Manipulação e análise de dados com o Postgis

Tópicos Avançados de Bases de Dados

Tabelas de exemplo

Tabela taxi_stands:

```
                    Table "public.taxi_stands"
    Column     |          Type           | Collation | Nullable | Default
---------------+-------------------------+-----------+----------+---------
 id            | integer                 |           | not null |
 name          | character varying(255) |           |          |
 location      | geometry(Point,4326)    |           |          |
 proj_location | geometry(Point,3763)    |           |          |
Indexes:
    "taxi_stands_pkey" PRIMARY KEY, btree (id)
    "taxi_stands_location_idx" gist (proj_location)
```

Notem que foi criado um índice espacial (GIST - Generalized Search Tree) para o atributo proj_location:

```
create index taxi_stands_location_idx on taxi_stands using GIST(proj_location);
```

Tabela de tracks

- A tabela de tracks tem um total de 102,267 tuplos, representando trajectos de táxi no dia 10 de Outubro de 2019;
- A estrutura da tabela é a seguinte:

```
                    Table "public.tracks"
   Column   |            Type             | Collation | Nullable | Default
------------+-----------------------------+-----------+----------+---------
 id         | integer                     |           | not null |
 ts         | integer                     |           |          |
 taxi       | character(8)                |           |          |
 state      | character varying(6)        |           |          |
 track      | geography(LineString,4326)  |           |          |
 proj_track | geometry(LineString,3763)   |           |          |
Indexes:
    "tracks_pkey" PRIMARY KEY, btree (id)
    "tracks_proj_track_idx" gist (proj_track)
```

Campos da tabela tracks

- id: identificador inteiro de cada trajecto, sendo a chave primária da tabela;

- ts: o timestamp de início de cada trajecto;

- taxi: o identificador único de cada táxi;

- state: a descrição do estado do táxi no trajecto (PICKUP, PAUSE, BUSY, FREE);

- track: a LINESTRING com a descrição do trajecto, com um ponto por segundo, em WGS84 (4326).

Criação da tabela tracks

- Na página de TABD foi colocado um ficheiro comprimido, tracks.tgz, que tem o SQL que tem os inserts para criarem a tabela; (note que se trata de um ficheiro muito grande, com mais de 1GB quando descompactado)

- Podemos criar a tabela tracks com o seguinte comando:

```
create table tracks (
  id int primary key,
  ts int,
  taxi char(8),
  state varchar(6),
  track geography(LINESTRING));
```

NOTA: o tipo geography corresponde a dados geométricos guardados no sistema de coordenadas WGS84, sendo portanto equivalente a geometry(LINESTRING,4326).

Alteração da tabela tracks

- Para facilitar a manipulação, é útil termos um atributo na tabela tracks em coordenadas projectadas no sistema 3763;
- Isto pode ser obtido pelo comando abaixo, tal como foi feito para tabela taxi_stands:

```
alter table tracks add proj_track geometry('LINESTRING',3763);

update tracks set proj_track = st_transform(track::geometry,3763);
```

NOTA: Convém igualmente criar um índice sobre o atributo proj_track:

```
create index tracks_proj_track_idx on tracks using GIST(proj_track);
```

Consultas SQL não espaciais

- Para se familiarizar com os datasets das tabelas taxi_stands e tracks formule as seguintes consultas sem componente espacial:

1. Obter a listagem com os nomes das diferentes praças de táxi;

2. Obter o número total de trajectos na tabela tracks;

3. Obter a percentagem de trajectos pelos diferentes valores do atributo state;

4. Obter o número total de táxis diferentes;

5. Obter por hora do dia o número total de viagens com state = ‘BUSY’

Consultas SQL espacial

- O PostGIS fornece um conjunto de funções que permitem a execução de consultas com componente espacial;

- O manual do PostGIS será uma consulta obrigatória, estando disponível em: https://postgis.net/docs/

- Em particular, o capítulo 8: PostGIS Reference, lista e explica as funções disponíveis.

Consultas SQL espacial

- Formule as seguintes consultas com componente espacial:

1. Listar os nomes e as coordenadas em texto da sua localização;

2. Obter a distância em metros entre as praças de táxis mais distantes entre si;

3. Obter o nome da praça de táxi mais a Norte, mais a Sul, mais a Oeste e mais a Este;

4. Obter o total em quilómetros de todos os trajectos;

5. Obter o total de quilómetros por estado do táxi (state).

Postgres e Python

- Como referimos, para a manipulação analítica das bases de dados passaremos a usar a linguagem de programação Python e o módulo psycopg2 para acesso ao Postgres/PostGIS.

  https://pypi.org/project/psycopg2/

  https://www.psycopg.org/docs/

- Para visualização, usaremos por enquanto a biblioteca matplotlib:

  https://matplotlib.org

  https://matplotlib.org/contents.html

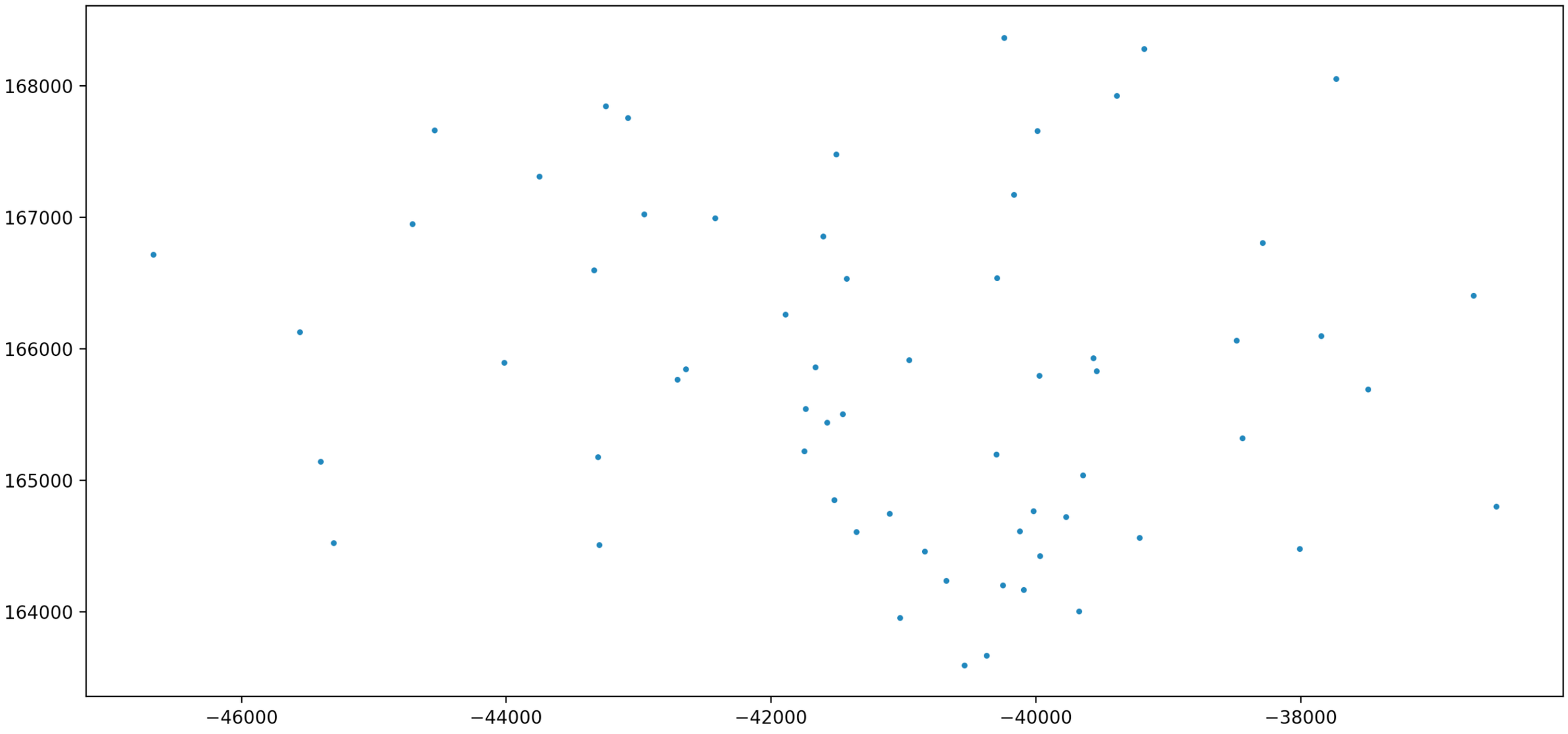
Exemplo: visualização de um mapa

- Vamos criar um pequeno programa para construir um mapa com a localização das praças de táxi:

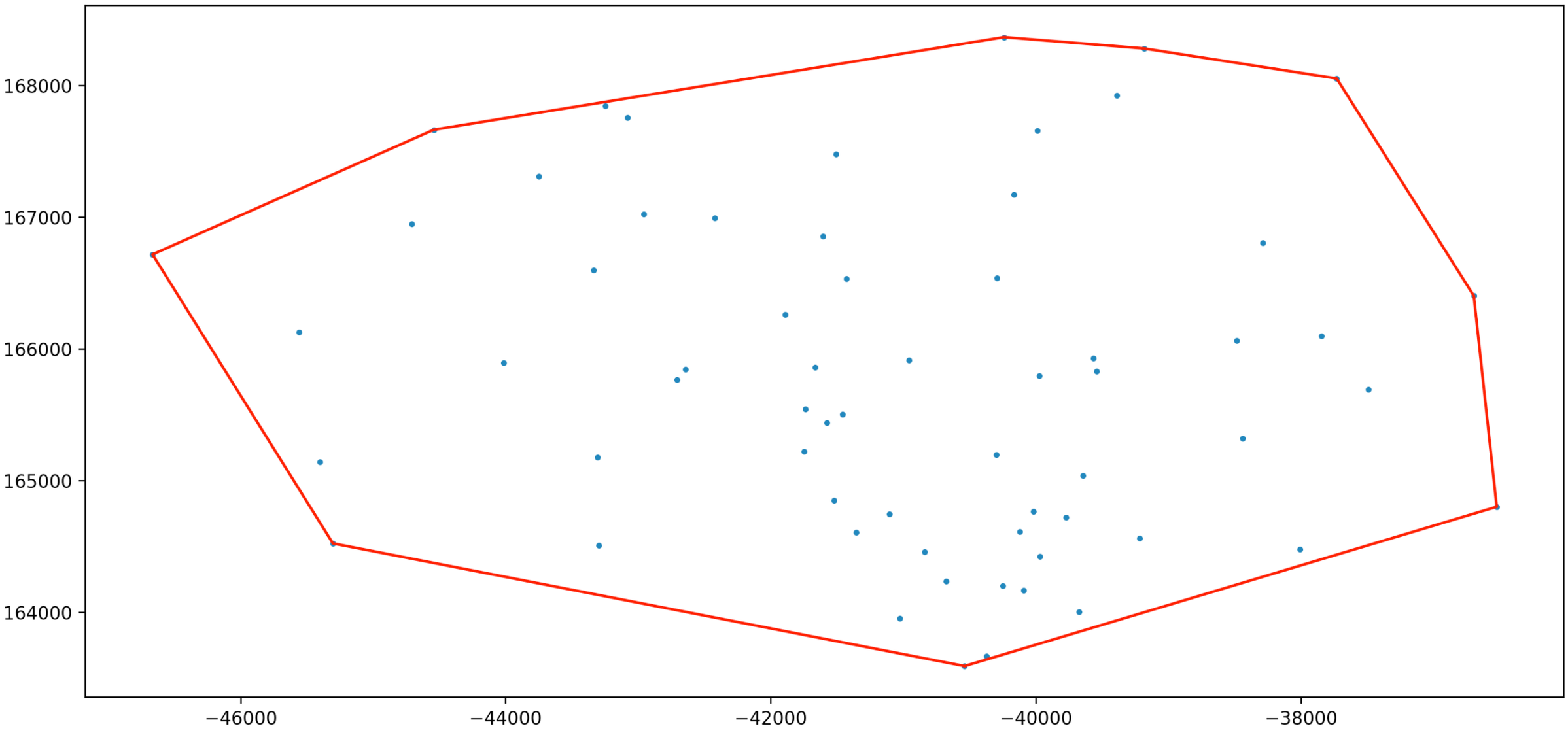
```
import matplotlib.pyplot as plt
import psycopg2

scale=1/30000
conn = psycopg2.connect("dbname=michelferreira user=michelferreira")
cursor_psql = conn.cursor()
sql = "select st_astext(proj_location) from taxi_stands"
cursor_psql.execute(sql)
results = cursor_psql.fetchall()
xs = []
ys = []
for row in results:
    point_string = row[0]
    point_string = point_string[6:-1]
    (x,y) = point_string.split()
    xs.append(float(x))
    ys.append(float(y))
width_in_inches = ((max(xs)-min(xs))/0.0254)*1.1
height_in_inches = ((max(ys)-min(ys))/0.0254)*1.1
fig = plt.figure(figsize=(width_in_inches*scale,height_in_inches*scale))
plt.scatter(xs,ys,s=5)
plt.show()
```

Mapa de praças de táxi



Um outro exemplo



Um outro exemplo

```
sql = "select st_astext(st_convexhull(st_collect(proj_location))) from taxi_stands"
cursor_psql.execute(sql)
results = cursor_psql.fetchall()

xs = []
ys = []

for row in results:
    print(row[0])
    points_string = row[0]
    points_string = points_string[9:-2]
    points = points_string.split(',')
    for point in points:
        (x,y) = point.split()
        xs.append(float(x))
        ys.append(float(y))
        print(x,y)

plt.plot(xs,ys, color='red')
plt.show()
```

Visualização de consultas espaciais

1. Calcular e mostrar o polígono convexo mais pequeno que contém 5 praças de táxi da tabela taxi_stands;

2. Mostrar todos os trajectos que se originam a menos de 150m da Praça de Táxis de Campanhã;

3. Mostrar todos os trajectos em ocupado que ficam dentro de um envelope rectangular que contenha todas as praças de táxi da tabela taxi_stands;