Геоданные в Postgres

Дмитрий Лебедев, BestPlace

PgConf Russia 2017

Геоданные и что с ними делать

- многоквартирные дома в городе
- продовольственные магазины
- контуры парков и зелёных зон
- места происшествий и дтп

Сколько интересного можно найти

- Транспортная доступность на велосипеде и общественном транспорте. Сколько выигрывает велосипед и где он нужен в городе?
- Насколько активны кварталы
- Где промзона распродана, а где жива

Содержание

1. Прототипирование. Быстрый ввод, обработка и показ результатов. Импорт.
2. Тонкости: загрузка данных из OpenStreetMap, системы координат.
3. Примеры исследований на PostGIS
4. Организация работы, устранение рутины и собственных ошибок

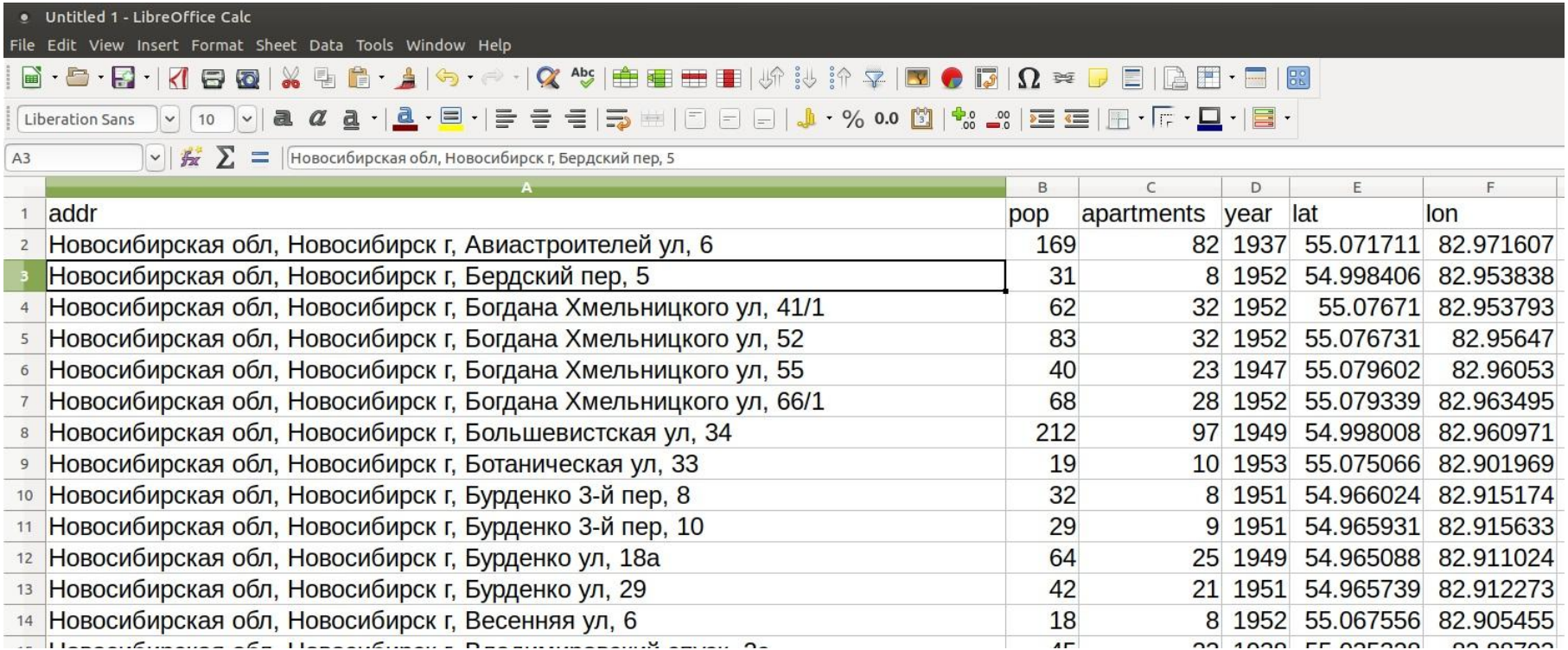
Простейшее исследование: плотность жителей

Создание БД (PostGIS v2.0+)

- Установка Postgis, в Ubuntu всё просто: **`sudo apt install postgis`**
- Создание БД для экспериментов:

```
create database gis_experiment;
\c gis_experiment
create extension postgis;
```

Ввод данных. CSV



	A	B	C	D	E	F
1	addr	pop	apartments	year	lat	lon
2	Новосибирская обл, Новосибирск г, Авиастроителей ул, 6	169	82	1937	55.071711	82.971607
3	Новосибирская обл, Новосибирск г, Бердский пер, 5	31	8	1952	54.998406	82.953838
4	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 41/1	62	32	1952	55.07671	82.953793
5	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 52	83	32	1952	55.076731	82.95647
6	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 55	40	23	1947	55.079602	82.96053
7	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 66/1	68	28	1952	55.079339	82.963495
8	Новосибирская обл, Новосибирск г, Большевистская ул, 34	212	97	1949	54.998008	82.960971
9	Новосибирская обл, Новосибирск г, Ботаническая ул, 33	19	10	1953	55.075066	82.901969
10	Новосибирская обл, Новосибирск г, Бурденко 3-й пер, 8	32	8	1951	54.966024	82.915174
11	Новосибирская обл, Новосибирск г, Бурденко 3-й пер, 10	29	9	1951	54.965931	82.915633
12	Новосибирская обл, Новосибирск г, Бурденко ул, 18а	64	25	1949	54.965088	82.911024
13	Новосибирская обл, Новосибирск г, Бурденко ул, 29	42	21	1951	54.965739	82.912273
14	Новосибирская обл, Новосибирск г, Весенняя ул, 6	18	8	1952	55.067556	82.905455

Ввод данных. CSV

- Импорт:

```
pg> create temporary table houseimport (addr text, pop
integer, apartments integer, year integer, lat float, lon
float);

pg> copy houseimport from '/tmp/novosib.csv' csv header;
```

Ввод данных. CSV

- Форматирование в геоданные:

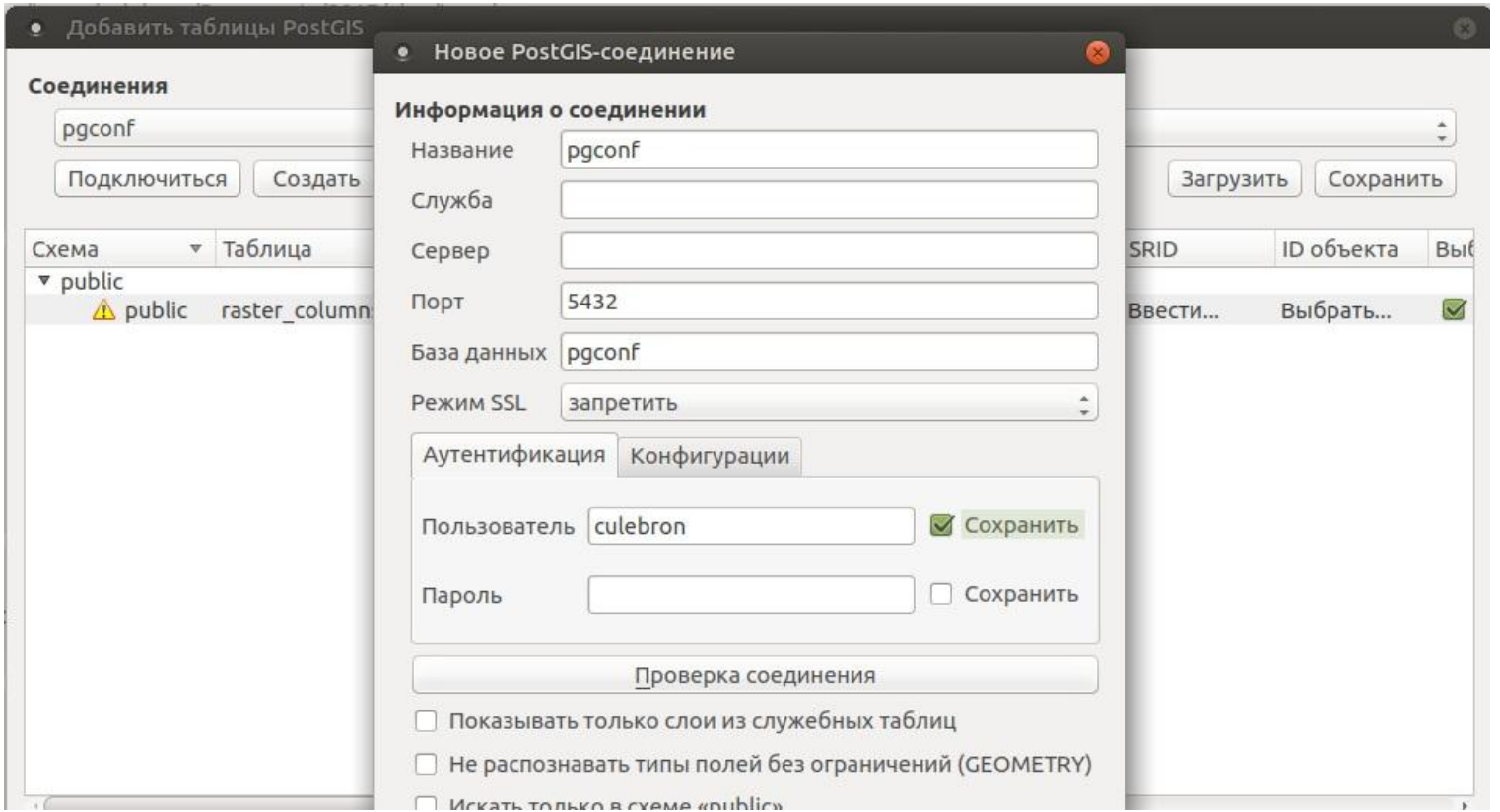
```
pg> create table houses (id serial primary key, addr text, pop
integer, geopos geometry);
```

```
pg> insert into houses (addr, pop, geopos)
select addr, pop, st_makepoint(lon, lat) from houseimport;
```

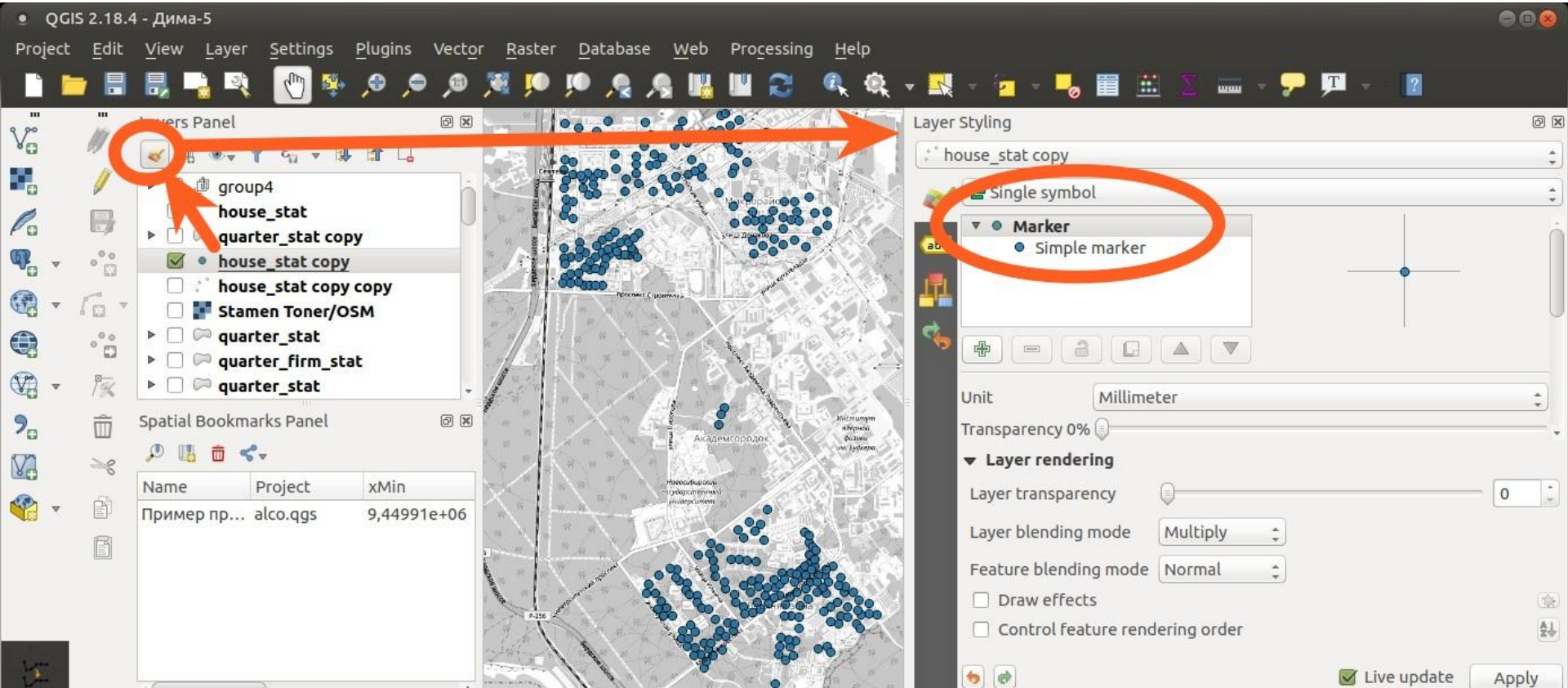
Десктопный ГИС абсолютно необходим

- Быстрый ручной ввод
- Нужно быстро видеть, что у тебя за данные - открывать CSV без импортов в Postgres
- Данные нужно быстро фильтровать и проверять
- Нужно смотреть много разных видов на одни и те же данные

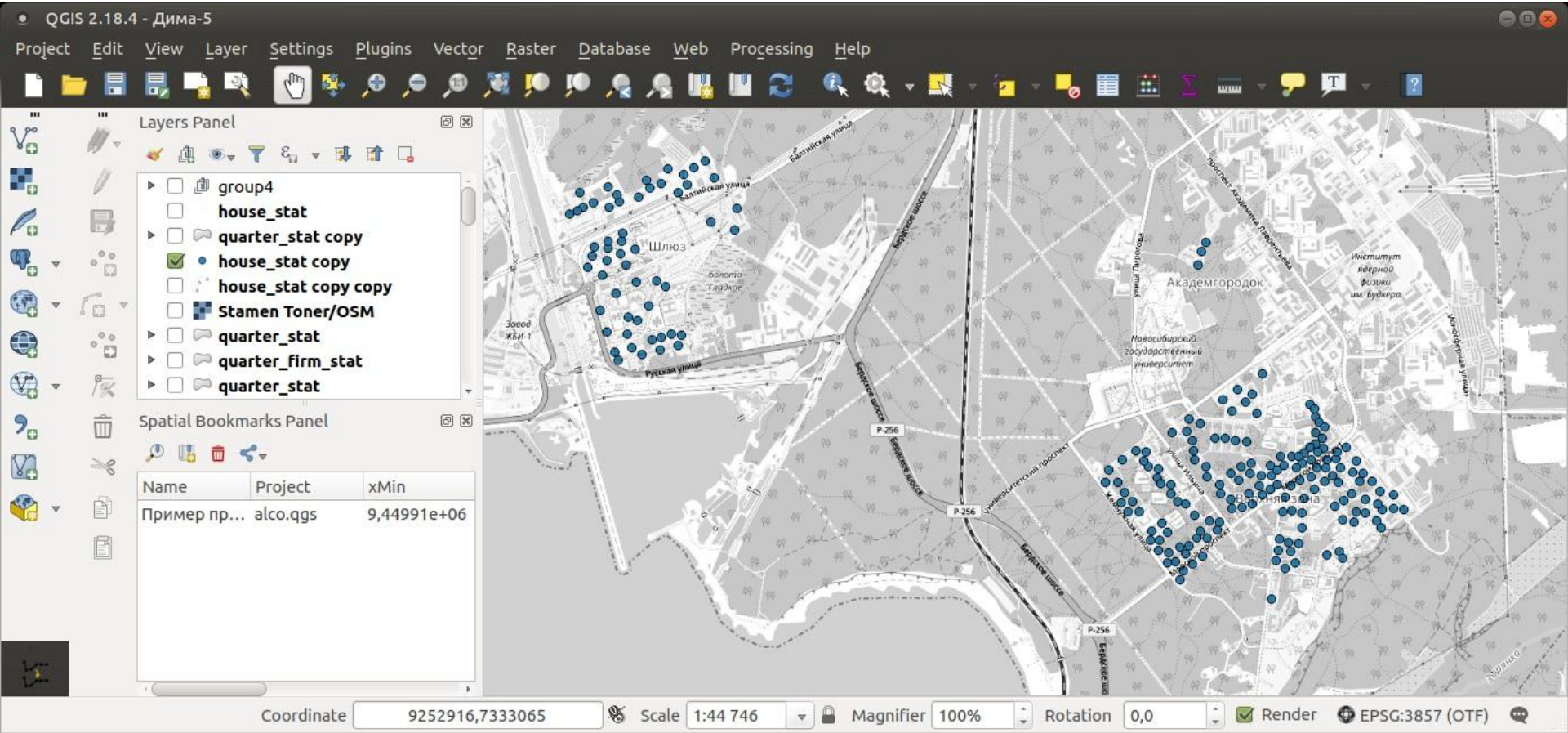
Визуализация в QGIS

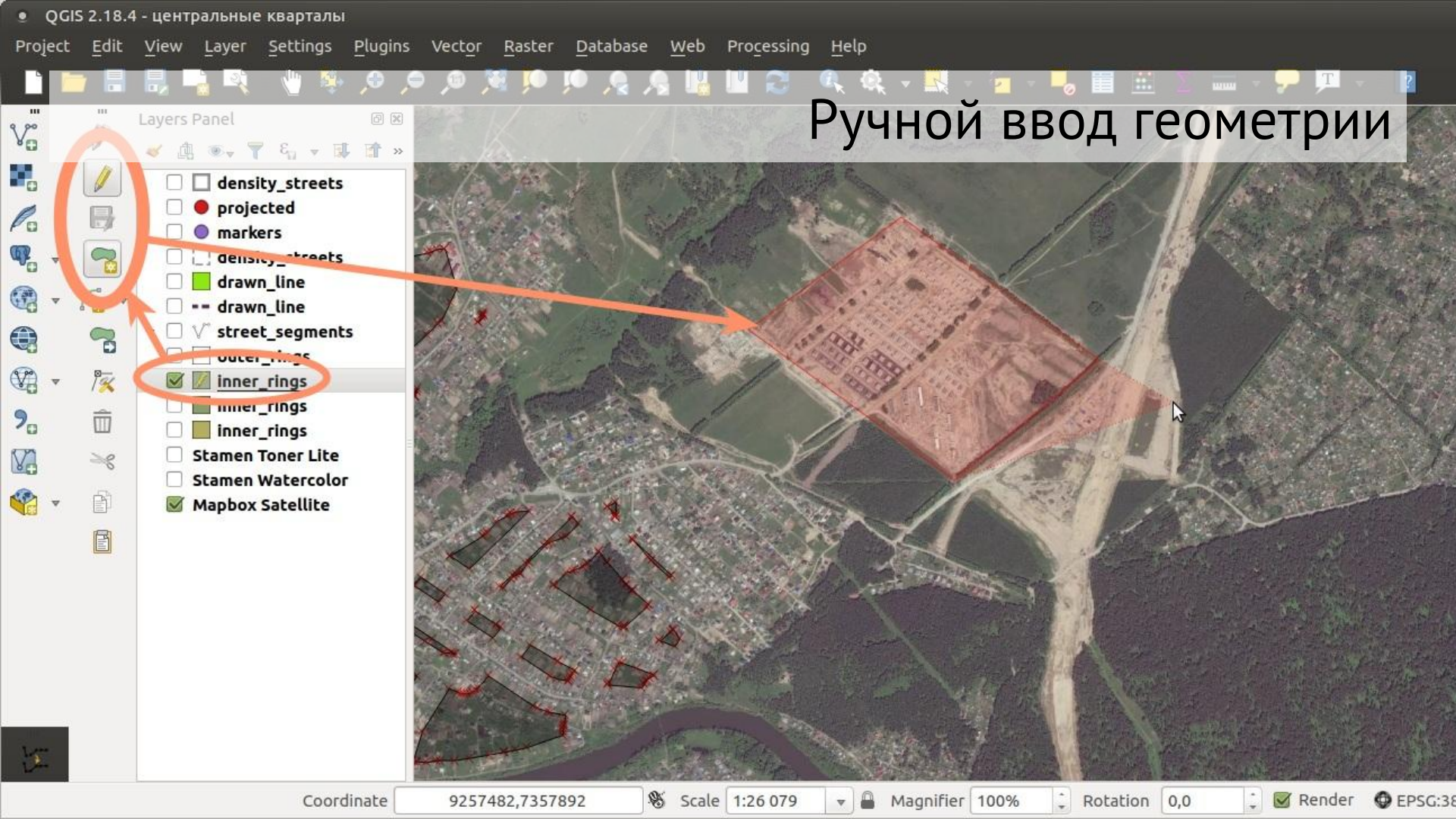


Визуализация в QGIS

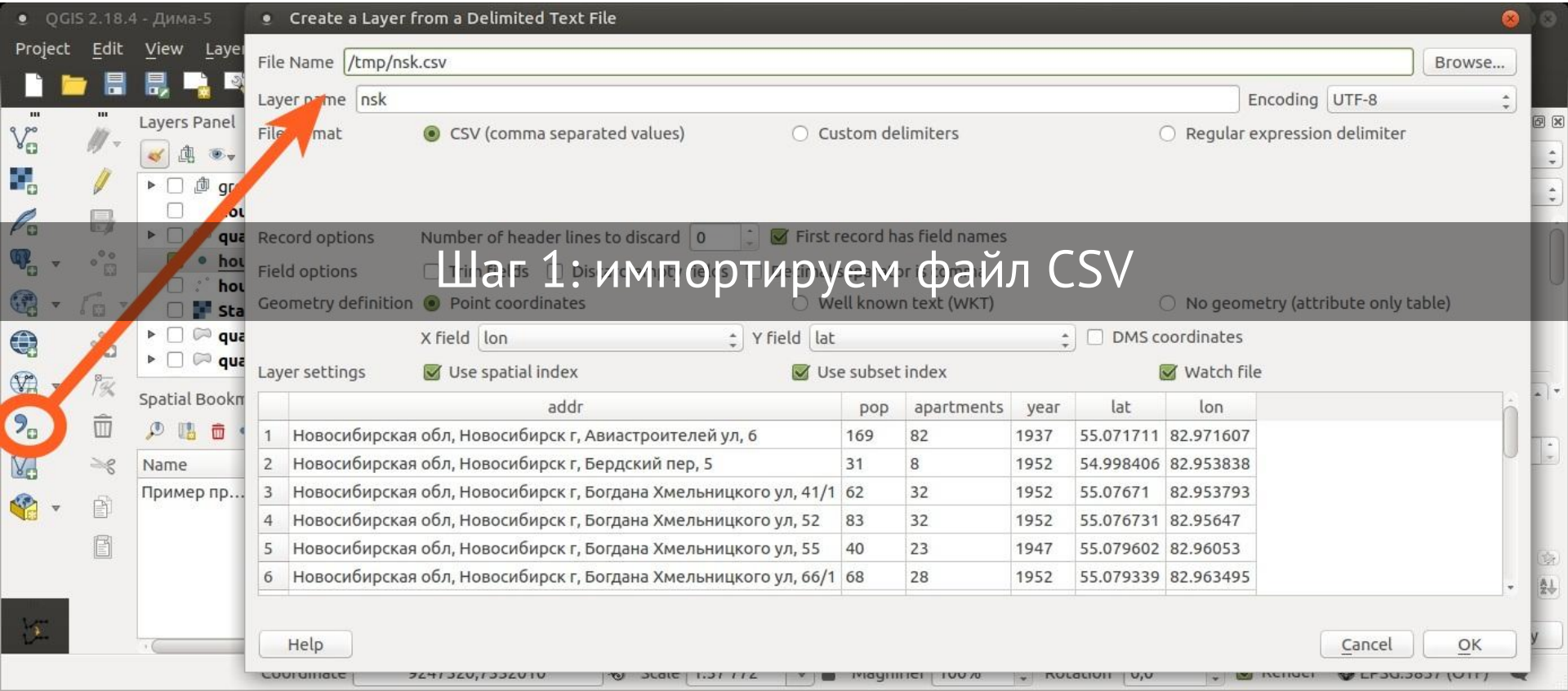


Визуализация: QGIS



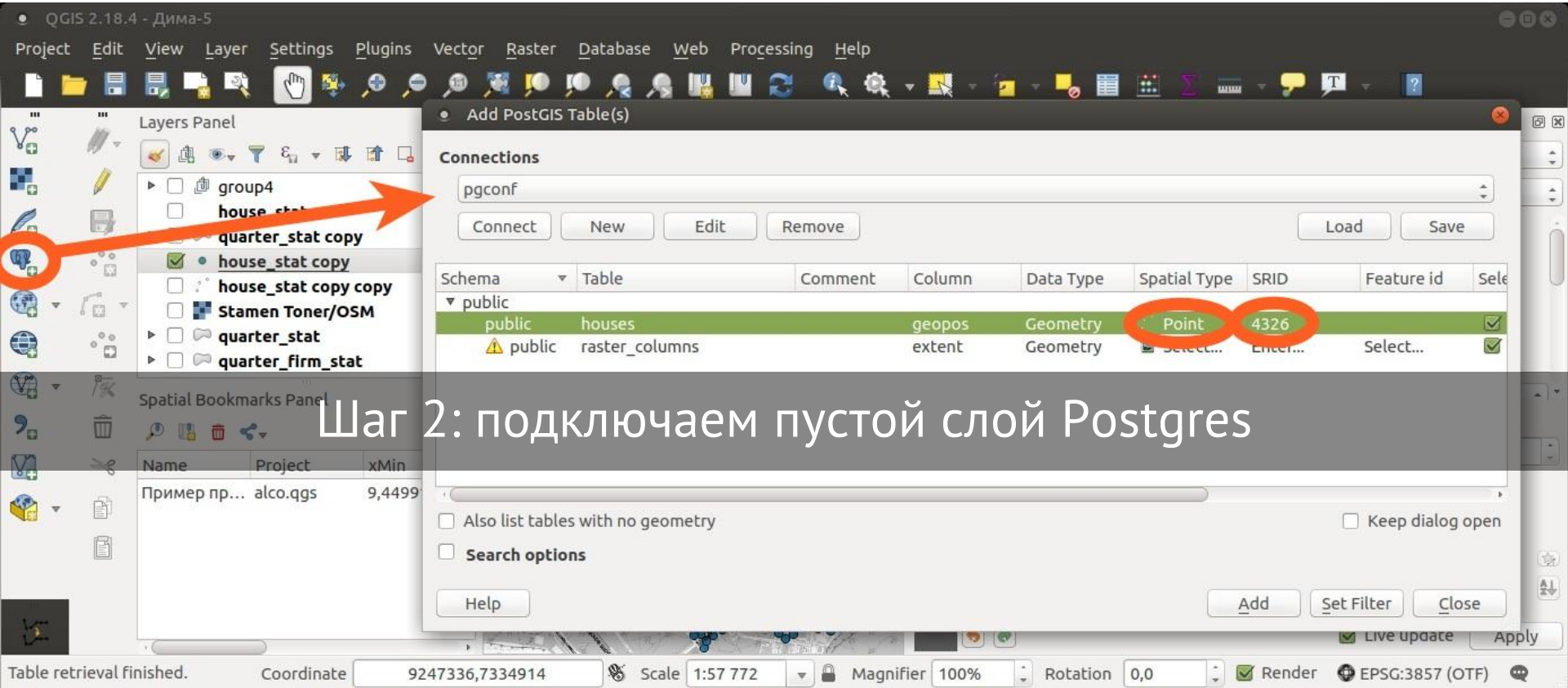

QGIS 2.18.4 - центральные кварталы
Project Edit View Layer Settings Plugins Vector Raster Database Web Processing Help
Layers Panel
density_streets
projected
markers
drawn_line
drawn_line
street_segments
inner_rings
inner_rings
Stamen Toner Lite
Stamen Watercolor
Mapbox Satellite
Ручной ввод геометрии
Coordinate 9257482,7357892
Scale 1:26 079
Magnifier 100%
Rotation 0,0
Render
EPSG:38

Лайфхак: импорт в PostGIS через QGIS

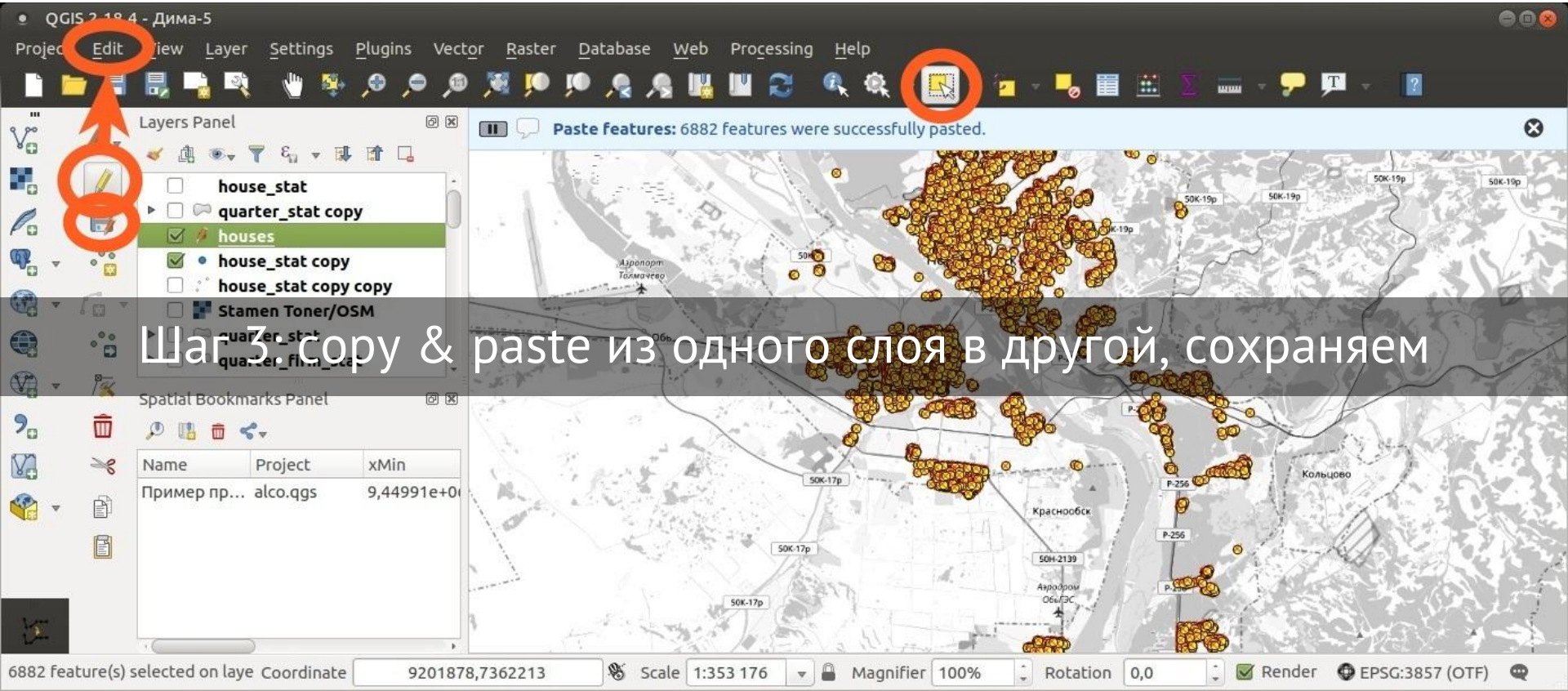


	addr	pop	apartments	year	lat	lon
1	Новосибирская обл, Новосибирск г, Авиастроителей ул, 6	169	82	1937	55.071711	82.971607
2	Новосибирская обл, Новосибирск г, Бердский пер, 5	31	8	1952	54.998406	82.953838
3	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 41/1	62	32	1952	55.07671	82.953793
4	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 52	83	32	1952	55.076731	82.95647
5	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 55	40	23	1947	55.079602	82.96053
6	Новосибирская обл, Новосибирск г, Богдана Хмельницкого ул, 66/1	68	28	1952	55.079339	82.963495

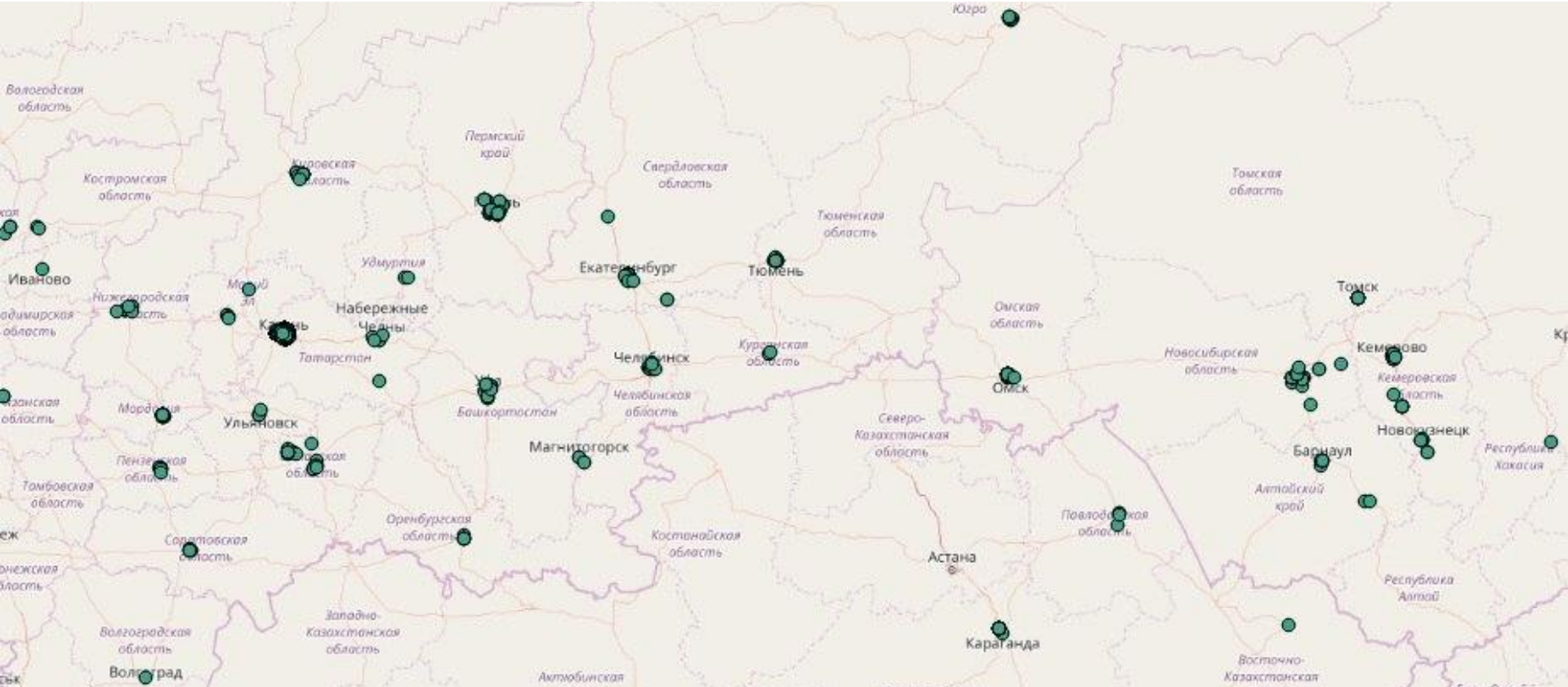
Лайфхак: импорт в PostGIS через QGIS



Лайфхак: импорт в PostGIS через QGIS



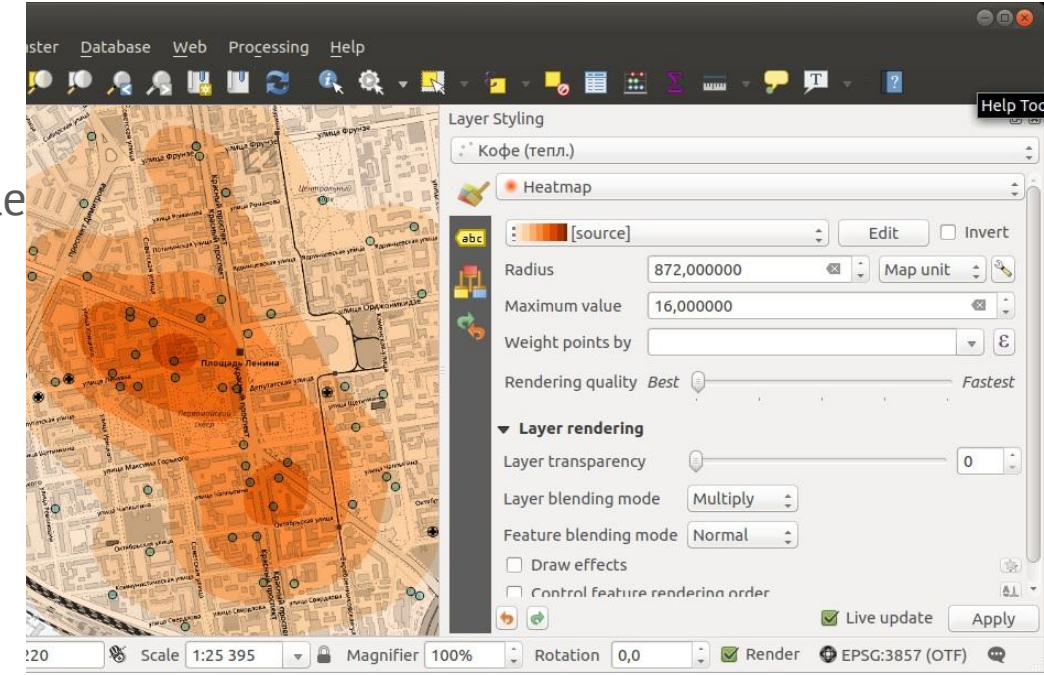
Проверка качества данных



Когда не надо PostGIS

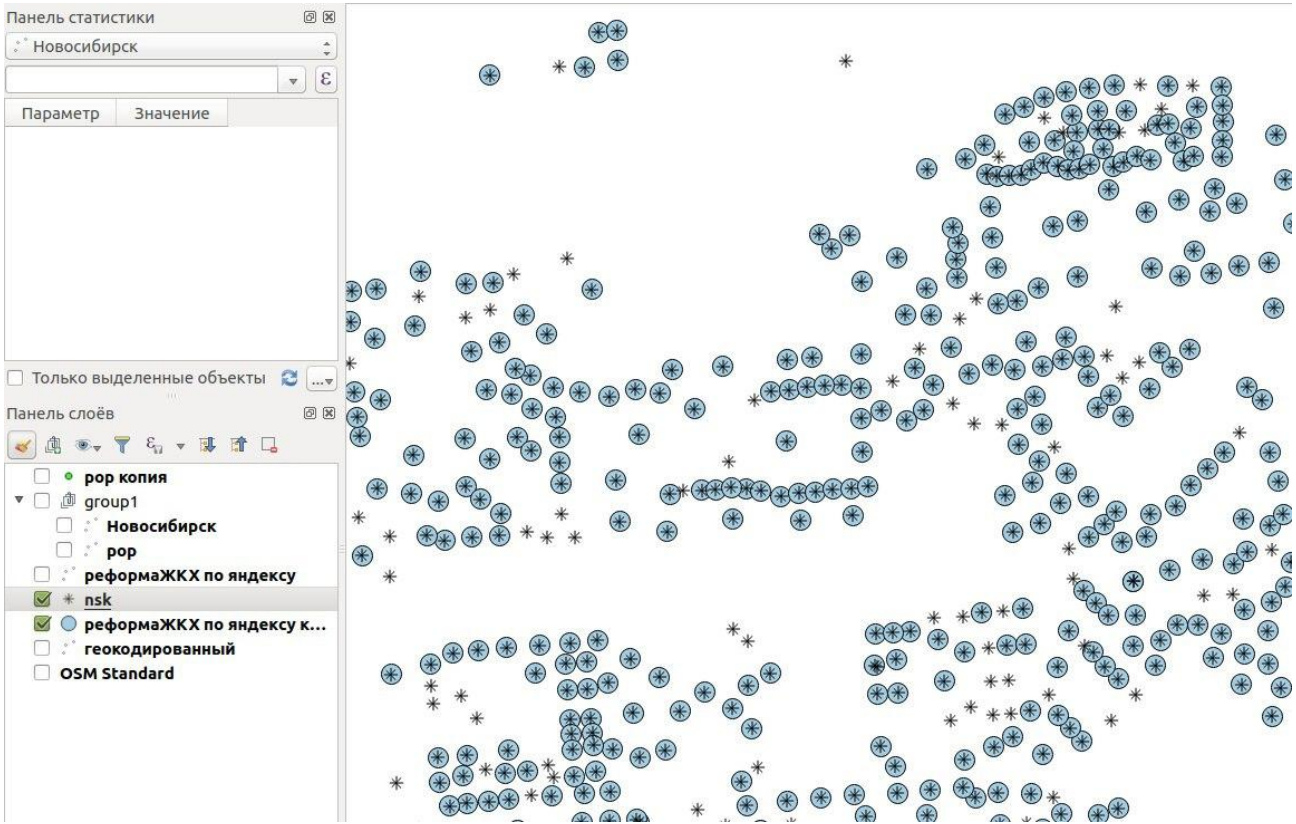
Просто проверка данных и визуализация:

- Получаем CSV, GeoJSON, Shapefile
- Открываем в QGIS
- Если GeoJSON/CSV, можно пересохранить в SQLite/GeoPackage (или по-старинке в Shapefile)
- Фильтруем, визуализируем



Когда не надо PostGIS

Сравнение качества геокодинга



Форматы импорта в PostGIS: CSV

```
pg> copy tablename from ‘my_file.csv’ csv header;
```

- широта/долгота в виде колонок -> st_makepoint
- колонка CSV в формтае Well-known-text (WKT) автоматически распознаётся при вставке в Postgres в geometry:

  POINT(82.9 55.1)
  LINESTRING(37.50387 55.80058, 37.50386 55.80055, 37.50294 55.80002)
  POLYGON(37.50387 55.80058, 37.50386 55.80055, 37.50294 55.80002, 37.50387 55.80058)

Форматы импорта в PostGIS: GeoJSON

- Библиотека OGR (`sudo apt install ogre-1.9-tools`)

```
$> ogr2ogr -f "PostgreSQL" PG:"dbname=pgconf
user=culebron" "source_data.json"
-nln houses -append
```

Форматы импорта в PostGIS: OpenStreetMap

Утилита **osm2pgsql**

Форматы файлов:

- OSM XML (.osm, .osm.bz2)
- ProtoBuffer (.pbf)

Источники данных:

- **Geofabrik**: download.geofabrik.de
- **GIS-Lab**:
  gis-lab.info/projects/osm_dump/
- **Overpass API**: overpass-api.de
- **Редактор JOSM** (через Overpass):
  josm.openstreetmap.de

Ручной ввод в PostGIS

1. **QGIS**
   + Прямое редактирование данных в таблицах
   - Создавать можно, редактировать геометрию очень неудобно
2. **JOSM**
   + Очень гибкий редактор, удобно рисовать и уточнять геометрию
   + Можно подкачивать данные OSM
   - Импорт в 7 таблиц через консольный инструмент `osm2pgsql`

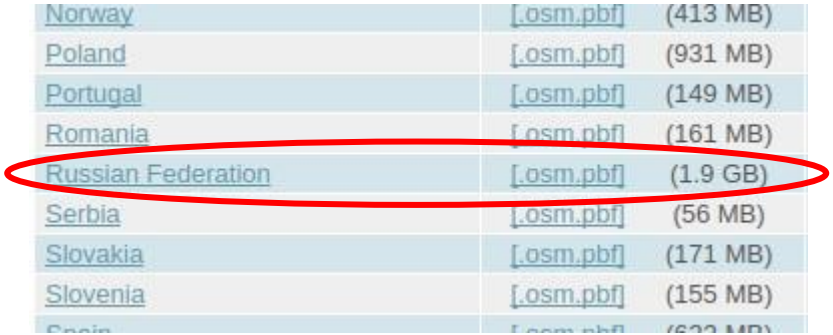
Часть 2. Тонкости

- OpenStreetMap
- Системы координат

OpenStreetMap

Проблемы, которые быстро встают при работе:

- Часто нужно много городов
- Дампы по стране целиком весят много
- Вручную выбирать неудобно
- Osm2pgsql импортирует не те тэги



Norway	[.osm.pbf]	(413 MB)
Poland	[.osm.pbf]	(931 MB)
Portugal	[.osm.pbf]	(149 MB)
Romania	[.osm.pbf]	(161 MB)
Russian Federation	[.osm.pbf]	(1.9 GB)
Serbia	[.osm.pbf]	(56 MB)
Slovakia	[.osm.pbf]	(171 MB)
Slovenia	[.osm.pbf]	(155 MB)
Spain	[.osm.pbf]	(633 MB)

OpenStreetMap

Вариант “дёшево и сердито”:

- Скачать дампы нужных регионов
- Каждый импортировать (очищая таблицы)
  только с теми колонками, которые нужны

- Выбрать нужную геометрию в отдельную таблицу
- Повторить для другого региона

OpenStreetMap

```
$ wget http://data.gis-lab.info/osm_dump/dump/latest/RU-TY.osm.pbf


$ osm2pgsql --slim --style my_import.style -d pgconf RU-TY.osm.pbf
-p osm --multi-geometry --cache-strategy sparse


$ psql -d pgconf
pg# insert into highways select * from osm_polygon where highway
is not null;
```

OpenStreetMap

My_import.style: запишем в базу только нужные колонки

```
way,node        name        text    linear
way             highway     text    linear
way             railway     text    linear
way,relation    rural       text    linear,polygon
way             service     text    linear
way             waterway    text    linear,polygon
```

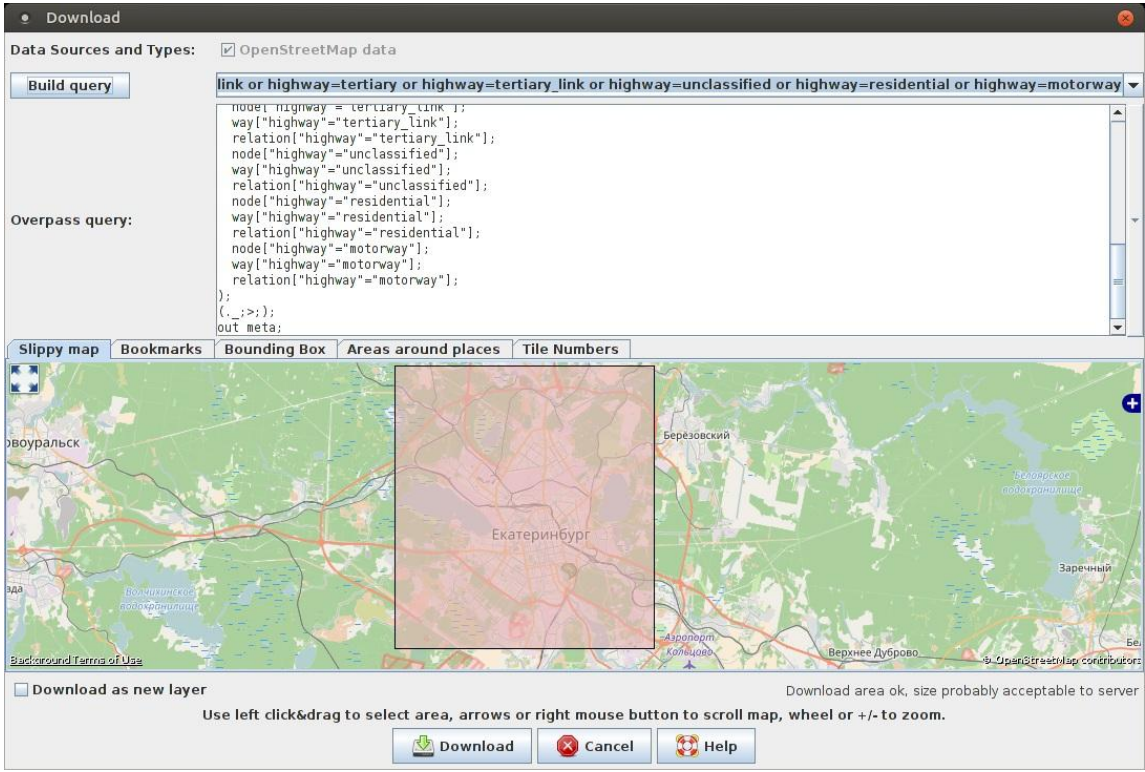
OpenStreetMap

Выбираем только элементы с нужными тэгами

Overpass API http://overpass-api.de/

В JOSM можно сделать вручную.

Сохранённый файл импортировать в osm2pgsql.



OpenStreetMap

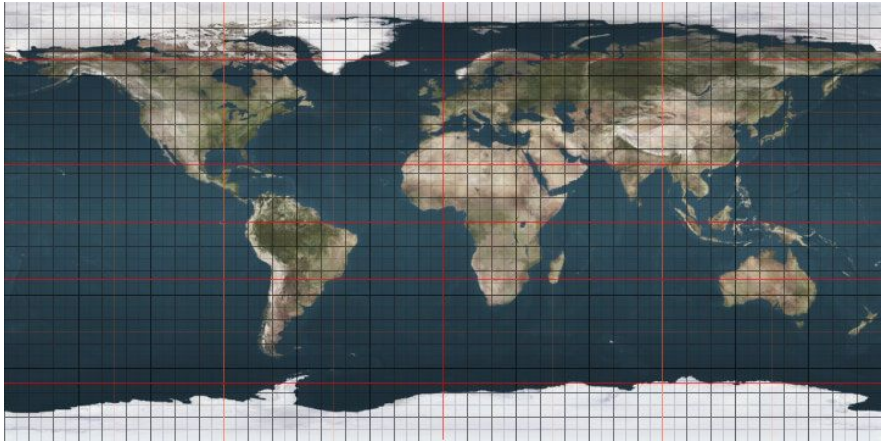
Есть скрипты для автоматического импорта из Overpass (но осторожно, сервер может вас заблокировать, а частыми запросами его можно уронить)

Запрос из JOSM вставляем в скрипт, выполняем в Python Requests, ответ пишем в bz2.

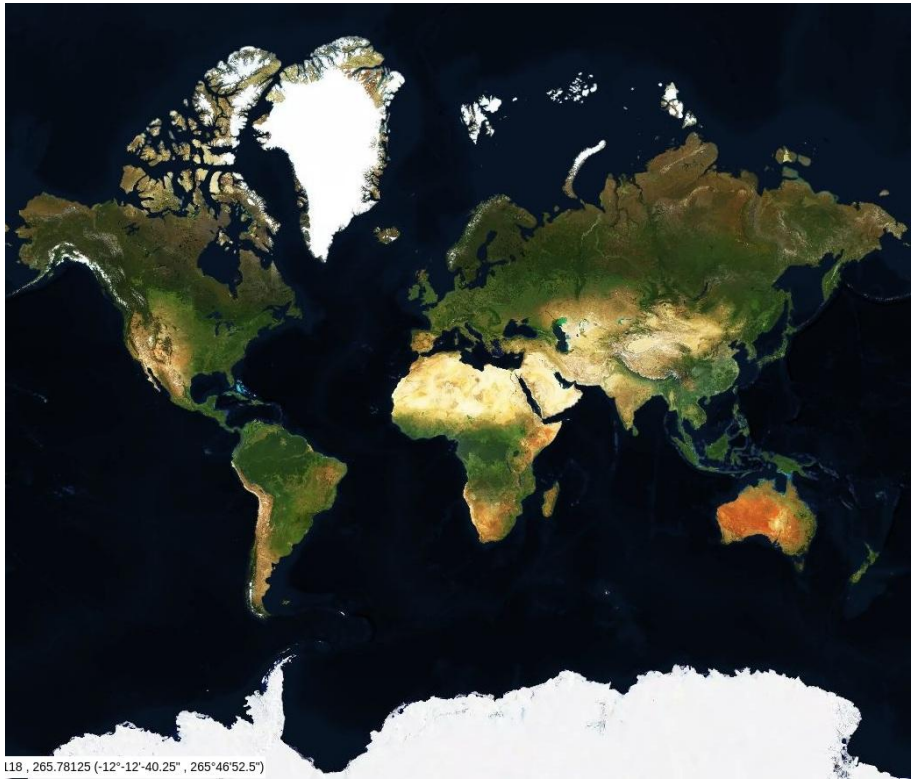
https://gist.github.com/culebron/9f1d98507919b5a0f9109688046a505d

Системы координат

Lon/Lat
EPSG **4326**



Web/Google Mercator
EPSG **3857** aka **900913**



Системы координат

```
# select st_astext(
    st_transform('SRID=4326;POINT(83 55)'::geometry, 3857));

# select st_astext(
    st_transform(
        st_setsrid('POINT(83 55)'::geometry, 4326), 3857));

"POINT(9239517.73584171 7361866.11305119)"
```

Системы координат

- Широта-долгота
  - Удобна: общепринятые географические координаты, читаема (можно помнить)
  - Неудобна: нельзя считать геометрию как в декартовых координатах, на карте всё искажённое
- Web/Google Mercator
  - Удобна: можно считать геометрию (буфер, расстояние и т.д.), округлять и равнять по сетке локально (в пределах, например, города)
  - Неудобно: размер нужно всё время масштабировать под широту объекта, нечитаема. (Но по большим числам координат видно, что это данная проекция)
- PostGIS требует одинаковых систем отчёта, не конвертирует сам
- Есть тип Geography, которая считает расстояния в метрах, храня объекты как на сфере (но не работает для всех функций)

Часть 4. Примеры исследований

1. Плотность улично-дорожной сети
2. Плотность населения и частные дома
3. Плотность жителей по кварталам
4. Активность кварталов и фасадов
5. Транспортная доступность парков города

1. Плотность улично-дорожной сети

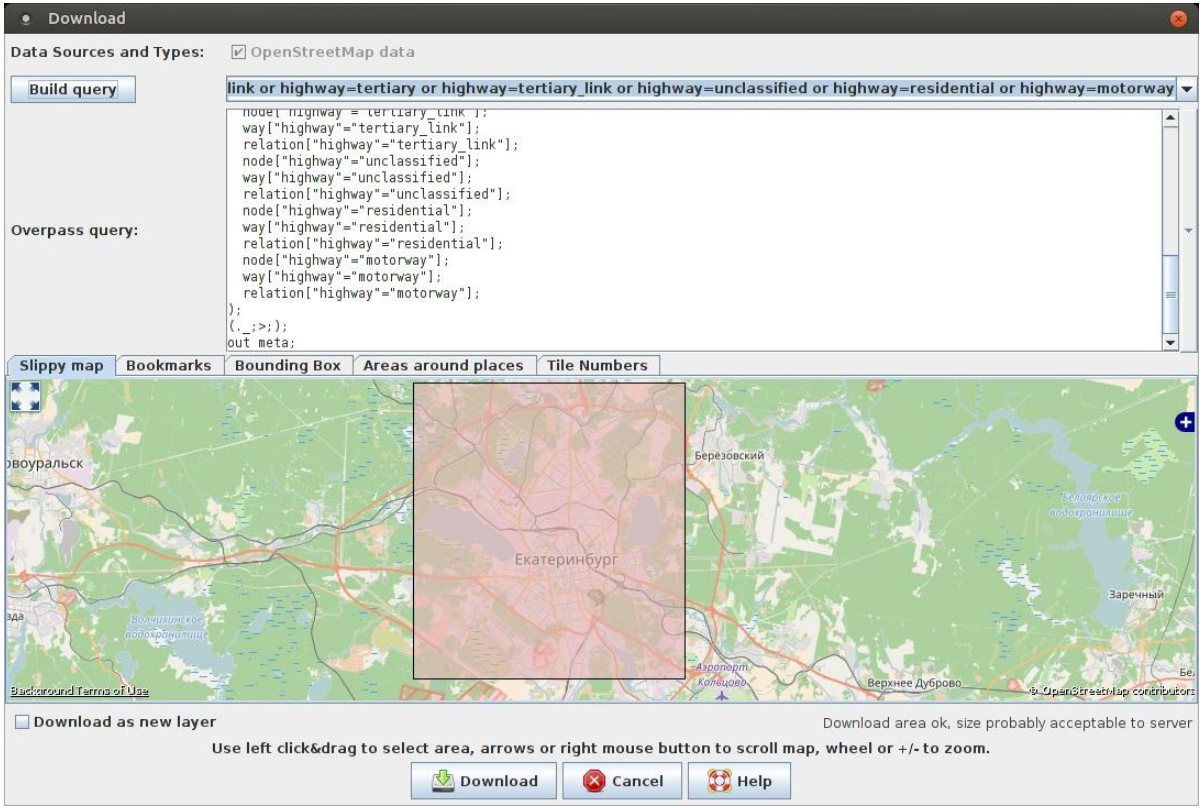
Сколько дорог на квадратный километр есть в городе?

Что считать?

1) Общий показатель по городу
2) Показатель по отдельным частям

1. Плотность УДС. Скачивание из OpenStreetMap

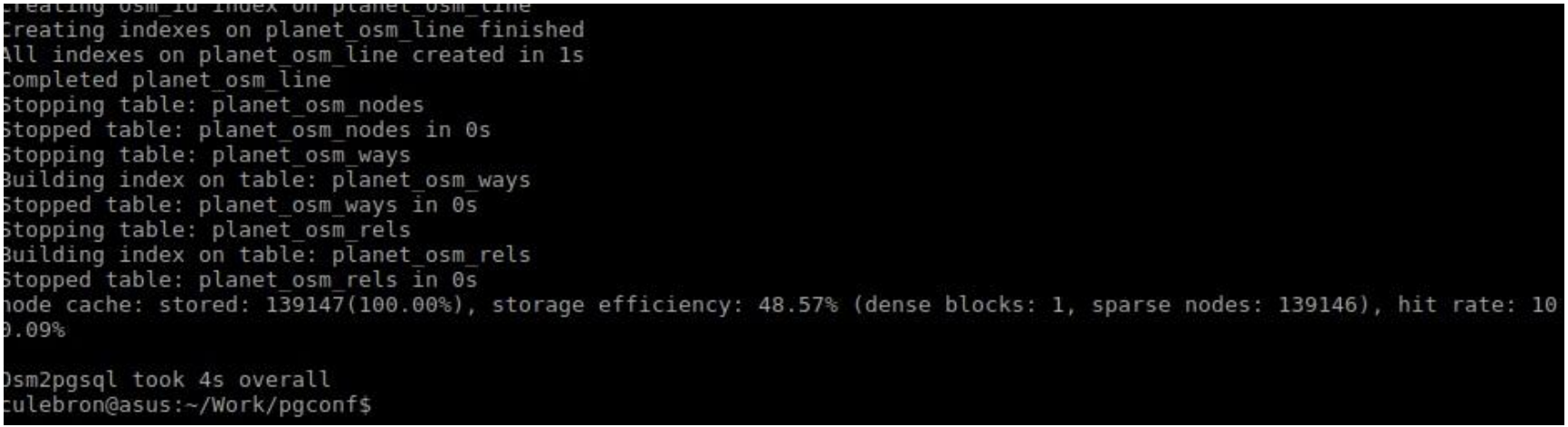
1. Скачиваем город в OpenStreetMap (плагин Overpass, скачиваем highway), сохраняем.



1. Плотность УДС. Импорт

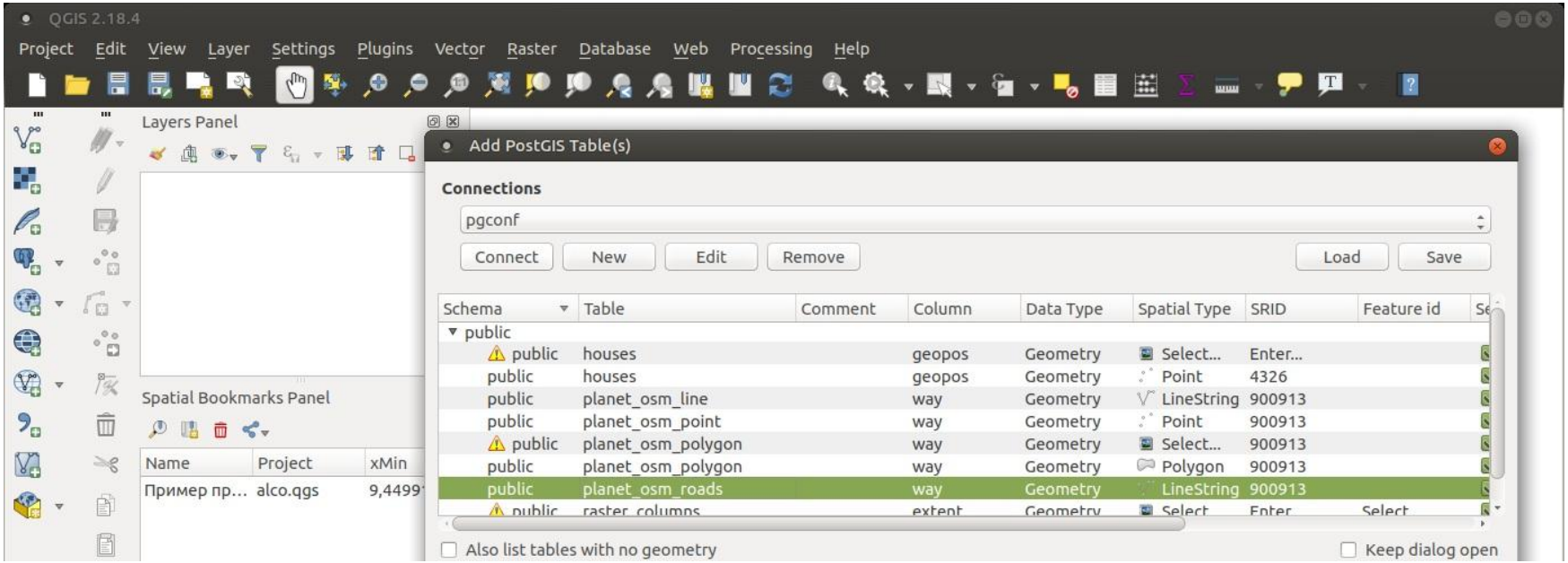
2. Импортируем в базу при помощи osm2pgsql:

```
$ osm2pgsql -d pgconf --multi-geometry -c Екатеринбург.osm.bz2 -s
```



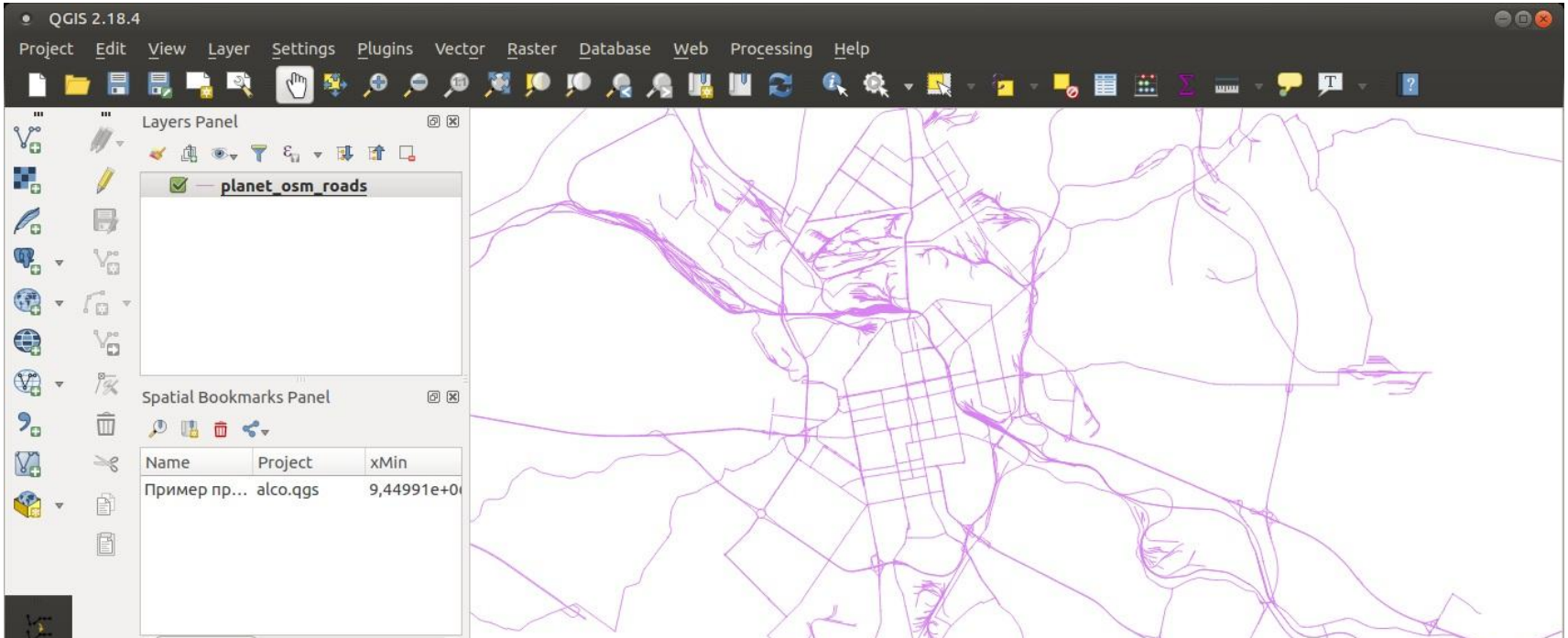
1. Плотность УДС. Визуализация

2. Проверяем данные в QGIS

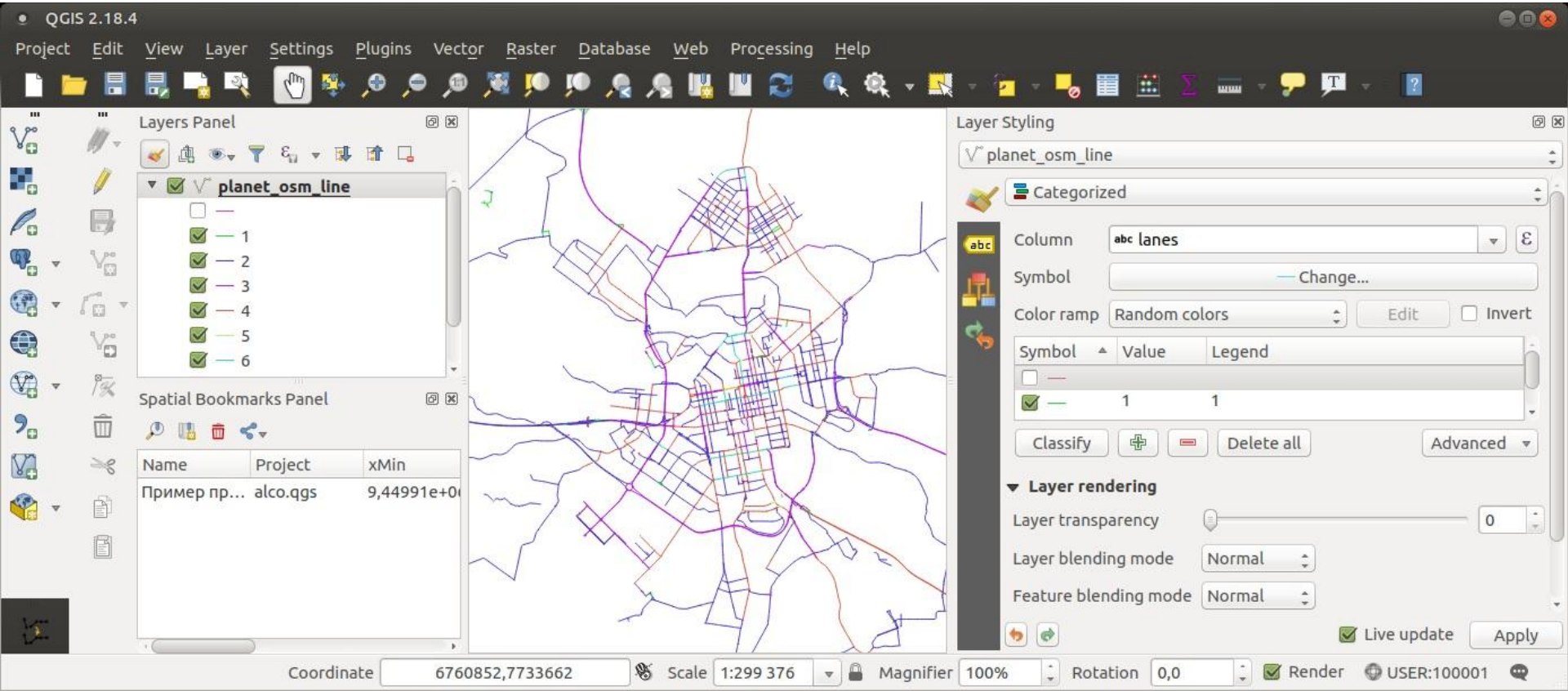


1. Плотность УДС. Проверка

2. Проверяем данные в QGIS. Похоже на правду.



1. Плотность УДС. Проверка, где есть данные



1. Плотность УДС. Рассчёты

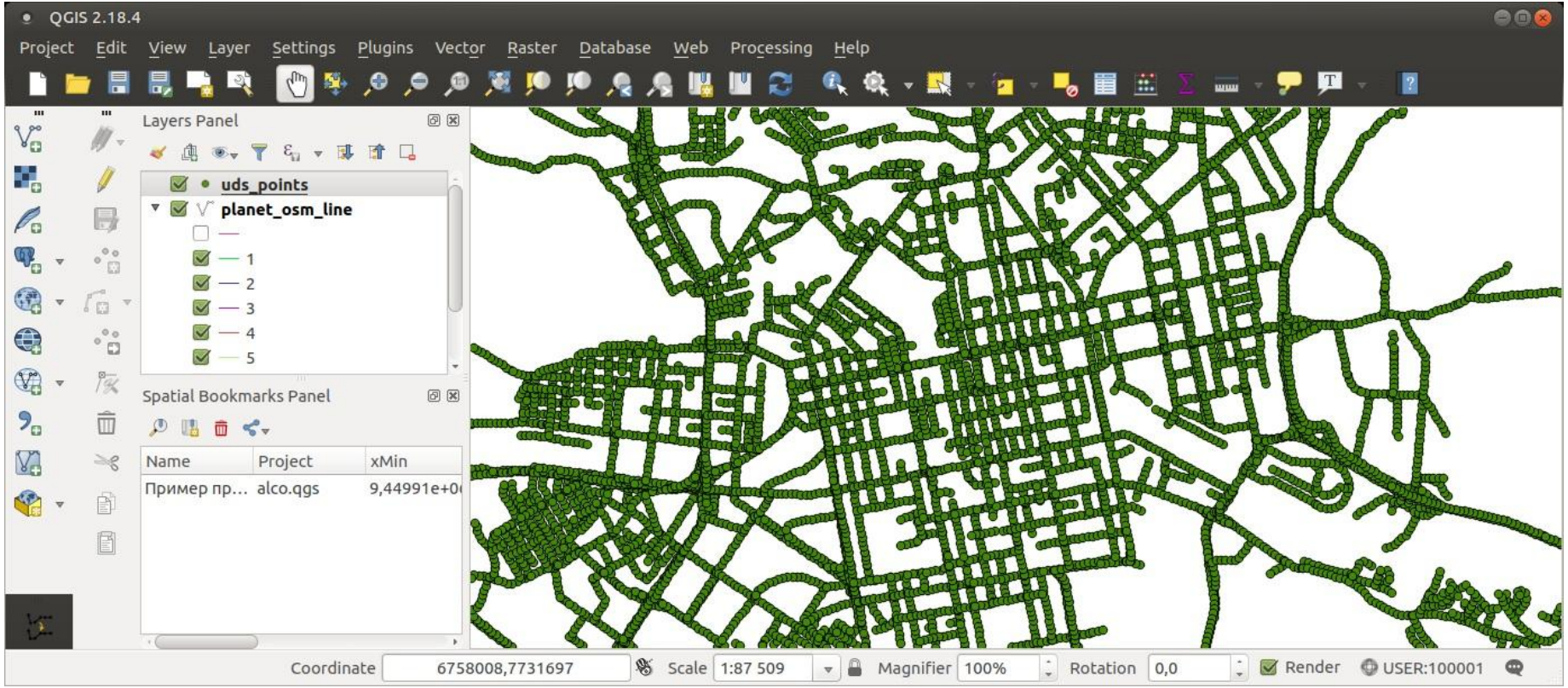
Стометровые вешки:

```
select osm_id, way,
    generate_series(0, st_length(way)::integer, 100) mark
from planet_osm_line
where
    highway in ('motorway', 'trunk', 'trunk_link', 'primary',
'primary_link', 'secondary', 'secondary_link', 'tertiary',
'unclassified', 'residential');
```

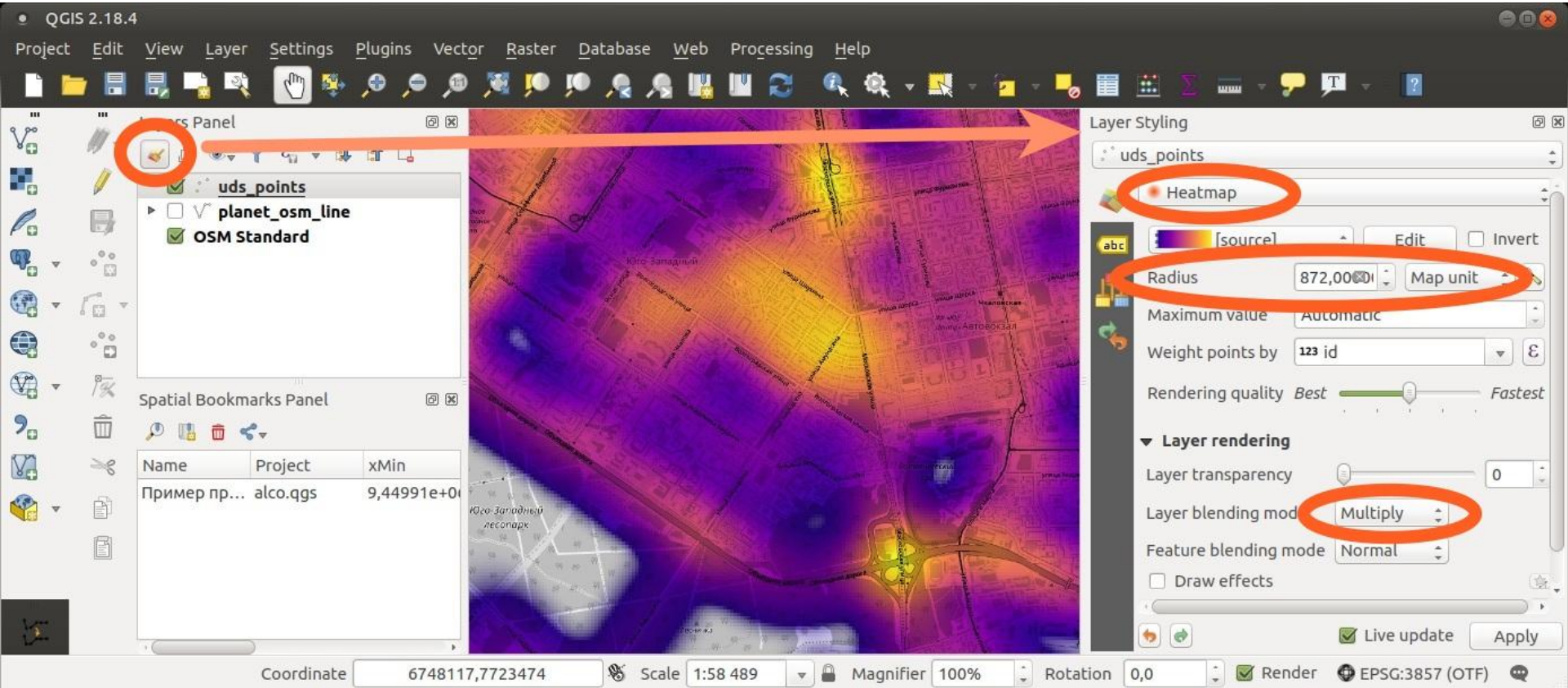
1. Плотность УДС. Рассчёты

```
create table uds_points as

with marks as (...) /* предыдущий запрос */

select row_number() over () id,
    st_locatealong(
        st_addmeasure(way, 0, st_length(way)),
        mark) point
from marks;
```

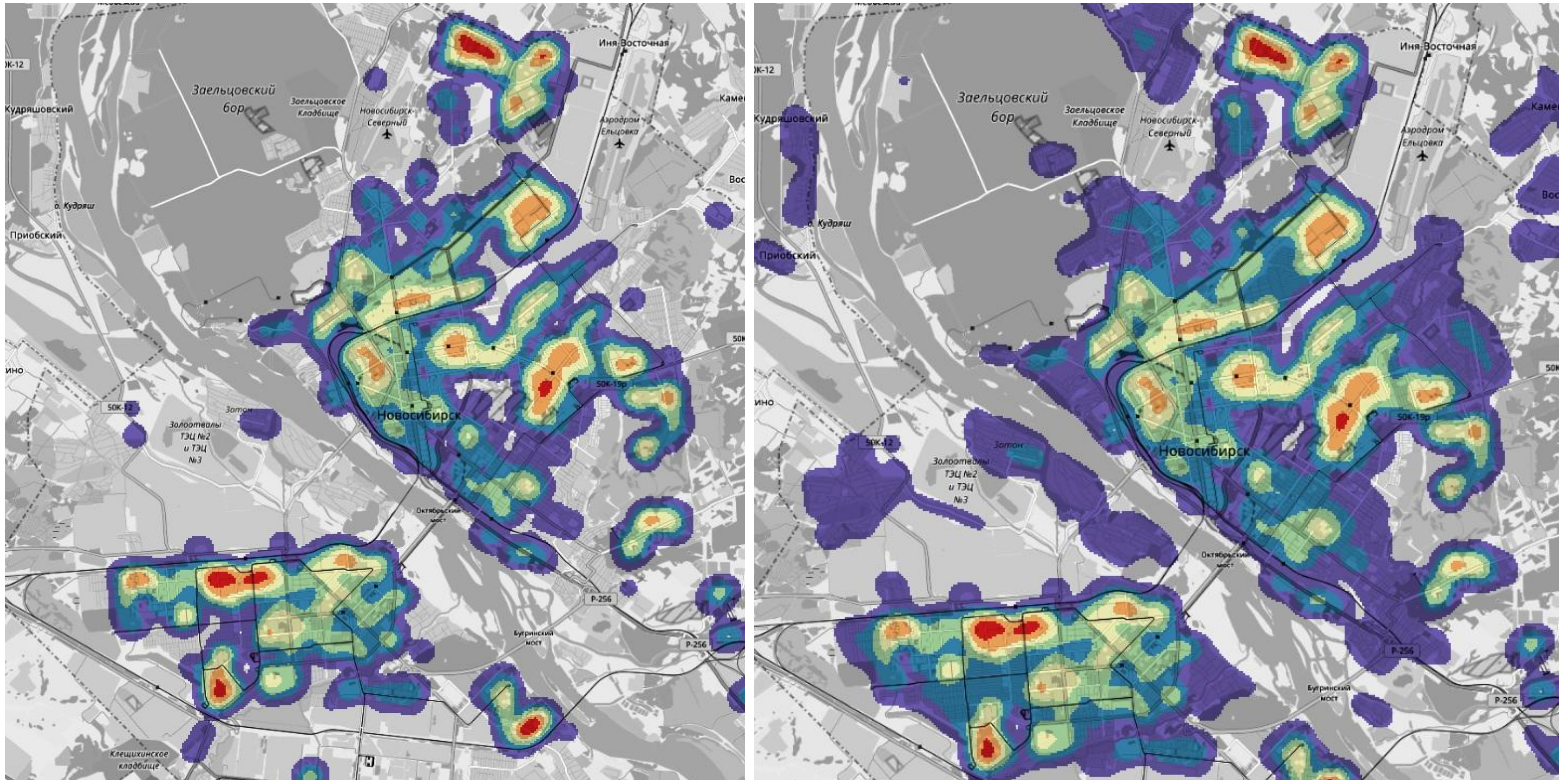
1. Плотность УДС. Визуализация



1. Плотность УДС. Результат



2. Плотность населения и частные дома



2. Плотность населения. Визуализация МКД

2. Плотность населения. Обрисовка ИЖС

2. Плотность населения. Интерполяция

Дороги:

```
select way from quarter_line
where highway is not null and
      highway not in ('pedestrian', 'footway', 'path', 'steps', 'track', 'service');
```

Линии вдоль дорог:

```
with roads as (...)
select (st_dump(st_boundary(st_union(st_buffer(way, 30.0))))).geom way
from roads;
```

2. Плотность населения. Интерполяция

Обрезка линий домов по полигонам ИЖС:

```
select b.osm_id, st_intersection(b.way, r.way) way
from houses_road_buff b, rural_polygon r
where st_intersects(b.way, r.way)) and r.residential='rural';
```

Расстановка точек с шагом (как в примере с УДС):

```
select osm_id, st_addmeasure(way, 0, st_length(way)) way,
st_length(way) len, generate_series(1, (st_length(way)/45)::integer-1) i
from houses_line_cropped;
```

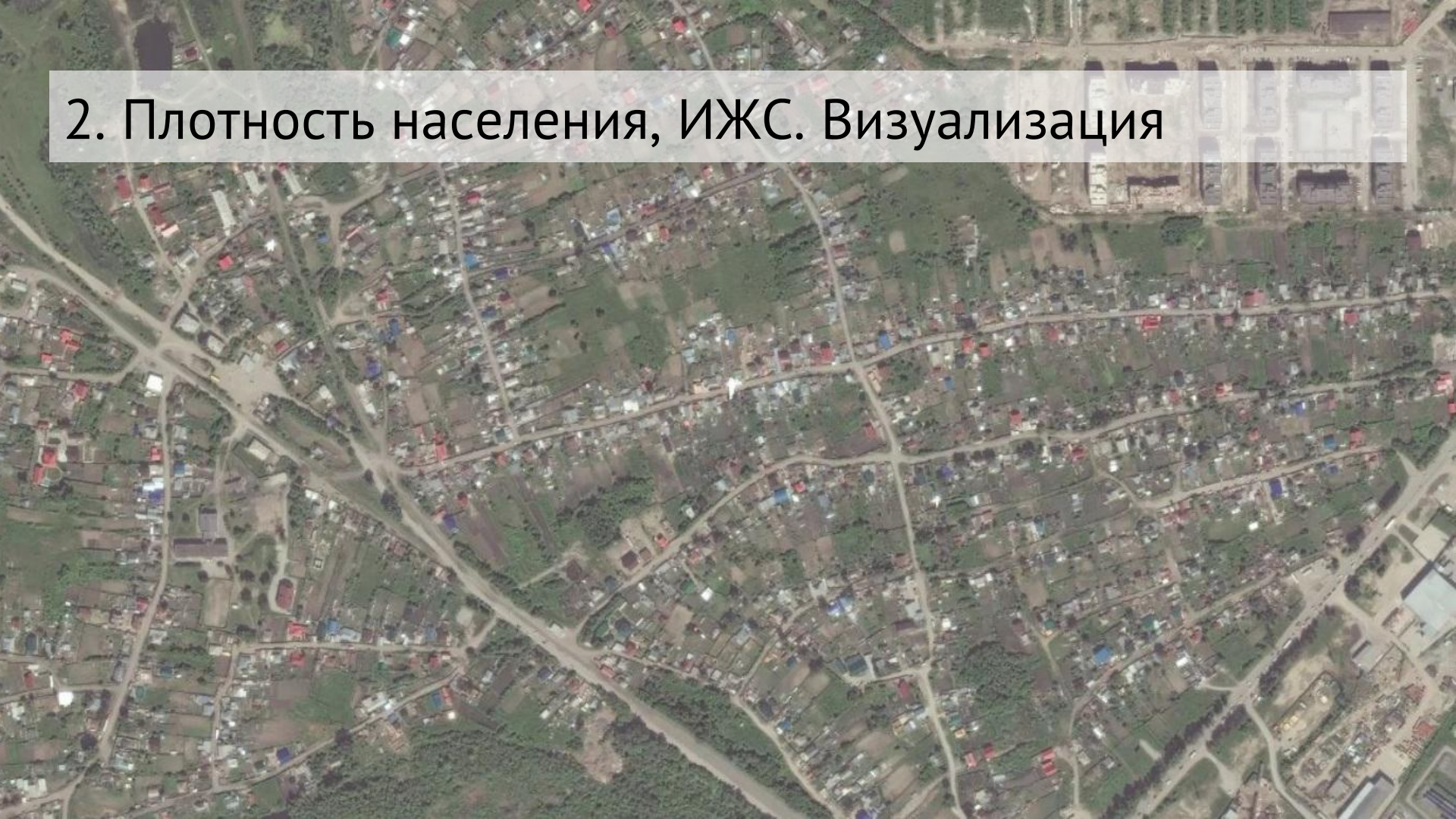
2. Плотность населения. Интерполяция

Сборка в одной таблице:

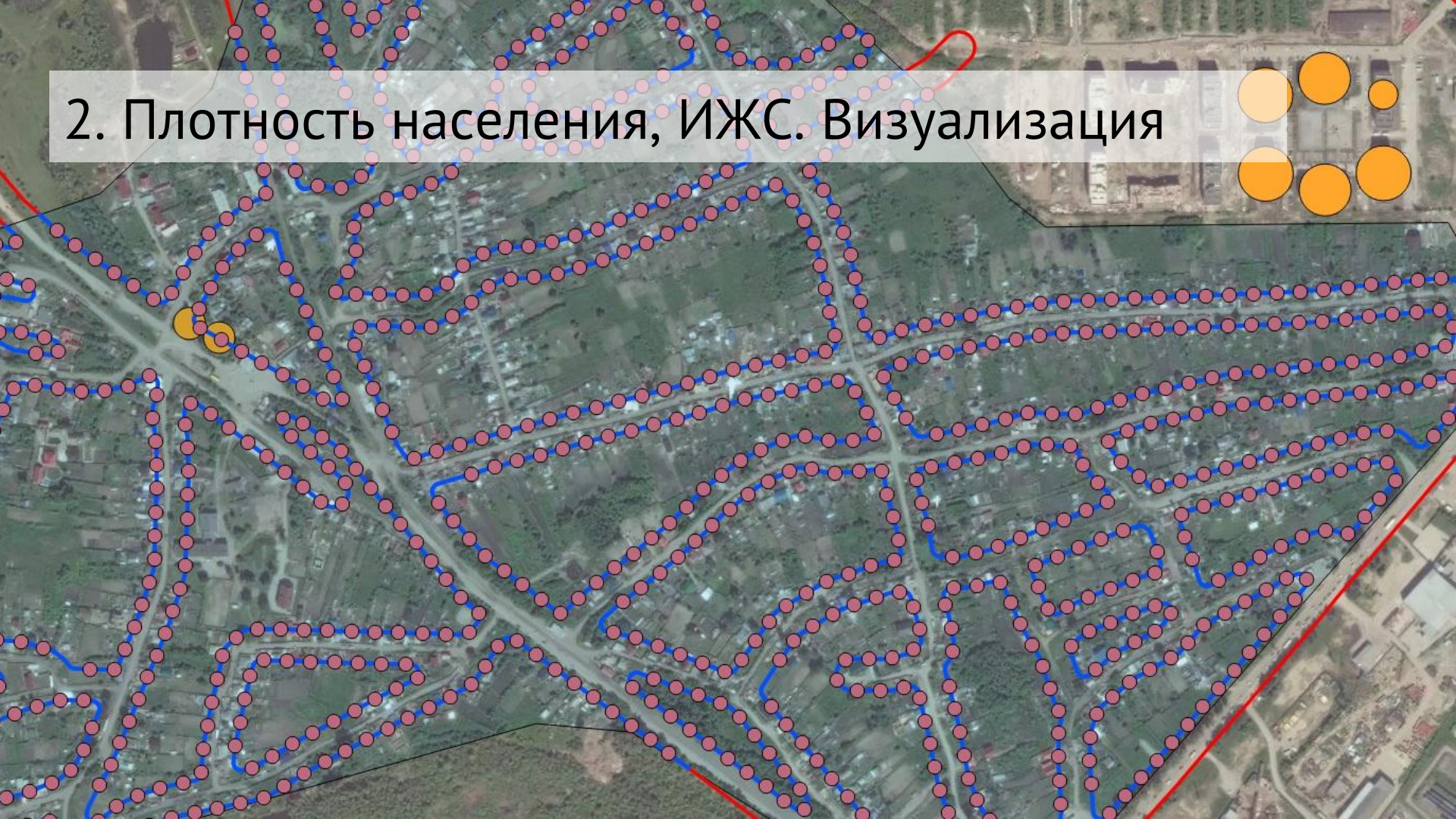
```
select row_number() over () id, st_locatealong(way, i*45) point
from houses_series;
```

Всё.

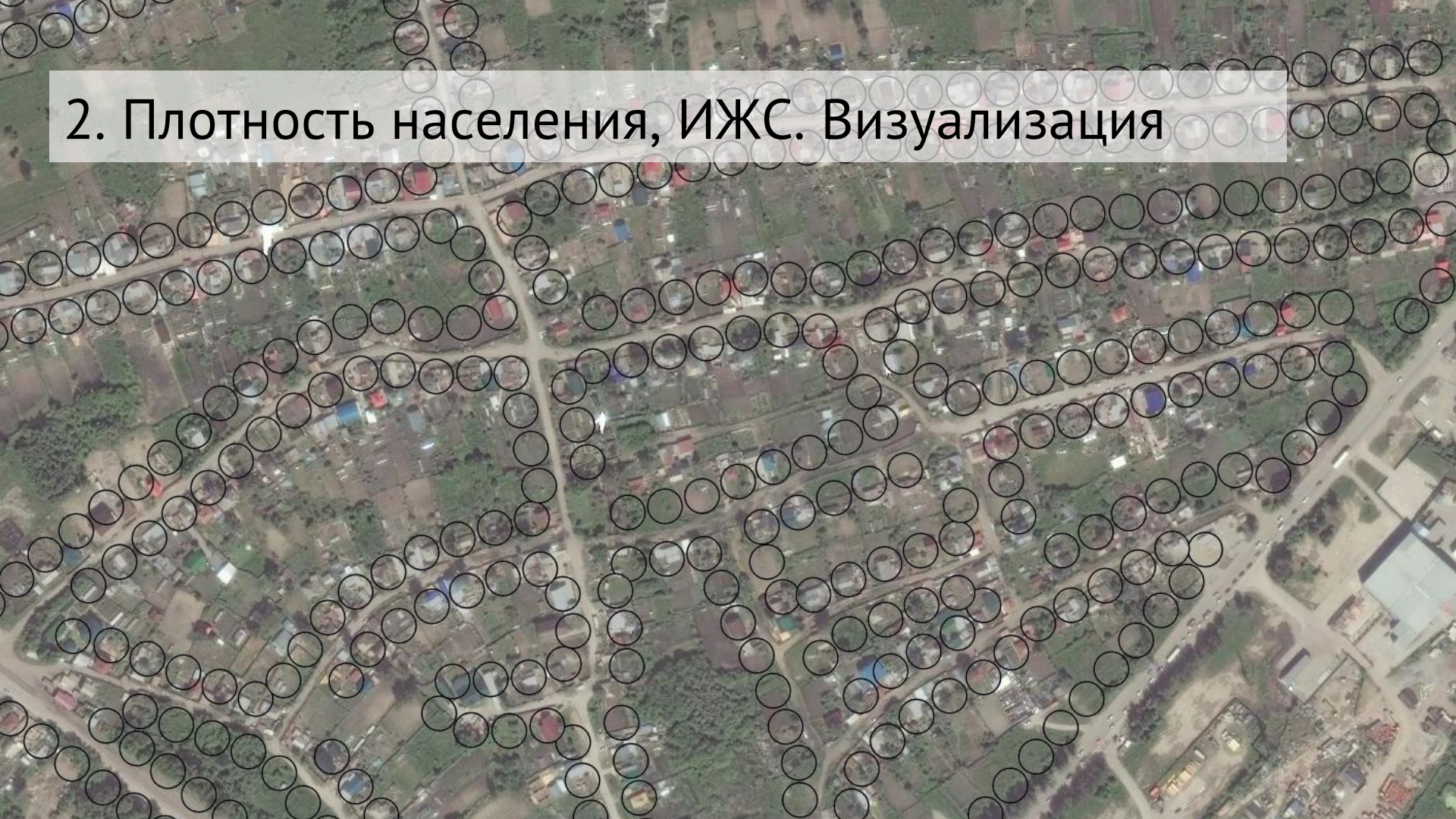
2. Плотность населения, ИЖС. Визуализация



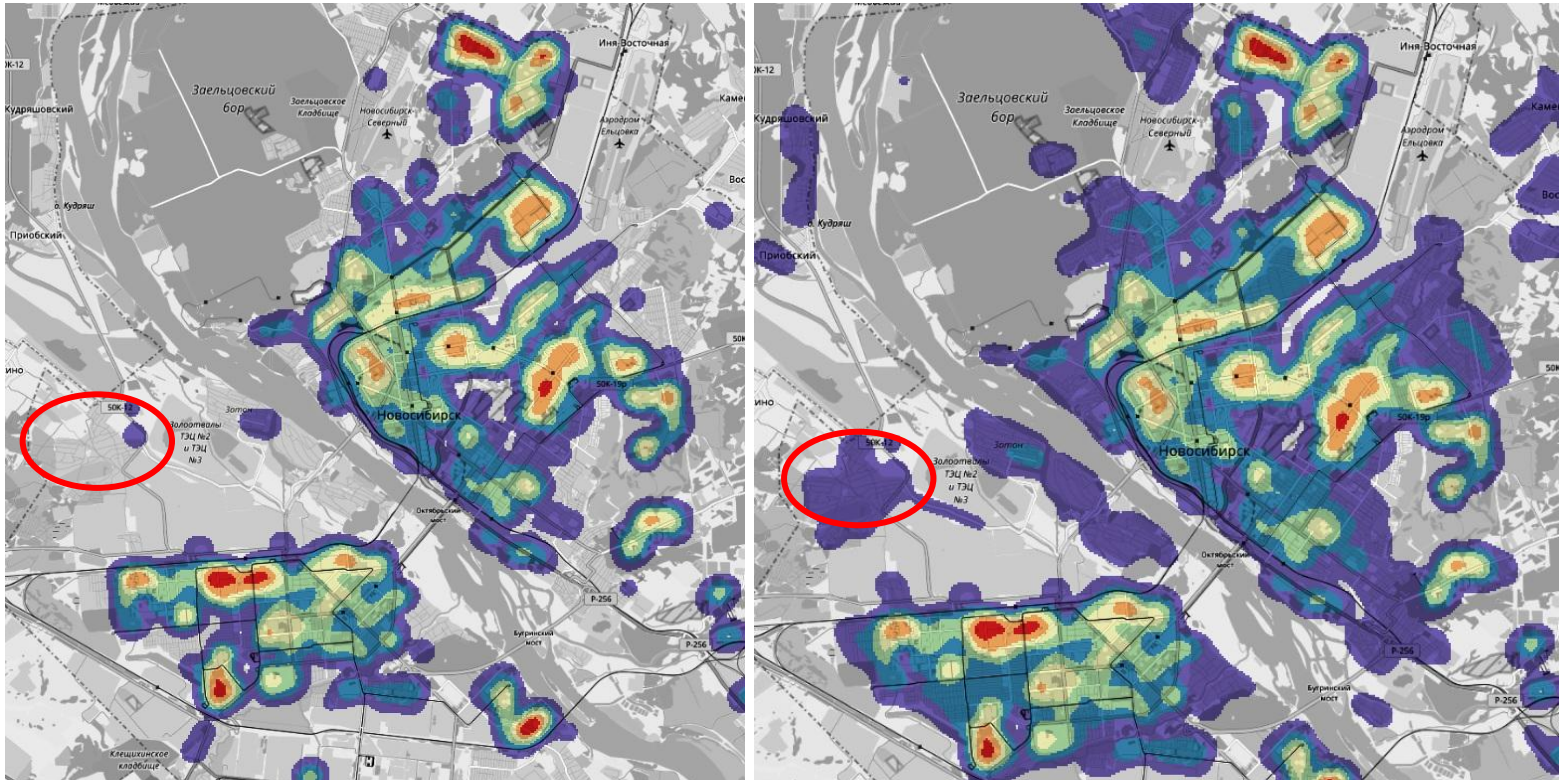
2. Плотность населения, ИЖС. Визуализация



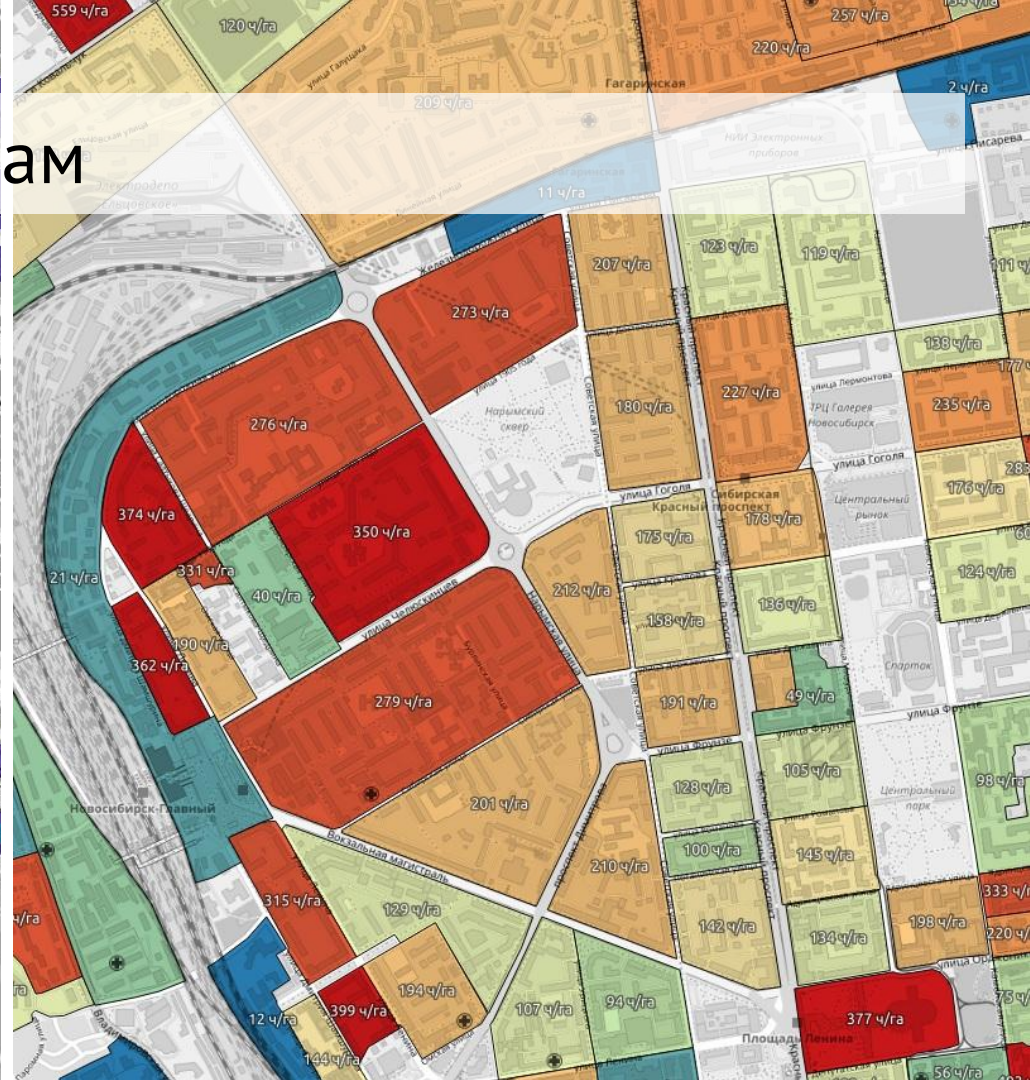
2. Плотность населения, ИЖС. Визуализация



2. Плотность населения, результат



3. Статистика по кварталам

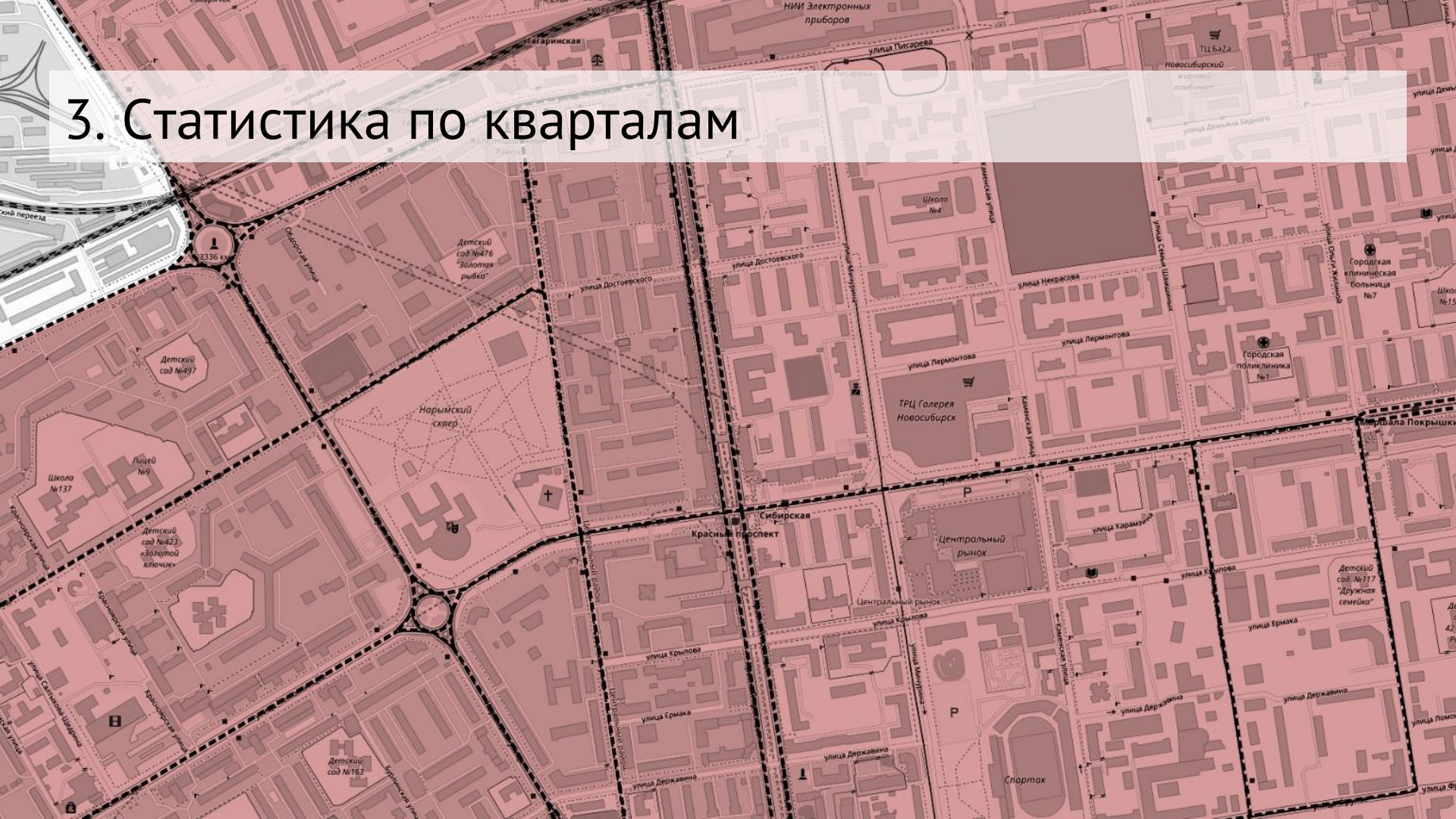


3. Статистика по кварталам. Нарезка

В примере 1 мы импортировали OSM с дорогами. Нарежем из них кварталы.

```
create table blocks (id serial primary key, way geometry);

insert into blocks (way)
select (st_dump(st_polygonize(st_union(way)))).geom way
from planet_osm_line
where highway in ('primary', 'trunk', 'primary_link',
    'trunk_link', 'secondary', 'secondary_link',
    'tertiary', 'unclassified', 'residential') or
    (railway='rail' and service is null) or
    waterway is not null;
```

3. Статистика по кварталам



3. Статистика по кварталам. Привязка значений

```
create table block_population as
select b.id, sum(pop), way
from blocks b, houses p
where st_within(point, b.way)
group by b.id, way;
```

3. Статистика по кварталам. Визуализация

4. Активность фасадов

Не активный квартал

Квартал с активным фасадом

Фото motors-nsk.ru и автора

4. Активность фасадов. Исходные данные

4. Активность фасадов. Обработка данных

Берём нарезанные кварталы и фильтруем их: уберём узкие (газоны между проезжими частям и клумбы кольцевых развязок)

```
create table nonempty_blocks as
select id, way
from blocks
where not st_isempty(st_buffer(way, -25));
```

4. Активность фасадов. Выборка

Сделаем “колбасу” по границе квартала:

```
create table blocks_selections as
with polys as (
    select id, st_buffer(poly, -30) inner_poly,
           st_buffer(poly, 20) outer_reach
    from blocks
    where not st_isempty(st_buffer(way, -20)))
select id, inner_poly, outer_reach,
    st_difference(outer_reach, inner_poly) bubble
from polys;
```

4. Активность фасадов. Полигоны

4. Активность фасадов. Выборка

Сделаем линии фасада:

```
create table blocks_lines as
select id, bubble,
       st_addmeasure(st_exteriorring(poly), 0, 1) block_line
from block_selections
where not st_isempty(st_buffer(way, -25));
```

4. Активность фасадов. Проекция

Проекция точек фирм на фасад:

```
select b.id,
       st_closestpoint(block_line, centroid) closest,
       st_interpolatepoint(block_line,
           centroid) * st_length(block_line) inter
   from block_lines b, firms f
   where st_within(centroid, bubble)
```

4. Активность фасадов. Проекция

4. Активность фасадов. Выборка

Разбиваем на отрезки:

```
with sequences as (
    select id, block_line,
        generate_series(0, st_length(block_line)::int, 25) pos
        from block_selections)
select st_linesubstring(block_line, pos, pos+25) ls,
       st_interpolatepoint(block_line, centroid)
        * st_length(block_line) inter
from sequences;
```

4. Активность фасадов. Выборка

Дальше ещё много кода.

В общем,

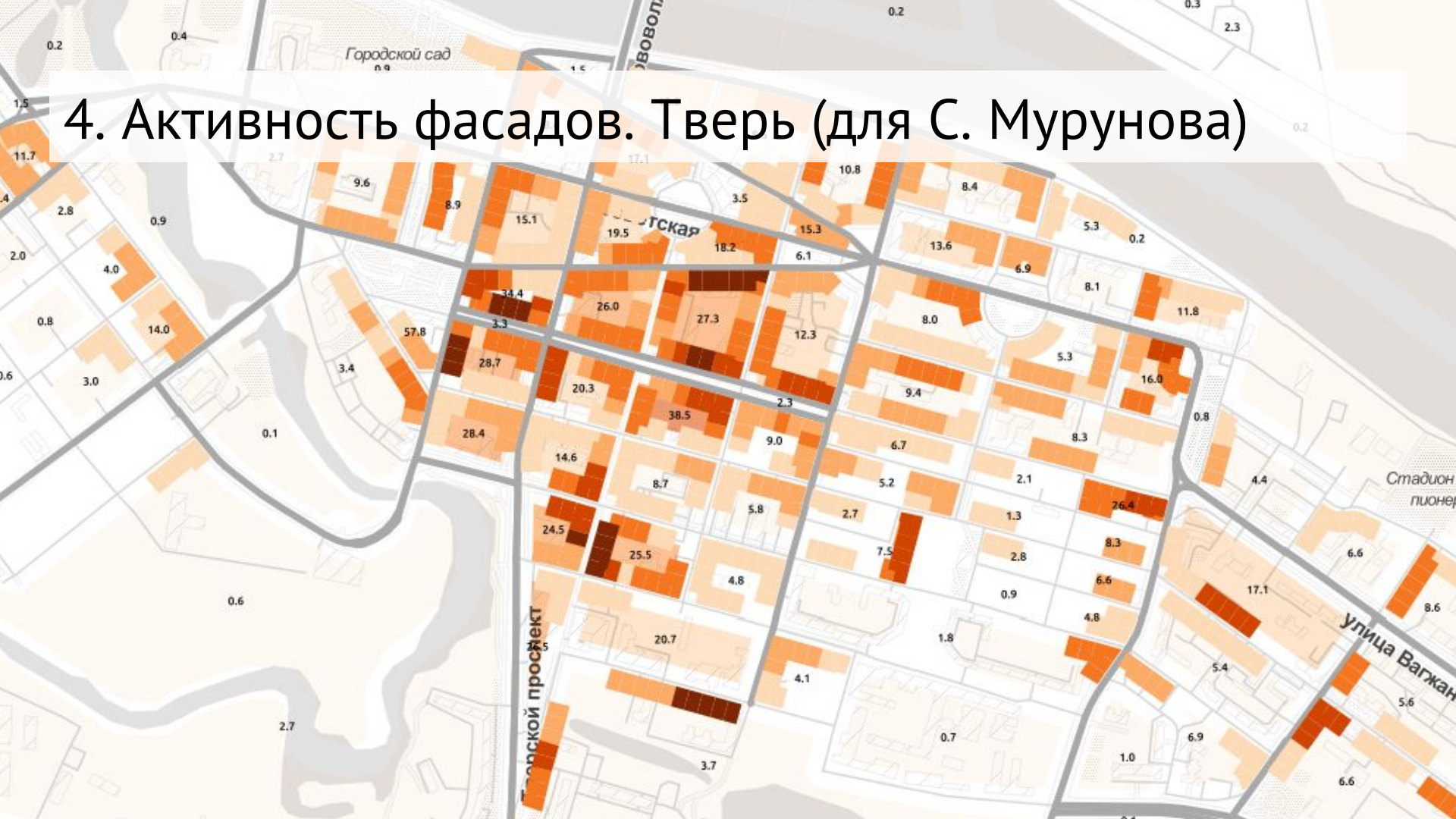
- группируем фирмы по номеру квартала и номеру отрезка
- рисуем нарезанные куски линий с числом фирм
- в моей реализации ещё есть сглаживание оконной функцией

Весь запрос - 5 Кб - по ссылке:

https://gist.github.com/culebron/545d5da90823066204d4b2a1f3cf87d7

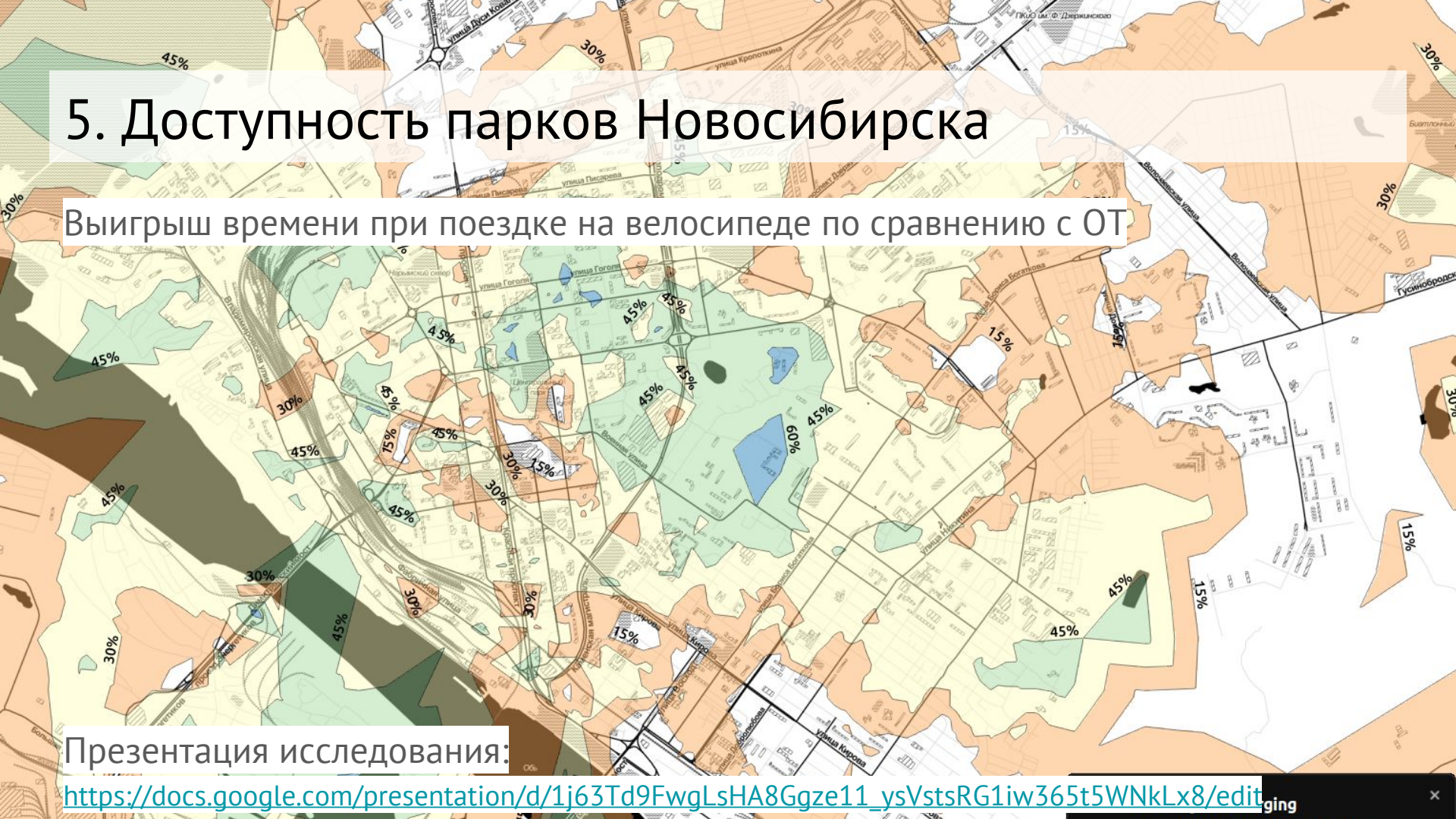
4. Активность фасадов. Результаты

4. Активность фасадов. Тверь (для С. Мурунова)



5. Доступность парков Новосибирска

Выигрыш времени при поездке на велосипеде по сравнению с ОТ

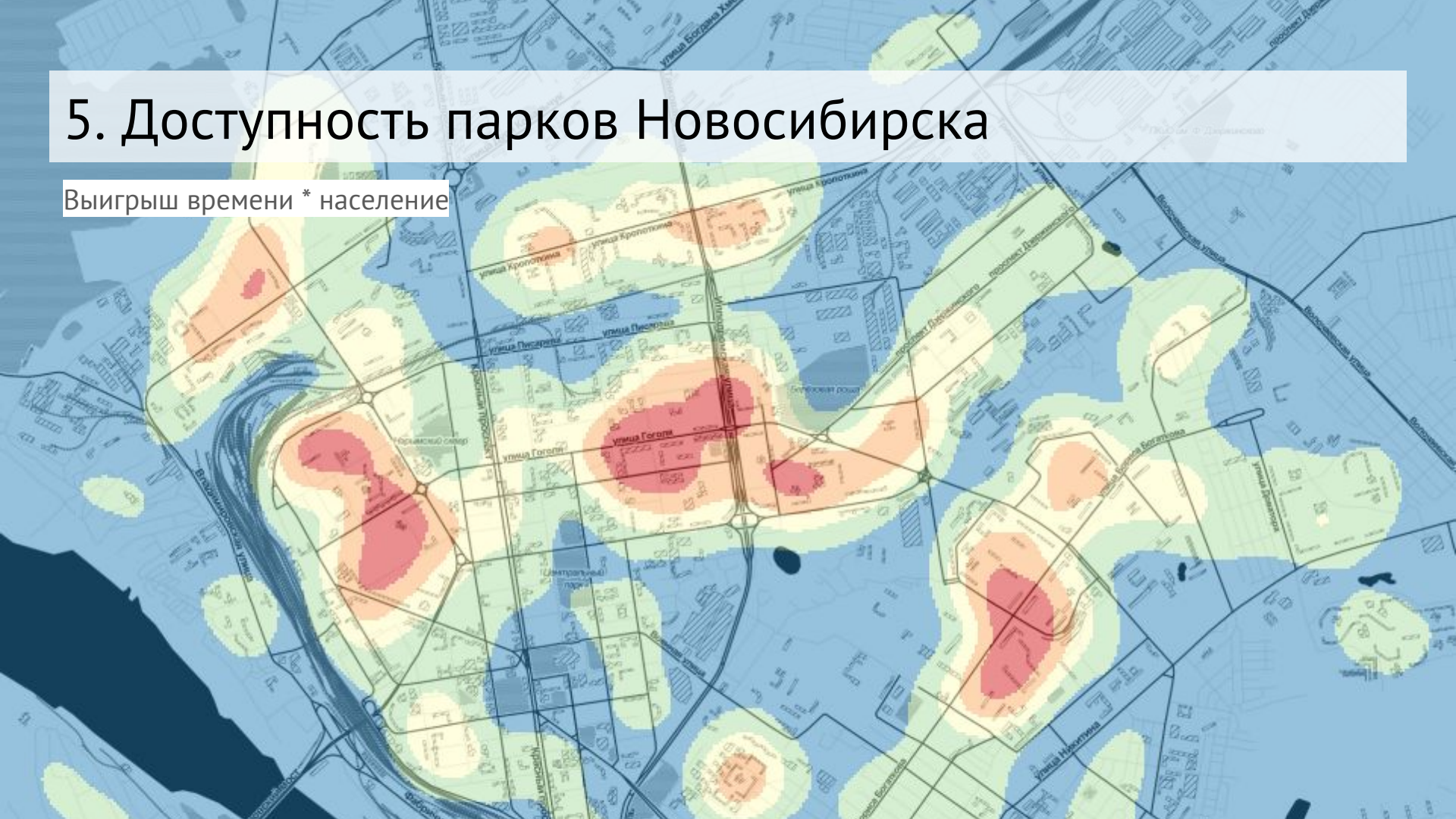


Презентация исследования:

https://docs.google.com/presentation/d/1j63Td9FwgLsHA8Ggze11_ysVstsRG1iw365t5WNkLx8/edit

5. Доступность парков Новосибирска

Выигрыш времени * население



5. Доступность парков Новосибирска

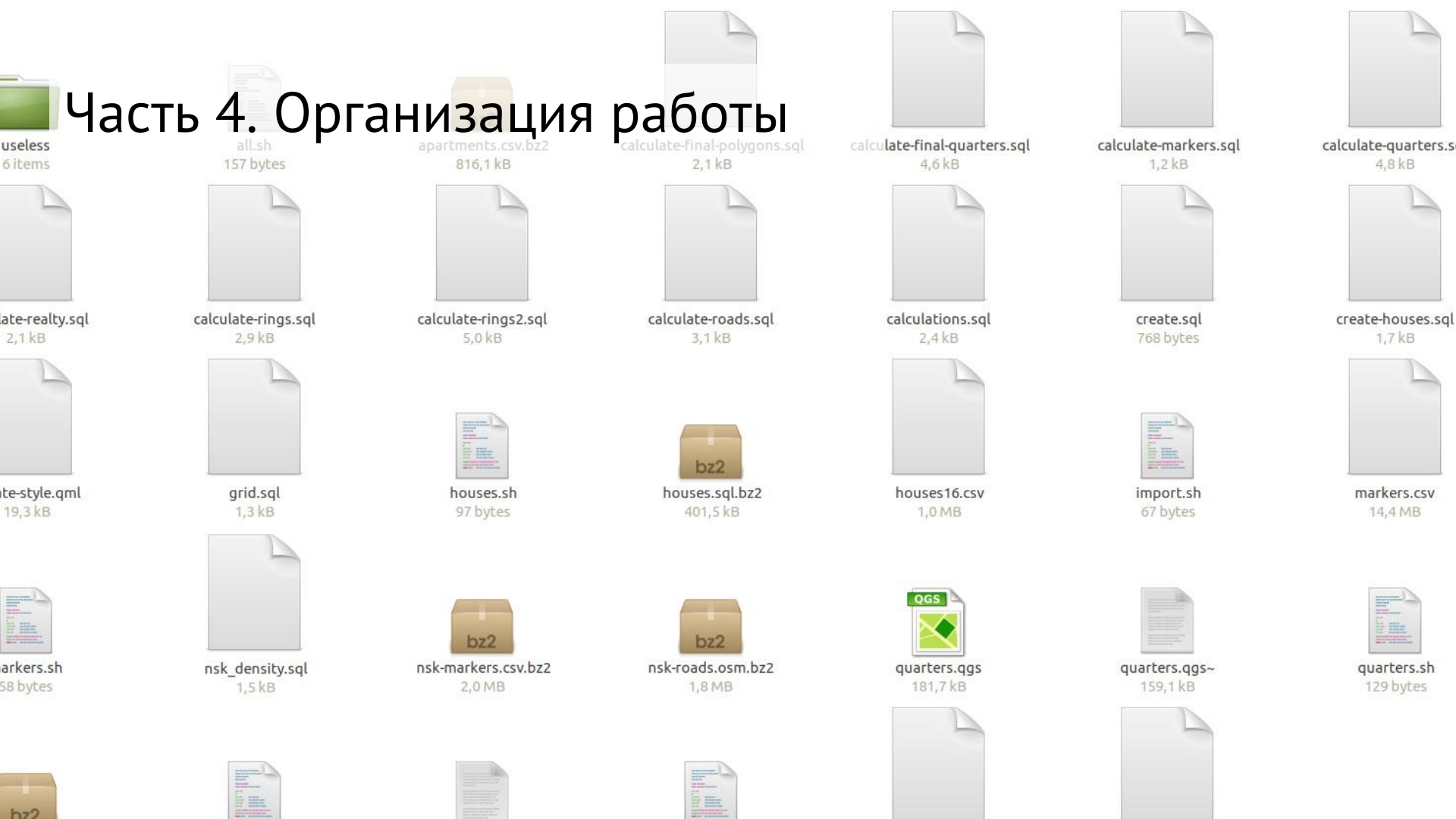
Среднее время поездки до парка

Парк	Время
Сквер Оперного театра	47
Сквер ГПНТБ	47
Набережная	48
Нарымский сквер	50
Центральный парк	52
Берёзовая роща	56
Монумент Славы	56
Тимирязевский сквер	60
ПКиО им. Ф. Дзержинского	61
Заельцовский парк	65
Сосновый бор	77
Бургинская роща	82
У моря обского	96

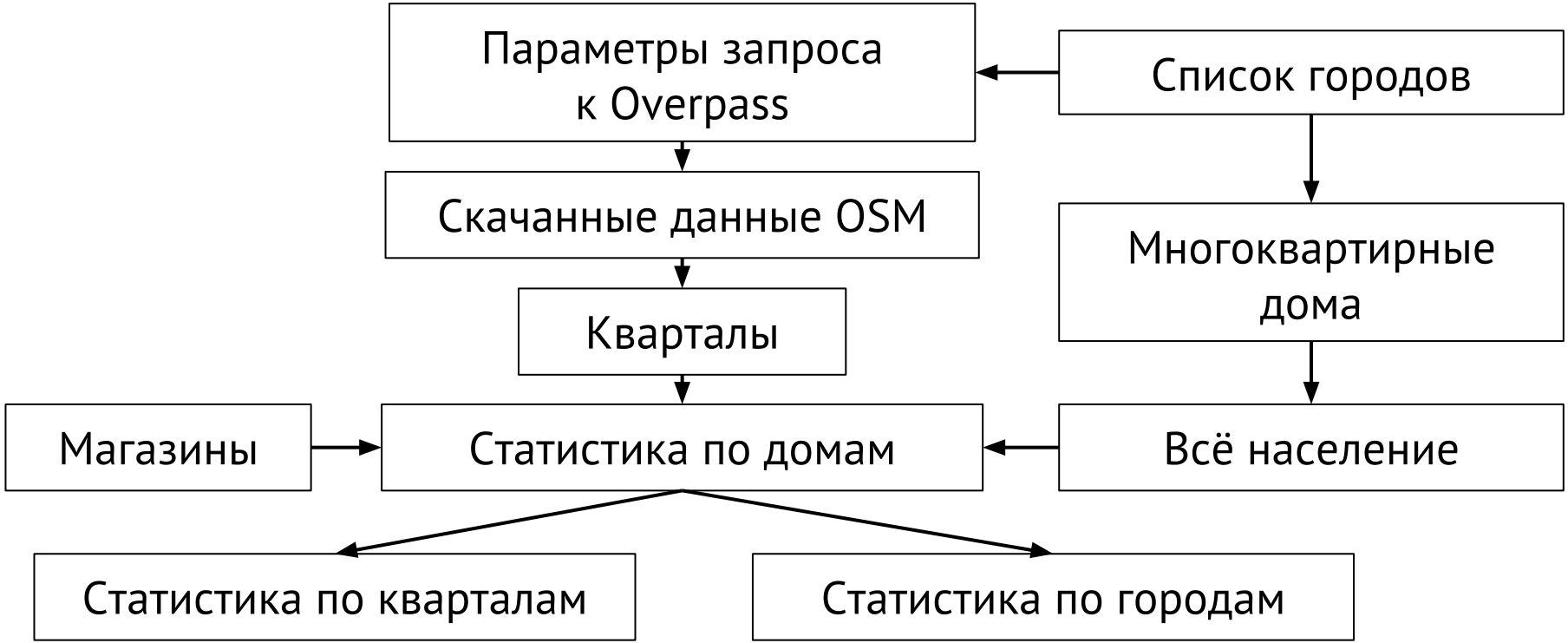
Полезные ссылки

- **NextGIS**
  Опенсорсная веб-платформа, интегрированная с PostGIS и QGIS
  http://nextgis.com/
- **Carto** (ранее CartoDB)
  Геохостинг с PostGIS и визуализацией
  https://carto.com/

Часть 4. Организация работы



Организация работы. Пример диаграммы данных



Организация работы

- Очень быстро ваш проект нельзя будет быстро пересчитать полностью
- И нельзя будет записать в скрипт последовательно

- Обработка данных - это **не последовательность, а дерево**
- Используйте менеджер заданий на ваш вкус, Makefile/Gulp, и т.д.
  - Он должен уметь хранить дерево задач и выполнить цепочку до той, что вы просите.
- Ввод первоначальных данных просто из файлов удобен - их можно держать в Git/Mercurial

Организация работы

Makefile:

```
db = coffee_beer
touch/create-db: touch/drop-db
	createdb $(db) -T template_postgis -O culebron && touch $@
touch/drop-db:
	dropdb $(db) --if-exists && touch $@
touch/create-house: sql/create-house.sql touch/create-db
	$(pdb) -f $< && touch $@
touch/insert-house: sql/import-house.sql $(codedfiles) touch/create-house
	for f in $(codedfiles); do a=$$(realpath "$$f"); $(pdb) -f $< -v fname="'$$a'";
done && touch $@
```

Организация работы

- Нужно минимизировать время итерации
  - Время редактирования и процедуры запуска
  - Время выполнения
- Если скрипты работают, ещё не значит, что всё в порядке
- Лучше сделать все этапы вычислений на небольшом наборе данных, чем один на большом
- Скрипты SQL придётся постоянно обновлять и дополнять. В идеале они должны выполняться как есть на БД, без “запускалок”
- Вместо `view, materialized view` лучше `create table`

Организация работы

- Ваш код будет постоянно меняться
- Ваши данные будут постоянно меняться
- И в данных, и в коде будут ошибки или новые решения
- Очень быстро нельзя будет переделать всё даже шелл-скриптом

Что нужно чтобы это не было больно?

Можно ли летать на велосипеде?*

* построить самолёт, приводимый в движение ногами

1979 год, Госсамер Альбатросс

Создатели самолёта выиграли приз Кремера, учреждённый в 1959 году

20 лет другие команды строили оптимальный самолёт.

Команда Пола МакКриди придумала как сделать быстро собираемый.

Спасибо за внимание!

Дмитрий Лебедев
Разработчик ГИС в BestPlace (http://bestplace.ai)

Мой личный ГИС-блог: https://www.facebook.com/planiformica

15.03.2017