以下のように PHP の情報が表示されれば、環境の準備は成功しています。（Docker Toolbox 使っている場合は localhost じゃなさそう）

図 1.1　phpinfo() の実行結果

「Not Found」の画面が表示されてしまう場合は、どこかで間違えてしまったようです。これまでの手順を見ながらやり直してみてください。

`docker run` をやり直すには、以下のコマンドを実行して一旦 Docker コンテナを削除する必要があります。

1zwDocker コンテナの削除

```
% docker rm grnbook
```

ありがちな間違いとして、Docker コンテナと docker コマンドを実行している（OS X などの）ホストマシンとで共有するディレクトリーをうまく設定できないことがあります。リスト 2.1 で volume オプションに$PWD を使用していますが、これは現在のディレクトリーを意味します。そのため、このコマンドを実行した時のディレクトリーが、PHP ファイルを描くべき場所として使用されます。事前にプロジェクトのディレクトリーに移動してから実行するようにしましょう。

1.3 作成する全文検索システムの概要

それでは、PHP と Mroonga を使って、全文検索システムを作っていきましょう。以下のようなシステムを作ることにします。

概要

登録済みの PDF ファイルを全文検索するシステムである

できること

検索対象となる文書をウェブ UI で登録することができる

ウェブ UI 上のテキストフィールドを使って検索することができる

実装方針

ウェブ UI は PHP を使用して作成する

全文検索用のデータは MySQL に格納する

PDF からテキストを抜き出す処理は本書では扱わない（Docker イメージ付属のツールを使う）

1.4 データベースの作成

まず、PHP を使って、MySQL にデータベースを作成します。作成の手順は通常の MySQL でのデータベース作成と同様です。Docker コンテナ内では既に MySQL が動作しており、root ユーザーでパスワード無しでログインできます。

ソースコードは以下のようになります。[*2]

リスト 1.3 create-database.php

```
<?php
define('DBNAME', 'pdfsearch');

$dbh = new PDO('mysql:host=localhost;charset=utf8', 'root', '');

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $result = $dbh->exec('CREATE␣DATABASE␣' . DBNAME);

    header("Location:␣{$_SERVER['REQUEST_URI']}");
}

$result = $dbh->query('SHOW␣DATABASES')->fetchAll();
$databases = array_map(function($row) {
```

[*2] 限定された状況での一度きりの処理のため、褒められない書き方をしている箇所もあります。実際のアプリケーションで使用する場合には、フレームワークなどのやり方に則った適切な方法でデータベースを操作してください。

```
    return $row[0];
}, $result);
?>
<!doctype html>
<html>
  <head>
    <title>データベースの作成</title>
  </head>
  <body>
    <h1>データベースの作成</h1>
    <?php if (! (in_array(DBNAME, $databases))): ?>
    <form method="post">
      <input type="submit" value="pdfsearchデータベースの作成">
    </form>
    <?php endif; ?>
    <h2>現在のデータベース一覧</h2>
    <ul>
    <?php foreach ($databases as $database): ?>
      <li><?php echo htmlspecialchars($database, ENT_QUOTES, 'UTF-8'); ?></li>
    <?php endforeach; ?>
    </ul>
  </body>
</html>
```

通常の MySQL に於けるデータベースと全く同じ手順でデータベースを作成すれば OK です。

うまくできたら、http://localhost:8080/create-database.php を表示して、実際に画面上のボタンを押してデータベースを作成してみましょう。「現在のデータベース一覧」に「pdfsearch」というデータベースが加わるはずです。

■コラム: PHP のデバッグ

PHP が期待通りに動作しない場合や画面が真っ白になって何が悪いか分からない場合は、`docker logs` コマンドでログを確認することができます。

リスト 1.4　docker logs で PHP のログを確認

```
% docker log pdfsearch
13.03.2016 09:41:25 INFO  -- [web:php] switch :starting [:unmonitored => :starting] (
reason: monitor by user)
160313 09:41:26 mysqld_safe Can't log to error log and syslog at the same time.  Remove
all --log-error configuration options for --syslog to take effect.
160313 09:41:26 mysqld_safe Logging to '/var/log/mysql/error.log'.
160313 09:41:27 mysqld_safe Starting mysqld daemon with databases from /var/lib/mysql
[Sun Mar 13 09:41:54 2016] 172.17.0.1:58804 [200]: /create-database.php
[Sun Mar 13 09:41:55 2016] 172.17.0.1:58808 [404]: /favicon.ico - No such file or
directory
[Sun Mar 13 09:42:01 2016] PHP Parse error:  syntax error, unexpected '$databases' (
T_VARIABLE) in /var/lib/pdfsearch/create-database.php on line 14
[Sun Mar 13 09:42:01 2016] 172.17.0.1:58812 [500]: /create-database.php - syntax error,
unexpected '$databases' (T_VARIABLE) in /var/lib/pdfsearch/create-database.php on line
14
```

`-f` オプションを付け

1zw

```
% docker logs -f pdfsearch
```

と実行すると、ログが出力する度にそのログが表示されます。Docker コンテナの中では PHP のビルトインサーバーが動作しているため、ログの内容もそれに準じます。

また、ブラウザーでエラー内容を確認したい場合には、PHP ファイルの冒頭で

1zw

```
<?php
ini_set('dispaly_errors', 'stdout');
```

と設定するとよいでしょう。

1.5 テーブルの作成

データベースが出来たので、以下のカラム持つ pdfs テーブル作りましょう。

表 1.1 作成する pdfs テーブル

カラム	内容	データ型	備考
path	システム内のパス（場所）	VARCHAR(255)	主キー。検索対象ではない
title	PDF 文書のタイトル	VARCHAR(255)	検索対象
content	PDF 内のテキスト	LONGTEXT	検索対象

やり方はデータベースの作成と同様です。

リスト 1.5 create-table.php

```
<?php
define('DBNAME', 'pdfsearch');
define('TABLE_NAME', 'pdfs');

$dsn = 'mysql:host=localhost;dbname=' . DBNAME . ';charset=utf8';
$dbh = new PDO($dsn, 'root', '');
$table = TABLE_NAME;

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $statement = <<<STATEMENT
CREATE TABLE `{$table}` (
    path    VARCHAR(255) PRIMARY KEY,
    title   VARCHAR(255),
    content LONGTEXT,
    FULLTEXT INDEX (title),
    FULLTEXT INDEX (content)
) ENGINE = Mroonga DEFAULT CHARSET utf8;
STATEMENT;
    $dbh->exec($statement);

    header("Location: {$_SERVER['REQUEST_URI']}");
}

$result = $dbh->query('SHOW TABLES')->fetchAll();
$tables = array_map(function($row) {
    return $row[0];
}, $result);

?>
<!doctype html>
<html>
  <head>
    <title>テーブルの作成</title>
  </head>
  <body>
    <h1>テーブルの作成</h1>
    <?php if (! (in_array(TABLE_NAME, $tables))): ?>
    <form method="post">
      <input type="submit" value="pdfsテーブルの作成">
    </form>
    <?php endif; ?>
    <h2>現在のテーブル一覧</h2>
    <ul>
    <?php foreach ($tables as $table): ?>
      <li><?php echo htmlspecialchars($table, ENT_QUOTES, 'UTF-8') ?></li>
    <?php endforeach; ?>
    </ul>
  </body>
</html>
```

http://localhost:8080/create-table.php にアクセスして pdfs テーブルを作成してみましょう。事前に pdfsearch テーブルを作っておかないとエラーになるので注意してください。

データベース作成時と違って、Mroonga を使ったテーブルを作る時にはいくつか注意点があります。

一番大事なのは、CREATE TABLE 文の最後に、ストレージエンジンとして Mroonga を指定することです。

1zw

```
CREATE TABLE (
-- ...
) ENGINE = Mroonga DEFAULT CHARSET utf8;
```

普段は InnoDB や MyISAM を指定しているところを切り替えるだけなので、難しいことはないでしょう。

次に、インデックスの作成方法が InnoDB などと異なります。PDF のタイトルや内容を完全一致で検索することはまずないでしょうから、通常のインデックスは貼る必要がありません。代わりに、全文検索のための特別なインデックスを作る、FULLTEXT INDEX 構文を使用します。これは、MySQL に元々備わっている、全文検索機能のための構文です。Mroonga でも用途は同じなので、全く同じ構文で使用できます。

それ以外は、通常の MySQL 使い方と同じです。主キーにはパスといった意味のある物ではなく、ただレコードを識別するためだけのサロゲートキーを使う人もいることでしょう。それでも構いません。