ORACLE®

MySQL 5.7 GIS

Norvald H. Ryeng
Software Engineer
MySQL Optimizer Team

Trondheim, March 2015

ORACLE

Agenda

1. What is GIS?
2. Basic concepts
3. GIS functions
4. Indexing and optimization
5. Displaying data in a web map
6. The future of MySQL GIS

ORACLE

What is GIS?

ORACLE

“The early days of GIS were very lonely. No-one knew what it meant.”

— Roger Tomlinson, “Father of GIS”

What is GIS?

Geocoding
Satellite imagery
Decision analysis
Routing
Raster data
2d
Network models
4d
Web maps
Vector data
Navigation
Monitoring
Topology
Hydrology
Cartography
Statistics
Positioning
3d
City planning
Remote sensing
Standards
Maps
Modelling
Projections
Overlays

"A geographic information system (GIS) is a system designed to capture, store manipulate, analyze, manage and present all types of spatial or geographical data."

— Wikipedia

What is GIS?

FalconView
SuperMap
OGR
Mapnik
QGIS
Kalypso
SAGA
AutoCAD
Google Earth
GRASS
uDig
Smallworld
ArcGIS
Bing Maps
MySQL
NetCAD
GDAL
ILWIS
Galileo
GPS
Google Maps
OpenLayers
OpenStreetMap
MySQL Workbench
MapFish
TerraView
GeoServer
MapServer

GIS in database management systems

- Geometric shapes
  - Points, lines, polygons
- Functions
  - Comparisons
  - Generate new shapes
  - Measures
  - Properties
- Data types
- Indexes
- Query optimization
- Standards
  - SQL/MM
  - OGC Simple Feature Access
- Coordinate systems

ORACLE

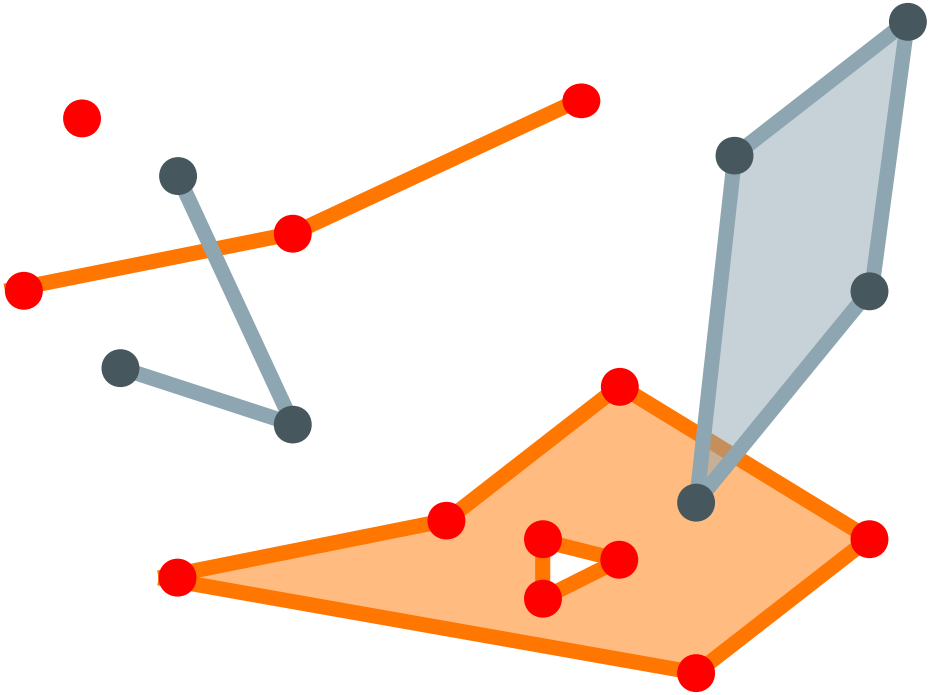
Basic concepts

“Geography is just physics slowed down, with a couple of trees stuck in it.”

— Terry Pratchett, in *The Last Continent*

Geometric objects

- Point
- LineString
- Polygon
- MultiPoint
- MultiLineString
- MultiPolygon
- GeometryCollection



Point

Point(0, 0)

ST_GeomFromText('POINT(0 0)')

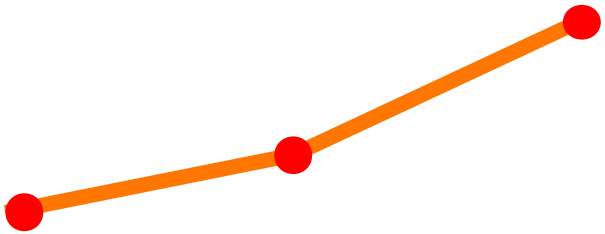
ST_GeomFromGeoJSON('{"type":"Point","coordinates":[0,0]}')

ST_PointFromGeohash('s000', 0)

ST_GeomFromWKB(0x0101000000000000000000000000000000000000)

LineString

- Two or more points



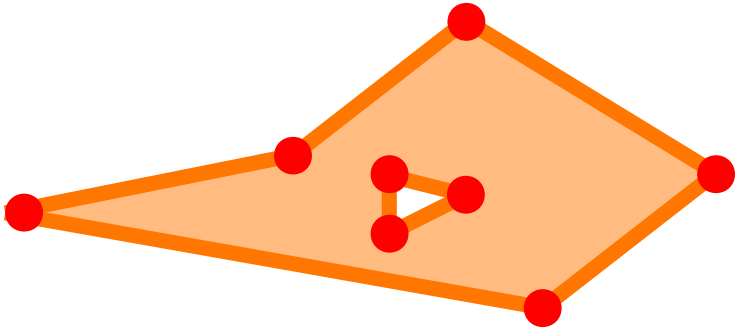
LineString(Point(0, 0), Point(1, 1))

ST_GeomFromText('LINESTRING(0 0, 1 1)')

ST_GeomFromGeoJSON('{"type":"LineString","coordinates":[[0,0],[1,1]]}')

Polygon

- One exterior ring
- Zero or more inner rings (holes)
- At least four points in each ring
  - Start and end point is the same



Inner ring

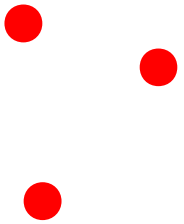
Polygon(LineString(Point(0, 0), Point(1, 0), Point(1, 1), Point(0, 0)))

ST_GeomFromText('POLYGON((0 0, 1 0, 1 1, 0 0), (0.2 0.1, 0.9 0.8, 0.9 0.1, 0.2 0.1))')

ST_GeomFromGeoJSON('{"type":"Polygon","coordinates":[[[0,0],[1,0],[1,1],[0,0]]]}')

MultiPoint

- One or more Point
  - Can't be empty



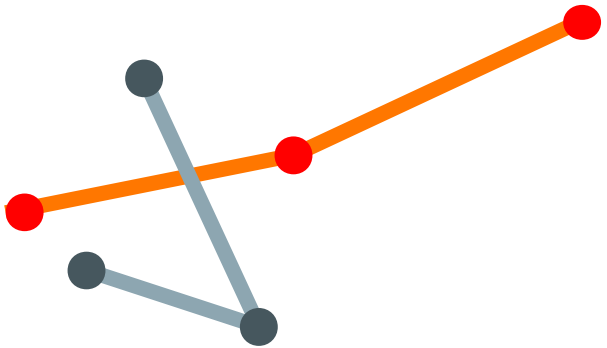
MultiPoint(Point(0, 0), Point(1, 1))

ST_GeomFromText('MULTIPOINT(0 0, 1 1)')

ST_GeomFromGeoJSON('{"type":"MultiPoint","coordinates":[[0,0],[1,1]]}')

MultiLineString

- One or more LineString
  - Can't be empty



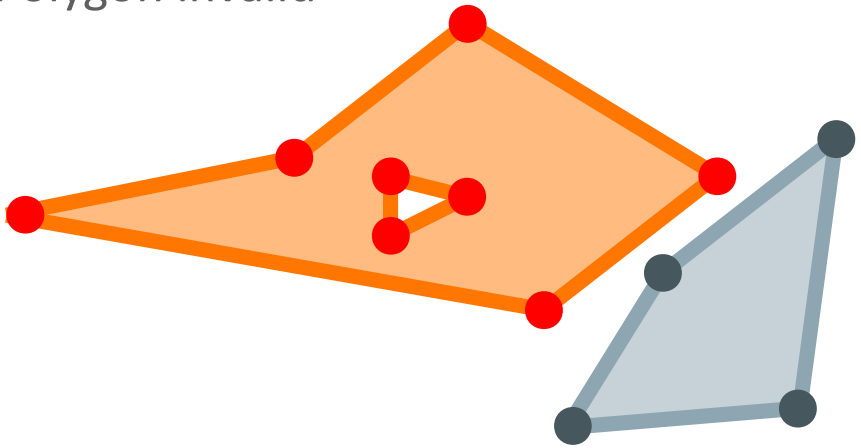
MultiLineString(LineString(Point(0, 0), Point(1, 1)), LineString(Point(2, 2), Point(3, 3)))

ST_GeomFromText('MULTILINESTRING((0 0, 1 1),(2 2, 3 3))')

ST_GeomFromGeoJSON('{"type":"MultiLineString","coordinates":[[[0,0],[1,1]],[[2,2],[3,3]]]}')

MultiPolygon

Will make the MultiPolygon invalid



- One or more Polygon
  - Can't be empty
- Polygons shouldn't overlap
  - May touch in a finite number of points

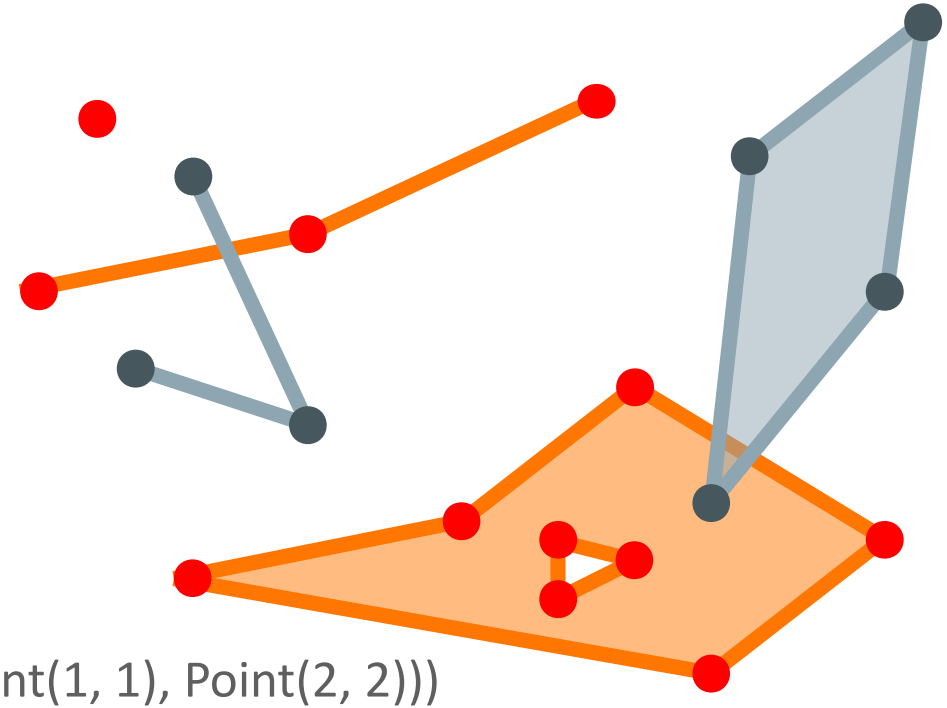
MultiPolygon(Polygon(LineString(Point(0, 0), Point(1, 0), Point(1, 1), Point(0, 0))))

ST_GeomFromText('MULTIPOLYGON(((0 0, 1 0, 1 1, 0 0)))')

ST_GeomFromGeoJSON('{"type":"MultiPolygon","coordinates":[[[[0,0],[1,0],[1,1],[0,0]]]]}')

GeometryCollection

- Zero or more geometries
  - May be empty
- No restrictions on overlapping



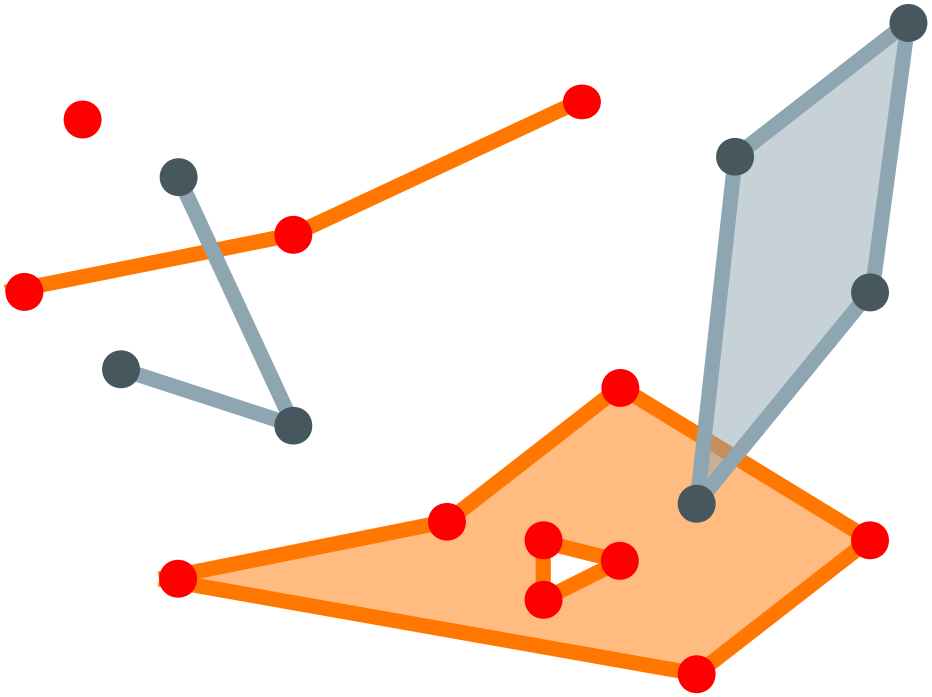
GeometryCollection(Point(0, 0), Linestring(Point(1, 1), Point(2, 2)))

ST_GeomFromText('GEOMETRYCOLLECTION(POINT(0 0), LINESTRING(1 1, 2 2))')

ST_GeomFromGeoJSON('{"type":"GeometryCollection","geometries":[{"type":"Point","coordinates":[0,0]},{"type":"LineString","coordinates":[[1,1],[2,2]]}]}')

Data types

- Geometry (may store any of the types below)
- Point
- LineString
- Polygon
- MultiPoint
- MultiLineString
- MultiPolygon
- GeometryCollection



Example

- Sightseeing in Trondheim
- Database of popular places to visit
  - A unique ID
  - Position (point)
  - Descriptive text

Example

```
CREATE TABLE sights (
   id INT AUTO_INCREMENT PRIMARY KEY,
   pos POINT NOT NULL,
   description VARCHAR(200)
) ENGINE=InnoDB;

INSERT INTO sights (pos, description) VALUES (
   Point(10.3958, 63.4269), 'Nidaros Cathedral'
);

SELECT ST_AsText(pos), description FROM sights;
```

ST_AsText(pos)	**description**
POINT(10.3958 63.4269)	Nidaros Cathedral

Coordinate systems

- Each geometry is in a spatial reference system (SRS)
  - Specified by SRID (integer)
  - Geometries in different SRSs can't be compared
- MySQL supports a Cartesian plane (SRID 0)
  - Default if no SRS is specified
    - Unless the import format defaults to another SRS, e.g., WGS 84 for GeoJSON
- Other reference systems
  - Typically defined by EPSG, e.g., WGS 84 (SRID 4326)
  - Computations are always done in SRID 0

ORACLE

Longitude and latitude

- OGC specifies X and Y axes, not longitude and latitude
- In GIS, ⟨longitude, latitude⟩ is the de facto standard
  - X = degrees East (negative for West)
  - Y = degrees North (negative for South)

Example

```
INSERT INTO sights (pos, description) VALUES (
   ST_GeomFromGeoJSON('{"type":"Point","coordinates":[10.4025,63.4194]}'),
   'Norwegian University of Science and Technology'
);

INSERT INTO sights (pos, description) VALUES (
   ST_GeomFromText('POINT(10.3948 63.4225)', 4326),
   'Student Society Building'
);

INSERT INTO sights (pos, description) VALUES (
   ST_GeomFromText('POINT(10.3951 63.4305)'),
   'Olav Tryggvason Monument'
);
```

Example

SELECT ST_AsText(*pos*), ST_SRID(*pos*), *description* FROM *sights*;

ST_AsText(pos)	**ST_SRID(pos)**	**description**
POINT(10.3958 63.4269)	0	Nidaros Cathedral
POINT(10.4025 63.4194)	4326	Norwegian University of Science and Tech ...
POINT(10.3969 63.428)	4326	Student Society Building
POINT(10.3951 63.4305)	0	Olav Tryggvason Monument

GIS functions

Functions

- SQL/MM style naming
  - ST_ prefix
- MBR functions
  - Minimum bounding rectangle instead of exact shape
- Geometry construction functions
  - Same name as data type

Functions

- Only defined for valid geometries
- The result is undefined for invalid geometries
  - A best effort result
  - A weird result
  - An error
- Check with ST_IsValid or use ST_Validate if unsure
- Mixing SRIDs is not allowed
  - Results in an error

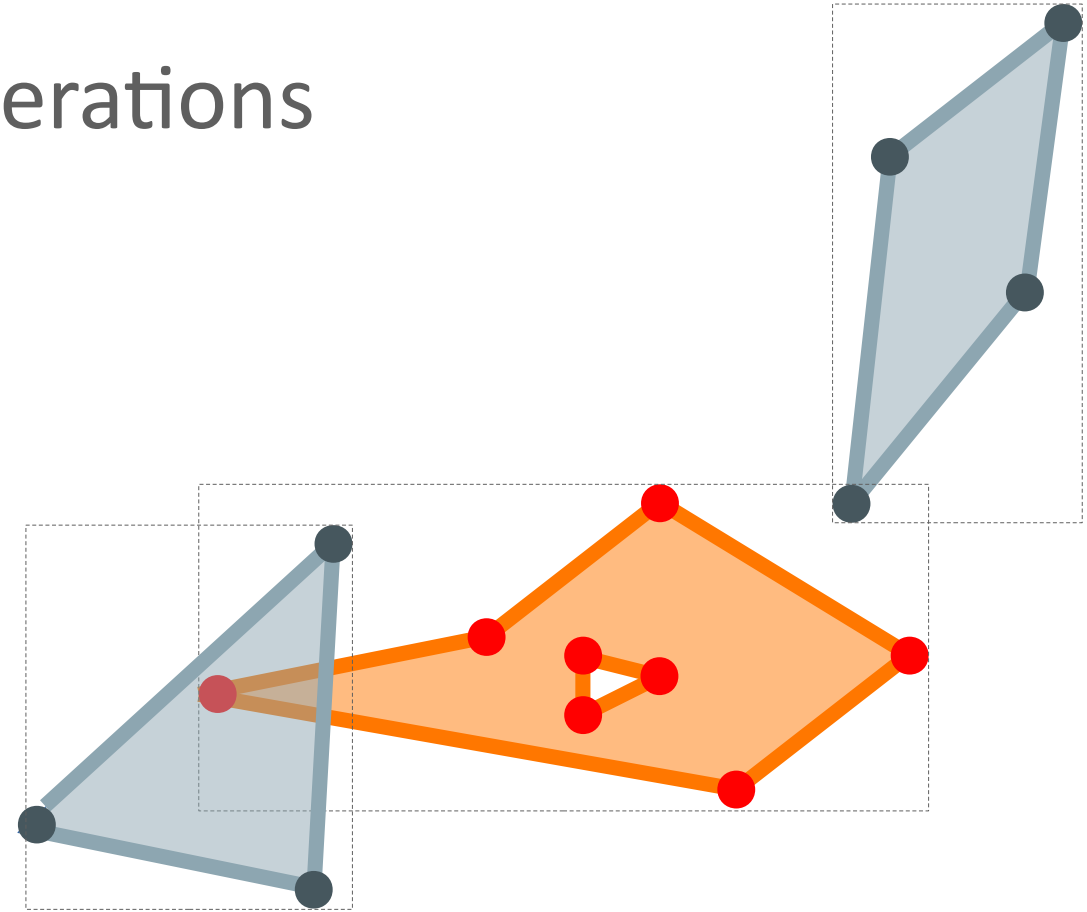
ORACLE

Functions

- Comparison
  - ST_Contains, ST_Crosses, ST_Disjoint, ST_Equals, ST_Intersects, ST_Overlaps, ST_Touches, ST_Within
- Produce new geometries
  - ST_Buffer, ST_Centroid, ST_ConvexHull, ST_Envelope, ST_MakeEnvelope, ST_Simplify, ST_Difference, ST_Intersection, ST_SymDifference, ST_Union

ORACLE

Set operations



Functions

- Measures
  - ST_Area, ST_Distance, ST_Distance_Sphere, ST_Length
- Extract properties
  - ST_Dimension, ST_EndPoint, ST_ExteriorRing, ST_GeometryN, ST_GeometryType, ST_InteriorRingN, ST_IsClosed, ST_IsEmpty, ST_IsSimple, ST_IsValid, ST_PointN, ST_SRID, ST_StartPoint, ST_X, ST_Y
- Helper functions
  - ST_LatFromGeohash, ST_LongFromGeohash, ST_Validate

ORACLE

Functions

- Import
  - ST_GeomCollFromTxt/ST_GeomCollFromText, ST_GeomCollFromWKB, ST_GeomFromGeoJSON, ST_GeomFromText, ST_GeomFromWKB, ST_LineFromText, ST_LineFromWKB, ST_MLineFromText, ST_MLineFromWKB, ST_MPointFromText, ST_MPointFromWKB, ST_MPolyFromText, ST_MPolyFromWKB, ST_PointFromGeohash, ST_PolyFromText, ST_PolyFromWKB
- Export
  - ST_AsBinary, ST_AsGeoJSON, ST_AsText, ST_Geohash

ORACLE

Example

```
SET @city_center = ST_GeomFromText(
   'POLYGON((10.3765 63.4292, 10.3847 63.4277, 10.3902 63.4247, 10.3986 63.4245,
   10.4013 63.4264, 10.4013 63.4283, 10.4072 63.4347, 10.4037 63.4354,
   10.3954 63.4350, 10.3799 63.4314, 10.3765 63.4292))'
);

SELECT description FROM sights
   WHERE ST_Within(pos, @city_center);
```

Example

```
SET @city_center = ST_GeomFromText(
   'POLYGON((10.3765 63.4292, 10.3847 63.4277, 10.3902 63.4247, 10.3986 63.4245,
   10.4013 63.4264, 10.4013 63.4283, 10.4072 63.4347, 10.4037 63.4354,
   10.3954 63.4350, 10.3799 63.4314, 10.3765 63.4292))'
);

SELECT description FROM sights
   WHERE ST_Within(pos, @city_center);
```

ERROR 3033 (HY000): Binary geometry function st_within given two geometries of different srids: 4326 and 0, which should have been identical.

Example

UPDATE *sights* SET *pos* = ST_GeomFromWKB(ST_AsBinary(*pos*));

SELECT *description* FROM *sights*
  WHERE ST_Within(*pos*, *@city_center*);

description
Nidaros Cathedral
Olav Tryggvason Monument

Example

Standing at the Olav Tryggvason Monument, which sights are nearby?

SELECT *description* FROM *sights*
  WHERE ST_Within(*pos*, ST_Buffer(Point(10.3951, 63.4305), 0.004));

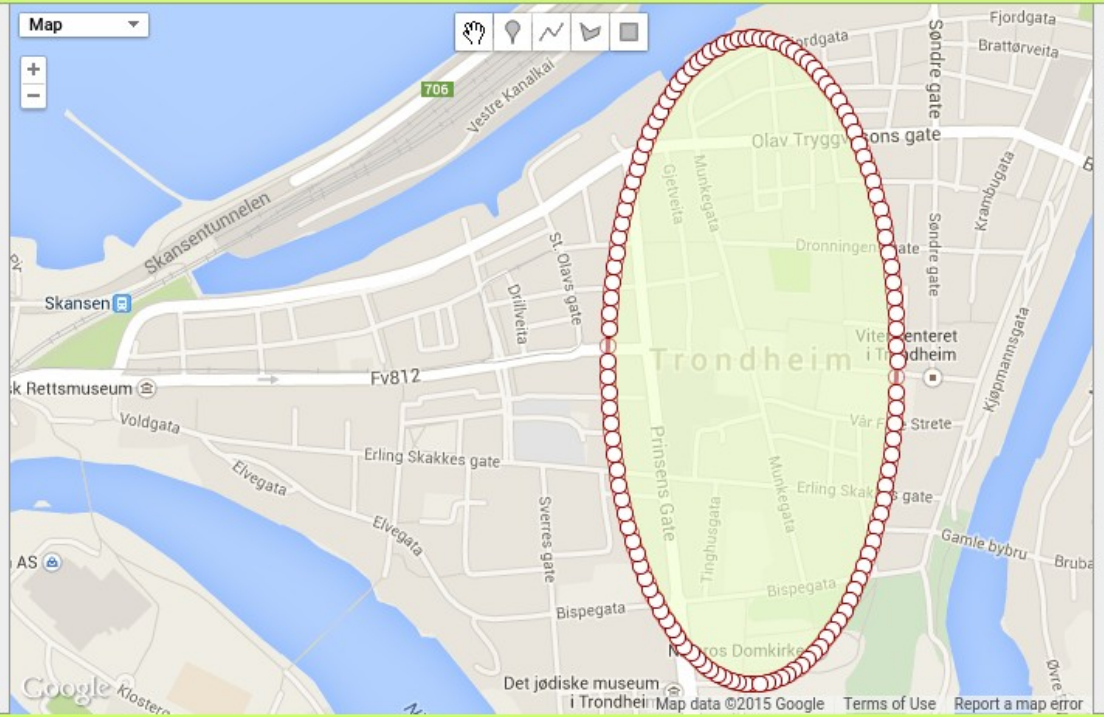
description
Nidaros Cathedral
Olav Tryggvason Monument

"nearby" = within 0.004 degrees

Fork me on GitHub

Wicket

It whispers WKT in your application's ear.



Wicket is a lightweight Javascript library that reads and writes Well-Known Text (WKT) strings. It can also be extended to parse and to create geometric objects from various mapping frameworks, such as Leaflet and the Google Maps API.

```
9612
63.426619874987225,10.39588036128
8064
63.42657685887839,10.395686921897
822
63.426543293960144,10.39549206856
1317
63.42651926109331,10.395296270697
308 63.42650481817518,10.3951
63.426500000000004))
```

Format for URLs

Clear Map

Map It!

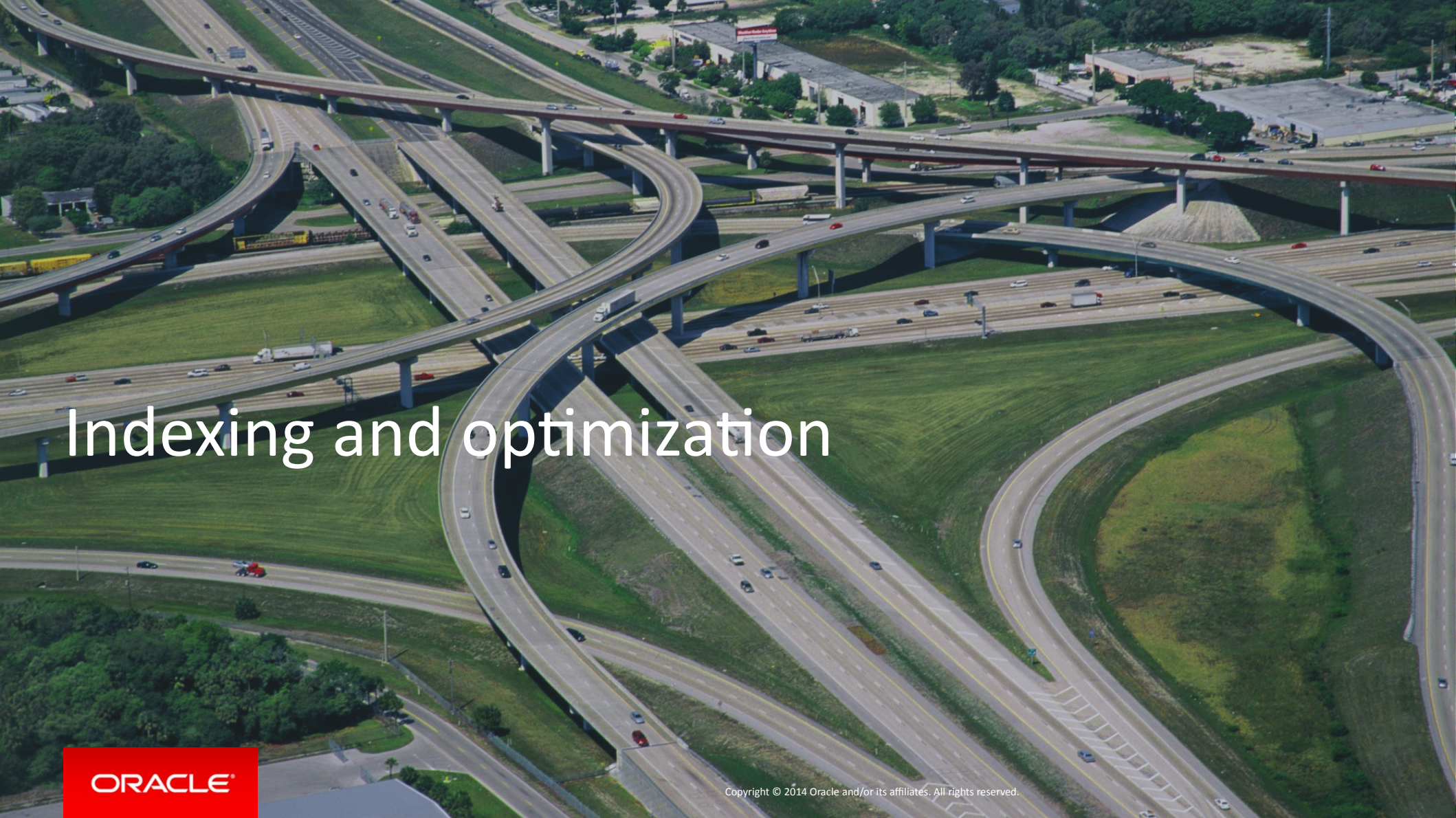
Home Contact Me "Fork me on GitHub!"

Example

Standing at the Olav Tryggvason Monument, how far is it to the Nidaros Cathedral?

```
SELECT ST_Distance_Sphere(Point(10.3951, 63.4305), Point(10.3958, 63.4269)) AS d;
```

d
401.8121469013054



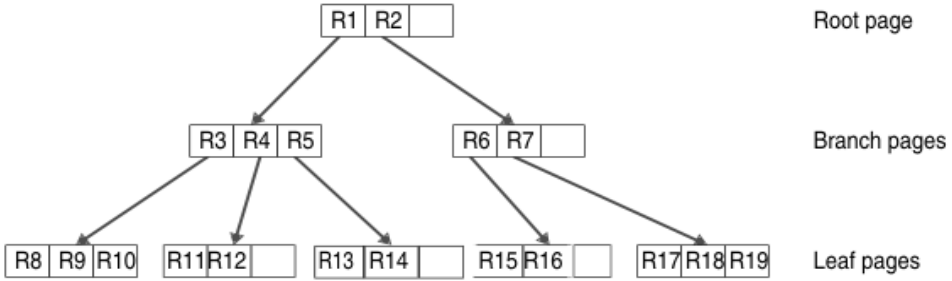
Indexing and optimization

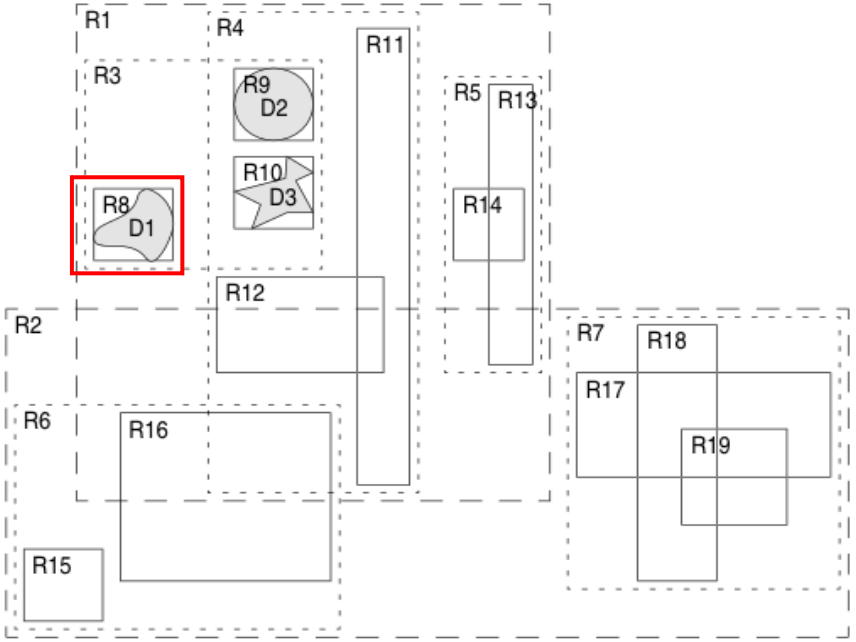
ORACLE

Indexes

- R-tree spatial indexes
  - 2d
  - Uses bounding boxes
  - Fully transactional
- Column must be NOT NULL

ALTER TABLE *table* ADD SPATIAL INDEX (*column*);





Optimization

- The optimizer automatically uses an R-tree index if it thinks it's beneficial
- The query must have a suitable WHERE clause
  - ST_Contains, ST_Crosses, ST_Disjoint, ST_Equals, ST_Intersects, ST_Overlaps, ST_Touches, ST_Within
  - MBRContains, MBRDisjoint, MBREquals, MBRIntersects, MBROverlaps, MBRTouches, MBRWithin

ORACLE

Example

```
SELECT description FROM sights WHERE ST_Within(pos, ST_Buffer(Point(10.3951, 63.4305), 0.004));
```

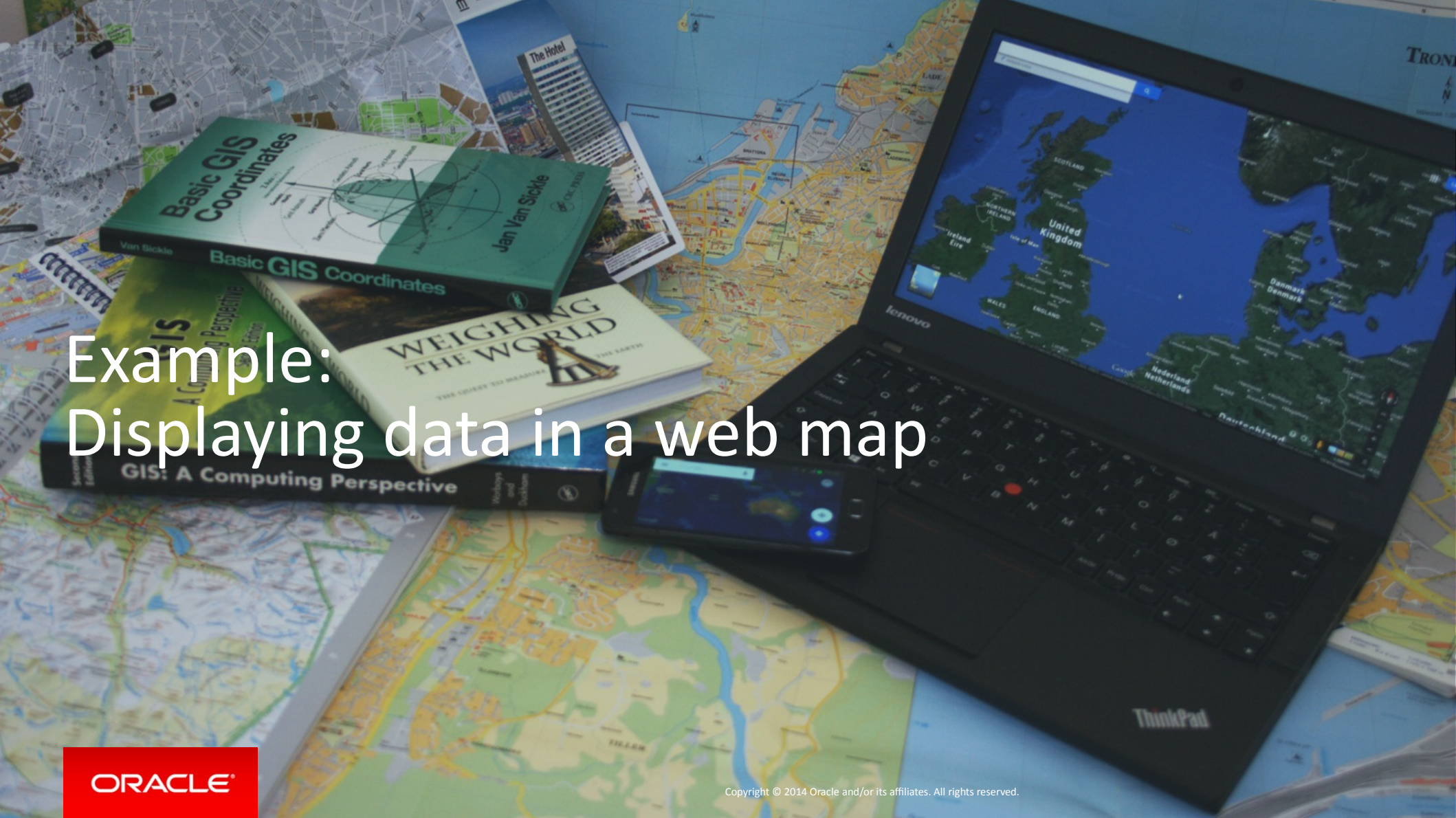
```
mysql> SHOW STATUS LIKE 'Handler_read%';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| Handler_read_first    | 1     |
| Handler_read_key      | 1     |
| Handler_read_last     | 0     |
| Handler_read_next     | 0     |
| Handler_read_prev     | 0     |
| Handler_read_rnd      | 0     |
| Handler_read_rnd_next | 16    |
+-----------------------+-------+
7 rows in set (0.00 sec)
```

Without index

```
mysql> SHOW STATUS LIKE 'Handler_read%';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| Handler_read_first    | 0     |
| Handler_read_key      | 1     |
| Handler_read_last     | 0     |
| Handler_read_next     | 3     |
| Handler_read_prev     | 0     |
| Handler_read_rnd      | 0     |
| Handler_read_rnd_next | 0     |
+-----------------------+-------+
7 rows in set (0.00 sec)
```

With index

Example: Displaying data in a web map



ORACLE

Displaying data in a web map

- A sightseeing map of Trondheim
- Use our sightseeing database from earlier examples
  - Add a few more points to make it more exciting
- Use OpenLayers to display the map
  - OpenStreetMap tiles
  - Add a marker for each place of interest

ORACLE

L

Any Linux distro will do

Ubuntu 14.04 LTS

A

Whichever Apache version came with my OS

M

MySQL 5.7 DMR from repo.mysql.com (5.7.6 or newer)

Sveta Smirnova's JSON UDFs

P

Whichever PHP version came with my OS

ORACLE

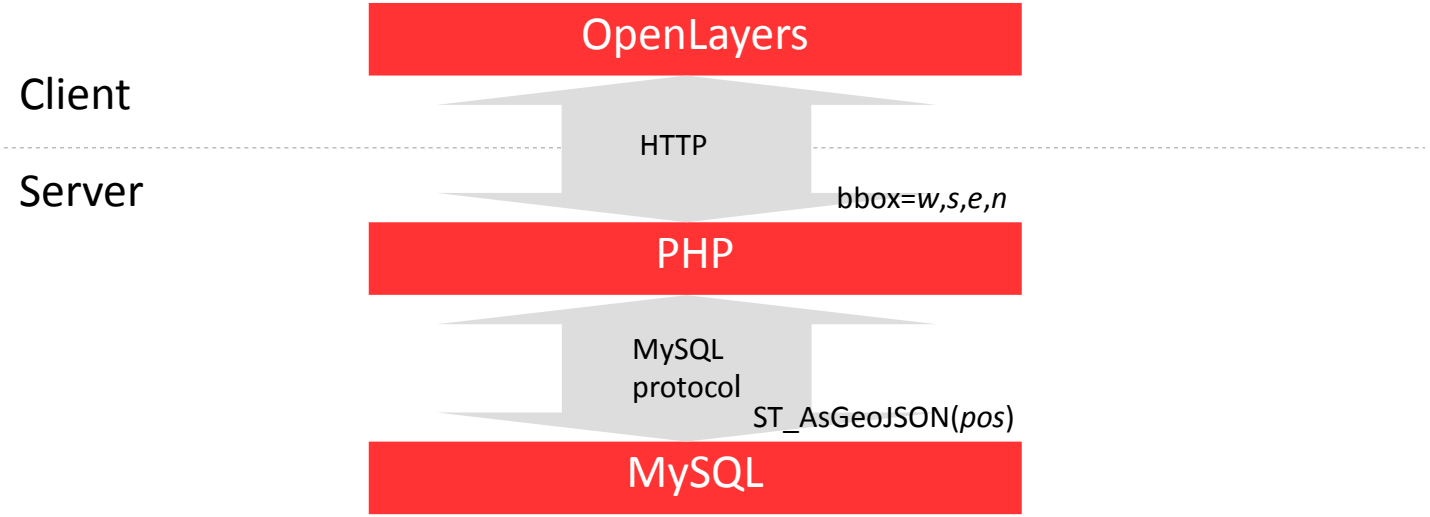
OpenLayers

- Simple setup, following an online guide
  - http://docs.openlayers.org/library/introduction.html
- Modifications
  - OpenStreetMap map layer
    - More detailed than the one in the guide
  - Local tile cache (in case I lose my network connection)

ORACLE

Connecting the parts

- Both MySQL and OpenLayers support GeoJSON
- Use a PHP script to query the database



```
<html><head><title>OpenLayers Example</title>
<script src="openlayers/OpenLayers.js"></script>
</head>
<body>
 <div style="width:100%; height:100%" id="map"></div>
 <script defer="defer" type="text/javascript">
  var map = new OpenLayers.Map('map');
  var osm_url = "http://localhost:8080/map/tiles.php?z=${z}&x={x}&y=${y}&r=mapnik"
  var osm = new OpenLayers.Layer.OSM('osm', [osm_url]);
 map.addLayer(osm);
 map.zoomToMaxExtent();
</script></body></html>
```

```
<html><head><title>OpenLayers Example</title>
<script src="openlayers/OpenLayers.js"></script>
</head>
<body>
 <div style="width:100%; height:100%" id="map"></div>
 <script defer="defer" type="text/javascript">
  var map = new OpenLayers.Map('map');
  var osm_url =
       "http://localhost:8080/map/tiles.php?z=${z}&x=
                                  {x}&y=${y}&r=mapnik"
  var osm = new OpenLayers.Layer.OSM('osm',
                                          [osm_url]);

  var geojson_format = new
                          OpenLayers.Format.GeoJSON();
```

```
var sights_layer = new
 OpenLayers.Layer.Vector("Sights",
 {
  strategies: [new OpenLayers.Strategy.BBOX()],
  protocol: new OpenLayers.Protocol.HTTP(
   {
    url: 'sights.php',
    format: geojson_format
   })
 });

 map.addLayers([osm, sights_layer]);
 map.zoomToMaxExtent();
 </script></body></html>
```

GeoJSON

```
{
    "type":"FeatureCollection",
    "features":
    [
        {
            "type":"Feature",
            "geometry":{"type":"Point","coordinates":[10.4017,63.4282]},
            "properties": {"description":"Old Town Bridge"}
        },
        {
            "type":"Feature",
            "geometry": {"type":"Point","coordinates":[10.3958,63.4269]},
            "properties": {"description":"Nidaros Cathedral"}
        }
    ]
}
```

GeoJSON

One row of the table

```
{
    "type":"FeatureCollection",
    "features":
    [
        {
            "type":"Feature",
            "geometry":{"type":"Point","coordinates":[10.4017,63.4282]},
            "properties": {"description":"Old Town Bridge"}
        },
        {
            "type":"Feature",
            "geometry": {"type":"Point","coordinates":[10.3958,63.4269]},
            "properties": {"description":"Nidaros Cathedral"}
        }
    ]
}
```

GeoJSON

ST_AsGeoJSON(*pos*)

```
{
    "type":"FeatureCollection",
    "features":
    [
        {
            "type":"Feature",
            "geometry":{"type":"Point","coordinates":[10.4017,63.4282]},
            "properties": {"description":"Old Town Bridge"}
        },
        {
            "type":"Feature",
            "geometry": {"type":"Point","coordinates":[10.3958,63.4269]},
            "properties": {"description":"Nidaros Cathedral"}
        }
    ]
}
```

JSON_APPEND

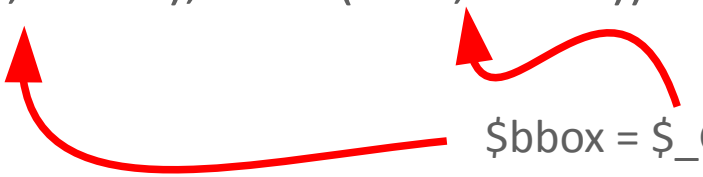
JSON_APPEND(*object*, *attribute_name*, *value*)

SELECT JSON_APPEND('{}', 'foo', '"bar"') AS *json*;

json
{"foo": "bar"}

Query

```
SELECT JSON_APPEND
(
   JSON_APPEND('{"type":"Feature"}', 'geometry', ST_AsGeoJSON(pos)),
   'properties',
   JSON_APPEND('{}', 'description', CONCAT('"', description, '"'))
) AS json
FROM sights
WHERE ST_Within(
   pos,
   ST_MakeEnvelope(Point(west, south), Point(east, north))
);
```



```
$bbox = $_GET['bbox'];
```

The future of MySQL GIS

ORACLE

General goals

- Provide adequate GIS for existing MySQL users
  - Growing mobile market and emerging IoT market both require spatial features
- Competing with PostGIS in the FOSS GIS DBMS market
  - For basic/common GIS use cases
- Competing with Microsoft SQL Server in the commercial GIS market

ORACLE

Future enhancements

- A non-flat Earth
  - Ellipsoidal Earth model
  - Projections
- OGC standard metadata tables (SPATIAL_REF_SYS, etc.)
- 3d and 4d support
  - 3dm, 3dz, 3dzm
- What else would **you** like to see?
  - Let us know!

ORACLE

“GIS is a form of digital mapping technology. Kind of like Google Earth, but better.”

— Arnold Schwarzenegger, Governor of California

Hardware and Software

Engineered to Work Together

ORACLE

Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE®