ANALISA KECEPATAN SPATIAL QUERY DATA SATELIT PADA BEBERAPA STORAGE ENGINE MYSQL DATABASE

I Ketut Swardika

Lab. Multimedia, Gd. EC, Program Studi Manajemen Informatika,
Jurusan Teknik Elektro, Politeknik Negeri Bali,
Bukit Jimbaran, P.O. Box 1064 Tuban Badung Bali, Phone: +62-081 339 336 460,
Email: swardika@pnb.ac.id

Abstrak: Penerapan *spatial database* untuk data *satellite remote sensing*, membutuhkan struktur data yang memungkinkan proses *spatial query* dilakukan dengan kecepatan tinggi, oleh karena volume data satelit biasanya sangat besar. Data-data satelit disimpan kedalam *spatial database* MySQL, karena MySQL adalah *open-source database server* yang sangat populer saat ini. Pemilihan tipe mesin penyimpan atau *storage engine* pada MySQL mempengaruhi *perfomance* sebuah *spatial query*. Pada paper ini, dilakukan pengujian kecepatan *spatial query* pada *storage engine* MySQL. Data pengujian berupa data radian beberapa *band* sensor MODIS pada satelit AQUA/TERRA. Sebuah *store procedure* dibuat yang akan memberikan informasi kecepatan *query*. *Spatial database* diimplementasikan dengan memilih *database table* dengan tipe *storage engine* yang mendukung kolom tipe *geometry*. Untuk mendapatkan *perfomance* yang baik, *storage engine* harus mendukung *spatial index*. *Spatial query* dilakukan dengan bantuan *spatial function* yang sudah terstandarisasi. Hasil test menunjukan tipe *storage engine* MyISAM secara umum memberikan kecepatan *spatial query* yang lebih cepat dari tipe *storage engine* InnoDB dan ditemukan *delay* yang besar pada kecepatan transfer InnoDB yang menyebabkan kecepatan *query* total menjadi lebih lama dari MyISAM. Telah dikembangkan tipe *storage engine* ARIA pada MariaDB yang memiliki keunggulan lebih baik dari MyISAM dan mendukung fitur transaksional seperti pada InnoDB.

Kata kunci: *spatial query perfomance, storage engine, database index.*

Analysis of speed of spatial queries of satellite data in MySQL Database Storage Engine

Abstract: *Spatial database applications for satellite remote sensing data which usually as big data, need a data structure that allows spatial query process performed at high speed. Spatial database contains satelite data created using MySQL server, as MySQL is the most today's popular open-source database server. The storage engine types of MySQL affect of performances of a spatial query, hence selection of storage engine type is crucial. In this paper, The researcher test the speed of spatial queries on MySQL storage engine types. Test data is the radiances of several bands of MODIS sensor of the satellite AQUA/TERRA. The rutin computer program, a store procedure is created to provides speed queries information. Spatial database is implemented by selecting a database table with storage engine that supports a column of type geometry. To get a good performance, storage engines must support spatial index. Spatial queries uses spatial function that has been standardized. Test results showed that in general the MyISAM storage engine provides speeds that are faster than InnoDB storage engine type. We found that a large delay of the transfer speed on InnoDB, that cause of total query speed becomes longer than MyISAM. Developer has developed the ARIA storage engine on MariaDB server that has advantages than MyISAM and supports features such as the InnoDB transactional.*

Keywords: *spatial query perfomance, storage engine, database index.*

I. PENDAHULUAN

1.1 Latar Belakang

Sejak diperkenalkannya *Global Positioning System* (GPS) untuk dapat digunakan oleh masyarakat secara umum, peralatan navigasi GPS sangat berkembang dan umum kita dapatkan pada peralatan *smartphone* [1]. Aplikasi yang menggunakan sensor GPS tidak lepas dari keberadaan peta-digital yang merupakan bagian dari *Spatial Information System* (SIS) [2]. Istilah SIS tidak sepopuler istilah *Geographic Information System* (GIS). Dengan GIS kita dengan mudah mendapatkan jawaban atau melakukan *query* terhadap atribut dari sebuah titik (*geometric point*) atau lokasi atau kunpulan dari *point-point* yang memiliki atribut yang sama dari sebuah tabel, misalnya dari tabel lokasi SPBU diseputar kota Denpasar. Terdapat perbedaan persepsi antara SIS

dengn GIS dalam melakukan *query*. SIS mengarah ke bentuk (*point, line-polyline, polygon* atau *vertex*) atau perhitungan geometrik, seperti *intersection, adjacency, nearest neighbour,* dan *range.* Untuk melakukan *query* lokasi SPBU diseputar kota Denpasar didapat dengan membatasi titik lokasi SPBU hanya yang ada pada potongan *polygon* dengan *id* Denpasar.

Penerapan SIS pada sistem data-data *satellite remote sensing*, membutuhkan pengembangan struktur data yang memungkinkan proses *query* dapat dilakukan dengan kecepatan tinggi, oleh karena volume data satelit biasanya sangat besar [3]. Data sensor *Moderate Resolution Imaging Spectroradiometer* (MODIS) pada satelit AQUA/TERRA NASA (*National Aeronautics and Space Administration*) yang tersimpan dalam format 16-bit dan terdiri dari 36 *band* dapat mencapai ukuran 1Gb untuk sebuah potongan *image* (*scene*). Untuk data MODIS spasial resolusi 1Km, sebuah lokasi (*point*) setara dengan luasan satu *pixel* seluas 1Km persegi. Dimensi satu *scene* sebesar 1354 x 2030 *pixel*, jadi akan terdapat 2.748.620 *record* pada tabel *database*. Untuk resolusi spasial sebesar 500m dan 250m, total *record* menjadi 5.497.240 dan 10.994.480 *record* [4]. *Query* dilakukan dengan mencari semua *point* (*record*) yang memenuhi kriteria, biasanya sebuah area atau *polygon*. Menjadi sangat penting untuk mengecek banyak *point* apakah *point-point* tersebut termasuk di dalam area tersebut apa tidak. Terdapat dua tipe umum struktur data yaitu: *tree structure* dan *block structure* [3]. Pada metode *block structure*, pemecahan kasus sebuah *point* dalam *polygon* dilakukan hanya pada *block* yang memiliki *polygon* tersebut, proses pencarian dilakukan. Pengenbangan struktur data mengarah ke struktur data *quadree*, yang mana digunakan tidak hanya untuk *query* kecepatan tinggi, tetapi juga untuk kompresi data [5].

MySQL adalah sebuah *open-source database server software* yang sangat populer saat ini di kalangan pengembang aplikasi maupun di lingkungan akademisi [6]. Dibandingkan dengan *spatial databases* yang sudah ada seperti, ESRI-GIS ataupun PostGIS, MySQL melengkapi fitur *spatial geometry* sesuai standar dari *Open Geospatial Consortium* sebagai *extensions*. MySQL mengimplementasikan *datatype geometric* dan *spatial functions* mengikuti spesifikasi dari OpenGIS *geometry model*. MySQL menggunakan struktur data *R-Trees* dengan *quadratic splitting* untuk *spatial index* pada kolom spasial. Spasial objek (*shape*, *line* dan *point*) digrup menggunakan *minimum bounding rectangle* (MBR) yang memungkinkan ukurannya menjadi sekecil mungkin. Walaupun *spatial index* tidak berdasarkan *actual geomteric*, MySQL memiliki banyak tipe *storage engine* (berupa tabel) yang mendukung fitur spasial, yaitu, MyISAM, InnoDB, NDB, BDB dan ARCHIVE [7]. Tipe *storage engine* NDB dan BDB hanya tersedia pada versi *cluster* dari MySQL. *Storage engine* ARCHIVE tidak mendukung file *index*, sehingga diganti menjadi *storage engine* ARIA dari MariaDB [8] yang kompatibel dengan MySQL. Sedangkan *storage engine* MRG_MyISAM, MEMORY dan CSV tidak mendukung tipe data geometri.

Pemilihan tipe mesin penyimpan atau *storage engine* pada MySQL *database server* memberikan keunggulan sekaligus kekurangannya. Implementasinya sangat tergantung dengan penerapannya. Pada tulisan ini, penulis melakukan pengujian kecepatan *spatial query* pada *storage engine* MySQL diatas. Data pengujian berupa data radian beberapa *band* MODIS. Sebuah *stored-procedure* (rutin program) nantinya akan melakukan perhitungan statistik NDVI (*Normalized Different Vegetation Index*) dari *band* MODIS tersebut [4]. Pengujian *query* NDVI dijalankan pada beberapa potongan *polygon* yang merupakan cakupan wilayah desa-desa di provinsi Bali. Proses *query* membutuhkan waktu proses dalam satuan *sec* (detik) yang akan dianalisa untuk dapat menarik simpulan.

1.2 Rumusan Masalah

Berdasarkan penjabaran latar belakang penelitian diatas, terdapat dua permasalahan yang ingin dipecahkan, yaitu:

1. Bagaimana cara mengimplementasikan *spatial database* dengan data radian *band* MODIS dan melakukan *spatial query* NDVI pada MySQL server?

2. Bagaimana pengaruh pemilihan tipe *storage engine* pada kecepatan *spatial query* pada MySQL server?

1.3 Tujuan

Tujuan dari penelitian ini adalah untuk membuat prosedur *spatial database* dengan data radian satelite MODIS dan melakukan pengujian pengaruh pemilihan tipe *storage engine* pada kecepatan *spatial query* pada MySQL *database server*. Prosedur berupa *script* rutin program komputer dan menganalisa hasil kecepatan *query* dengan metode statistik dari beberapa kali pengujian.

II. METODOLOGI PENELITIAN

2.1 Data

a) Data Satelit Radian MODIS

Digunakan data MODIS Level 1B *Atmospheric Product* berupa radian refleksi permukaan bumi dengan resolusi spasial 500m dengan dimensi total 2708 (baris) x 4060 (kolom). Satu *scene* data dipilih dengan tutupan awan yang paling minimal. Didapat satu *scene* pada bulan September (musim kemarau) dengan kondisi tutupan awan paling minimal diatas pulau Bali. Data (file: MYD02QHKM-A2011265 22 September 2011) di-*download* langsung dari NASA

MODIS data center [9]. Data diproses dengan beberapa tahapan yaitu, geo-referensi, *croping*, *masking* sampai format konversi menjadi tabel *spatial database* MySQL dengan *field* empat *band* MODIS (R-B-G-IR).

b) Data Polygon Point Bali

Data polygon Bali berupa data spasial batas wilayah desa-desa pada kecamatan, kota, kabupaten yang ada di Provinsi Bali. Data didapat dari organisasi *geospatial* OpenStreetMap (OSM). Data dalam format ESRI *shape file*, dikonversi ke tabel *spatial database* MySQL. Data point Bali berupa data lokasi-lokasi desa, kecamatan, kota, kabupaten yang ada di Provinsi Bali. Data didapat dan diproses sama dengan data polygon Bali. Data menunjukkan lokasi pusat desa atau kota dengan atribut tipe kota dan populasi.

2.2 Metode Perancangan

Spatial database dibuat dengn nama dbspatial, dengan *collation* utf8_general_ci. *Collation* adalah *set-rule* untuk membandingkan karakter set *encoding*. *Geometry* pada *spatial database* menggunakan proyeksi *latitude*, *lomgitude* dengan *datum* WGS84. Pada *database* terdapat tiga tabel dengan nama; balidesa, baliplace dan baliraddata dengan struktur tabel dan *index* seperti dibawah ini (Tabel 1 sampai dengan Tabel 3). *Stored-Procedure* merupakan rutin program tersimpan yang berisi *spatial query* yang akan menghitung besarnya nilai NDVI pada setiap *point* pada *polygon* terpilih.

A. Struktur Tabel

1. Tabel balidesa

Tabel 1a. Struktur Tabel balidesa

Column	Type	Null
*OGR_FID	int(11)	No
SHAPE	geometry	No
admncode	varchar(10)	Yes
provcode	varchar(2)	Yes
pkabcode	varchar(4)	Yes
pkkecode	varchar(7)	Yes
aream	double(19,2)	Yes
perim	double(19,2)	Yes
provname	varchar(30)	Yes
kabname	varchar(30)	Yes
kecname	varchar(30)	Yes
desaname	varchar(64)	Yes

*(unique, AutoIncrement)

Tabel 1b. Struktur Index balidesa

Keyname	Type	Unique	Packed	Column
OGR_FID	BTREE	Yes	No	OGR_FID
SHAPE	SPATIAL	No	No	SHAPE (32)

2. Tabel baliplace

Tabel 2a. Struktur Tabel baliplace

Column	Type	Null
*OGR_FID	int(11)	No
SHAPE	geometry	No
name	varchar(48)	Yes
type	varchar(16)	Yes
population	int(10)	Yes

*(unique, AutoIncrement)

Tabel 2b. Struktur Index baliplace

Keyname	Type	Unique	Packed	Column
OGR_FID	BTREE	Yes	No	OGR_FID
SHAPE	SPATIAL	No	No	SHAPE (32)

3. Tabel baliraddata

Tabel 3a. Struktur Tabel baliraddata

Column	Type	Null
*OGR_FID	int(11)	No
SHAPE	geometry	No
x	int(11)	Yes
y	int(11)	Yes
lat	double(19,5)	Yes
lon	double(19,5)	Yes
b1	double(19,5)	Yes
b2	double(19,5)	Yes
b3	double(19,5)	Yes
b4	double(19,5)	Yes

*(unique, AutoIncrement)

Tabel 3b. Struktur Index baliraddata

Keyname	Type	Unique	Packed	Column
OGR_FID	BTREE	Yes	No	OGR_FID
SHAPE	SPATIAL	No	No	SHAPE (32)

B. MODIS NDVI

Band atau kanal MODIS resolusi 500m terdiri dari empat kanal yaitu; band_1 (620-670nm) kanal merah, band_2 (841-876nm) kanal inframerah dekat, band_3 (459-479nm) kanal biru, band_4 (545-565nm) kanal hijau. Untuk menghitung indek vegetasi (NDVI) dilakukan dengan rumus sebagai berikut:

$$NDVI = \frac{band2 - band1}{band2 + band1} \quad (1)$$

Di mana band2 adalah kanal inframerah dekat atau *near infrared* (NIR) dan band1 adalah kanal merah. Komposit Band_1, band_4 dan band_3 digunakan untuk pembuatan R-G-B *true color image* MODIS.

C. Stored-Procedure

Pada *Stored-Procedure* terdapat set parameter penentu pemilihan lokasi, dari Provinsi sampai dengan Desa. Setelah set parameter lokasi ditentukan, Perintah SELECT dilakukan untuk mendapatkan data geometri *polygon* (SHAPE) pada tabel balidesa. Selanjutnya perintah SELECT kedua dijalankan guna menseleksi *point-point* yang ada didalam *polygon*. ST_CONTAINS(*par1*, *par2*) [10] adalah *built-in* fungsi spasial yang akan memberikan kondisi *true*(1) atau *false*(0), jika *polygon par1* mencangkupi *point par2*. Untuk dapat melakukan *query* pada banyak *polygon*, misalnya pada sebuah kecamatan yang terdiri dari banyak desa atau *polygon*, diperlukan fungsi ST_UNION [10] untuk menggabungkan banyak *polygon* tersebut menjadi satu sebelum melakukan *query* dengan fungsi ST_CONTAINS.

Listing Stored-Procedure diberikan dibawah ini.

```
DELIMITER $$
USE `dbspatial`$$
DROP PROCEDURE IF EXISTS
`spatial_query`$$
DROP TEMPORARY TABLE IF EXISTS
`gabung`$$
CREATE DEFINER=`root`@`localhost`
PROCEDURE `spatial_query`(carkab
CHAR(30),carkec CHAR(30),cardesa
CHAR(30))
BEGIN
SET @kab=carkab; SET @kec=carkec;
SET@desa=cardesa;

CREATE TEMPORARY TABLE gabung (FID
INT PRIMARY KEY AUTO_INCREMENT)
SELECT SHAPE FROM balidesa WHERE
kabname=@kab OR kecname=@kec OR
desaname=@desa
ORDER BY OGR_FID;

SET @jum= (SELECT COUNT(*) FROM
gabung);
SET @poly=(SELECT SHAPE FROM gabung
WHERE FID=1);

SET @n=2;
WHILE @n <= @jum DO
 SET @npoly=(SELECT SHAPE FROM
gabung WHERE FID=@n);
 SET @poly=(SELECT
ST_UNION(@poly,@npoly));
 SET @n=@n+1;
END WHILE;

DROP TEMPORARY TABLE IF EXISTS
gabung;
SELECT OGR_FID, b1, b2, (b2-b1)/(b2+b1)
AS NDVI, ASTEXT(SHAPE) FROM
baliraddata WHERE ST_CONTAINS(@poly,
SHAPE) ORDER BY OGR_FID;

END$$
DELIMITER ;
```

2.3 Test-Data

Test-data, diperlihatkan pada Tabel 4, digunakan untuk melakukan pengujian apakah *stored-procedure spatial query* diatas berjalan dengan benar. Test-data berupa data (*input-output*) yang akan dibandingkan dengan data keluaran *stored-procedure spatial query* diatas.

Tabel 4. Tabel Input Output Test-Data

Input	@desa='RENON'
Ouput	OGR_FID=672, POLYGON(115.2428206707018- 8.690632936112529,115.24359918186947 -8.690545931957486,115.24360667529092 -8.690654938608382,115.24334717156836 -8.69166544409235,115.24285869101465 - 8.693290938194774,115.24244667856368 -8.6948174336387,115.24208068276118 - 8.696080930106447,115.24194318602136 -8.696424445644606,115.24140916658766 -8.697723447854237,115.24066168522558 -8.699311449703721) (Total 1 row , tabel balidesa) OGR_FID= 83467 83468 83507 83508 83509 83510 83653 83654 83655 83656 83728 83729 83730 83731 83811 83812 83909 (Total 17 row, tabel baliraddata)

Pada Tabel 4 diatas, dengan *input* sama dengan 'RENON' pada variabel nama desa, *output* harus menunjukkan POLYGON Desa Renon saja dan memberikan POINTS yang ada pada POLYGON

tersebut dengan OGR_FID sebanyak 17 POINT. Dengan parameter masukan (*input*) hanya pada kolom desa sama dengan RENON, maka *stored-procedure* dijalankan dengan perintah:

```
CALL spatial_query (‘<blank>’,’<blank>’,’RENON);
```

2.4 Metode Analisis

Tabel-tabel *database* dengan tipe *storage engine* yang berbeda-beda, disiapkan dengan cara menduplikasi, diperlihatkan pada tabel 5 dibawah ini,. Dengan tipe *storage engine* yang berbeda-beda, ukuran (Size) setiap tabel bervariasi. *Spatial query* yang merupakan *stored-procedure* di atas dijalankan dan hasil waktu proses dicatat dan dianalisis. Karena dijalankan pada mesin (*hardware*) yang sama dan *engine* MySQL versi 5.7 yang sama berjalan pada PC Windows7, parameter yang memengaruhi kecepatan *query* dapat dipastikan terbatas pada tipe *Storage Engine* saja.

Tabel 5. Kondisi Tabel Pada Masing Masing Tipe *Storage Engine*

Table	Rows	Type	Size
balidesa	711	MyISAM	783.3 KiB
balidesa_ar	711	ARIA	888 KiB
balidesa_in	711	InnoDB	1.5 MiB
baliplace	993	MyISAM	151.3 KiB
baliplace_ar	993	ARIA	192 KiB
baliplace_in	993	InnoDB	112 KiB
baliraddata	88,979	MyISAM	19.3 MiB
baliraddata_ar	88,979	ARIA	16.7 MiB
baliraddata_in	88,979	InnoDB	11.5 MiB
9 tables	272,049	--	50.1 MiB

Oleh karena tipe *storage engine* ARIA hanya terdapat pada MariaDB versi 10.1 yang digunakan (kompatibilitas dengan MySQL) dan tipe *storage engine* InnoDB tidak mendukung spatial index pada MariaDB maka, analisis hasil akan dibandingkan antar dua tipe *storage engine* yaitu antar MyISAM dengan InnoDB pada MySQL dan MyISAM dengan ARIA pada MariaDB. Tidak terdapat perbedaan ukuran besar file tabel tipe *storage engine* MyISAM yang dibuat pada MySQL server dengan yang dibuat di MariaDB, sehingga sebagai hipotesa awal dapat disebutkan tabel-tabel tersebut identik.

Untuk mendapatkan berbedaan waktu kecepatan proses (*KP*) yang signifikan, maka *query* diberikan beban yang cukup (*query* pada wilayah yang cukup luas) dengan cara parameter *input* berupa wilayah kabupaten yang ada di Provinsi Bali. Untuk mendapatkan hasil yang valid, *spatial query* dijalankan dengan kriteria masukan kabupaten, sehingga terdapat $n = 9$, sebanyak jumlah kabupaten. *Spatial query* juga diulang (iterasi) sebanyak $k = 5$ untuk mendapatkan hasil yang konvergen. Hasilnya dirata-ratakan secara komulatif dengan rumus seperti berikut ini:

$$\overline{\square\square_{\square}} = \frac{1}{\square * \square} \sum_{1}^{\square * \square} \square\square_{(\square,\square)} \qquad (2)$$

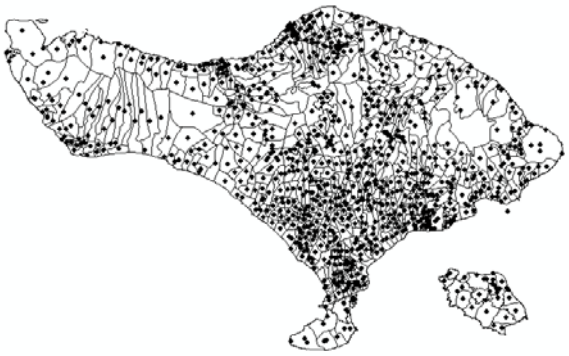
$\overline{\square\square_{\square}}$ adalah total kecepatan proses rata-rata untuk setiap tipe *storage engine* (*E*).

Parameter kecepatan *query* terdiri dari tiga kriteria yaitu: *execution*, *transfer* dan *total time* (*sec*). *Total time* merupakan penjumlahan dari *execution* dan *transfer time*. *Execution time* adalah selisih waktu saat *client* mengirim *query* ke server dengan waktu *query* dijalankan sampai server memberikan status ke *client*. *Transfer time* adalah waktu yang dibutuhkan *client* untuk mendisplay hasil pada *screen*. Waktu eksekusi (*execution*) adalah proses internal yang berhubungan dengan algoritma proses pada struktur data yang paling berpengaruh pada tipe *storage engine*, sehingga bobot analisa kecepatan diberikan lebih besar. Secara umum kecepatan akan dilihat pada *total time*. Analisa juga akan memperhitungkan ketiga kriteria kecepatan diatas dengan memisahkan hasil rata-rata per kriteria kecepatan.

III. HASIL DAN PEMBAHASAN

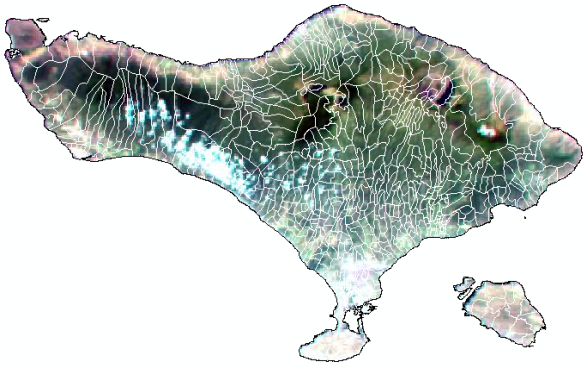
3.1 Data Visualisasi

Data-data yang digunakan, seperti pada sub-bab 2, setelah dikonversi ke *database* MySQL divisualisasikan ke dalam ArcMap (*software* GIS dari ESRI) [11] sehingga mudah dalam menginterpretasikan data, ditunjukkan pada Gambar-1 dibawah ini. Setiap potongan POLYGON pada gambar merupakan data SHAPE(*geometry*) pada tabel balidesa dan setiap POINT merupakan data SHAPE(*geometry*) pada tabel baliplace. Dengan parameter proyeksi (*latitude*, *longitude*) dan datum (WGS84) [11] yang sama antara peta ArcMap dengan tabel *database*, letak POLYGON desa dan POINT kota pada Gambar-1 terlihat tepat. Begitu juga untuk data POINT pada tabel baliraddata, menunjukkan letak posisi tepat pada peta.



Gambar 1. Visualisasi Data pada Peta ArcMap

Sedangkan data radian MODIS yang pada mulanya berupa data *raster image*, menjadi data *vector* berupa POINT pada *spatial database* yang dilengkapi dengan atribut yang memuat informasi pada POINT tersebut. *True-color* R-G-B *image* data radian satelit MODIS level 1B setelah melalui tahapan geo-referensi, *croping* dan *masking*, di-overlay dengan data polygon desa Bali (garis putih) diperlihatkan pada Gambar-2 berikut ini.



Gambar 2. *True-color* R-G-B *image* satelit MODIS

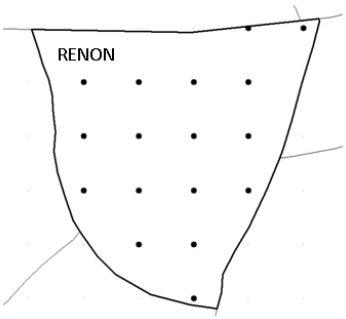
Terjadi perubahan format data dari *raster image* radian MODIS (Gambar 2) menjadi data *vector* POINT seperti yang ditunjukkan pada Gambar 3. Pada data raster, satuan terkecil berupa *pixel*, mewakili area persegi seluas 500m persegi, sedangkan pada data vector, POINT merupakan titik pusat dari pixel tersebut. Direpresentasikan hanya sebuah titik yang menandakan lokasinya, sedangkan data informasi yang lain termuat dalam atribut kedalam tabel.

3.2 Hasil Test-Data

Dengan data test sesuai dengan Tabel 4 diatas, *stored-procedure* memberikan *output query* sesuai dengan Test-Data. Terdapat 17 rows pada output dari tabel baliraddata dengan OGR_FID yang sesuai. *Output query*, ditunjukkan pada Tabel 6 divisualisasikan pada peta dengan bantuan ArcMap untuk dapat dengan mudah dalam menginterpretasikan data. Hasilnya ditunjukkan pada Gambar-3 di bawah ini. Garis hitam tebal merupakan POLYGON desa RENON dan titik-titik hitam merupakan POINTS data band-radian MODIS. Semua POINT yang terseleksi, sebanyak 17 titik, sesuai kriteria.

Tabel 6. Output Spatial Query

OGR_FID	b1	b2	NDVI	ASTEXT(SHAPE)
83467	55.73	76.48	0.157	POINT(115.24 -8.69)
83468	54.58	76.96	0.170	POINT(115.24-8.69)
83507	54.51	68.85	0.116	POINT(115.23-8.69)
83508	52.15	71.72	0.158	POINT(115.24-8.69)
83509	50.29	69.28	0.159	POINT(115.24-8.69)
83510	52.55	75.90	0.182	POINT(115.24-8.69)
83653	52.78	63.09	0.089	POINT(115.24-8.69)
83654	48.30	60.23	0.110	POINT(115.23-8.69)
83655	50.20	59.50	0.085	POINT(115.23-8.69)
83656	55.90	62.43	0.055	POINT(115.23-8.69)
83728	53.02	58.62	0.050	POINT(115.23-8.69)
83729	50.13	63.14	0.115	POINT(115.23-8.69)
83730	49.48	71.77	0.184	POINT(115.23-8.69)
83731	55.21	78.26	0.173	POINT(115.24-8.69)
83811	49.23	76.36	0.216	POINT(115.23-8.70)
83812	52.31	69.88	0.144	POINT(115.23-8.70)
83909	52.48	66.88	0.121	POINT(115.23-8.70)



Gambar 3. Peta Hasil Test-Data

Seperti pada gambar 3 diatas, posisi POINT yang terseleksi tepat berada didalam POLYGON desa RENON. Tidak ada satupun POINT yang terseleksi ada diluar POLYGON. Ini menunjukkan fungsi spatail ST_CONTAINS berkerja dengan sangat baik.

Seperti dijelaskan pada referensi *user-guide* MySQL/ MariaDB, fungsi ST_CONTAINS() menggunakan metode yang lebih baik dari fungsi CONTAINS() dalam menentukan POINT dalam POLYGON. Dimana fungsi ST_CONTAINS() menggunakan metode *object shapes*, sedangkan fungsi CONTAINS() menggunakan *object bounding rectangles*. Metode *Object shapes* membutuhkan tabel database yang mendukung *spatial index*.

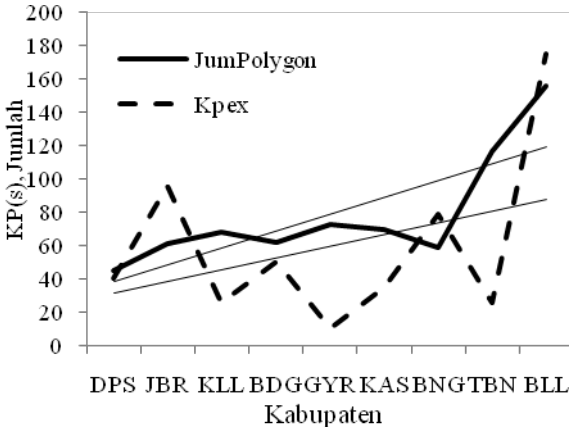
3.3 Hasil Test Kecepatan Spatial Query

Tabel 7 sampai dengan Tabel 10, menunjukan hasil pengukuran kecepatan *spatial query* pada masing-masing tipe *storage engine*. Tabel 7 dan tabel 8 merupakan hasil test pada MySQL server, sedangkan Tabel 9 dan Tabel 10 merupakan hasil test pada MariaDB server (kompatibel dengan MySQL server).

A. Korelasi Jumlah Objek

Pada tabel-tabel tersebut, terdapat sembilan Kabupaten dengan parameter jumlah POLYGON atau desa dan jumlah POINT yang bervariasi. Dilihat hubungan antara jumlah POLYGON dengan jumlah POINT. Sebagai hypotesa awal, semakin banyak jumlah POLYGON semakin luas wilayah kabupaten maka semakin banyak jumlah POINT. Garis *trend*, menunjukan kedua parameter tersebut meningkat, namum terdapat varian di mana jumlah POLYGON berkurang, jumlah POINT meningkat. Dapat dianalisis, bahwa terdapat Kabupaten yang memiliki jumlah desa yang banyak dengan luasan wilayah yang sempit atau sebaliknya.

Grafik berikut menunjukan hubungan antara jumlah POINT yang meningkat (selaras dengan jumlah POLYGON) terhadap kecepatan eksekusi (KP*ex*). Garis linier trend kecepatan eksekusi meningkat dan tidak memotong garis trend jumlah POLYGON. Jumlah POINT lebih banyak mempengaruhi kecepatan eksekusi dari pada jumlah POLYGON.

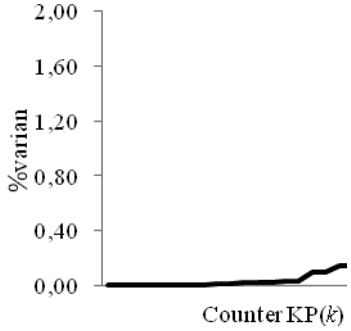


Gambar 4. Grafik pengaruh jumlah POINT terhadap $KP_{EX}(S)$

Pada grafik terlihat, nilai KP(s) terkecil atau tercepat ada pada Kabupaten Gianyar, namum jumlah POLYGON terkecil dan jumlah POINT terkecil ada pada Kabupaten Kota Denpasar.

B. Konvergensi Iterasi

Dengan tujuan didapat iterasi varian kurang dari 5% atau 95% *confidence level*, grafik berikut ini menunjukan %varian masih dibawah 5%. Jumlah iterasi sebanyak lima kali, sudah cukup memberikan hasil iterasi yang konvergen.



Gambar 5. Grafik prosentase varian iterasi

3.4 Analisa Kecepatan Spatial Query

Seperti yang ditunjukkan pada Tabel 7 dan Tabel 8, kecepatan *spatial query* (KP_E) untuk MyISAM sebesar 46.15s lebih cepat dari InnoDB yang sebesar 102.21s. Walaupun keduanya menggunakan *spatial index* pada kolom *geometry*, terdapat perbedaan KP_E yang signifikan. Kalau dilihat lebih detail, pada InnoDB kecepatan eksekusi (KP_{ex}) memberikan hasil yang lebih cepat, sebesar 17.3s lebih cepat dari MyISAM. Namum, pada proses transfer hasil *query* ke *client*, kecepatan tranfer (KP_{tr}) untuk InnoDB terdapat keterlambatan yang mencolok, lebih dari 100s dibandingkan dengan MyISAM. Hal ini yang menyebabkan kecepatan *query* total (KP_{tt}) menjadi sangat lambat pada InnoDB. Kemungkinan besar lambatnya kecepatan tranfer pada InnoDB dibandingkan dengan MyISAM disebabkan oleh adanya *report* transaksional yang harus diproses dan disampaikan ke *client*. Di mana fitur ini tidak terdapat pada MyISAM. Namun, MyISAM memiliki *perfomance* yang bagus dan ringan.

Sedangkan pada Tabel 9 dan Tabel 10 kecepatan *spatial query* (KP_E) untuk ARIA sebesar 35.146s lebih cepat dari MyISAM yang sebesar 35.824s. Selisih kecepatan hanya sebesar 0.678s, kurang dari 1s. Namum hal ini bisa sangat berpengaruh jika data yang ditangani sangat besar. Dilihat lebih detail pada ketiga kecepatan *query*, yaitu KP_{ex} ,KP_{tr} dan KP_{tt} ,ARIA unggul pada ketiga kategori tersebut. Khusus pada KP_{ex}, ARIA unggul lebih dari 1s secara signifikan dari MyISAM.

Tabel 7.Hasil Test Kecepatan Query pada MyISAM-MySQL server

Iterasi (k)	Jembrana (59 Desa, 13350 Point)			Tabanan (117 Desa, 13596 Point)			Badung (62 Desa, 6066 Point)			Gianyar (68 Desa, 5754 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	2 m 19	1.053	2 m 20	22.847	1.069	23.916	42.562	0.469	43.031	8.341	0.437	8.778
2	2 m 19	1.053	2 m 20	22.518	1.071	23.589	42.291	0.455	42.747	8.145	0.440	8.586
3	2 m 19	1.062	2 m 20	22.518	1.056	23.575	44.213	0.454	44.667	8.155	0.435	8.591
4	2 m 19	1.039	2 m 20	22.518	1.080	23.598	42.193	0.453	42.647	8.519	0.437	8.957
5	2 m 18	1.052	2 m 19	22.486	1.062	23.548	42.360	0.465	42.826	8.142	0.438	8.580
KP(s)	138.800	1.052	139.800	22.577	1.068	23.645	42.724	0.459	43.184	8.260	0.437	8.698

Iterasi (k)	Klungkung (61 Desa, 4704 Point)			Bangli (73 Desa, 8234 Point)			Karangasem (70 Desa, 13055 Point)			Buleleng (156 Desa, 20379 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	19.924	0.368	20.292	13.191	0.649	13.840	38.120	1.029	39.149	5 m 32	1.609	5 m 34
2	20.020	0.359	20.380	12.573	0.638	13.212	38.041	1.045	39.086	5 m 29	1.610	5 m 30
3	19.999	0.361	20.361	12.733	0.639	13.373	38.192	1.019	39.212	5 m 29	1.605	5 m 31
4	20.016	0.356	20.373	12.568	0.664	13.232	37.943	1.022	38.965	5 m 28	1.590	5 m 30
5	20.015	0.358	20.373	12.684	0.637	13.321	38.165	1.028	39.193	5 m 27	1.602	5 m 29
KP(s)	19.995	0.360	20.356	12.750	0.645	13.396	38.092	1.029	39.121	329.000	1.603	330.800

Iterasi (k)	Denpasar (45 Desa, 2000 Point)		
	Execution (s)	Tranfer (s)	Total (s)
1	3.963	0.140	4.103
2	3.884	0.139	4.024
3	3.904	0.140	4.045
4	3.881	0.139	4.020
5	3.878	0.136	4.015
KP(s)	3.902	0.139	4.041

SUMMARY

KP_E	46.15	s
KPex	68.46	s
KPtr	0.75	s
KPtt	69.23	s

Tabel 8. Hasil Test Kecepatan Query pada InnoDB-MySQL server

Iterasi (k)	Jembrana (59 Desa, 13350 Point)			Tabanan (117 Desa, 13596 Point)			Badung (62 Desa, 6066 Point)			Gianyar (68 Desa, 5754 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	19.306	2 m 0	2 m 19	28.764	53.259	1 m 22	58.439	2 m 18	3 m 16	42.940	48.012	1 m 30
2	18.824	1 m 59	2 m 17	28.840	53.885	1 m 22	58.216	2 m 17	3 m 16	42.171	4.988	1 m 27
3	18.974	1 m 59	2 m 18	29.578	54.931	1 m 24	59.003	2 m 18	3 m 17	43.799	46.384	1 m 30
4	18.930	1 m 58	2 m 17	28.733	53.589	1 m 22	58.516	2 m 17	3 m 16	43.534	45.534	1 m 29
5	19.197	2 m 1	2 m 20	29.483	54.817	1 m 24	59.230	2 m 19	3 m 18	44.522	46.452	1 m 30
KP(s)	19.0462	119.4	138.2	29.079	54.096	82.8	58.680	137.8	196.6	43.393	38.274	89.2

Iterasi (k)	Klungkung (61 Desa, 4704 Point)			Bangli (73 Desa, 8234 Point)			Karangasem (70 Desa, 13055 Point)			Buleleng (156 Desa, 20379 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	2 m 51	44.807	3 m 35	8.797	1 m 9	1 m 17	32.288	2 m 32	3 m 4	22.153	5 m 8	5 m 30
2	2 m 51	45.427	3 m 36	8.662	50.760	59.423	31.925	2 m 32	3 m 4	22.174	5 m 6	5 m 28
3	2 m 51	44.557	3 m 36	6.908	50.461	57.369	32.321	2 m 33	3 m 5	22.171	5 m 7	5 m 29
4	2 m 53	45.062	3 m 38	6.980	51.758	58.739	32.256	2 m 32	3 m 4	22.073	5 m 7	5 m 29
5	2 m 52	44.616	3 m 36	6.942	50.704	57.646	32.053	2 m 33	3 mi 5	22.173	5 m 7	5 m 29
KP(s)	171.600	44.894	216.200	7.658	54.537	62.035	32.169	152.400	184.400	22.149	307.000	329.000

Iterasi (k)	Denpasar (45 Desa, 2000 *Point)*		
	Execution (s)	Tranfer (s)	Total (s)
1	1 m 16	7.955	1 m 24
2	1 m 16	7.975	1 m 24
3	1 m 17	7.998	1 m 25
4	1 m 17	8.006	1 m 25
5	1 m 17	7.986	1 m 25
KP(s)	76.600	7.984	84.600

SUMMARY

KP_E	102.21	s
KPex	51.15	s
KPtr	101.82	s
KPtt	153.67	s

Tabel 9. Hasil Test Kecepatan Query pada MyISAM-MariaDB server

Iterasi (k)	Jembrana (59 Desa, 13350 Point)			Tabanan (117 Desa, 13596 Point)			Badung (62 Desa, 6066 Point)			Gianyar (68 Desa, 5754 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	59.861	0.898	1 m 0	20.804	0.919	21.724	46.943	0.401	47.345	7.150	0.380	7.531
2	59.455	0.897	1 m 0	20.125	0.917	21.042	47.347	0.405	47.753	7.140	0.390	7.530
3	1 m 0	0.905	1 m 1	20.149	0.933	21.083	47.384	0.399	47.783	7.126	0.384	7.510
4	1 m 0	0.901	1 m 1	20.248	0.927	21.175	46.942	0.401	47.343	7.140	0.383	7.523
5	1 m 0	0.897	1 m 1	20.063	0.914	20.978	47.178	0.400	47.579	7.101	0.403	7.504
KP(s)	59.863	0.900	60.600	20.278	0.922	21.200	47.159	0.401	47.561	7.131	0.388	7.520

Iterasi (k)	Klungkung (61 Desa, 4704 Point)			Bangli (73 Desa, 8234 Point)			Karangasem (70 Desa, 13055 Point)			Buleleng (156 Desa, 20379 Point)		
	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)	Execution (s)	Tranfer (s)	Total (s)
1	28.786	0.310	29.096	11.555	0.552	12.107	46.785	0.878	47.664	4 m 12	1.361	4 m 13
2	28.467	0.310	28.777	11.185	0.553	11.738	46.800	0.880	47.681	4 m 17	1.368	4 m 18
3	28.576	0.311	28.888	11.130	0.556	11.686	46.658	0.878	47.537	4 m 12	1.366	4 m 14
4	28.675	0.312	28.988	11.174	0.550	11.725	46.333	0.885	47.218	4 m 13	1.364	4 m 14
5	29.038	0.314	29.353	11.246	0.553	11.800	46.546	0.890	47.437	4 m 11	1.370	4 m 12
KP(s)	28.708	0.311	29.020	11.258	0.553	11.811	46.624	0.882	47.507	253.000	1.366	254.200

Iterasi (k)	Denpasar (45 Desa, 2000 *Point)*		
	Execution (s)	Tranfer (s)	Total (s)
1	4.434	0.128	4.562
2	4.379	0.125	4.504
3	4.554	0.138	4.693
4	4.296	0.125	4.421
5	4.394	0.124	4.518
KP(s)	3.902	0.139	4.041

SUMMARY

KP_E	35.82	s
KPex	53.10	s
KPtr	0.65	s
KPtt	53.72	s

Tabel 10. Hasil Test Kecepatan Query pada ARIA-MariaDB server

Iterasi (k)	Jembrana (59 Desa, 13350 Point)			Tabanan (117 Desa, 13596 Point)			Badung (62 Desa, 6066 Point)			Gianyar (68 Desa, 5754 Point)		
	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)
1	59.642	0.549	1 m 0	19.698	0.563	20.262	46.551	0.252	46.803	6.772	0.236	7.008
2	59.590	0.548	1 m 0	19.653	0.563	20.216	46.419	0.244	46.663	6.768	0.237	7.005
3	59.674	0.547	1 m 0	19.812	0.573	20.386	46.502	0.241	46.744	6.713	0.237	6.951
4	59.754	0.548	1 m 0	19.443	0.565	20.008	46.447	0.242	46.690	6.763	0.239	7.003
5	59.993	0.561	1 m 0	19.628	0.569	20.197	46.469	0.240	46.710	6.867	0.244	7.112
KP(s)	59.731	0.551	60.000	19.647	0.567	20.214	46.478	0.244	46.722	6.777	0.239	7.016

Iterasi (k)	Klungkung (61 Desa, 4704 Point)			Bangli (73 Desa, 8234 Point)			Karangasem (70 Desa, 13055 Point)			Buleleng (156 Desa, 20379 Point)		
	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)	Execution (s)	Transfer (s)	Total (s)
1	28.847	0.192	29.040	10.682	0.336	11.018	45.872	0.541	46.414	4 m 10	0.833	4 m 11
2	28.331	0.189	28.521	10.753	0.338	11.091	45.838	0.537	46.376	4 m 10	0.843	4 m 11
3	28.412	0.190	28.602	10.571	0.336	10.908	45.761	0.544	46.305	4 m 10	0.836	4 m 11
4	28.334	0.193	28.528	10.802	0.338	11.141	45.914	0.540	46.454	4 m 10	0.839	4 m 11
5	28.372	0.193	28.566	10.703	0.342	11.045	46.119	0.551	46.671	4 m10	0.843	4 m 10
KP(s)	28.459	0.191	28.651	10.702	0.338	11.041	45.901	0.543	46.444	248.000	0.839	250.800

Iterasi (k)	Denpasar (45 Desa, 2000 Point)		
	Execution (s)	Transfer (s)	Total (s)
1	4.282	0.074	4.356
2	4.286	0.076	4.363
3	4.382	0.075	4.458
4	4.366	0.074	4.441
5	4.445	0.074	4.520
KP(s)	4.352	0.075	4.428

SUMMARY

KP_E	35.15	s
KPex	52.23	s
KPtr	0.39	s
KPtt	52.81	s

Walaupun *perfomance* kecepatan *query* pada MyISAM lebih cepat dari InnoDB, namum InnoDB memiliki kemampuan transaksional yang tidak dimiliki oleh MyISAM. Keterbatasan tersebut diperbaiki oleh ARIA dengan mengembangkan kemampuan transaksional yang dapat diset aktif atau tidak aktif. Jika dibandingkan antara MyISAM pada MySQL dengan MyISAM pada MariaDB, pada MariaDB MyISAM unggul lebih dari 10s dibandingkan MySQL. Terlihat bahwa MariaDB memang dikembangkan untuk mengungguli *perfomance* dari MySQL.

IV. SIMPULAN

1. *Spatial database* diimplementasikan dengan memilih *database table* dengan tipe *storage engine* yang mendukung kolom tipe *geometry*. *Geometry* objek dapat berupa *point*, *line*, *polygon* dan lain-lain. Kolom *geometry* diberi nama SHAPE. Untuk mendapatkan *perfomance* yang baik, *storage engine* harus mendukung *spatial index*. ObjectID menjadi OGR_FID di-index sebagai *primary key* yang bersifat unik. Atribut yang lain dapat dilengkapi dengan mengacu pada kolom OGR_FID dan SHAPE atau lokasi geografi. Atribut berupa band-band radian MODIS dan NDVI didapat dari perhitungan kolom-kolom radian. *Spatial query* dilakukan dengan bantuan *spatial function* yang sudah terstandarisasi.

2. Pemilihan tipe *storage engine* perlu diperhitungkan untuk mendapatkan *perfomance query* yang cepat. Implementasi pada *spatial database* secara umum tidak memerlukan fitur transaksional. Tipe *storage engine* MyISAM secara umum memberikan kecepatan *spatial query* yang lebih cepat dari tipe *storage engine* InnoDB. Ditemukan *delay* yang besar pada kecepatan transfer InnoDB yang menyebabkan kecepatan *query* total menjadi lebih lama dari MyISAM. Telah dikembangkan tipe *storage engine* ARIA pada MariaDB yang memiliki keunggulan lebih baik dari

MyISAM dan mendukung fitur transaksional seperti pada InnoDB.

DAFTAR PUSTAKA

[1] U.S-National Research Council, "*The global positioning system: a shared national asset: recommendations for technical improvements and enhancements"*, National Academies Press, 1995, p.16, ISBN 0-309-05283-1.

[2] Michela Bertolotto, *"Lecture Note on Spatial Information System"*, University College Dublin Press, 2002, pp.34.

[3] Japan Association on Remote Sensing, "*Remote Sensing Note"*, NASDA (JAXA), Japan, 2000, chapter 13, pp,246-258.

[4] Rees, W.G., *"Physical Principles of Remote Sensing, Second Editon"*, Cambridge University Press, Cambridge-UK, 2001, chapter.11, p.273.

[5] Hanan Samet and Robert Webber, *"Storing a Collection of Polygons Using Quadtrees"*, ACM Transactions on Graphics, InfoLAB, 1985, pp.182-222.

[6] Paul DuBois, *"MySQL (4th Edition)"*, Addison-Wesley Professional; 4 edition (September 8, 2008), 2008, pp.1224.

[7] Shashi Shekhar and Sanjay Chawla, *"Spatial Databases: A Tour"*, Prentice Hall, 2003 (ISBN 0-13-017480-7).

[8] MariaDB Foundation, "*MariaDB Knowlegde Base"*, https://mariadb.com/kb/en/, MariaDB.org.

[9] NASA-MODIS Distributed Archived Center, https:// ladsweb.nascom.nasa.gov/data/. Level 1 and Atmosphere Archive and Distribution System. NASA USA.

[10] Adam Piokowski, "*MySQL Spatial and PostGIS Implementations of Spatial Data Standards"*, Electronic Journal of Polish Agricultural Universities, Vol.14-1, 2011.

[11] Jo Wood, Jason Dykesm, Aidan Slingsby dan Keith Clarke, "*Interactive Visual Exploration of A large Spatio-Temporal Dataset: Reflections on A geovisualization Mashup"*, IEEE Transactions on Visualization and Computer Graphic, Vol.13 No.6, 2007.

Ucapan Terima Kasih

Terima kasih kepada NASA MODIS Level 1 and Atmosphere Archive and Distribution System (https:// ladsweb.nascom.nasa.gov/data/) atas data MODIS Aqua file ID: 2012077112342 dan kepada openstreetmap.org atas data polygon, city Provinsi Bali.(http://openstreetmap.org)