

IPGEO system for pure360

Project description	2
Database design	3
database Indexes	3
APPLICATION DESIGN	4
Data import	4
The REST API	5

Project IPGEO for pure 360

Project description

Introduction

This task is intended for prospective PHP Developers to demonstrate capability in the following skills:

- Retrieving and working with remote resources
- Building and populating databases
- Structuring PHP applications
- Designing and serving REST APIs
- Use of Github (public or private repo) or other public code repository service
- Documenting web services and applications

Output

The output of the task should include the following:

1. A process for constructing a database to hold the GeoLite Country reference data
2. A process that checks if the GeoLite Country database is populated and if not, downloads the GeoLite Country database file from <http://geolite.maxmind.com/download/geoip/database/GeoIPCountryCSV.zip> and populates the database from the data file
3. An API that supports a restful GET endpoint that returns the country when supplied with an IP address (GET /locationByIP?IP=127.0.0.1)
4. An accessible online source code repository containing the output of this task and appropriate documentation

Support

Please don't hesitate to ask if any of the instructions are unclear to you. Questions can be sent to the following address and will be answered as quickly as possible:

michael.pinnell@pure360.com

Database design

mysql server details

server: dty.mine.nu:3306

user: pure360

password: Pure360

database: pure360


table: ipgeo

sql file that create the table on the database: ipgeo.sql

The table ipgeo has the data from the cvs. This table we will be used to query data for our REST API. To do this faster two indexes got created for columns a and b (that have the ip information in numeric form).

The information that we store on the database is.

- ip1 , as variable , tell us from what ip the subnet starts
- ip2 , as variable , tell us from what ip the subnet stops
- a , as big integer , has the ip1 in numeric form
- b , as big integer , has the ip2 in numeric form
- country code , 3 char var
- country , var

Name	Type	Length	Decimals	Not Null	Key
id	int	11	0	<input checked="" type="checkbox"/>	
ip1	varchar	15	0	<input type="checkbox"/>	
ip2	varchar	15	0	<input type="checkbox"/>	
a	bigint	10	0	<input type="checkbox"/>	
b	bigint	10	0	<input type="checkbox"/>	
countryCode	varchar	4	0	<input type="checkbox"/>	
country	varchar	50	0	<input type="checkbox"/>	

database Indexes

To make the database query faster we use two unique type indexes for columns a and b.

Its a unique type, btree method, as we are dealing with unique numbers.

APPLICATION DESIGN

Data import

This php software imports the data on the online zipped cvs to a mysql database.

Located:

/import.php

How to run:

From the terminal type

```
php -f import.php
```

This application will check theres data on the database, then if the cvs exsist, then if we need to download the file and unzip ip.

To unzip we use the ziparchive php class. <http://php.net/manual/en/class.ziparchive.php>

We use **1 query to insert all data** and we filter the input of the country name.

Once the import process get completed then there a **verification phase** that will check if the records on the CVS is equal to the number of records on the database.

If a step fail you will see the step that failed. Its better to run from terminal.

First run example, with no data at all.

```
pro:pure360 djazz$ php -f import.php
Connecting to database ... DONE
No data has been found on the database
Deleting all records from the database before adding data from the cvs
Downloading, and unzipping the file ...
Please wait, im downloading file ... DONE
Unzipping the file ... GeoIPCountryCSV.zip extracted to /Users/djazz/pure360 DONE
Please wait, inserting data to database ... DONE
VERIFICATION Stage ... PASSED
pro:pure360 djazz$
```

Running after truncating the ipgeo table but we still have the cvs file.

```

pro:pure360 djazz$ php -f import.php
Connecting to database ... DONE
No data has been found on the database
Deleting all records from the database before adding data from the cvs
Downloading, and unzipping the file ...
The file has already been downloaded and unzipped
Please wait, inserting data to database ... DONE
VERIFICATION Stage ... PASSED
pro:pure360 djazz$

```

Running in case that its all done.

```

pro:pure360 djazz$ php -f import.php
Connecting to database ... DONE
The data is already on the database
pro:pure360 djazz$

```

The REST API

How to run:

from the testing server run

`/rest.php?ip=64.35.63.67`


It uses `$_GET[ip]` as input to get the ip to check, then it runs this mysql query and we have the answer in a json output.

The query the brings the data from ipgeo and its using the a and b indexes

```

$result = mysqli_query($connection,"SELECT * FROM ipgeo USE INDEX (a,b)
WHERE `a` <= $iplong AND `b` >= $iplong LIMIT 0,1");

```

Name	Type	Length	Decimals	Not Null	Key
id	int	11	0	<input checked="" type="checkbox"/>	
ip1	varchar	15	0	<input type="checkbox"/>	
ip2	varchar	15	0	<input type="checkbox"/>	
a	bigint	10	0	<input type="checkbox"/>	
b	bigint	10	0	<input type="checkbox"/>	
countryCode	varchar	4	0	<input type="checkbox"/>	
country	varchar	50	0	<input type="checkbox"/>	

If we use the REST API from our browser we will get this for an output.

A screenshot of a web browser window. The address bar shows 'localhost/test.php?ip=62.35.63.67'. The page content displays a JSON array: [{"0":"12959","id":"12959","1":"62.34.0.0","ip":"62.34.0.0","2":"62.35.255.255","ip2":"62.35.255.255","3":"1042415616","a":"1042415616","4":"1042546667","b":"1042546667","5":"FR","countryCode":"FR","6":"France","country":"France"}].

```
[{"0":"12959","id":"12959","1":"62.34.0.0","ip":"62.34.0.0","2":"62.35.255.255","ip2":"62.35.255.255","3":"1042415616","a":"1042415616","4":"1042546667","b":"1042546667","5":"FR","countryCode":"FR","6":"France","country":"France"}]
```

The REST API will also check if the \$_GET input is valid, by converting the ip to a number and checking if it is really a number.

In case that's a wrong ip or the ip couldn't be found on the database, we will receive an error.