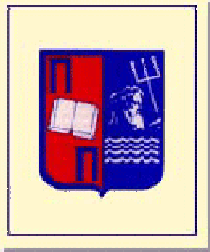


Εργαστηριακές ασκήσεις μαθήματος

Εισαγωγή στο PostGIS



Δρ. Ηλίας Φρέντζος
efrentzo@unipi.gr

PostgreSQL



- Λογισμικό ανοικτού κώδικα (open source), κάτω από BSD license
 - Copyright του Πανεπιστημίου της Καλιφόρνια
 - εκδόσεις για Linux, και Windows
- Τρέχουσα έκδοση: 8.4
- Υποστηρίζει χωρικά δεδομένα:
 - Γεωμετρικοί τύποι δεδομένων
 - Χωρικοί δείκτες
 - Γεωμετρικοί τελεστές / συναρτήσεις
 - Συστήματα αναφοράς



- Επέκταση της PostgreSQL για χωρικά δεδομένα
 - Τύποι δεδομένων βάσει OpenGIS Consortium
 - Ειδικοί τελεστές για την σύνταξη ερωτημάτων
 - Γεωμετρικές συναρτήσεις και τελεστές
 - Δυνατότητα οπτικοποίησης δεδομένων
 - Με το Quantum GIS (QGis) , UDig, GeoServer κ.α.
 - Προγραμματιστικά

Geometry και Well Known Text



- Ο τύπος Geometry δεν είναι πρόσφορος για ερμηνεία
 - «01060000203408000001000000010300000001000000070000008BC3E53F90901C416C1C7573A103504109A69A955B901C412032DD7CB70350418B00E19E04911C419FDD48A3B90350419B060D012B911C4166D1451AAB035041FC5482022B911C41178E9219AB03504173B28BD23E911C41AA125FB4A30350418BC3E53F90901C416C1C7573A1035041»
- Μορφότυπος WellKnownText (WKT)
 - Σημείο: 'POINT(X Y)'
 - Γραμμή: 'LINESTRING(X1 Y1, X2 Y2, ...)'
 - Πολύγωνο: 'POLYGON((X1 Y1, X2 Y2, .., X1 Y1), (Xn,Yn, Xn+1 Yn+1,.., Xn Yn))'

Geometry και Well Known Text



- WKT από Geometry παράγονται με τη συνάρτηση AsText
 - AsText(geometry)

Query - gis_mashup_ve on postgres@localhost:5432 *

```
SELECT lt_id, AsText(lt_geometry) FROM lights
```

Output pane

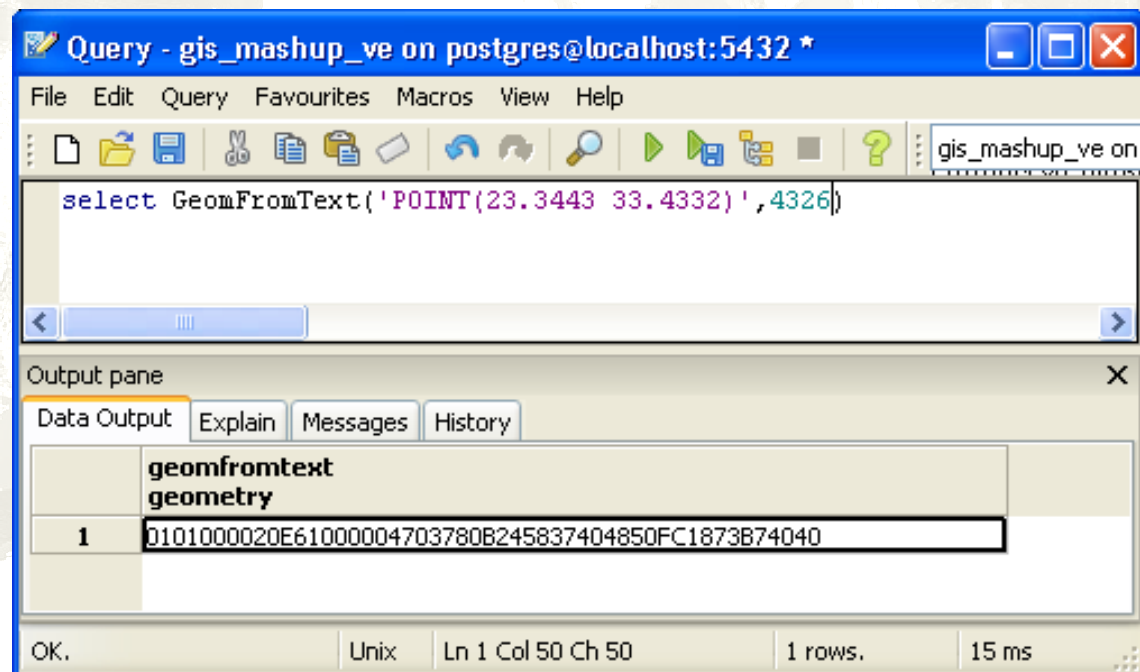
	lt_id integer	astext text
1	38	POINT(23.5956716537476 38.4597554267177)
2	1	POINT(23.596658706665 38.4687105113576)
3	2	POINT(23.596658706665 38.4687105113576)
4	3	POINT(23.5968089103699 38.4603939123387)
5	4	POINT(23.5947811603546 38.4610911993896)
6	5	POINT(23.5943627357483 38.4596126067921)
7	6	POINT(23.5963582992554 38.4583944253473)
8	7	POINT(23.5975813865662 38.4620237052721)
9	8	POINT(23.5983216762543 38.4612550188635)
10	9	POINT(23.5994964838028 38.4609441815963)
11	10	POINT(23.6017656326294 38.4604317172312)
12	11	POINT(23.6006391048431 38.4615322509717)
21	20	POINT(23.5954141616821 38.4615532533608)

OK. Unix Ln 1 Col 39 Ch 39 38 rows. 140 ms

Geometry και Well Known Text



- Γεωμετρίες παράγονται από Text με την συνάρτηση GeomFromText
 - GeomFromText(WellKnownText, [SRID])



Γεωμετρικοί τελεστές



+	Translation	<	Does not extend to the right of?
-	Translation	>	Does not extend to the left of?
*	Scaling/rotation	<<	Is left of?
/	Scaling/rotation	>>	Is right of?
#	Point or box of intersection	<^	Is below?
#	Number of points in path or polygon	>^	Is above?
@-@	Length or circumference	?#	Intersects?
@@	Center	?-	Is horizontal?
##	Closest point to first operand on second operand	?-	Are horizontally aligned?
<->	Distance between	?	Is vertical?
&&	Overlaps?	?	Are vertically aligned?
		?-	Is perpendicular?
		?	Are parallel?
		~	Contains?
		@	Contained in or on?
		~=	Same as?

Γεωμετρικές Συναρτήσεις



area(object)	double precision	area	area(box '((0,0),(1,1))')
box_intersect(box, box)	box	intersection box	box_intersect(box '((0,0),(1,1))', box '((0.5,0.5),(2,2))')
center(object)	point	center	center(box '((0,0),(1,2))')
diameter(circle)	double precision	diameter of circle	diameter(circle '((0,0),2.0)')
height(box)	double precision	vertical size of box	height(box '((0,0),(1,1))')
isclosed(path)	boolean	a closed path?	isclosed(path '((0,0),(1,1),(2,0))')
isopen(path)	boolean	an open path?	isopen(path '[(0,0),(1,1),(2,0)]')
length(object)	double precision	length	length(path '((-1,0),(1,0))')
npoints(path)	integer	number of points	npoints(path '[(0,0),(1,1),(2,0)]')
npoints(polygon)	integer	number of points	npoints(polygon '((1,1),(0,0))')
pclose(path)	path	convert path to closed	pclose(path '[(0,0),(1,1),(2,0)]')
popen(path)	path	convert path to open	popen(path '((0,0),(1,1),(2,0))')
radius(circle)	double precision	radius of circle	radius(circle '((0,0),2.0)')
width(box)	double precision	horizontal size of box	width(box '((0,0),(1,1))')

PostGIS και Συστήματα Αναφοράς



- Το PostGIS υποστηρίζει πλήθος συστημάτων αναφοράς
 - Πίνακας spatial_ref_sys
 - SRID

The screenshot shows the pgAdmin III interface. On the left, the 'Object browser' pane displays the database structure, including the 'gis_mashup' database and the 'spatial_ref_sys' table. The main window shows the 'Edit Data' view for the 'spatial_ref_sys' table. The table contains 12 rows of data, each representing a spatial reference system (SRID). The columns are: sr_id, auth_name, auth_srid, srs_text, and proj4text. The data is as follows:

sr_id	auth_name	auth_srid	srs_text	proj4text
98	2097	EP5G	PROJCS["Korean 1985 / Korea Central Belt", GE...	+proj=merc +lat_0=38 +lon_0=127 +k=1
99	2098	EP5G	PROJCS["Korean 1985 / Korea West Belt", GE...	+proj=merc +lat_0=38 +lon_0=125 +k=1
100	2099	EP5G	PROJCS["Qatar 1948 / Qatar Grid", GEOGCS["C...	+proj=merc +lat_0=25.38236111111111 +l...
101	2100	EP5G	PROJCS["GGRS87 / Greek Grid", GEOGCS["GR...	+proj=merc +lat_0=38 +lon_0=24 +k=0.9...
102	2101	EP5G	PROJCS["Lake / Maracabo Grid M1", GEOGCS["...	+proj=merc +lat_1=10.16666666666667 +lat...
103	2102	EP5G	PROJCS["Lake / Maracabo Grid M2", GEOGCS["L...	+proj=merc +lat_1=10.16666666666667 +lat...
104	2103	EP5G	PROJCS["Lake / Maracabo Grid M3", GEOGCS["L...	+proj=merc +lat_1=10.16666666666667 +lat...
105	2104	EP5G	PROJCS["Lake / Maracabo La Rosa Grid", GEO...	+proj=merc +lat_1=10.16666666666667 +lat...
106	2105	EP5G	PROJCS["NZGD2000 / Mount Eden Circuit 2000...	+proj=merc +lat_0=-36.87972222222222
107	2106	EP5G	PROJCS["NZGD2000 / Bay of Plenty Circuit 200...	+proj=merc +lat_0=-37.76111111111111
108	2107	EP5G	PROJCS["NZGD2000 / Poverty Bay Circuit 2000...	+proj=merc +lat_0=-38.62444444444444
109	2108	EP5G	PROJCS["NZGD2000 / Hawkes Bay Circuit 2000...	+proj=merc +lat_0=-39.65083333333333
110	2109	EP5G	PROJCS["NZGD2000 / Taranaki Circuit 2000", G...	+proj=merc +lat_0=-39.13555555555556
111	2110	EP5G	PROJCS["NZGD2000 / Tuhirangi Circuit 2000", G...	+proj=merc +lat_0=-39.51222222222222
112	2111	EP5G	PROJCS["NZGD2000 / Wanganui Circuit 2000", G...	+proj=merc +lat_0=-40.24194444444444

PostGIS και Συστήματα Αναφοράς



	Σύστημα Συντεταγμένων	Τύπος	Μονάδες μέτρησης	SRID
1	Ελληνικό Γεωδαιτικό Σύστημα Αναφοράς 1987 (ΕΓΣΑ87)	Προβολικό	Μέτρα	2100
2	UTM Ζώνη 34 επί του WGS84	Προβολικό	Μέτρα	32634
3	UTM Ζώνη 35 επί του WGS84	Προβολικό	Μέτρα	32635
4	WGS84	Γεωγραφικό	Μοίρες	4326
5	Mercator Spheric	Γεωγραφικό	Μέτρα	900913

Εργαστήριο στο PostGIS



- <http://isl.cs.unipi.gr/db/courses/gis/lab/>
- Κατεβάστε το postgresql_lab_data.zip
- Αποσυμπιέστε το zip στον φάκελο
P:\postgresql
- Περιέχει
 - ένα shape file
 - δύο αρχεία sql (scripts)

Βήμα 1ο: Δημιουργία καινούργιας βάσης δεδομένων



```
createdb -T template_postgis -E UTF8  
lab_spatial_db
```

ή

```
SET ROLE postgres;
```

```
CREATE DATABASE lab_spatial_db  
TEMPLATE=template_postgis  
ENCODING='UTF8'  
TABLESPACE=pg_default;
```

Βήμα 2ο: Σύνδεση στη βάση



```
\c lab_spatial_db
```

Βήμα 3ο: Δημιουργία σχήματος με την ονομασία lab:

```
CREATE SCHEMA lab;
```

Βήμα 4ο: Δημιουργία πίνακα σημείων ενδιαφέροντος με την ονομασία landmarks:



```
CREATE TABLE lab.Landmarks(  
    landmark_id integer primary key,  
    landmark_name varchar(50),  
    landmark_geometry geometry);
```

Βήμα 5ο: Δημιουργία πίνακα δρόμων με την ονομασία roads:

```
CREATE TABLE lab.roads(  
    road_id integer primary key,  
    road_name varchar(50),  
    road_geometry geometry);
```


Βήμα 6ο: Δημιουργία πίνακα γεωτεμαχίων με την ονομασία parcels :



```
CREATE TABLE lab.parcels(  
    parcel_id integer primary key,  
    parcel_name varchar(50),  
    parcel_geometry geometry);
```

Βήμα 7ο: Δημιουργία πίνακα ζωνών με την ονομασία zones:

```
CREATE TABLE lab.zones(  
    zone_id integer primary key,  
    zone_name varchar(50),  
    zone_geometry geometry);
```

Βήμα 8ο: Δημιουργία πίνακα χωρικών αντικειμένων με χρήση των προτύπων του Open GIS Consortium.



```
CREATE TABLE lab.dummy(  
dummy_id integer primary key,  
dummy_name varchar(50));
```

```
AddGeometryColumn(<schema_name>, <table_name>,  
<column_name>, <srid>, <type>, <dimension>)
```

Το SRID είναι το σύστημα προβολής των δεδομένων. Για ΕΓΣΑ'87, SRID = 2100

```
SELECT AddGeometryColumn('lab', 'dummy',  
'dummy_geometry', 2100, 'GEOMETRY', 2);
```

Βήμα 9ο: Δημιουργία χωρικών ευρετηρίων GiST επάνω στους πίνακες landmarks, roads, parcels, zones



```
CREATE INDEX landmarks_spatial_index  
ON lab.landmarks USING  
GIST(landmark_geometry);
```

```
CREATE INDEX roads_spatial_index  
ON lab.roads USING GIST(road_geometry);
```

```
CREATE INDEX parcels_spatial_index  
ON lab.parcels USING GIST(parcel_geometry);
```

```
CREATE INDEX zones_spatial_index  
ON lab.zones USING GIST(zone_geometry);
```


Βήμα 10ο: Εισαγωγή δεδομένων στον πίνακα σημείων ενδιαφέροντος με διαδοχικές εντολές insert:



```
INSERT INTO lab.landmarks  
VALUES(1, 'Landmark1',GeomFromText('POINT(468346  
4198876)', 2100));  
INSERT INTO lab.landmarks  
VALUES(2, 'Landmark2',GeomFromText('POINT(468518  
4198644)', 2100));
```

Βήμα 11ο: Εισαγωγή δεδομένων στον πίνακα δρόμων με διαδοχικές εντολές insert:

```
INSERT INTO lab.roads  
VALUES(1, 'Road1',GeomFromText('LINESTRING(469180  
4199872, 469171 4199855)', 2100));  
INSERT INTO lab.roads  
VALUES(2, 'Road2',GeomFromText('LINESTRING(468994  
4199855, 469092 4199882)', 2100));
```



Βήμα 12ο: Εισαγωγή δεδομένων στον πίνακα ζωνών με διαδοχικές εντολές insert:

```
INSERT INTO lab.zones
VALUES(1, 'Zone1', GeomFromText('POLYGON((467646 4198420,
467734 4197970, 467962 4198009, 467882 4198464, 467646
4198420))', 2100));
INSERT INTO lab.zones
VALUES(2, 'Zone2', GeomFromText('POLYGON((467882 4198464,
467962 4198009, 468248 4198055, 468224 4198156, 468264
4198163, 468233 4198347, 468199 4198522, 467882
4198464, 467882 4198464))', 2100));
```

Βήμα 13ο: Εισαγωγή δεδομένων από ASCII αρχείο sql.

```
DELETE FROM lab.roads;
DELETE FROM lab.landmarks;
```

Εισαγωγή από sql αρχείο (script): `psql -d [database] -f [filename]`

Επομένως, στην περίπτωση μας έχουμε:

```
psql -d lab_spatial_db -f P:\landmarks.sql
psql -d lab_spatial_db -f P:\roads.sql
```




Βήμα 14ο: Εισαγωγή δεδομένων από shape file.

```
shp2pgsql -s 2100 [shapefile] [schema].[table] |  
psql -d [database] -U [user]
```

Επομένως, στην περίπτωση μας έχουμε:

```
shp2pgsql -s 2100 P:\parcels.dbf lab.parcels_temp |  
psql -d lab_spatial_db -U postgres
```

Βήμα 15ο: Μεταφορά δεδομένων :

```
INSERT INTO lab.parcels (parcel_id, parcel_geometry)  
SELECT id, the_geom FROM lab.parcels_temp;
```

```
DROP TABLE lab.parcels_temp;
```


Βήμα 16ο: Εμφάνιση των δεδομένων



```
SELECT * FROM lab.landmarks;
```

Μετατροπή κατά την εμφάνιση σε WKT format

```
SELECT landmark_id, landmark_name,  
AsText(landmark_geometry) FROM lab.landmarks;
```

Βήμα 17ο: Οπτικοποίηση των δεδομένων. Εκκινήστε την εφαρμογή Quantum GIS

Settings > Project Properties

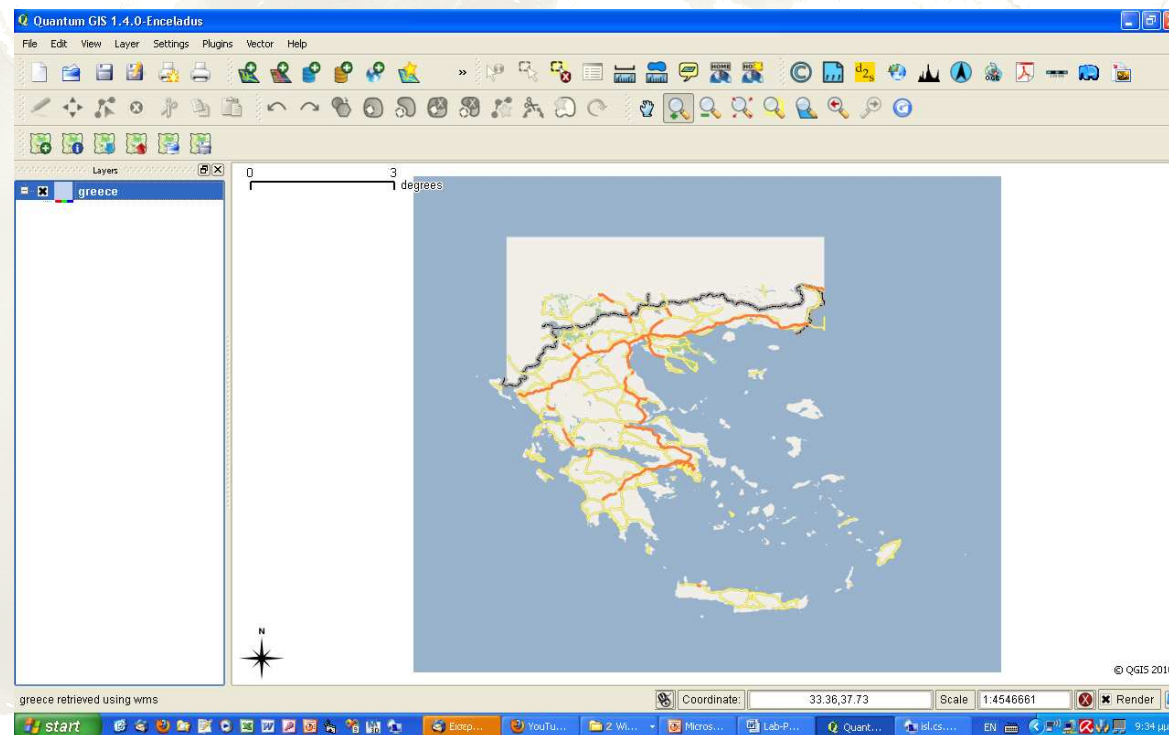
Name: *lab_spatial_db*
host: *localhost*
database: *lab_spatial_db*
port: *5432*
username: *postgres*
password: *dbpassword*

Βήμα 18ο: Εμφάνιση WMS Layer



Name: *infolab.wms*

URL: *http://isl.cs.unipi.gr:8080/geoserver/wms*



Βήμα 19ο: Πραγματοποιήστε μία ερώτηση επιλογής με βάση ένα σημείο:



```
SELECT parcel_id FROM lab.parcels WHERE  
parcel_geometry && GeomFromText('POINT(468010  
4198347)',2100);
```

Βήμα 20ο: Πραγματοποιήστε μία ερώτηση επιλογής με βάση την απόσταση από σημείο:

```
FROM lab.roads WHERE road_geometry &&  
Expand(GeomFromText('POINT(468010 4198347)',2100),  
100) AND Distance(GeomFromText('POINT(468010  
4198347)',2100),road_geometry)< 100;
```

```
SELECT * FROM lab.roads WHERE Distance(GeomFromText  
( 'POINT(468010 4198347)',2100),road_geometry)< 100;
```

```
CREATE VIEW lab.expand_test AS  
SELECT landmark_id, Expand(landmark_geometry, 20) AS  
expanded_geometry FROM lab.landmarks;
```


Βήμα 21ο: Ερώτηση επιλογής με παραλληλόγραμμο:



```
SELECT * FROM lab.landmarks
WHERE box( '467350,4198000,467686,4198277' ) &&
landmark_geometry;
```

-ή-

```
SELECT * FROM lab.landmarks
WHERE Contains(GeomFromText( 'POLYGON((467350
4198000, 467350 4198277, 467686 4198277, 467686
4198000, 467350 4198000))',2100), landmark_geometry);
```

```
SELECT * FROM lab.landmarks
WHERE landmark_geometry @
GeomFromText( 'POLYGON((467350 4198000, 467350
4198277, 467686 4198277, 467350 4198000))', 2100);
```

-ή-

```
SELECT * FROM lab.landmarks
WHERE Contains(GeomFromText( 'POLYGON((467350
4198000, 467350 4198277, 467686 4198277, 467350
4198000))', 2100), landmark_geometry);
```

Βήμα 22ο: Ερώτηση σύνδεσης (σημείο – πολύγωνο):



```
SELECT landmark_id, landmark_geometry
FROM lab.landmarks, lab.parcels
WHERE landmark_geometry @ parcel_geometry;
```

```
SELECT landmark_id, landmark_geometry
FROM lab.landmarks, lab.parcels
WHERE Contains(parcel_geometry, landmark_geometry);
```

Βήμα 23ο: Ερώτηση σύνδεσης (πολύγωνο - πολύγωνο):

```
SELECT parcel_id, parcel_geometry
FROM lab.parcels, lab.zones
WHERE contains(zone_geometry, parcel_geometry) AND
zone_name='Zone1'
```

Βήμα 24ο: Ερώτημα χωρικής αυτοσύνδεσης.



```
SELECT L1.landmark_id, L1.landmark_geometry,  
       L2.landmark_id, L2.landmark_geometry,  
       distance(L1.landmark_geometry,  
L2.landmark_geometry)  
FROM lab.landmarks L1, lab.landmarks L2  
WHERE distance(L1.landmark_geometry,  
L2.landmark_geometry)<50  
       AND L1.landmark_id<>L2.landmark_id;
```

Βήμα 25ο: Χρήση συναρτήσεων:

```
SELECT parcel_id, parcel_name, area(parcel_geometry)  
FROM lab.parcels;
```

```
SELECT road_id, road_name, length(road_geometry)  
FROM lab.roads WHERE length(road_geometry)>100;
```


Βήμα 26ο: Buffer χωρικών αντικειμένων



```
SELECT parcel_id, parcel_geometry, road_id  
FROM lab.parcels, lab.roads  
WHERE buffer(road_geometry,6) && parcel_geometry;
```

```
SELECT parcel_id, parcel_geometry, road_id  
FROM lab.parcels, lab.roads  
WHERE intersects(buffer(road_geometry,6),  
parcel_geometry);
```

```
CREATE VIEW lab.buffer_selection AS  
SELECT DISTINCT parcel_id, parcel_geometry  
FROM lab.parcels, lab.roads  
WHERE intersects(buffer(parcel_geometry,6),  
road_geometry) AND road_id>=1200 AND road_id<1300;
```

Βήμα 27ο: Συναθρήσεις



```
SELECT  
ConvexHull(collect(landmark_geometry))  
FROM lab.landmarks;
```

και

```
SELECT  
ConvexHull(collect(landmark_geometry))  
FROM lab.landmarks  
GROUP BY (landmark_id/10)::int;
```