



Use APEX to Visualize Spatial Data

- Display data on maps with minimal efforts
-

Øyvind Isene
@OyvindIsene



~~<http://oisene.blogspot.com>~~

<https://enesi.no/>



<http://sysco.no>



<http://www.bicon.no>

500+ Technical Experts Helping Peers Globally

ORACLE®
ACE PROGRAM



3 Membership Tiers

- Oracle ACE Director
- Oracle ACE
- Oracle ACE Associate

bit.ly/OracleACEProgram

Connect:

✉ oracle-ace_ww@oracle.com

f Facebook.com/oracleaces

t [@oracleace](https://twitter.com/oracleace)



Nominate yourself or someone you know: acenomination.oracle.com

Agenda

- Quick intro to APEX
- Find some cool data - Import them to Oracle
- Use Apex to build a nice web page.



I'm a DBA - so why APEX?

- Fun to create something on my own
- It is fast!
- DBA should recommend APEX
 - it gives you less pain
 - Fewer moving parts than
 - We know the technology behind it
- With APEX I can report easily to management
- Customers love it



OK, but why spatial?

- People love maps
- People move around and like to see it in 2D
- Boring data can be enriched and look great on maps
- Lots of people don't know how easy you can work with geographic data in Oracle



Licenses

- Not my favourite subject
- Locator — subset of Spatial included with your db license
- Locator includes what you need for fun
- You can do a lot of advanced stuff with extra Spatial and Graph license
- Test in lab and check
DBA_FEATURE_USAGE_STATISTICS



Installation

- Use Docker or Virtual Box with Vagrant
- Or download Oracle Database App Development VM for VirtualBox
- Or apex.oracle.com
- Or your friend's cloud?



A note on Docker

- Learning Docker is a good investment
- Set up your own lab with little hassle
- Focus on your task - not installation
- Excellent support from Oracle
 - <https://github.com/oracle/docker-images>
- Also check out work by Gerald Venzl
- See a short intro at the end.



Let's get started

In APEX



Create a new empty page


Create a static region with the following source:

```
<div id="mapRegion" style="width:100%;height:430px;"></div>
```



You'll need this later

Add JavaScript and CSS

- From  mapbox
- Add JavaScript and CSS URLs to page

ORACLE Application Express App Builder SQL Workshop Team Development Packaged Apps HROUG

Application 101 \ Page Designer

Page 2: Map testing

- Pre-Rendering
- Regions
 - Content Body
 - MapQuest Map
 - Attributes
- Post-Rendering

Map testing

PAGE HEADER

PAGE NAVIGATOR

SPENDCLUB BAR

CONTENT BODY

- MapQuest Map
 - COPY EDIT PREVIOUS NEXT
 - ITEMS
 - REGION CONTENT
 - SUB-REGIONS

Regions Items Buttons

Background Calendar Chart Classic Report Classic Report Based on Functions Help Text Interactive Grid

Page

Filter Properties

Template Options Use Template Defaults

CSS Classes

Media Type

Navigation Menu

Override User Interface Level Yes No

Navigation

Cursor Focus Do not focus cursor

Warn on Unsaved Changes Yes No

JavaScript

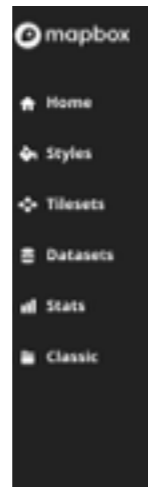
File URLs

<https://api.tiles.mapbox.com/mapbox.js/v3.1.1/mbox.js>

BICON
BUSINESS INTELLIGENCE
powered by SYSSO

The screenshot displays the Oracle Application Express Page Designer interface. The top navigation bar includes 'ORACLE Application Express', 'App Builder', 'SQL Workshop', 'Team Development', and 'Packaged Apps'. The main workspace is titled 'Application 105 \ Page Designer' and shows a page layout with components like 'PAGE HEADER', 'PAGE NAVIGATION', 'SPREADSHEET BAR', and 'CONTENT BODY'. A 'MapQuest Map' component is highlighted in the content body. The left sidebar shows a tree view with 'Page 1: Map testing' selected, and a green arrow points to the 'Regions' folder. The right sidebar shows the 'Page' properties panel, with a green arrow pointing to the 'CSS' section, which lists a 'File URL' for 'https://api.mapbox.com/mapbox.js/v5.2.0/mapbox.css'. The bottom of the interface features a 'Regions' tab and a toolbar with icons for 'Breakdown', 'Calendar', 'Chart', 'Classic Report', 'Classic Report Based on Function', 'Map Test', and 'Interactive Grid'. The BICON logo is visible in the bottom right corner.

Create a user at Mapbox



Account

Profile Security A

1 token

Create a new token

Default Public Token +

Modified a few seconds ago

pk.eyJ1Ijoib212LzZkOjE1Iiw1Y3I6Im9kZ310XWZjYk9jZmFwa2IzMHk1bGkiFQ.M0y4y0Pa6hF906A-d0LCx

4 scopes stylestyles stylesread fontsread datasetsread

Actually, you can do this later and use the public token shown here

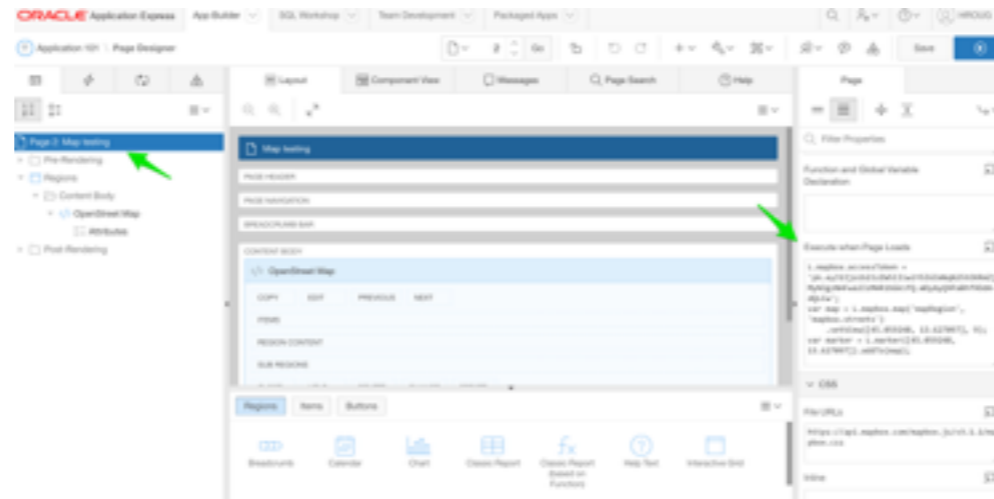


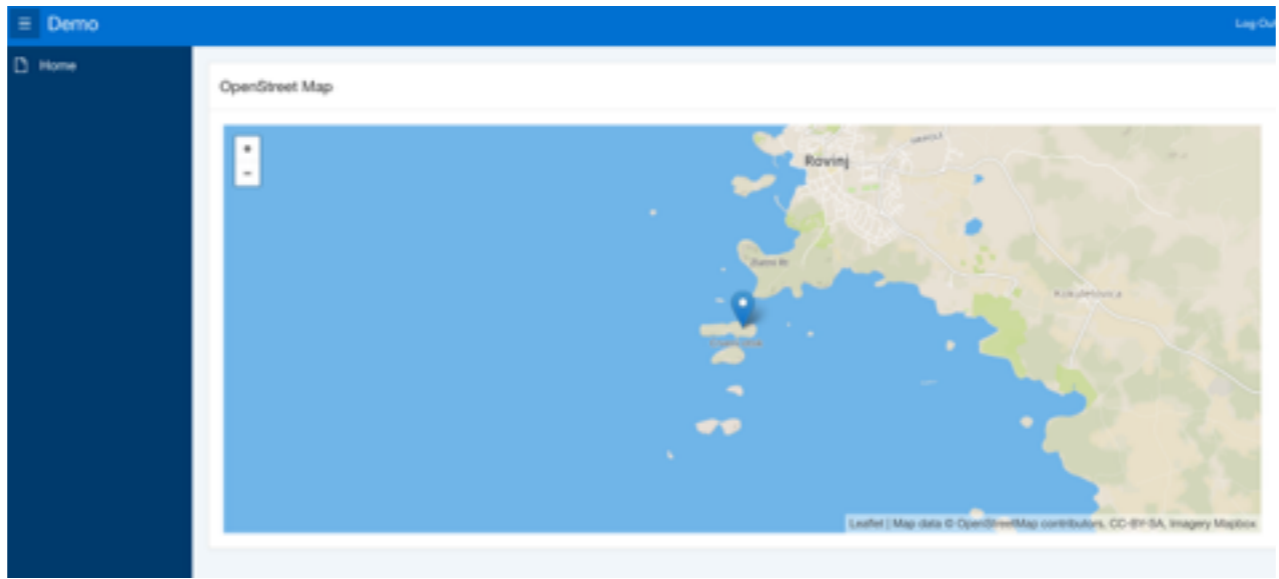
Some code...

```
L.mapbox.accessToken =  
'pk.eyJ1Ijoib2lzMW5lIiwiaSI6ImNqN25tOXRmZjMyN3gzNHFwa2IzMHR1bGkiLCJ0.WDyAyQ9PaBhf9Dda-dQLCw';  
var map = L.mapbox.map('mapRegion', 'mapbox.streets')  
    .setView([45.059248, 13.627097], 9);  
var marker = L.marker([45.059248, 13.627097]).addTo(map);
```

Your id from <div ...>

Add it here





It works, let us map some data
from the database, of course



Find a cool data set

- data.gov.hr - open data from your government
- not in English

Example with air pollution

- <http://iszz.azo.hr/iskzl/koordinat.htm>
- Data downloaded as Excel file
- Open it and copy out the table, save in new file
- Import with SQL Developer

Koordinater av måle stasjoner

Export til Excel

Legend:
 G, S, Y, T) koordinater i Gauss-Kruger koordinatsystem
 X, Y = (deg, min, sek) grader, minutter og sekunder i WGS84-prosjeksjon (geogrid)

Nettverk	Stasjon	Gauss-X	Gauss-Y	Gauss-Z	X, deg	X, min	X, sek	Y, deg	Y, min	Y, sek	X	Y
Statnett for kontinuerlig luftkvalitetsmåling	Beovo	46	15	352	46	15	8,80	15	29	38,30	46,150076	15,488628
Statnett for kontinuerlig luftkvalitetsmåling	HRK (gje 710)	45	16	574	45	16	10,80	16	6	57,30	45,271133	16,110970
Statnett for kontinuerlig luftkvalitetsmåling	HRK210C an			110	45	29	36,89	15	22	54,47	45,483225	15,391121
Statnett for kontinuerlig luftkvalitetsmåling	KOPČAK RT	45	18	83	45	18	32,30	18	30	4,70	45,298038	18,034628
Statnett for kontinuerlig luftkvalitetsmåling	Kuhovec			107	45	18	44,80	16	46	50,40	45,479056	16,786607
Statnett for kontinuerlig luftkvalitetsmåling	OPUZEN (Zelma Hrvatska)	45	17	80	45	17	21,40	17	22	57,40	45,283758	17,366025
Statnett for kontinuerlig luftkvalitetsmåling	Sejane an	504620	495490	109	45	16	31,85	18	41	55,57	45,296763	18,498768
Statnett for kontinuerlig luftkvalitetsmåling	PAZ	45	14	860	45	14	36,30	14	27	49,47	45,233481	14,433484
Statnett for kontinuerlig luftkvalitetsmåling	PUTYICE LARZ	44	15	704	44	15	37,80	15	30	35,20	44,261033	15,483778
Statnett for kontinuerlig luftkvalitetsmåling	PSLAC (Zelma Hrvatska)	45	15	104	44	1	15,34	15	30	36,30	44,021094	15,516171
Statnett for kontinuerlig luftkvalitetsmåling	RIŠAN an	503700	545380	10	45	19	35,30	14	29	47,85	45,31975	14,488633
Statnett for kontinuerlig luftkvalitetsmåling	RIŠAN-2	45	14	109	45	19	14,86	14	29	5,84	45,330794	14,483911
Statnett for kontinuerlig luftkvalitetsmåling	Šušak an	503570	543890	104	45	27	26,25	14	22	18,38	45,439125	14,388326
Statnett for kontinuerlig luftkvalitetsmåling	ŠUHONSKI BRZD - an midlertidig målestasjon				45	8	53,51	18	1	23,33	45,134111	18,023147
Statnett for kontinuerlig luftkvalitetsmåling	ŠUHONSKI BRZD-1				45	8	34,10					17,8901

This is the web page after Google translated it to Norwegian...

oracle12c-vagrant-ORCLPD 41 else

Tables (Filtered)

- APEX\$TEAM_DEV_...
- EBA_SPATIAL_ADC
- EBA_SPATIAL_AOI
- EBA_SPATIAL_COI
- EBA_SPATIAL_DEF
- EBA_SPATIAL_IMA
- EBA_SPATIAL_NOT

New Table...
Open
Import Data...
Import Using Oracle SQL Connector...
Refresh
Apply Filter...


STATIONS

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 MREZA	VARCHAR2(128 BYTE)	Yes	(null)	1	(null)
2 POSTAJA	VARCHAR2(144 BYTE)	Yes	(null)	2	(null)
3 GAUSS_X	NUMBER(9,8)	Yes	(null)	3	(null)
4 GAUSS_Y	NUMBER(9,8)	Yes	(null)	4	(null)
5 GAUSS_H	NUMBER(5,8)	Yes	(null)	5	(null)
6 X_SEC	NUMBER(4,8)	Yes	(null)	6	(null)
7 X_MIN	NUMBER(4,8)	Yes	(null)	7	(null)
8 X_SEC	NUMBER(6,2)	Yes	(null)	8	(null)
9 Y_SEC	NUMBER(4,8)	Yes	(null)	9	(null)
10 Y_MIN	NUMBER(4,8)	Yes	(null)	10	(null)
11 Y_SEC	NUMBER(6,2)	Yes	(null)	11	(null)
12 X	NUMBER(16,6)	Yes	(null)	12	(null)
13 Y	NUMBER(16,6)	Yes	(null)	13	(null)

Script Output | Query Result

SQL | Fetched 50 rows in 0:007 seconds

MREZA	POSTAJA	GAUSS_X	GAUSS_Y	GAUSS_H	X_DEC	X_MIN	X_SEC
1 Državna mreža za trajno praćenje kvalitete zraka DESINIC		46	25	252	46	10	8.6
2 Državna mreža za trajno praćenje kvalitete zraka HM (otok Vis)		43	36	574	43	1	52.8
3 Državna mreža za trajno praćenje kvalitete zraka KARLOVAC-1		(null)	(null)	158	45	29	36.69
4 Državna mreža za trajno praćenje kvalitete zraka KOPAČKI BIT		45	38	83	45	41	52.9
5 Državna mreža za trajno praćenje kvalitete zraka KUTINA-1		(null)	(null)	187	45	28	44.6
6 Državna mreža za trajno praćenje kvalitete zraka OPUZEN (Delta Neretve)		43	17	68	43	0	31.42
7 Državna mreža za trajno praćenje kvalitete zraka OSIJEK-1		5046288	6554958	189	45	33	31.65



JavaScript and JSON

- JS library wants data in JSON
- Not too difficult to generate with PL/SQL
- 12c comes with rich JSON support
 - A good reason to upgrade!

JSON support in 12c

- Generate JSON with SQL
- Store JSON in your database
- Enables schema-on-read (“store now — think later”)
- Check out “JSON Developers Guide”
 - even if you’re a DBA

*Every new database release
is an opportunity to simplify code*



Useful functions

- JSON_OBJECT
 - creates a JSON-object from column(s)
- JSON_ARRAYAGG
 - aggregate function
 - JSON-array of objects or column(s)

Return a list of points

```
select json_object('stations' value json_arrayagg(json_longlat )) json_arr
from (
select json_object('id' value id,
  'postaja' value apex_escape.html(postaja),
  'lat' value x, 'lng' value y) json_longlat
from stations
where id < 20
);
```

APEX is full of useful packages
that makes life easier



One annoying bug

- Functions can return VARCHAR2 (max 4000 bytes) or CLOB
- But due to bug 25186856, CLOB doesn't work :(
- See *Database Readme*
- A rewrite not too complicated

Consider extended data types

- Increase limit from 4000 to 32767 bytes for VARCHAR2 (for 12c)
- This is done by default in the cloud
- For on-prem database you'll need to do it.
- See Tim Halls post: <https://oracle-base.com/articles/12c/extended-data-types-12cR1>



PL/SQL to be called from APEX

```
create or replace function get_json return varchar2 is
  l_json varchar2(32767);
  i pls_integer;
begin
  i:=0;
  for piece in ( select json_object('id' value id,'mreza'
    value apex_escape.html(mreza),
    'postaja' value postaja,'lat' value x,
    'lng' value y null on null) longlat
  from stations) loop
    if i>1 then
      l_json := l_json || ',' || piece.longlat;
    else
      l_json := piece.longlat;
    end if;
    i := i + 1;
  end loop;
  return '{"stations": [' || l_json || ']}';
end;
/
```

Back to APEX example

- Add a hidden item to store the JSON-array
- Add a process to be executed *Before Header*

```
begin  
  :P3_JSON := get_json();  
end;
```

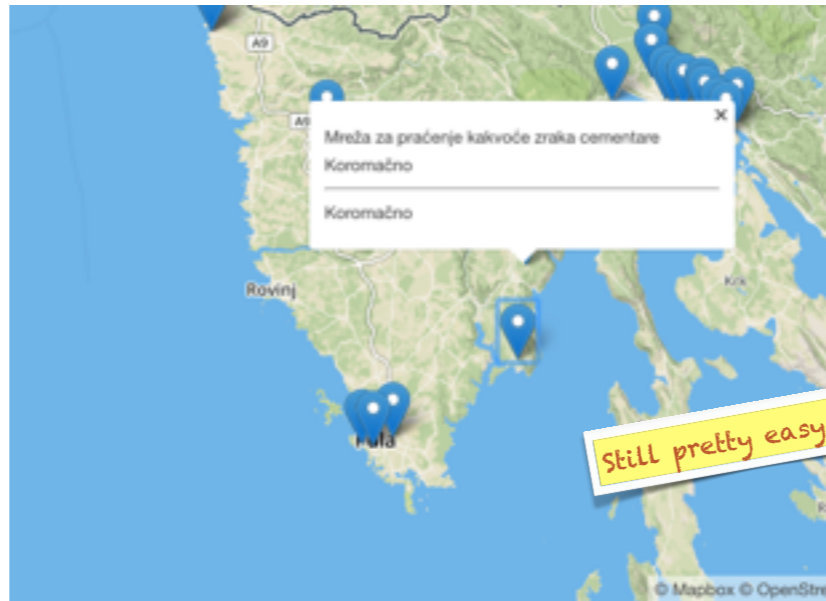

Add to previous "Execute when Page loads"

```
4 var popupOptions = {
5   offset: new L.Point(3,-25)
6 };
7
8 var json = JSON.parse($v("P3_JSON"));
9 var postajas = json.stations;
10 var marker, name, text, id ,lat, lng, mreza;
11 var Markers = [];
12
13 for (var i = 0; i < postajas.length; i++) {
14   id = postajas[i].id;
15   lat = Number(postajas[i].lat);
16   lng = Number(postajas[i].lng);
17   name = postajas[i].postaja;
18   mreza = postajas[i].mreza;
19   text = "<span class='postaja'" + mreza + "<hr'" + name + "</span'";
20   marker = L.marker([lat,lng]);
21   marker.bindPopup(text,popupOptions);
22   marker.id = id;
23   marker.name = name;
24   Markers.push(marker);
25   marker.addTo(map);
26 }
27
28 window.removeMarkers = function() {
29   Markers.forEach (function(e){
30     map.removeLayer(e);
31   });
32   Markers = [];
33 }
```


```
var popupOptions = {
  offset: new L.Point(3,-25)
};
```

```
var json = JSON.parse($v('P3_JSON'));
var postajas = json.stations;
var marker, name, text, id ,lat, lng, mreza;
var Markers = [];
```

```
for (var i = 0; i < postaja.length; i++) {
  id = [i].id;
  lat = Number(gs[i].lat);
  lng = Number(gs[i].lng);
  name = gs[i].postaja;
  mreza = gs[i].mreza;
  text = '<span class="postaja">' + mreza + "<hr>" + name + '</span>';
  marker = L.marker([lat,lng]);
  marker.bindPopup(text,popupOptions);
  marker.id = id;
  marker.name = name;
  Markers.push(marker);
```



Add more Stuff

- Add circles, polygons, etc
- Check out API documentation from 
- <https://www.mapbox.com/mapbox.js/api/v3.1.1/>

What about Spatial?

When you need to analyse your stuff



SDO_GEOMETRY

- Datatype to store spatial objects in database
- From a point to complex objects
- Needed for spatial analysis
- Spatial applications use this datatype
- Their data can be displayed in APEX.

Convert your data to SDO_GEOMETRY

```
alter table stations add geom_location sdo_geometry;  
-- 8307 for WGS 84 with order lon,lat  
update stations  
set geom_location=sdo_geometry(2001, 8307,  
sdo_point_type(y,x,null),null,null);
```

Type of map for these data, 8307 is common

2001 for points

Columns in table (longitude, latitude) from dataset
Slightly confusing names. Look at data to see what is what



“What is the minimum distance to next station,
for each station?”

```
select postaja, round(min(distance)) distance
from (
  select a.POSTAJA, SDO_GEOM.sdo_distance(
    a.geom_location,
    SDO_DIM_ARRAY(
      SDO_DIM_ELEMENT('long', -180,180,1),
      SDO_DIM_ELEMENT('lat',-90,90,1)),
    b.geom_location,
    SDO_DIM_ARRAY(
      SDO_DIM_ELEMENT('long', -180,180,1),
      SDO_DIM_ELEMENT('lat',-90,90,1))) distance
  from stations a, stations b
  where a.id != b.id)
group by POSTAJA
order by 2 ;
```



POSTAJA	DISTANCE
Ksaverska cesta	19
ZAGREB PPI PM2,5 - Ksaverska cesta	19
Umag sediment	20
Umag, Ulica Eduardo Pascali	20
SLAVONSKI BROD - privremena pokretna postaja	105
SLAVONSKI BROD-2	105
Mirogojska cesta	303
Vrh Martinšćice	410
Kostrena - Martinšćica	410
Vrhovec	640
Prilaz baruna Filipovića	640
Domobranska ulica 2	748
Dr.V.Mačeka 48	748
KUTINA-1	776
Vatrogasni dom (K2) - Kutina	776
KARLOVAC-1	795
RIJEKA-2	831
Banija 18	832
Gripe	867
Split-1	867

Default unit is meter for geodetic data

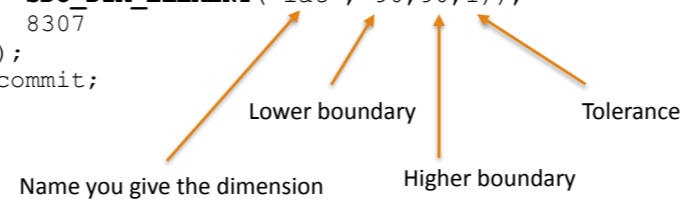


Spatial Index

- Special index to speed up search
- Oracle needs some data about the data — *metadata* before an index can be created
- Insert one row in USER_SDO_GEOM_METADATA for each column (aka *layer*)
- Not all functions requires a spatial index (previous example)
- All *Spatial Operators* do!



```
insert into user_sdo_geom_metadata
(TABLE_NAME,COLUMN_NAME,DIMINFO,SRID)
values ('STATIONS','GEOM_LOCATION',SDO_DIM_ARRAY(
  SDO_DIM_ELEMENT('long', -180,180,1),
  SDO_DIM_ELEMENT('lat',-90,90,1)),
8307
);
commit;
```



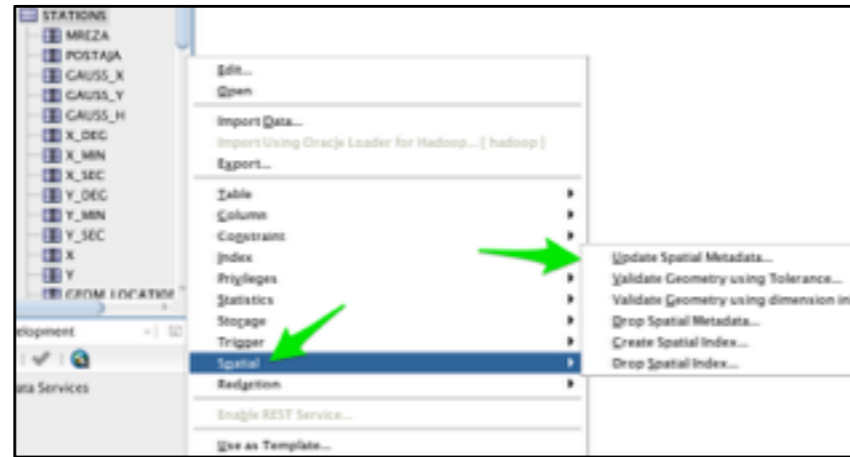
Tip: Shift-F4 in SQL Developer

Put cursor on object_type
Hit Shift-F4

```
...sql SDO_DIM_ARRAY - oracle12c-vagrant-0
Code Profiles Details Errors Dependencies Grants Ref
1 create or replace TYPE SDO_DIM_ARRAY
2
3 AS VARRAY(4) OF SDO_DIM_ELEMENT
```

```
...sql SDO_DIM_ARRAY SDO_DIM_ELEMENT
Code Profiles Details Errors Dependencies Grants Reference
1 create or replace TYPE SDO_DIM_ELEMENT
2
3 AS OBJECT (
4     SDO_DIMNAME    VARCHAR(64),
5     SDO_LB         NUMBER,
6     SDO_UB         NUMBER,
7     SDO_TOLERANCE  NUMBER )
```

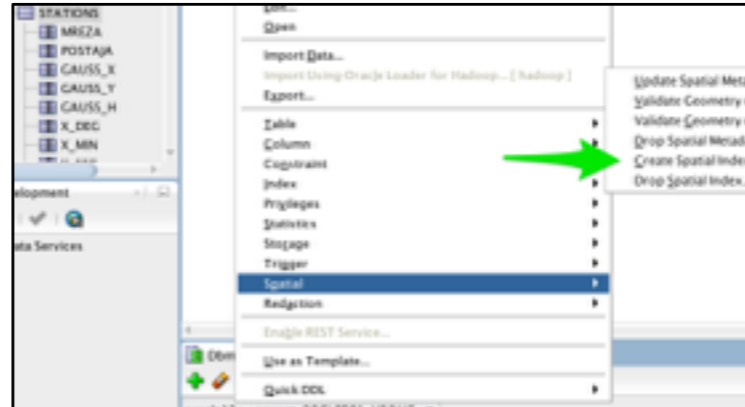
Or use menus in SQL Developer



Create the Index

```
create index stations_geom_location_si  
on stations(geom_location)  
indextype is MDSYS.SPATIAL_INDEX;
```

Package APEX_SPATIAL can
also help with this.



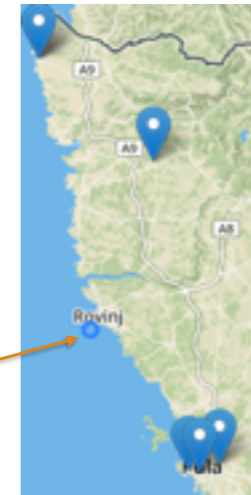
Spatial Operators

- Used as normal operators in WHERE clause
- Filter rows as early before further processing
- Require *Spatial Index*

Find the three closest stations to us

```
select postaja, round(sdo_nn_distance(1)) distance_km,  
       geom_location  
from stations  
where  
sdo_nn(geom_location,  
        sdo_geometry(2001, 8307,  
                     sdo_point_type(13.627097, 45.059248, null), null, null),  
        'sdo_num_res=3 unit=km', 1) = 'TRUE'  
order by 2;
```

Our position



POSTAJA	DISTANCE_KM	GEOM_LOCATION
Pula Fižela	27	[MDSYS.SDO_GEOMETRY]
Ulica Kamenjak (Dječji vrtić)	28	[MDSYS.SDO_GEOMETRY]
VIŠNJAN	28	[MDSYS.SDO_GEOMETRY]

TO_GEOJSON

- Convert SDO_GEOMETRY to GEO_JSON
- GEO_JSON supported in JS API
- SDO_UTIL.TO_GEOJSON
- Example with previous query
- **Warning:** GeoJSON inverts the order (lon, lat)

```

select postaja,round(sdo_nn_distance(1)) distance_km,
       sdo_util.to_geojson(geom_location)location_json
from stations
where
sdo_nn(geom_location,
       sdo_geometry(2001, 8307,
                    sdo_point_type(13.627097,45.059248,null),null,null),
       'sdo_num_res=3 unit=km',1) = 'TRUE'
order by 2;

```

POSTAJA	DISTANCE_KM	LOCATION_JSON
Pula Fižela	27	{ "type": "Point", "coordinates": [13.816858, 44.862469] }
Ulica Kamenjak (Dječji vrtić)	28	{ "type": "Point", "coordinates": [13.83925, 44.858889] }
VIŠNJAN	28	{ "type": "Point", "coordinates": [13.749778, 45.200000] }

Again, 12c makes this so much easier



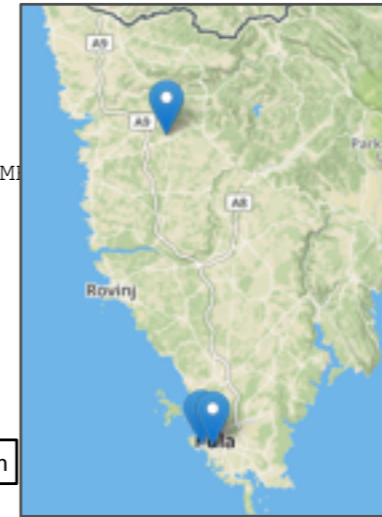
Code before header

```
begin
select '[' || listagg(geo_json,',')
      within group ( order by id) || ']' into :P4_JSON
from (
select id, sdo_util.to_geojson(geom_location) geo_json
from stations
where
sdo_nn(geom_location,
sdo_geometry(2001, 8307,
sdo_point_type(13.627097,45.059248,null),null,null),
'sdo_num_res=3 unit=km',1) = 'TRUE'
);
end;
```



JS on load

```
L.mapbox.accessToken =  
'pk.eyJ1Ijoib2lzMW51IiwiaSI6ImNqN25tOXRmZjMyN3gzNHFwa2IzMjMwLWdQLCw';  
var map = L.mapbox.map('map2Region', 'mapbox.streets')  
    .setView([45.059248, 13.627097], 9);  
  
var geo_json = JSON.parse($v('P4_JSON'));  
var layer = L.geoJson(geo_json).addTo(map);
```



GeoJSON uses lon, lat

Here the order is lat, lon

Add to GeoJSON

- to_geojson returns the geometries-part of GeoJSON
- Add other columns for better UX

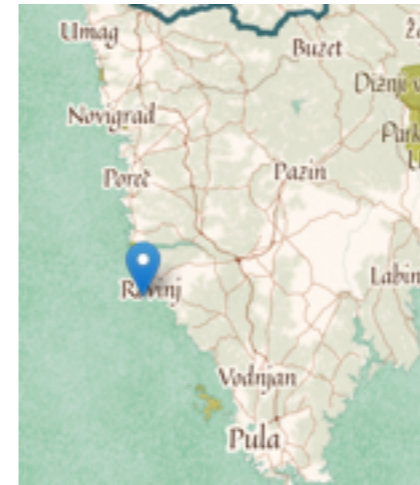
```
{ "type": "FeatureCollection",
  "features": [
    { "type": "Feature",
      "geometry": { "type": "Point", "coordinates": [102.0, 0.5] },
      "properties": { "prop0": "value0" }
    },
    { "type": "Feature",
      "geometry": {
        "type": "LineString",
        "coordinates": [
          [102.0, 0.0], [103.0, 1.0], [104.0, 0.0], [105.0, 1.0]
        ]
      },
      "properties": {
        "prop0": "value0",
        "prop1": 0.0
      }
    },
    { "type": "Feature",
      "geometry": {
        "type": "Polygon",
        "coordinates": [
          [ [100.0, 0.0], [101.0, 0.0], [101.0, 1.0],
            [100.0, 1.0], [100.0, 0.0] ]
        ]
      },
      "properties": {
        "prop0": "value0",
        "prop1": { "this": "that" }
      }
    }
  ]
}
```

GeoJSON example



Try different map styles

- Parameter to L.mapbox.map
- Try mapbox.pirates, mapbox.pencil, mapbox.comic, etc



Beer Data from Untappd

My drinking on a map





- Supporting users can download their data
- Position for each venue and other information.
- As earlier, easy to import with SQL Developer

1	COLUMN_NAME	DATA_TYPE
1	BEER_NAME	VARCHAR2(200 BYTE)
2	BREWERY_NAME	VARCHAR2(128 BYTE)
3	BEER_TYPE	VARCHAR2(128 BYTE)
4	BEER_ADV	NUMBER(6, 2)
5	BEER_IBU	NUMBER(5, 0)
6	TASTING_COMMENT	VARCHAR2(500 BYTE)
7	VENUE_NAME	VARCHAR2(128 BYTE)
8	VENUE_CITY	VARCHAR2(50 BYTE)
9	VENUE_STATE	VARCHAR2(128 BYTE)
10	VENUE_COUNTRY	VARCHAR2(50 BYTE)
11	VENUE_LAT	NUMBER(8, 4)
12	VENUE_LNG	NUMBER(12, 7)
13	RATING_SCORE	NUMBER(5, 2)
14	CREATED_AT	DATE
15	CHECKIN_URL	VARCHAR2(128 BYTE)
16	BEER_URL	VARCHAR2(128 BYTE)
17	BREWERY_URL	VARCHAR2(128 BYTE)
18	BREWERY_COUNTRY	VARCHAR2(128 BYTE)
19	BREWERY_CITY	VARCHAR2(50 BYTE)
20	BREWERY_STATE	VARCHAR2(50 BYTE)



Add spatial column

```
alter table untappd
add location sdo_geometry;

update untappd set location=
  sdo_geometry(2001, 8307,
    sdo_point_type(venue_lng, VENUE_LAT, null), null, null)
where venue_lng is not null
and venue_lat is not null;
commit;
```

Code fetching data

```
select json_object('type' value 'FeatureCollection'  
, 'features' value json_arrayagg(feature)) into :P5_JSON  
from (  
  select json_object('type' value 'Feature',  
    'properties' value json_object('description' value beer_name  
    , 'title' value venue_name),  
    'geometry' value cast (sdo_util.to_geojson(location)  
      as varchar2(4000)) format json) feature  
  from untappd  
  where venue_city='Birmingham'  
  and created_at > date '2016-01-01')  
;
```

JSON_OBJECT() does not
handle CLOB well.

JavaScript code

```
L.mapbox.accessToken =  
'pk.eyJ1Ijoib2lzZW51IiwiaSI6ImNqN25tOXRmZjMyN3gzNHFWa2IzMHR1bGkiLCJ0.WDyAyQ9PaBhf9DdA-  
dQLCw';  
  
var map = L.mapbox.map('map2Region', 'mapbox.streets')  
    .setView([52.489471, -1.898575], 9);  
  
var geo_json = JSON.parse($v('P5_JSON'));  
L.mapbox.featureLayer(geo_json).addTo(map);
```



Summary - You need

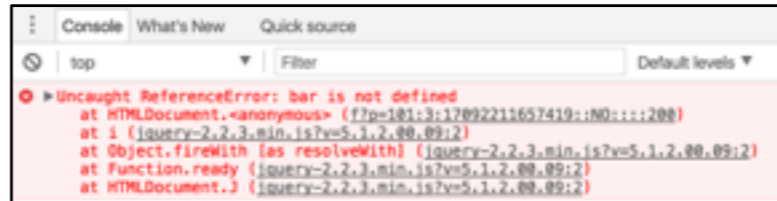
- One JavaScript library / API needed
 - Use Leaflet og Mapbox
- A tile server that serves you map tiles
 - OpenStreetMap, Mapbox, Mapquest
 - Automatic in Mapbox
- A process to load data into JSON
- JavaScript that renders it on the map



Tips at the end
for troubleshooting and fun

When your page is blank

- Developer Tool in Chrome
- “F12” in IE
- Check console for error messages
- Pretty easy to spot typos



```
Console | What's New | Quick source
top | Filter | Default levels
Uncaught ReferenceError: bar is not defined
at HTMLDocument.<anonymous> (ftp-101:3:17892211657419:1:80)
at 1 (jquery-2.2.3.min.js?v=5.1.2.00.09:2)
at Object.fireWith [as resolveWith] (jquery-2.2.3.min.js?v=5.1.2.00.09:2)
at Function.ready (jquery-2.2.3.min.js?v=5.1.2.00.09:2)
at HTMLDocument.J (jquery-2.2.3.min.js?v=5.1.2.00.09:2)
```



Log to console

- `console.log()`
- Verify data and logic
- If data are not shown it may be wrong order
 - RFC 5870

Conclusion



- Apex is easy and fun
- Good support for JSON in 12c
- Display easily spatial and “normal” data
- Show your data and get attention!



Crash introduction to Docker

Spend time on learning, not installation



Oracle on Docker in few steps

1. Download Docker:

- www.docker.com/docker-mac
- www.docker.com/docker-windows
- www.docker.com/docker-oracle-linux

2. Download Oracle database software

- otn.oracle.com

3. Clone Dockerfiles by Oracle from Github

- `git clone github.com/oracle/docker-images`



Build and start with

```
4. ./buildDockerImage.sh -v 12.2.0.1 -e
```

```
5. docker run -p 1521:1521 \  
-p 8080:8080 \  
-name oracle-ee \  
oracle/database:12.2.0.1-ee
```



Up and running

- SYS and SYSTEM password is generated each time a container is created
- Password printed out to screen
- Database is created during first run.
- Use SQL Developer Command Line for easy testing



Or perhaps Vagrant?

- If you know Virtual Box, Vagrant may be even easier.
- Find ready to use stuff here: <https://github.com/gvenzl/vagrant-boxes>