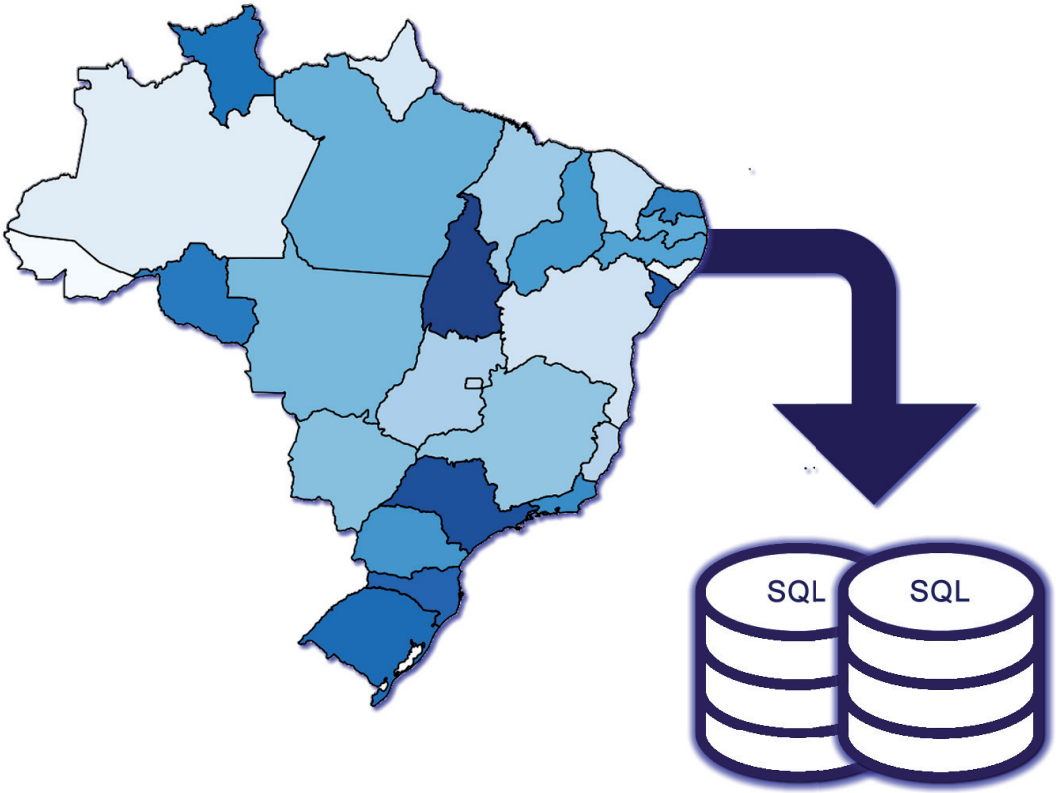
Documentos

ISSN 1518-4277
Setembro, 2017

212

Metodologia Simplificada de Conversão de SHP para SQL

Metodologia para Conversão de Arquivo *Shape* para Importação em Banco de Dados MySQL



Embrapa

ISSN 1518-4277
Setembro, 2017

Empresa Brasileira de Pesquisa Agropecuária
Embrapa Milho e Sorgo
Ministério da Agricultura, Pecuária e Abastecimento

Documentos 212

Metodologia Simplificada de Conversão de SHP para SQL

Metodologia para Conversão de Arquivo *Shape* para Importação em Banco de Dados MySQL

Ricardo Nunes Nery
Elena Charlotte Landau
André Hirsch
Daniel Pereira Guimarães

2ª edição revista e ampliada

Embrapa Milho e Sorgo
Sete Lagoas, MG
2017

Esta publicação está disponível no endereço:
https://www.embrapa.br/milho-e-sorgo/publicacoes

Embrapa Milho e Sorgo
Rod. MG 424 Km 45
Caixa Postal 151
CEP 35701-970 Sete Lagoas, MG
Fone: (31) 3027-1100
Fax: (31) 3027-1188
www.embrapa.br/fale-conosco

Revisão de texto: Antonio Claudio da Silva Barros
Normalização bibliográfica: Rosângela Lacerda de Castro
Tratamento de ilustrações: Tânia Mara Assunção Barbosa
Editoração eletrônica: Tânia Mara Assunção Barbosa
Foto(s) da capa: Ricardo Nunes Nery

2ª edição revista e ampliada
Formato digital (2017)

Dados Internacionais de Catalogação na Publicação (CIP)
Embrapa Milho e Sorgo

Metodologia simplificada de conversão de SHP para SQL: metodologia para conversão de arquivo shape para importação em banco de dados MySQL / Ricardo Nunes Nery ... [et al.]. -- 2. ed. rev. ampl. -- Sete Lagoas: Embrapa Milho e Sorgo, 2017.
28 p. : il. -- (Documentos / Embrapa Milho e Sorgo, ISSN 1518-4277; 212).

1. Linguagem de programação. 2. Banco de dados. 3. Sistema de Informação Geográfica. 4. Sensoriamento remoto. I. Nery, Ricardo Nunes. II. Série.

CDD 005.133 (21. ed.)

Autores

Ricardo Nunes Nery
Graduado em Sistemas de Informação; bolsista FAPEMIG/CNPq na Embrapa Milho e Sorgo, Sete Lagoas, MG e graduando em Engenharia Agronômica da Universidade Federal de São João del-Rei – Campus Sete Lagoas, Rodovia MG 424 – Km 47. CEP: 35701-970 Sete Lagoas, MG, ricardonunesnery@yahoo.com.br

Elena Charlotte Landau
Bióloga, DSc. Pesquisadora em Zoneamento Ecológico-Econômico, Geotecnologias e Agroclimatologia, Pesquisadora na Embrapa Milho e Sorgo, Rod MG 424 Km 45, Zona Rural, CEP 35701-970, Sete Lagoas, MG, charlotte.landau@embrapa.br

André Hirsch
Professor Adjunto da Universidade Federal de São João del-Rei – Campus Sete Lagoas, Rodovia MG 424 – Km 47, CEP 35701-970 Sete Lagoas, MG, hirsch_andre@ufsj.edu.br

Daniel Pereira Guimarães
Eng.-Florestal, D.Sc. em Manejo Florestal, Pesquisador da Embrapa Milho e Sorgo, Sete Lagoas, MG, Rod MG 424 Km 45, Zona Rural, CEP 35701-970, Sete Lagoas, MG, daniel.guimaraes@embrapa.br

Apresentação

O formato "shape" ou SHP representa um conjunto de arquivos digitais frequentemente usado para elaboração de mapas temáticos utilizando sistemas de informações geográficas, para representação de informações como localização geográfica, tipo de feição vetorial (ponto, linha, polígono) e atributos associados. SQL (Structured Query Language ou Linguagem de Consulta Estruturada) é a linguagem de pesquisa declarativa padrão para banco de dados relacional. A organização de informações em Sistemas Gerencias de Banco de Dados (SGBD) oferece inúmeros benefícios, mas, em alguns casos, transformar a informação para a entrada no SGBD é um desafio. Neste documento é descrita a metodologia para realizar a transformação de dados vetoriais armazenados em arquivos SHP para a inclusão através do SQL em um SGBD utilizando a linguagem de programação PHP (PHP: Hypertext Preprocessor), possibilitando a geração de conteúdo para acesso dinâmico.

Antonio Alvaro Corsetti Purcino
Chefe-Geral
Embrapa Milho e Sorgo

Sumário

Metodologia Simplificada de Conversão de SHP para SQL

Ricardo Nunes Nery

Elena Charlotte Landau

André Hirsch

Daniel Pereira Guimarães

Introdução

É grande a utilização do Sistema Gerenciador de Banco de Dados (SGBD) MySQL na *web*, pois grande parte das necessidades operacionais de um sistema *web* são viáveis com ele. Segundo Powers (2013), o MySQL foi considerado o SGBD de código aberto, no ano 2013, mais escolhido pelos desenvolvedores. O MySQL é um servidor de banco de dados SQL (*Structured Query Language* - Linguagem Estruturada para Pesquisas), considerado por MySQL AB (2006) muito rápido, multitarefa e multi-usuário.

O arquivo shapefile (SHP), ou arquivo *shape*, foi criado pela empresa ESRI, e possui ampla utilização para armazenamento da posição, formato e atributos de feições geográficas (ESRI, 2015). Desta forma é um formato de arquivo importante e muito comum na utilização de Sistema de Informação Geográfica (SIG).

Desenvolvido desde 2002, o Quantum GIS, ou QGIS, é um SIG de código aberto capaz de desempenhar inúmeras tarefas, podendo ser destacada a sua capacidade de suportar um grande número de formatos de dados *raster* e vetoriais (QGIS PROJECT, 2015).

O gvSIG é um SIG também de código aberto, e é considerado por Carrera (2015) como um sistema capaz de orientar, editar, analisar e gerir informações do ponto de vista das relações espaciais. O gvSIG é capaz de realizar conexões com base de dados espaciais, além de outras funcionalidades.

A denominação PHP é um acrônimo contínuo de "*PHP: Hypertext Preprocessor*", que pode ser considerada uma linguagem de *script* utilizada no desenvolvimento de páginas HTML (*HyperTextMarkupLanguage*) na *web* (ACHOUR, 2017). As linguagens de *script* podem ser compreendidas como um conjunto de códigos executados por um programa.

A Linguagem de Marcação Kheyhole, ou *KeyholeMarkupLanguage* (KML) é considerada por Wilson (2008) uma gramática XML (*eXtensibleMarkupLanguage*) capaz de codificar e armazenar informações geográficas organizadas pela leitura de diversos *softwares*.

O objetivo deste trabalho foi desenvolver um *script* em PHP para transformar arquivos SHP em código SQL, permitir inserção do código em um banco de dados relacional e possibilitar consultas SQL de pontos em polígonos. Considerando sua aplicação no setor de sensoriamento remoto, a metodologia apresentada é capaz de organizar em um SGBD as informações contidas em um *shapefile*, permitindo assim:

consultas e interação com linguagens e programas utilizando as informações vetoriais, incluindo registros com múltiplos polígonos.

Procedimentos para Conversão

Os procedimentos para a conversão de um arquivo SHP para código SQL não são diretos. Para tal operação é necessária a transformação do arquivo SHP em arquivo KML e posteriormente a conversão de KML para SQL. Desta forma, segue a metodologia para execução destas tarefas.

Conversão de SHP em KML

A conversão de SHP em KML pode ser intermediada por vários softwares, sendo o QGIS escolhido por ser um SIG de código aberto e de fácil utilização. A versão utilizada foi a 2.8.2 '*Wien*' no idioma inglês. Ao abrir o sistema, o usuário deverá carregar uma camada vetorial, carregando o arquivo SHP através do Menu: "*Layer*>*AddLayer*>*AddVectorLayer*...". O acesso ao menu pode ser visualizado na Figura 1.

Após acionar a camada vetorial, o sistema irá exibir uma tela para o usuário escolher o tipo de fonte, codificação e local do arquivo. O usuário deverá selecionar o arquivo SHP no sistema de arquivos através do botão "*Browse*" e acionar o botão "*Open*", como pode ser observado na Figura 2.

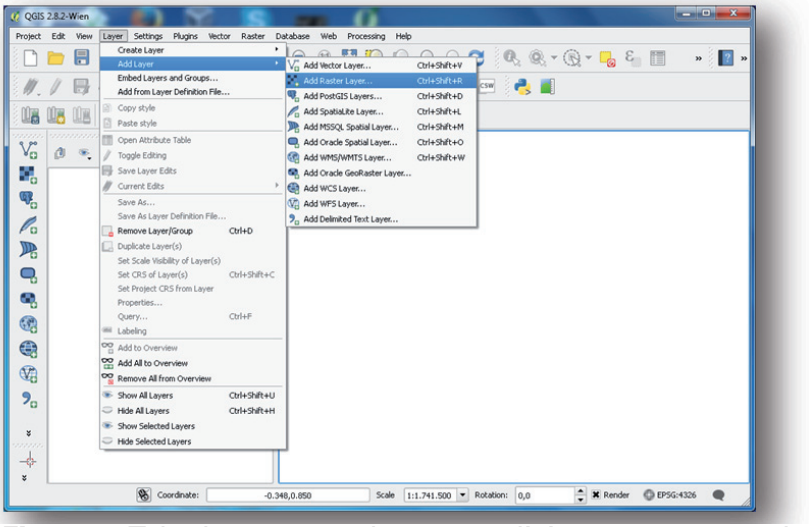


Figura 1. Tela demonstrando como adicionar uma camada vetorial.

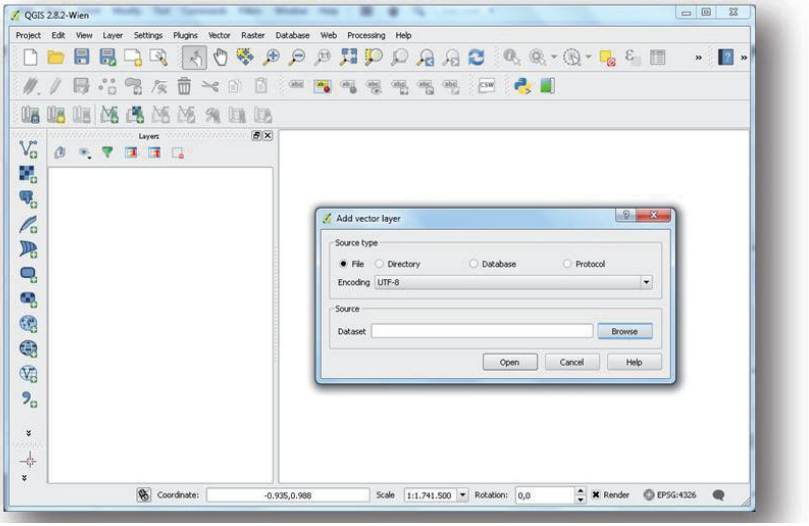


Figura 2. Tela para selecionar o arquivo.

Caso o QGIS não reconheça o sistema de coordenadas de referência do arquivo selecionado, será apresentada uma tela para a sua seleção. Neste caso, o usuário deverá saber qual foi o utilizado na confecção do arquivo *Shape.* Após selecionar o sistema de coordenadas, o usuário deverá acionar o botão "*OK*". A Figura 3 mostra a tela de seleção de sistema de coordenadas de referência.

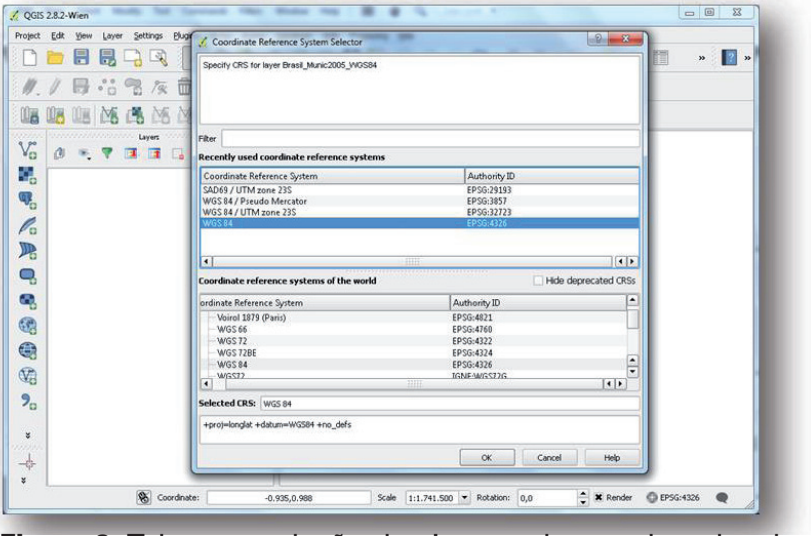


Figura 3. Tela para seleção do sistema de coordenadas de referência.

Após estes procedimentos, a camada será adicionada ao QGIS e o usuário poderá observar se o arquivo está correto. O próximo passo será a conversão do arquivo SHP em arquivo KML. O usuário deverá acionar a camada inserida com o botão direito do *mouse* e selecionar a opção "*SaveAs...*", conforme pode ser observado na Figura 4.

A opção de "Salvar Como...", ou "*SaveAs...*", irá possibilitar a escolha do tipo de arquivo; neste caso, a opção desejada é o tipo KML. Na tela de "Salvar Como..." o usuário deverá escolher na opção "*Format*" o formato do arquivo "*KeyholeMarkupLanguage* [KML]", escolher local e nome do arquivo na opção "*Browse*" e em seguida acionar o botão "*OK*". As opções e os botões podem ser observados na Figura 5. O QGIS irá armazenar o arquivo KML no local escolhido e este arquivo será utilizado para a conversão em arquivo SQL.

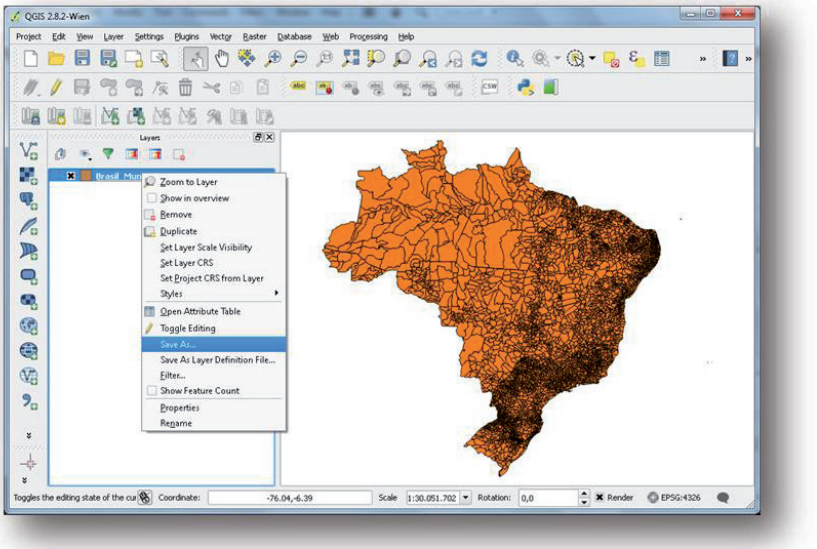


Figura 4. Tela com acionamento do "Salvar Como..." de uma camada.

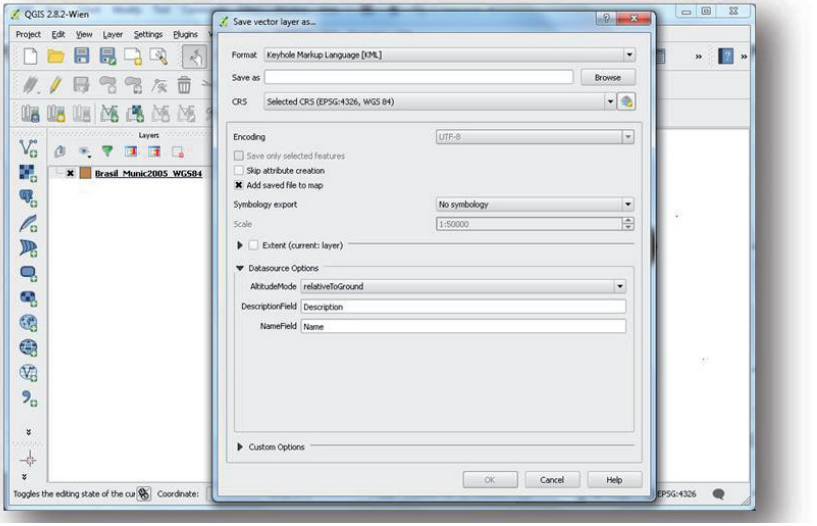


Figura 5. Tela para "Salvar Como..." de uma camada.

Conversão de Arquivo KML em Código SQL

A conversão do arquivo KML em código SQL foi realizada por um *script* na linguagem PHP. Para a utilização do *script* é necessário realizar a instalação do PHP na versão 5.6.8, existindo muitas possibilidades para esta instalação. Um exemplo simplificado da instalação do PHP no sistema operacional *Windows* pode ser o "*ApacheFriends* XAMPP Versão 5.6.8". O XAMPP é capaz de instalar o PHP, MySQL e outros *software* requeridos. Após a instalação do PHP, será possível utilizar o *script* em PHP para a conversão do arquivo KML em código SQL.

A primeira versão do *script* não suportava arquivos com múltiplos polígonos dentro de uma mesmo registro (exemplo do município do Rio de Janeiro na Figura 6), desta forma foi necessária uma reestruturação para trabalhar com esse tipo de registro. O *script* em PHP para conversão de arquivo KML em código SQL possui 105 linhas e pode ser observado na Figura 7.

A configuração para o funcionamento do *script* deve ser feita na linha 5, onde a variável "$xml" deverá receber o caminho do arquivo KML. Neste exemplo, o caminho do arquivo KML é: "kml".DIRECTORY_SEPARATOR."brazilContorno.kml"; desta forma o arquivo KML se encontra em uma pasta que está localizada junto ao *script* PHP. A utilização da constante "*DIRECTORY_SEPARATOR*" é nativa do PHP e garante que independentemente do sistema operacional utilizado irá adicionar ao caminho a barra correta, por isso, a sua utilização neste exemplo. Caso seja necessário, o usuário pode inserir o caminho completo do arquivo, indicando assim a unidade de disco e demais pastas, por exemplo: "C:/xampp/htdocs/script/kml/brazilContorno.kml".

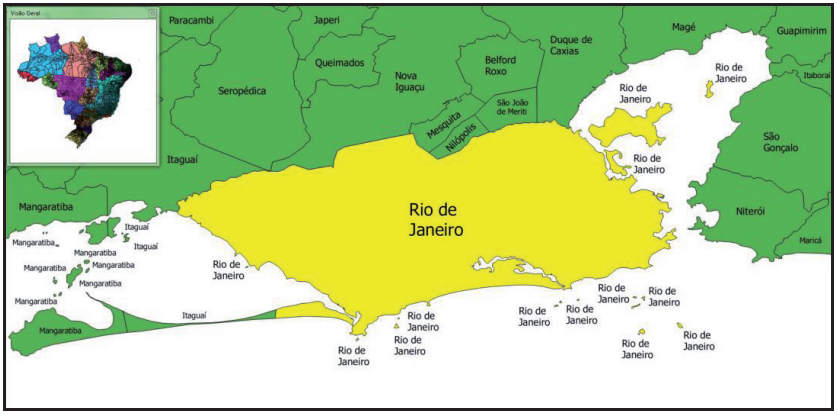


Figura 6. Exemplo de um registro com multigeometria através do municipio do rio de janeiro, representado por 18 poligonos na escala do mapa.

```
<?php
header('Content-Type: application/octet-stream');
header('Content-Disposition: attachment; filename="kmlToSql.sql"');

$xml = simplexml_load_file("kml".DIRECTORY_SEPARATOR."brazilContorno.kml"); // Local do Arquivo KML
$nomeTabela = uniqid();
$intNumeroPoligonos=0;
$stringCampos = "";
foreach ($xml->children()->children() as $root) {
 if($root->getName()=="Folder"){
  foreach ($root->children() as $placeMark) {
   if($placeMark->getName()=="Placemark"){
    foreach ($placeMark->children() as $Data) {
     if($Data->getName()=="ExtendedData"){
      if($intNumeroPoligonos==0){
       echo "CREATE TABLE IF NOT EXISTS `".$nomeTabela."` ( `id_"
       .$nomeTabela."` int(11) NOT NULL,`coordenadas_".$nomeTabela."` polygon NOT NULL,";
      }
      $stringCreateTable = "";
      $stringValores = "";
      $stringCampos = "";
      foreach ($Data->children()->children() as $ExtendedData) {
       if($intNumeroPoligonos==0){
        $stringCreateTable .= "`".$ExtendedData->attributes()->name."` varchar(350) DEFAULT NULL,";
        $stringCampos .= "`".$ExtendedData->attributes()->name."`,";
        $stringValores .= "'".iconv('UTF-8', 'ASCII//IGNORE', $ExtendedData)."',";
       }else{
        $stringCampos .= "`".$ExtendedData->attributes()->name."`,";
        $stringValores .= "'".iconv('UTF-8', 'ASCII//IGNORE', $ExtendedData)."',";
       }
      }
      if($intNumeroPoligonos==0){
       $stringCreateTable = substr($stringCreateTable, 0, -1);
       echo $stringCreateTable;
       echo ") ENGINE=InnoDB AUTO_INCREMENT=0 DEFAULT CHARSET=latin1;".PHP_EOL;
       echo "ALTER TABLE `".$nomeTabela."` ADD PRIMARY KEY (`id_".$nomeTabela."`);".PHP_EOL;
      }
     }else if($Data->getName()=="MultiGeometry"){ // Arquivo KML com Multi Poligonos
      foreach ($Data->children() as $PolyGon) {

       foreach ($PolyGon->children() as $PolyGons) {
        if($PolyGons->getName()=="outerBoundaryIs"){
         foreach ($PolyGons->children()->children() as $coordinates) {
          if($coordinates->getName()=="coordinates"){
           $coordenadasSQL = str_replace(",", "|", $coordinates);
           $coordenadasSQL = str_replace(" ", ",", $coordenadasSQL);
           $coordenadasSQL = str_replace("|", " ", $coordenadasSQL);

           $arrayCoordenas = explode(",",$coordenadasSQL);
           $coordenadasSQL = "";
           foreach($arrayCoordenas as $coordenada){
            $coordenadaTrocada = explode(" ",$coordenada);
            $coordenadasSQL .= $coordenadaTrocada[0]." ".$coordenadaTrocada[1].",";
           }
           $coordenadasSQL = rtrim (  $coordenadasSQL , "," );
           $stringValores = rtrim (  $stringValores , "," );
           $stringCampos = rtrim (  $stringCampos , "," );

           $intNumeroPoligonos++;
           $tempTexto = "INSERT INTO `".$nomeTabela."` (`id_".$nomeTabela."`,`coordenadas_"
           .$nomeTabela."`,".$stringCampos.") VALUES ("
           .$intNumeroPoligonos.",GeomFromText('POLYGON(("
           .$coordenadasSQL."))'),".$stringValores.");".PHP_EOL;;
          }

         }
        }
       }
      }else if($Data->getName()=="Polygon"){ // Arquivo KML Simples
       foreach ($Data->children() as $PolyGon) {
        if($PolyGon->getName()=="outerBoundaryIs"){
         foreach ($PolyGon->children()->children() as $coordinates) {
          if($coordinates->getName()=="coordinates"){
```

Figura 7. Código em PHP para conversão de arquivo KML em código SQL - conversor.php.

O *script* em PHP deverá ser acionado por um navegador *web*, pois ao final da execução irá gerar um arquivo SQL. O acionamento do *script* "conversor.php" pode ser observado na Figura 8 e após o processamento o sistema irá gerar um arquivo SQL, como pode ser observado na Figura 9.

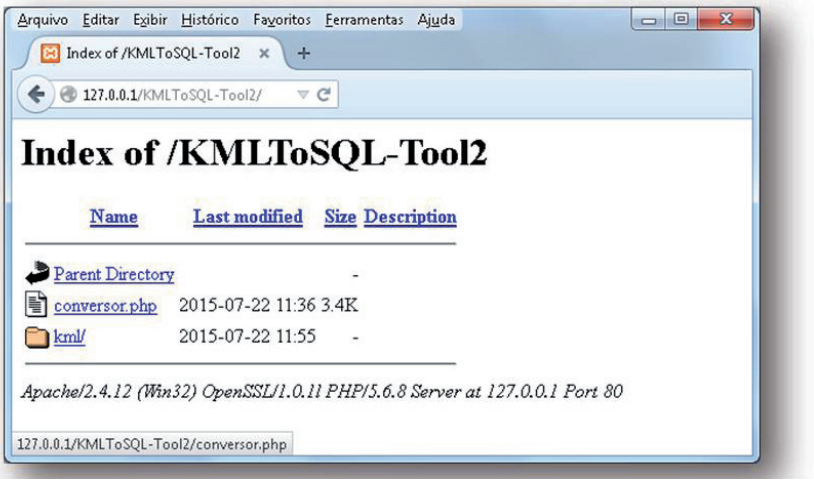


Figura 8. Tela com listagem de arquivos e link para "conversor.php".

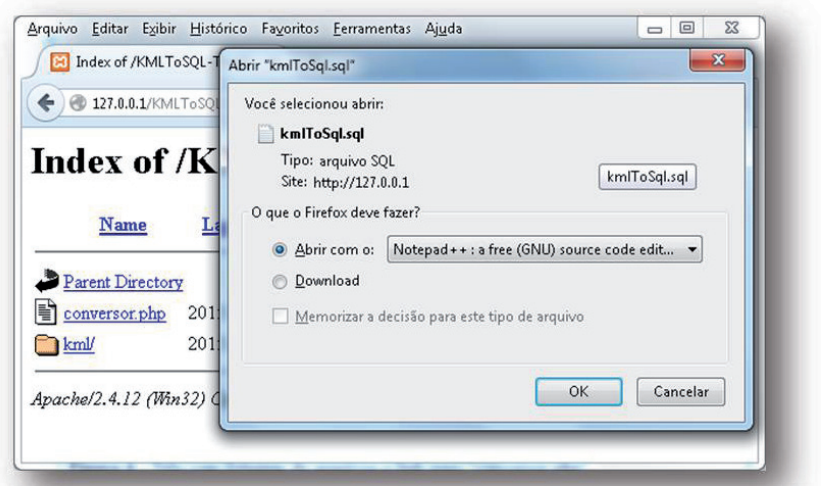


Figura 9. Tela com a opção para salvar o código SQL.

Visualização do Resultado

Visualização do KML

Utilizando o QGIS e o *script* desenvolvido em PHP é possível obter o código SQL. O código SQL pode ser importado em um SGBD como o MySQL. Após desenhar dois polígonos, dentro do Estado de Minas Gerias, no programa QGIS e realizar o procedimento de conversão de arquivo SHP para arquivo KML, podemos observar o resultado de um arquivo KML no *software Google Earth* versão 7.1.2.2041 na Figura 10.

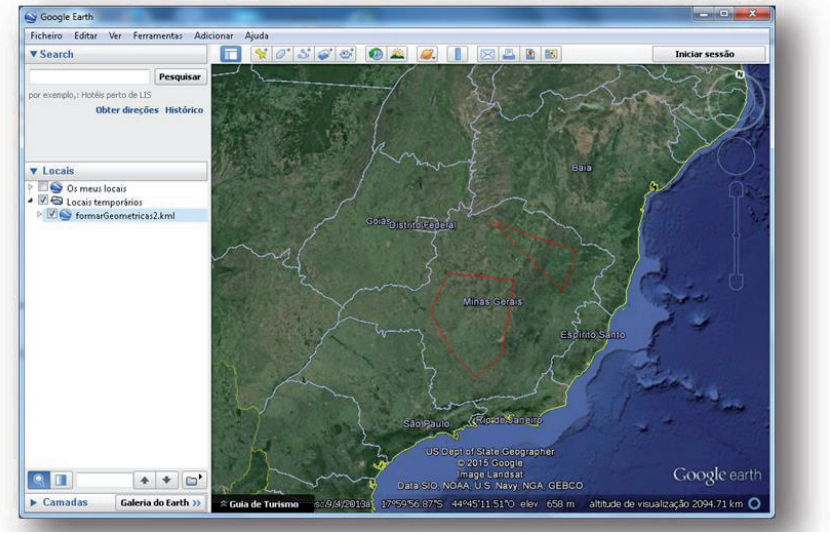


Figura 10. Tela do Google Earth com um arquivo KML aberto.

Visualização do Código SQL

O *script* em PHP é capaz de realizar a conversão de arquivos KML. Desta forma, este arquivo KML aberto no *Goole Earth* pode ser observado após a conversão para código SQL, conforme exemplificado na Figura 11.

```
CREATE TABLE IF NOT EXISTS `55afea3104455` ( `id_55afea3104455` int(11) NOT NULL,`coordenadas_55afea3104455` polygon NOT NULL,`id` varchar(350) DEFAULT NULL,`nome` varchar(350) DEFAULT NULL) ENGINE=InnoDB AUTO_INCREMENT=0 DEFAULT CHARSET=latin1;
ALTER TABLE `55afea3104455` ADD PRIMARY KEY (`id_55afea3104455`);
INSERT INTO `55afea3104455` (`id_55afea3104455`,`coordenadas_55afea3104455`,`id`,`nome`) VALUES (1,GeomFromText('POLYGON((-45.064234276840551 -15.361096343348455,-41.444257448792051 -16.054283395527953,-41.790850974881806 -17.671719850613453,-45.064234276840551 -15.361096343348455))'),"1","triangulo");
INSERT INTO `55afea3104455` (`id_55afea3104455`,`coordenadas_55afea3104455`,`id`,`nome`) VALUES (2,GeomFromText('POLYGON((-46.566139556562796 -17.633209458825704,-43.793391347844789 -17.556188675250201,-43.908922523208048 -20.521488842906948,-44.833171926114041 -21.484248637600697,-45.949973287958791 -20.675530410057945,-47.066774649803541 -19.212135522123447,-46.566139556562796 -17.633209458825704))'),"2","poligono qualquer");
ALTER TABLE `55afea3104455` MODIFY `id_55afea3104455` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=2;


/* Numero Poligonos: 2*/
```

Figura 11. Código em SQL para inserção de polígonos no SGBD.

O código SQL pode ser importado em uma base de dados no MySQL, e após a importação o *software*gvSIG versão 1.12.0 é capaz de realizar a conexão com a base de dados, exibindo a informação armazenada no SGBD. Para a visualização da tabela importada no gvSIG, basta abrir o *software* e criar um novo projeto, acessando o botão "*New*", conforme a Figura 12, e pressionar duas vezes o botão esquerdo do *mouse* sobre o nome que aparecerá dentro do quadrado denominado "*View*".

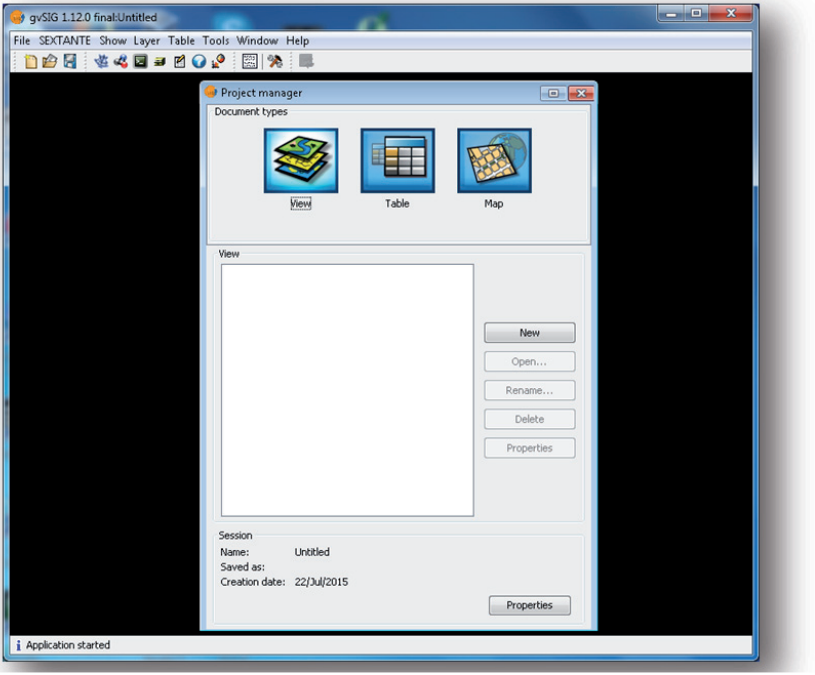


Figura 12. Tela inicial do gvSIG.

A abertura do projeto possibilitará a inclusão de camadas conexão ao SGBD, que podem ser acessadas através do Menu: "*View>AddLayer...*", conforma a Figura 13 demonstra.

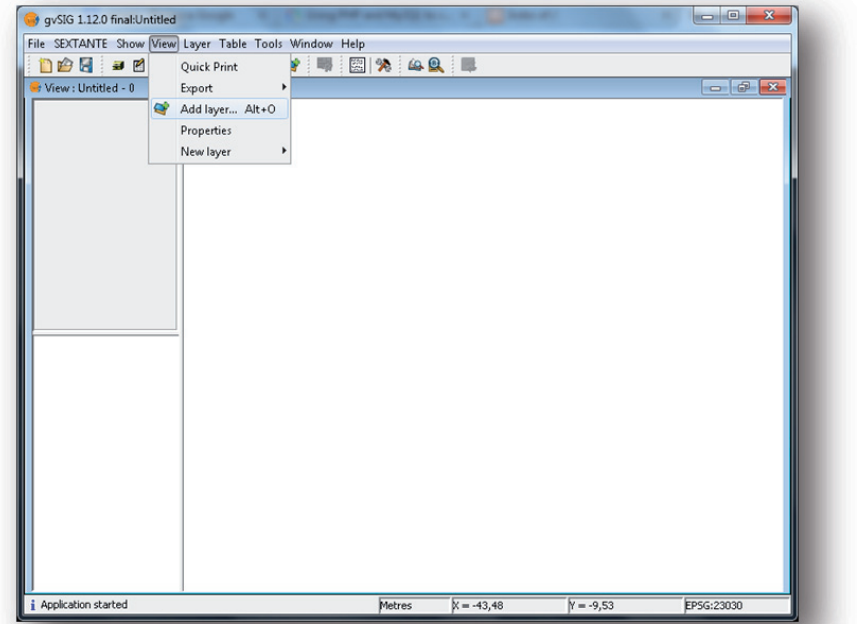


Figura 13. Tela para inserção de camada no gvSIG.

Ao acionar a inclusão de uma nova camada, o sistema irá exibir a tela de inclusão de camada. Nesta tela o usuário deverá acionar a aba "GeoDB" e o botão "*New Connection...*", conforme exemplificado na Figura 14.

As configurações para a inclusão da conexão SGBD deverão ser realizadas com as informações do servidor onde foi instalado o MySQL. As configurações padrões após a instalação do XAMPP podem ser observadas na Figura 15, lembrando que o principal nesta tela é selecionar o "*Driver*" para "*MySQL Spatial*" e o "*Database name*" que irá variar conforme o nome da base de dados criada no MySQL.

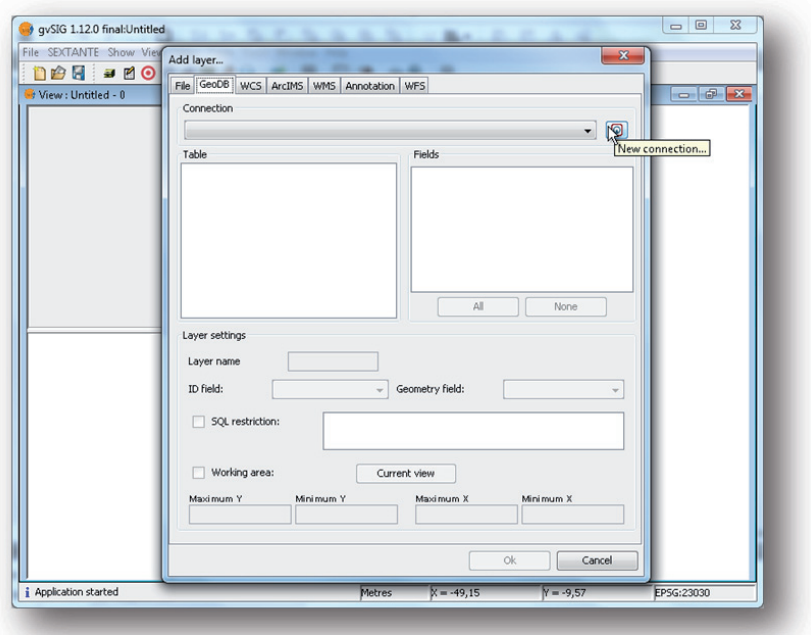


Figura 14. Tela para inserção de conexão com SGBD no gvSIG.

Caso a conexão seja realizada com sucesso, o sistema irá exibir dentro do campo "*Table*" as tabelas do SGBD conectado, e após selecionar a tabela o sistema irá carregar o nome das colunas no campo "*Fields*". O campo "*Layer settings*" irá demonstrar as configurações da camada. O código gerado em SQL e importado no SGBD preenche todos os campos no gvSIG automaticamente, como pode ser observado na Figura 16. Após a seleção da tabela, basta acionar o botão "*OK*" e o sistema irá adicionar a camada para visualização.

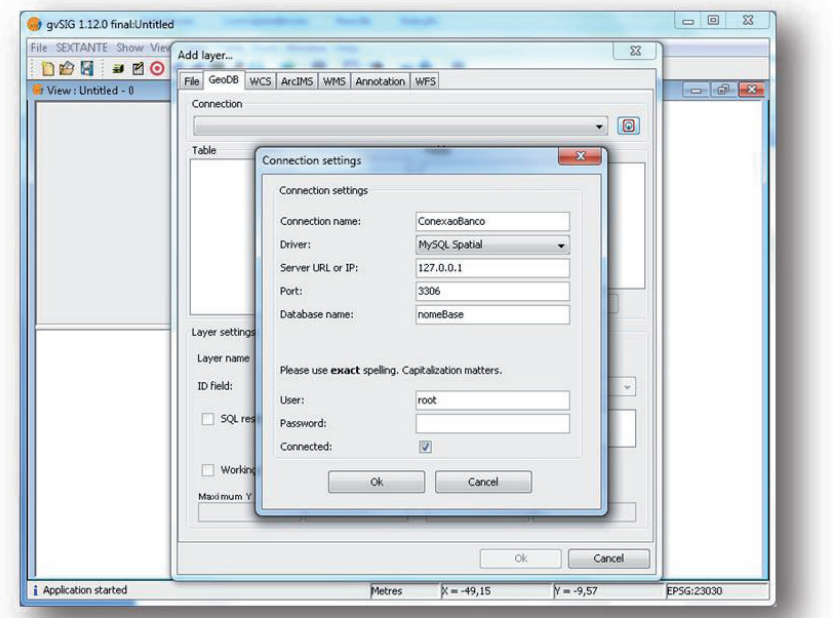


Figura 15. Tela de configuração de conexão SGBD no gvSIG.

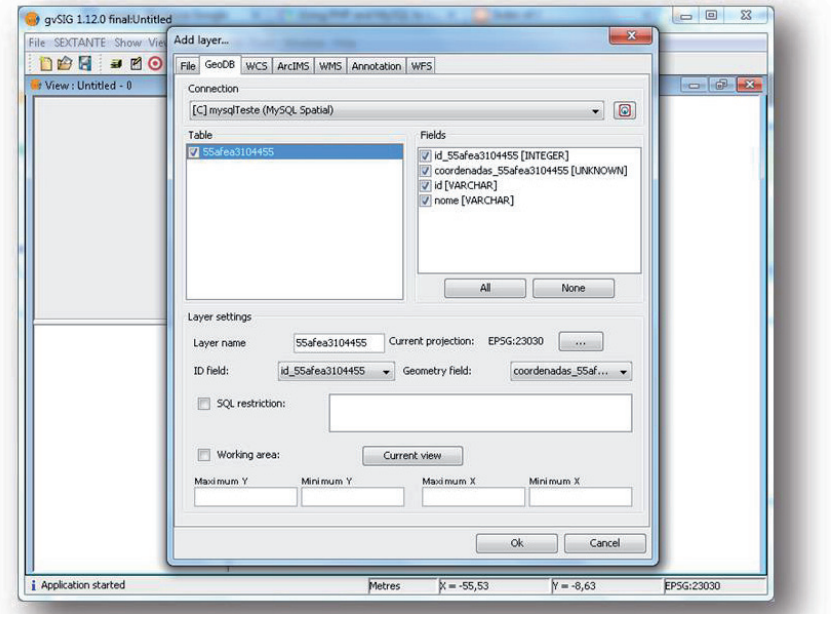


Figura 16. Tela para seleção da tabela após conexão no SGBD do gvSIG.

A exibição dos polígonos extraídos do SGBD pode ser observada na Figura 17.

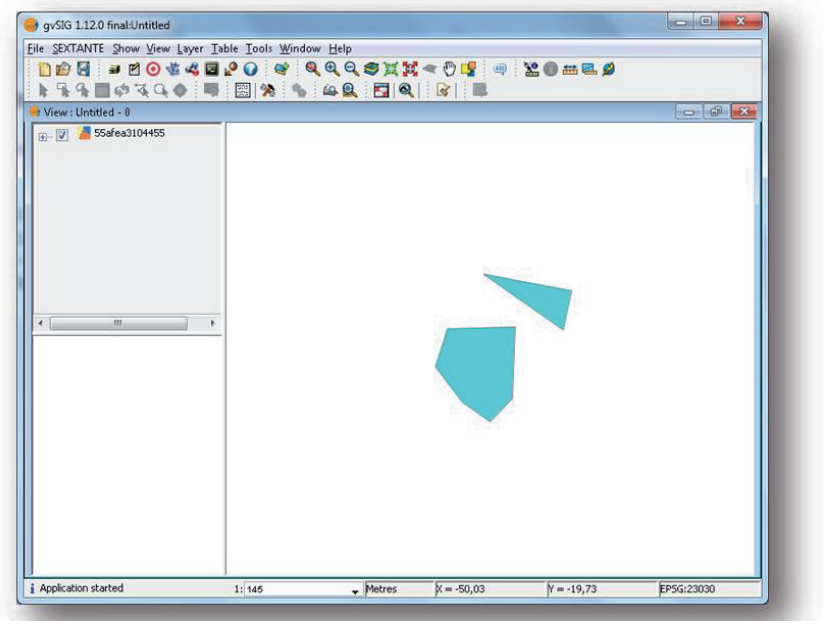


Figura 17. Tela com resultado após a conexão com o SGBD dos polígonos no gvSIG.

Consultas SQL para Polígonos Armazenados no SGBD MySQL

A consulta SQL a um polígono, armazenado no MySQL, pode ser de um ponto com latitude e longitude em graus decimais. Desta forma, o SGBD será capaz de retornar ao usuário os polígonos onde o ponto está inserido no seu interior. Por exemplo, um ponto com a latitude em graus decimais -19,8170 e longitude -43,9560 está localizado no município de Belo Horizonte - MG. Caso a base de dados tenha cadastrado os municípios do Brasil, o retorno do código exemplificado na Figura 18 seria o município de Belo Horizonte. Neste caso, a consulta está sendo feita ao polígono referente o código SQL

contido na Figura 11, e o resultado esperado é o polígono denominado "polígono qualquer".

```
SELECT * FROM 55afea3104455
WHERE ST_Contains( `coordenadas_55afea3104455` , point( - 43.9560 , - 19.8170 ) )
LIMIT 0 , 30
```

Figura 18. Código SQL para consulta de um ponto dentro de um ou mais polígonos.

A função ST_Contains utilizada para realização da consulta foi implementada pela Oracle na versão 5.6 do MySQL. Em alguns casos, a atualização do MySQL no servidor pode ser um problema de logística e a solução pode ser a substituição da função ST_Cotains nativa por uma função matemática implementada no banco. Serneels (2009) publicou em um debate em um fórum do MySQL a função "GISWithin", capaz de realizar o procedimento equivalente. A função matemática possui desempenho inferior, ou seja, demora mais tempo para ser executada em relação à função nativa do MySQL, mas a funcionalidade é semelhante. O código da função matemática "GISWithin" pode ser visualizado no Anexo A - Função *GISWithin*. A utilização da função GISWithink pode ser observada na Figura 19.

```
SELECT * FROM 55afea3104455
WHERE GISWithin( GeomFromText(CONCAT('POINT(',-43.9560,' ',-19.8170,')')), coordenadas_55afea3104455 )
LIMIT 0 , 30
```

Figura 19. Código SQL para consulta de um ponto dentro de um ou mais polígonos com a função matemática GISWithin.

Considerações Finais

A linguagem de programação PHP permite interação e conversão de tipos de arquivo, como o *script* obtido neste trabalho. A comunicação entre SGBD e SIG possui grande avanço atualmente, mas há ocorrência de erros, e a dificuldade em obter resultados com arquivos de grande porte se torna um desafio. Desta forma, um *script* onde o usuário tem maior controle dos procedimentos justifica esta iniciativa.

As consultas SQL de pontos em polígonos simplificam as relações espaciais dentro de um SGBD; desta forma, a utilização desta tecnologia abre um horizonte de possibilidades e trabalhos.

Referências

ACHOUR, M. **Manual do PHP**. Disponível em: <http://www.php.net/manual/pt_BR/>. Acesso em: 23 mar. 2017.

CARRERA, M. **Manual de usuário gvSIG 2.2.** Valencia: gvSIGasociaciónValencia, 2015. 422 p. Disponível em: <http://downloads.gvsig.org/download/gvsig-desktop/dists/2.2.0/docs/gvsig-2.2.0-doc-1.0.0-es.pdf>. Acesso em: 23 mar. 2017.

ESRI. **Shapefiles**: ajuda do ArcGIS online. New York, 2015. Disponível em: <http://doc.arcgis.com/pt-br/arcgis-online/reference/shapefiles.htm>. Acesso em: 23 mar. 2017.

MySQL AB. **Manual de Referência do MySQL 4.1**. Upsala, 2006. Disponível em: <http://downloads.mysql.com/docs/refman-4.1-pt.a4.pdf>. Acesso em: 23 mar. 2017.

POWERS, S. Readers' Choice Awards 2013. **Linux Journal**, 2013. Disponível em: <http://www.linuxjournal.com/rc2013?page=31>. Acesso em: 23 mar. 2017.

QGIS PROJECT. Equipe de Desenvolvimento. **QGIS UserGuide**: versão 2.8. 2015. 347 p. Disponível em: <http://docs.qgis.org/2.8/pdf/pt_BR/QGIS-2.8-UserGuide-pt_BR.pdf>. Acesso em: 23 mar. 2017.

SERNEELS, G. **Point In Polygon:** a MBRWithin() alternative. 2009. MySQL Forums. Disponível em: <http://forums.mysql.com/read.php?23,286574,286574>. Acesso em: 23 mar. 2017.

WILSON, T. (Ed.). **OGC KML**. [S.l]: Open Geospatial Consortium, 2008. 252 p. Disponível em: <http://portal.opengeospatial.org/files/?artifact_id=27810>. Acesso em: 23 mar. 2017.

Anexo A - Função *GISWithin*

```
DELIMITER //

CREATE FUNCTION GISWithin(pt POINT, mp MULTIPOLYGON) RETURNS INT(1) DETERMINISTIC
BEGIN

DECLARE str, xy TEXT;
DECLARE x, y, p1x, p1y, p2x, p2y, m, xinters DECIMAL(16, 13) DEFAULT 0;
DECLARE counter INT DEFAULT 0;
DECLARE p, pb, pe INT DEFAULT 0;

SELECT MBRWithin(pt, mp) INTO p;
IF p != 1 OR ISNULL(p) THEN
RETURN p;
END IF;

SELECT X(pt), Y(pt), ASTEXT(mp) INTO x, y, str;
SET str = REPLACE(str, 'POLYGON((',");
SET str = REPLACE(str, '))', ");
SET str = CONCAT(str, ',');

SET pb = 1;
SET pe = LOCATE(',', str);
SET xy = SUBSTRING(str, pb, pe - pb);
SET p = INSTR(xy, ' ');
SET p1x = SUBSTRING(xy, 1, p - 1);
SET p1y = SUBSTRING(xy, p + 1);
SET str = CONCAT(str, xy, ',');

WHILE pe > 0 DO
SET xy = SUBSTRING(str, pb, pe - pb);
SET p = INSTR(xy, ' ');
SET p2x = SUBSTRING(xy, 1, p - 1);
SET p2y = SUBSTRING(xy, p + 1);
IF p1y < p2y THEN SET m = p1y; ELSE SET m = p2y; END IF;
IF y > m THEN
IF p1y > p2y THEN SET m = p1y; ELSE SET m = p2y; END IF;
IF y <= m THEN
IF p1x > p2x THEN SET m = p1x; ELSE SET m = p2x; END IF;
IF x <= m THEN
IF p1y != p2y THEN
SET xinters = (y - p1y) * (p2x - p1x) / (p2y - p1y) + p1x;
END IF;
IF p1x = p2x OR x <= xinters THEN
SET counter = counter + 1;
END IF;
END IF;
END IF;
END IF;
SET p1x = p2x;
SET p1y = p2y;
SET pb = pe + 1;
SET pe = LOCATE(',', str, pb);
END WHILE;

RETURN counter % 2;

END;
//
DELIMITER ;
```

CGPE - 13922

MINISTÉRIO DA
AGRICULTURA, PECUÁRIA
E ABASTECIMENTO

BRASIL
GOVERNO FEDERAL