PostGIS t&t:
рассказ по заявкам

Дорофей Пролесковский
Минск, Беларусь

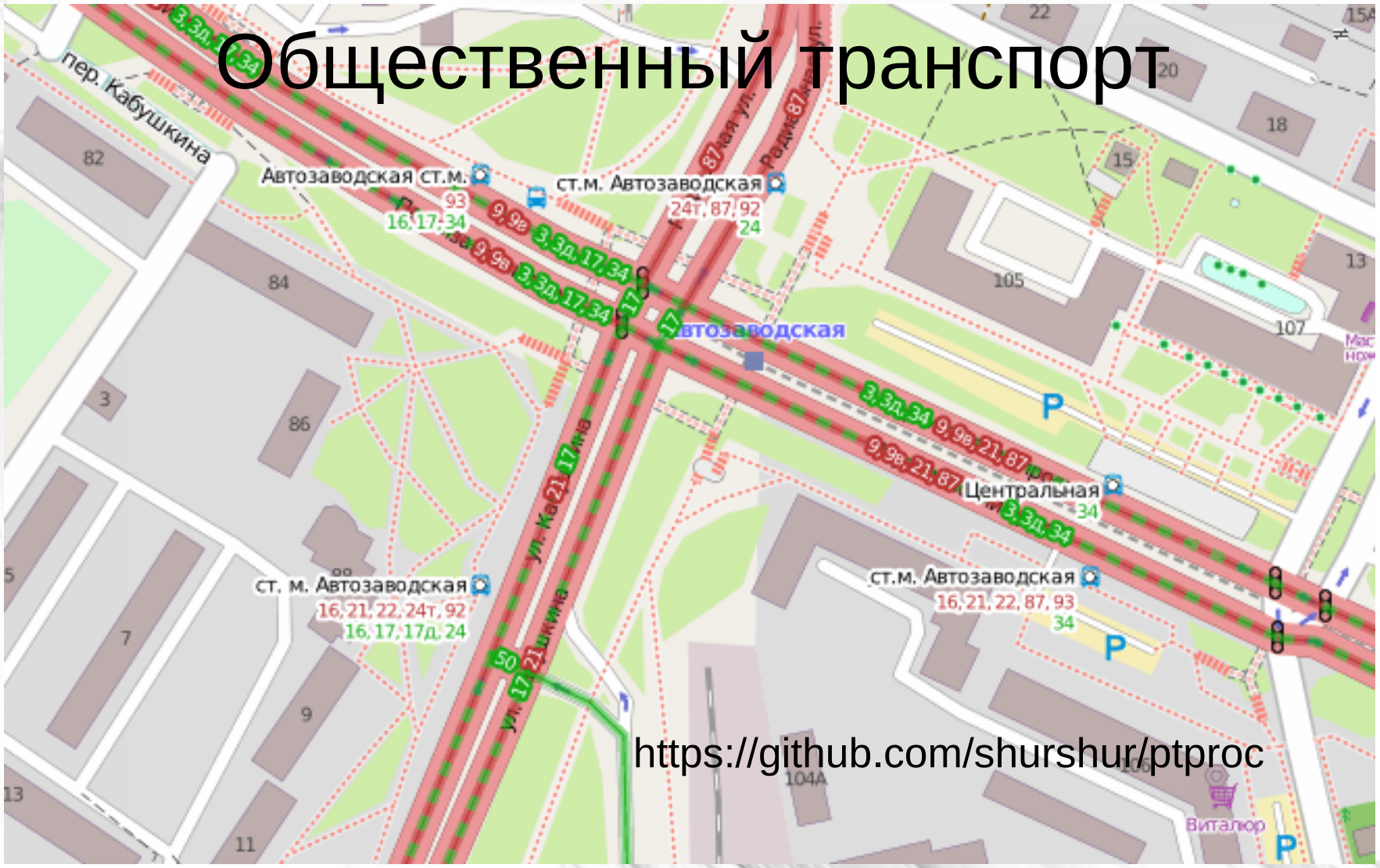
Альтернативы PostGIS

- MySQL: spatial?...
- Oracle Spatial: free?..
- MS SQL: no comments
- Поточная обработка: 8 часов на любой запрос?

Mapnik

- Популярный быстрый GIS-рендерер
- Позволяет встраивать в стиль запросы к PostgreSQL
- Умеет рисовать на cairo- и agg-surface

Общественный транспорт



komap

KOthic MApcss Processor

- Отвязывает стиль от конкретного рендерера
- Содержит в себе все необходмиые work-around'ы
- Упрощает разработку стиля

i18n

```
172   if locale == "en":
173     columnmap["name"] = ("""COALESCE("name:en","int_name",
        replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replace
        (replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replace(replac
        e(replace(replace(replace(replace(translate("name",'абвгдезиклмнопрстуфьАБВГДЕЗИКЛМНОПР
        СТУФЬ','abvgdeziklmnoprstuf'ABVGDEZIKLMNOPRSTUF'),'х','kh'),'Х','Kh'),'ц','ts'),'Ц','T
        s'),'ч','ch'),'Ч','Ch'),'ш','sh'),'Ш','Sh'),'щ','shch'),'Щ','Shch'),'ъ','"'),'Ъ','"'),'
        ё','yo'),'Ё','Yo'),'ы','y'),'Ы','Y'),'э','e'),'Э','E'),'ю','yu'),'Ю','Yu'),'й','y'),'Й
        ','Y'),'я','ya'),'Я','Ya'),'ж','zh'),'Ж','Zh')) AS name""",('name:en','int_name',))
174   elif locale:
175     columnmap["name"] =  ('COALESCE("name:'+locale+'", "name") AS name',('name:'+locale,))
```

ST_ForceRHR

SELECT %s, ST_ForceRHR(way)
as way from planet_osm_polygon where …

ORDER BY

- OSM – текстовые метаданные, но иногда и числа
- order by
  (CASE
    WHEN "%s" ~ E'^[[:digit:]]+([.][[:digit:]]+)?$'
     THEN to_char(
        CAST ("%s" AS FLOAT),
        '0000000000000.9999999999')
   else "%s" end)
   nulls last

ST_PointOnSurface

Точка, гарантированно находящаяся внутри полигона

select ST_PointOnSurface as way, ...

Не бойтесь строить индексы!

```
create index on planet_osm_line (man_made);
create index on planet_osm_line ("name:en");
create index on planet_osm_line (name);

create index on planet_osm_point (place);
create index on planet_osm_point (int_name);
create index on planet_osm_point ("name:ru");
create index on planet_osm_point ("name:en");
create index on planet_osm_point (name);
create index on planet_osm_point ("natural");
create index on planet_osm_point ("addr:street");

create index on planet_osm_polygon (admin_level);
create index on planet_osm_polygon (landuse);
create index on planet_osm_polygon ("name:en");
create index on planet_osm_polygon (name);
create index on planet_osm_polygon ("natural");
```

Индексные выборки

- Самая быстрая выборка по произвольному полю – проверка на равенство:
  select “name” from planet_osm_point
  where amenity='pub';

- Самая быстрая геометрическая выборка - && - автоматически используется PostGIS, когда возможно

(ST_Intersects, ST_DWithin...)

COALESCE

- Возвращает первое непустое поле
  COALESCE(name || ' (' || ref || ')', name, ref)

- Улица Строителей (P41)
- Улица Строителей
- P41

ST_Buffer(.... ,0)

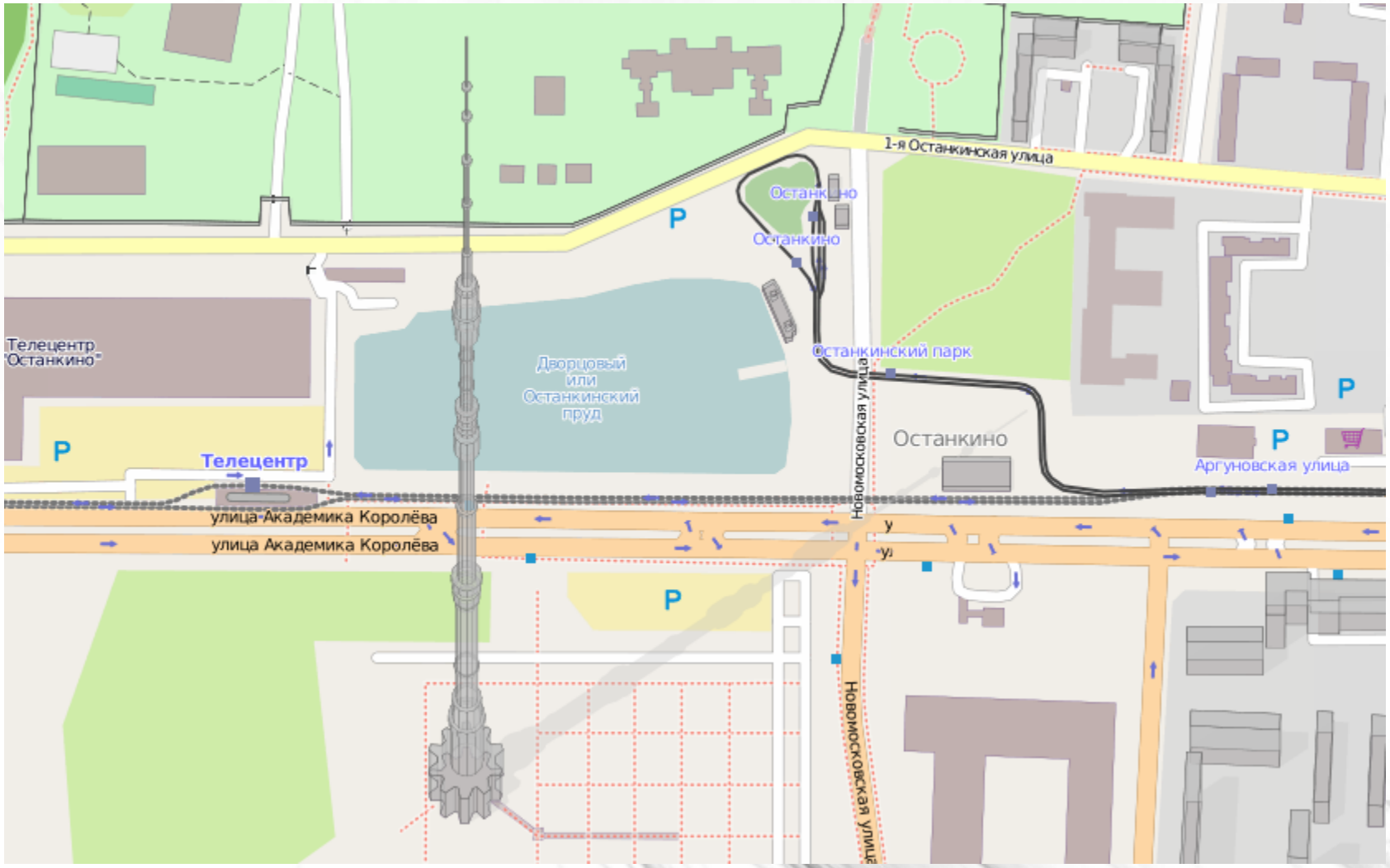
gis=> select **ST_IsValid**(way) from planet_osm_polygon where **not ST_IsValid(way)** limit 1;
NOTICE: Self-intersection at or near point 1.11887e+07 1.56352e+06
NOTICE: Self-intersection at or near point 1.11887e+07 1.56352e+06
 st_isvalid
----
 f
(1 row)

gis=> select ST_IsValid(**ST_Buffer(**way,0**)**) from planet_osm_polygon where not ST_IsValid(way) limit 1;
NOTICE: Self-intersection at or near point 1.11887e+07 1.56352e+06
 st_isvalid
----
 t
(1 row)

Where way && !bbox!

- Mapnik > 0.7.1
- Значительно ускоряет сложно вложенные выборки
- Автоматически подставляется в простые запросы

3D-здания


1-я Останкинская улица
Останкино
Останкино
Телецентр "Останкино"
Дворцовый или Останкинский пруд
Останкинский парк
Новомосковская улица
Останкино
Аргуновская улица
Телецентр
улица Академика Королёва
улица Академика Королёва
Новомосковская улица
P

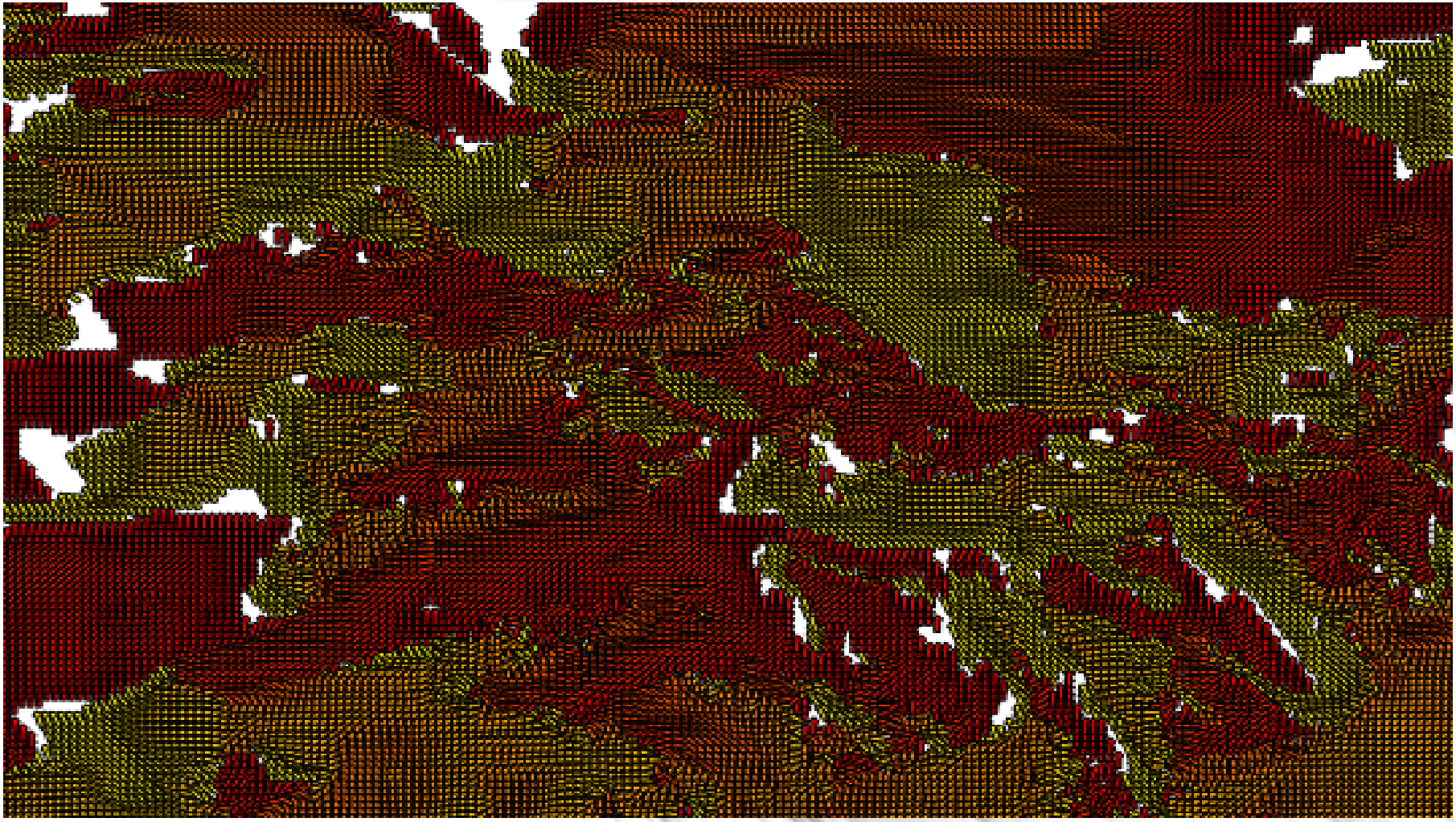
Напоминание:

Рассказ ведётся по заявкам и с учётом предложений, высказанных слушателями

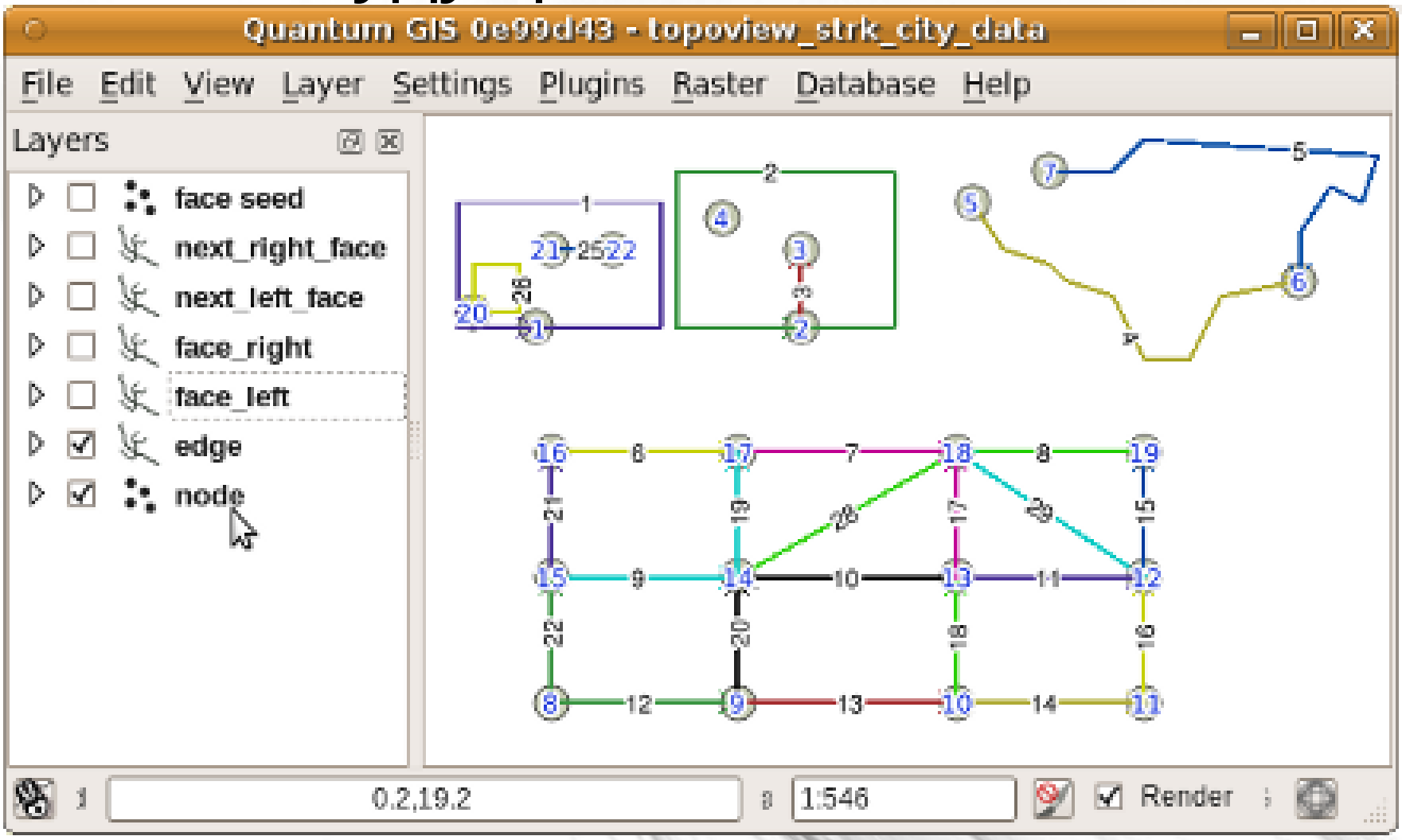
```
(
with buildings as (select (ST_Dump(ST_Union(way))).geom as way, minhei, hei from (select  way,
                (CASE WHEN z."height" ~ E'^(-)?[[:digit:]]+([.][[:digit:]]+)?$' THEN CAST (z."height" AS FLOAT) ELSE
                    CASE WHEN z."building:levels" ~ E'^(-)?[[:digit:]]+([.][[:digit:]]+)?$' THEN CAST (z."building:levels" AS FLOAT) * 3 ELSE 0 END END) as hei,
                 (CASE WHEN z."min_height" ~ E'^(-)?[[:digit:]]+([.][[:digit:]]+)?$' THEN CAST (z."min_height" AS FLOAT) ELSE
                    CASE WHEN z."building:min_level" ~ E'^(-)?[[:digit:]]+([.][[:digit:]]+)?$' THEN CAST (z."building:min_level" AS FLOAT) * 3 ELSE 0 END END) as minhei

                    from
(select  ST_Buffer(way,0) as way, "building:levels", "height", "building:min_level", "min_height" from

planet_osm_polygon  where (building is not NULL or "building:part" is not NULL or "man_made" in ('chimney', 'tower') or (area is not NULL and barrier in ('fence','wall') ) ) and
(height is not NULL or "building:levels" is not NULL) and way &amp;&amp; expand(!bbox!,700)

union

select ST_Buffer(way,10), "building:levels", "height", "building:min_level", "min_height" from

planet_osm_point  where (building is not NULL or "man_made" in ('chimney', 'tower') ) and (height is not NULL or "building:levels" is not NULL) and way &amp;&amp; expand(!
bbox!,500)

union


select ST_Buffer(way,CASE WHEN "width" ~ E'^(-)?[[:digit:]]+([.][[:digit:]]+)?$' THEN CAST ("width" AS FLOAT) ELSE 0.1 END, 'endcap=flat' ), "building:levels", case when
"height" is not NULL then height else '2.5' end as height, "building:min_level", "min_height" from

planet_osm_line  where (building is not NULL or "man_made" in ('chimney', 'tower')  or barrier in ('fence','wall','hedge')) and way &amp;&amp; expand(!bbox!,500)

union
select ST_Buffer(ST_Boundary(way),0.1), "building:levels", case when "height" is not NULL then height else '2.5' end as height, "building:min_level", "min_height" from

planet_osm_polygon where ( barrier in ('fence','wall')) and "area" is NULL and way &amp;&amp; expand(!bbox!,500)

) z

union

select b.way, case when a.residential = 'urban' then 9*1 else 3*1 end as hei, 0 as minhei from planet_osm_polygon b join planet_osm_polygon a on ( b.building is not NULL and
b."building:levels" is  NULL and b.height is NULL and ST_DWithin(b.way, a.way, 0) and  (a.residential is not NULL or a.landuse='garages') and b.way &amp;&amp; !bbox!)


)  p  group by minhei, hei )
```

```
		-- Walls fill
select (ST_ConvexHull(ST_Collect(ST_MakeLine(ST_Translate(p2,0,minhei),ST_Translate(p1,0,minhei)), ST_MakeLine(ST_Translate(p2,0,hei),ST_Translate(p1,0,hei))))) as way, LEAST(ST_Y(p1),ST_Y(p2)) as lea, 1 as hv,
case when ST_X(p1)&amp;gt;ST_X(p2) then 'yes' else 'no' end as visible,
ST_Azimuth(p2,p1)/3.14159265358*180 as azim, minhei, hei, 'area' as fill

from (
  select (ST_PointN(way,generate_series(1, ST_NPoints(way)-1))) as p1,
       (ST_PointN(way,generate_series(2, ST_NPoints(way)))  ) as p2, hei*2 as hei, minhei*2 as minhei
         from (select (ST_Dump(ST_Boundary(ST_ForceRHR(ST_SimplifyPreserveTopology(way,.4))))).geom as way,minhei, hei from buildings) p) a

		-- Roofs
union select ST_Translate(way,0,hei) as way, ST_YMin(way) as lea, 2 as hv, 'yes-h' as visible, -500 as azim, minhei, hei, 'area' as fill
		from (select case when o.way IS NOT NULL then ST_Difference(n.way,o.way) else n.way end as way, n.minhei *2 as minhei, n.hei*2 as hei
from buildings n LEFT OUTER JOIN (select ST_Union(way) as way, minhei from buildings group by minhei) o on (n.hei = o.minhei)  ) p


		-- Walls lines
union

select (ST_MakeLine(ST_Translate(p2,0,minhei),ST_Translate(p2,0,hei))) as way, LEAST(ST_Y(p1),ST_Y(p2)) as lea, 1 as hv,
case when ST_X(p1)&amp;gt;ST_X(p2) then 'yes-v' else 'no-v' end as visible,
ST_Azimuth(p2,p1)/3.14159265358*180 as azim, minhei, hei, 'line' as fill


from (
  select (ST_PointN(way,generate_series(1, ST_NPoints(way)-1))) as p1,
       (ST_PointN(way,generate_series(2, ST_NPoints(way)))  ) as p2, hei*2 as hei, minhei*2 as minhei
         from (select (ST_Dump(ST_Boundary(ST_ForceRHR(ST_SimplifyPreserveTopology(way,.4))))).geom as way,minhei, hei from buildings) p) a

		-- Roofs lines
union select ST_Translate(way,0,hei) as way, ST_YMin(way) as lea, 2 as hv, 'no-h' as visible, -500 as azim, 0 as minhei, hei, 'line' as fill
		from (select case when o.way IS NOT NULL then ST_SymDifference(n.way,o.way) else n.way end as way, n.hei*2 as hei
from (select ST_Union(way) as way, hei from buildings group by  hei) n LEFT OUTER JOIN (select ST_Union(way) as way, minhei from buildings group by minhei) o on (n.hei = o.minhei)  ) p


		-- Bottom lines
union
select way, ST_YMin(way) as lea, 3 as hv, 'no-h' as visible, -500 as azim, minhei, hei, 'line' as fill
		from (select  n.way as way, n.minhei *2 as minhei, n.hei*2 as hei
from buildings n where n.minhei=0 ) p

		-- Shadows

union
select

ST_Difference(a.way,  b.way) as way,

 0 as lea, 3 as hv,
'shadow' as visible,
-500 as azim, 0 as minhei, 0 as hei, 'area' as fill


 from ( select ST_Buffer(ST_Collect(way),0) as way from(



 select ST_ConvexHull(ST_Collect(ST_MakeLine(ST_Translate(p2,minhei*1.5,minhei),ST_Translate(p1,minhei*1.5,minhei)),
 ST_MakeLine(ST_Translate(p2,hei*1.5,hei),ST_Translate(p1,hei*1.5,hei))))       as way


 from (
   select (ST_PointN(way,generate_series(1, ST_NPoints(way)-1))) as p1,
        (ST_PointN(way,generate_series(2, ST_NPoints(way)))  ) as p2, hei as hei, minhei as minhei
          from (select (ST_Dump(ST_Boundary(ST_ForceRHR(ST_SimplifyPreserveTopology(way,.4))))).geom as way,minhei, hei from buildings) p) a


union
select ST_Translate(way, hei*1.5, hei) as way   from buildings



) a ) a, (select ST_Buffer(ST_Collect(way),0) as way from buildings where minhei=0 ) b



order by minhei, lea DESC,fill,hv, hei
```

Будущее: растры



Будущее: Топология



Спасибо за внимание

Ваши вопросы?

me@komzpa.net