de Lau

Portfolio website van Vincent de Lau

Kruimeltjes: Artikelen » GeoIP HOWTO

Home
Portfolio
Artikelen
GeoIP HOWTO
PHP Duotone
Fun stuff
Links
Contact

HOW-TO Import the MaxMind GeoIP Free Country CSV file into MySQL and save diskspace

Copyright and disclaimer

VINCENT DE LAU DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL VINCENT DE LAU BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Version history

Version 1.0 (2004-11-22) Initial release, simple HTML version with little formatting

Version 1.1 (2005-03-01) corrected some errors (file was never publicaly available and got lost on my computer

Version 2.0 (2005-03-17) Converted document to XML, using XSLT to generate XHTML and CSS for formatting

Version 2.1 (2006-10-02) small correction: properly escaped the `mysqlimport` command

Introduction

After reading an article on geopositioning in PHP Magazine (www.php-mag.net), I visited the MaxMind website to get myself a copy of the free country database to use in my custom logging module. On the site I stumbled upon the CSV file download and, curious as I am, downloaded a copy.

Since the website warned about large filesizes and prevered usage of the binary package, I looked into the file to see how wel an optimization job the had done. So I started importing the data into MySQL and do some optimization.

To my big surprise, the total size of this database was smaller than the size of the binary database. After closer inspection, I also found out that my database contained more information, since the binary release doesn't contain the country names. These are stored in the MaxMind GeoIP API's.

This HOW-TO describes how to import the CSV file that can be downloaded from the MaxMind website into MySQL and how to optimize this data to save diskspace. Before we start of, I'd like to point out that I can make no promises on performance. The only thing I am sure of is that this approach saves some kB on diskspace and might open some other interesting options for you.

In this document, I'm making the assumption that you know how to do some basic stuff in MySQL or SQL in general. If you don't, go Google for a MySQL tutorial or visit the MySQL website. The steps are quite simple, so if you know how to do a query using the mysql command line tool, you are fine. If you have DBA skills, this stuff might be a bit 'duh!'.

Overview

The process consists of a few steps.

- Step 0 is to download the CSV file from the MaxMind website.
- Step 1 is to make a MySQL database and set up some tables.
- Step 2 is to import the CSV file into MySQL
- Step 3 is to optimize the data
- Step 4 is our final clean-up step

After these steps, I'll show some PHP code to query the database.

Step 0: Download the CSV

Visit the MaxMind website and get yourself a copy of the MaxMind GeoIP free country CSV. Save it in a location where you can find it again and unzip.

http://www.maxmind.com/app/geoip_country

Step 1: Create the database and set up the tables

To start, we need a database. You can add the tables to an excisting database or create a new one:

```
CREATE DATABASE geoip;
```

```
USE geoip;
```

Before we start creating tables, take a look at the downloaded CSV in your favorit editor/viewer. You should see some lines like:

```
"80.247.159.0","80.247.159.255","1358405376","1358405631","NG","Nigeria"
"80.247.160.0","80.247.175.255","1358405632","1358409727","IE","Ireland"
"80.247.192.0","80.247.203.255","1358413824","1358416895","NL","Netherlands"
"80.247.204.0","80.247.205.255","1358416896","1358417407","BE","Belgium"
"80.247.206.0","80.247.207.255","1358417408","1358417919","NL","Netherlands"
```

Each row describes a range of IP addresses and the country it is allocated to. The format of each line is:

```
{start_ip},{end_ip},{start_long},{end_long},{cc},{country name}
```

Before we can import this into MySQL, we need to make a table to fit it's content. The SQL query to do so would be:

```
CREATE TABLE csv (
  start_ip CHAR(15) NOT NULL,
  end_ip CHAR(15) NOT NULL,
  start INT UNSIGNED NOT NULL,
  end INT UNSIGNED NOT NULL,
  cc CHAR(2) NOT NULL,
  cn VARCHAR(50) NOT NULL
);
```

This table is just for importing the data. We will empty/drop it when done. As you can see in the CSV, there is a lot of duplicate data. You don't have to look far in the file to find the country code for for instance the Netherlands. We are going to extract that data into a seperate table:

```
CREATE TABLE cc (
  ci TINYINT UNSIGNED NOT NULL PRIMARY KEY AUTO_INCREMENT,
  cc CHAR(2) NOT NULL,
  cn VARCHAR(50) NOT NULL
);
```

Why the `ci` field? Well, since the database contains about 64K lines, saving a byte on rowlength saves us about 64KB! The `TINYINT` field is one byte, while the `cc` field uses two.

We also will strip the ASCII form IP notation. Why? First, to store an ASCII representation of an IP address, you need 15 byte. (4*3 digits + 3 dots) If we store the binary value we use 4 byte per address. In PHP, conversion is easely done using the functions `ip2long()` and `long2ip()`. The format of the table will be:

```
CREATE TABLE ip (
  start INT UNSIGNED NOT NULL,
  end INT UNSIGNED NOT NULL,
  ci TINYINT UNSIGNED NOT NULL
);
```

The resulting row length is 10 bytes (for some reason MySQL adds one byte). The table size will be around 64K*10B = 640KB. We will not add any indexes, since one index will be larger that the whole table.

Step 2: import the CSV file into MySQL

First rename GeoIPCountryWhois.csv to csv.csv. The reason for doing this, is so that the MySQL import utility knows which table to use. Next we run the MySQL import utility. Of course you also use another utility to import the contents of the csv file into the csv table.

```
mysqlimport --fields-terminated-by=","
  --fields-optionally-enclosed-by="\""
  --lines-terminated-by="\n"
  --host=<hostname>
  --user=<username>
  --password=<password>
  geoip csv.csv
geoip.csv: Records: 63809  Deleted: 0  Skipped: 0  Warnings: 0
```

Step 3: optimize the data

Ok, lets go fill the final tables. First we need to get the country codes into the seperate table. To accomplish this we do the following query:

```
INSERT INTO cc SELECT DISTINCT NULL,cc,cn FROM csv;
```

For the SQL newby: this query will `INSERT` the data returned by the `SELECT` statement `INTO` the `cc` table. We `SELECT` the unique (`DISTINCT`) rows `NULL,cc,cn` with `NULL` as the value to trigger `AUTO_INCREMENT`.

The next step is to get the resulting information into the ip table:

```
INSERT INTO ip SELECT start,end,ci FROM csv NATURAL JOIN cc;
```

For the SQL newby: this query will `INSERT` the data returned by the `SELECT` statement `INTO` the `ip` table. We `SELECT` the values `start,end,ci` `FROM` the tables `csv` and `cc`, using a `NATURAL JOIN`. This means that the tables will be "linked" using equaly named fields, in this case `cc`. This is roughly equal to `FROM csv,cc WHERE csv.cc=cc.cc`.

```
mysql> use geoip;
Database changed
mysql> INSERT INTO cc SELECT DISTINCT NULL,cc,cn FROM csv;
Query OK, 231 rows affected (0.23 sec)
Records: 231  Duplicates: 0  Warnings: 0
mysql> INSERT INTO ip SELECT start,end,ci FROM csv NATURAL JOIN cc;
```

```
Query OK, 63809 rows affected (2.81 sec)
Records: 63809  Duplicates: 0  Warnings: 0
mysql>
```

Step 4: final clean-up

Ok, you now have the data in the database! But we also have the imported CSV file in our database. You have three options: keep the data (uses some space...), remove the data but leave the table or remove the whole table.

remove data: `DELETE FROM csv;`

remove table: `DROP TABLE csv;`

Usage: the SQL select query

To get the country code for a specific IP address, you first need to convert it to a integer. In PHP, the ip2long() function does exactly that. Next you need to perform a SQL query like:

```
SELECT cc, cn
FROM ip NATURAL JOIN cc
WHERE {insert your IP interger here} BETWEEN start AND end
```

The following piece of code are some PHP functions to get data from the database:

```
<?php
function getALLfromIP($addr,$db) {
  // this sprintf() wrapper is needed, because the PHP long is signed by default
  $ipnum = sprintf("%u", ip2long($addr));
  $query = "SELECT cc, cn FROM ip NATURAL JOIN cc WHERE ${ipnum} BETWEEN start AND end";
  $result = mysql_query($query, $db);
  if((! $result) or mysql_numrows($result) < 1) {
    //exit("mysql_query returned nothing: ".(mysql_error()?mysql_error():$query));
    return false;
  }
  return mysql_fetch_array($result);
}
function getCCfromIP($addr,$db) {
  $data = getALLfromIP($addr,$db);
  if($data) return $data['cc'];
    return false;
}
function getCOUNTRYfromIP($addr,$db) {
  $data = getALLfromIP($addr,$db);
  if($data) return $data['cn'];
    return false;
}
```

```
function getCCfromNAME($name,$db) {
  $addr = gethostbyname($name);
  return getCCfromIP($addr,$db);
}
function getCOUNTRYfromNAME($name,$db) {
  $addr = gethostbyname($name);
  return getCOUNTRYfromIP($addr,$db);
}
?>
```

To use this, use the folowing PHP code:

```
<?php
require('geoip.inc.php');

$db = mysql_connect("localhost","username","password") or die ("mysql_connect() failed: " . mysql_error());
mysql_select_db("geoip",$db) or die ("mysql_select_db() failed: " . mysql_error());

$remote = $_SERVER['REMOTE_ADDR'];

echo "<p>".getCCfromIP($remote,$db)."</p>\n";
echo "<p>".getCOUNTRYfromIP($remote,$db)."</p>\n";

?>
```