Objetivos:

- Criar camada no QGIS a partir de arquivos GeoPackage (GPKG) e Shapefile;
- Criar camadas temporárias vetoriais no QGIS.

Observações importantes:

- Para reproduzir os exemplos e fazer os exercícios, é necessário utilizar o ambiente Python integrado ao QGIS (Python Console ou Editor de Scripts do QGIS). Muitas das classes utilizadas (como `QgsVectorLayer` e `QgsProject`) são exclusivas do ambiente do QGIS;
- A documentação oficial da API Python do QGIS (PyQGIS) está disponível em: https://qgis.org/pyqgis/3.40 - selecione a versão que corresponde ao seu QGIS instalado (nesta aula, utilizaremos a versão 3.40).

I. Contextualização

Ao trabalhar com dados geoespaciais no QGIS, é comum utilizarmos camadas vetoriais que representam elementos como cidades, estradas, rios, ou unidades administrativas. Por meio da biblioteca **PyQGIS**, é possível criar e manipular essas camadas diretamente via código Python, o que abre espaço para automação de processos, personalização de análises e integração com outras ferramentas.

Conceitos Fundamentais

- **Camada Vetorial**: Representação digital de dados espaciais em três tipos principais de geometria: ponto, linha e polígono;
- **GeoPackage (GPKG)**:
  - Formato moderno baseado em banco de dados SQLite;
  - Suporta múltiplas camadas em um único arquivo .gpkg (vetoriais e raster);
  - Recomendado pelo OGC (Open Geospatial Consortium).
- **Shapefile**:
  - Formato tradicional amplamente utilizado;
  - Composto por múltiplos arquivos: .shp, .shx, .dbf, e geralmente .prj.
- **Camada temporária**:
  - Criada na memória durante a execução da aplicação;
  - Descartada ao encerrar a sessão (a menos que seja salva explicitamente).
- **Feições (features)**:
  - Elementos que compõem a camada;
  - Cada feição possui geometria (ponto, linha ou polígono) e atributos (ex: nome, população).

Arquitetura de dados no QGIS

O QGIS utiliza um modelo hierárquico para gerenciar dados espaciais:

```
Projeto QGIS
├── Camadas Vetoriais
│   ├── Feições (Features)
│   │   ├── Geometria (Point/Line/Polygon)
│   │   └── Atributos (Fields)
│   └── Sistema de Referência de Coordenadas (CRS)
└── Camadas Raster
```

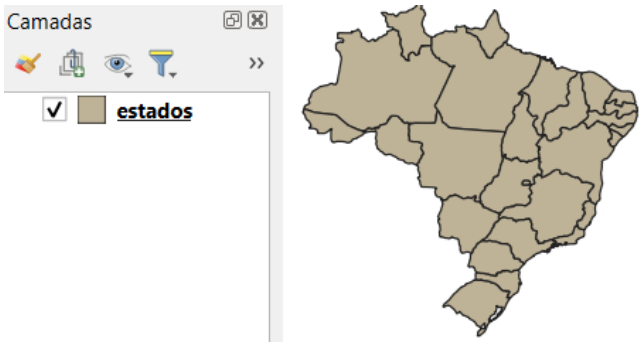
II. Criando camadas a partir de arquivos

Usando GeoPackage (GPKG)

Vamos carregar uma camada armazenada no formato GeoPackage.

Exemplo 1:

```
# Define o caminho completo para o arquivo GeoPackage
caminho = "D:/bdgeo/uf/uf.gpkg"

# Define o nome que será atribuído à camada dentro do QGIS (nome de exibição)
nome_camada = "estados"

# Cria uma camada vetorial a partir do arquivo GeoPackage utilizando o driver "ogr"
camada = QgsVectorLayer(caminho, nome_camada, "ogr")

# Verifica se a camada foi carregada corretamente (validação)
if not camada.isValid():
    # Informa erro caso o caminho esteja incorreto ou o arquivo seja inválido
    print("Erro ao carregar a camada.")
else:
    # Adiciona a camada ao painel de camadas do QGIS (interface gráfica)
    QgsProject.instance().addMapLayer(camada)
```



Comentários:

- Evite caminhos de arquivos que possuem espaços, traços e caracteres especiais no nome das pastas;
- O driver "`ogr`" é capaz de abrir uma variedade de formatos vetoriais, incluindo GeoPackage, Shapefile e GeoJSON;

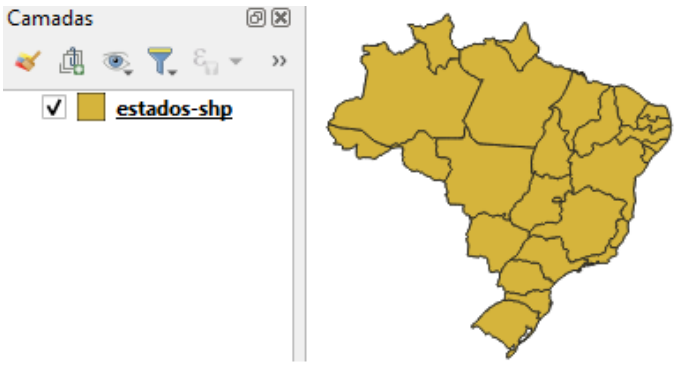
- O método `QgsProject.instance().addMapLayer()` adiciona a camada à interface gráfica do QGIS.

Usando Shapefile

Vamos carregar uma camada armazenada no formato Shapefile.

Exemplo 2:

```
# Define o caminho completo para o arquivo shapefile (.shp)
caminho = "D:/bdgeo/uf/uf.shp"

# Define o nome que será atribuído à camada dentro do QGIS (nome de exibição)
nome_camada = "estados-shp"

# Cria uma camada vetorial a partir do shapefile utilizando o driver "ogr"
camada = QgsVectorLayer(caminho, nome_camada, "ogr")

# Verifica se a camada foi carregada corretamente (validação)
if not camada.isValid():
    # Informa erro caso o caminho esteja incorreto ou o arquivo seja inválido
    print("Erro ao carregar a camada.")
else:
    # Adiciona a camada ao painel de camadas do QGIS (interface gráfica)
    QgsProject.instance().addMapLayer(camada)
```



Comentários:

- Todos os arquivos componentes do Shapefile devem estar na mesma pasta: .shp, .dbf, .shx e preferencialmente .prj para a referência espacial.

III. Criando camadas temporárias (memória)

Camadas temporárias são úteis para testar scripts, realizar análises momentâneas ou gerar dados em tempo real, sem a necessidade de salvamento imediato.

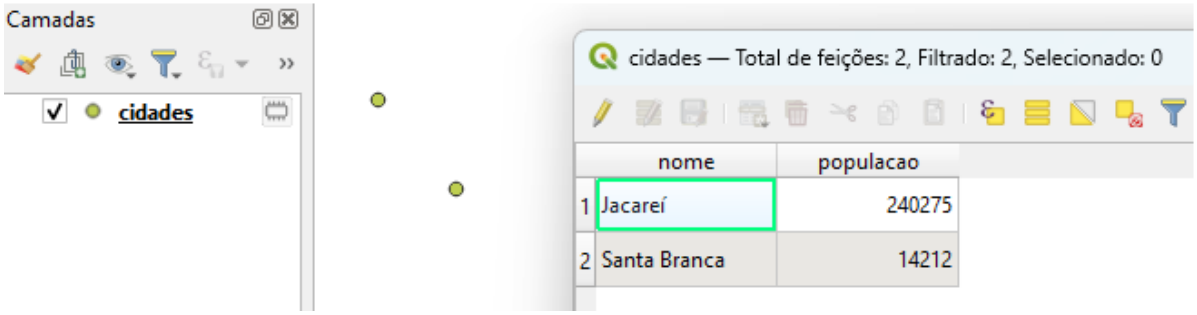
Criando camada com pontos

Neste exemplo, criamos uma camada temporária do tipo **ponto** (`Point`) com duas feições (Jacareí e Santa Branca) e dois atributos: `nome` e `populacao`.

Exemplo 3:

```
# Define o nome da camada
nome_camada = "cidades"

# Cria uma camada vetorial temporária do tipo ponto, com sistema de referência WGS 84
camada = QgsVectorLayer("Point?crs=EPSG:4326", nome_camada, "memory")

# Obtém o provedor de dados da camada, responsável por manipular feições e atributos
provider = camada.dataProvider()

# Adiciona o campo "nome" (tipo texto) à camada
provider.addAttributes([QgsField("nome", QVariant.String)])

# Adiciona o campo "populacao" (tipo inteiro) à camada
provider.addAttributes([QgsField("populacao", QVariant.Int)])

# Atualiza a definição de campos da camada para refletir os novos atributos
camada.updateFields()

# Define os pontos com coordenadas geográficas
jcr = QgsPointXY(-45.96642, -23.30555)   # Jacareí
stb = QgsPointXY(-45.88441, -23.39755)   # Santa Branca

# Converte os pontos QgsPointXY para objetos QgsGeometry, exigidos pelas feições
jcr_geom = QgsGeometry.fromPointXY(jcr)
stb_geom = QgsGeometry.fromPointXY(stb)

# Cria uma feição para Jacareí e define sua geometria e atributos
jcr_feicao = QgsFeature(camada.fields())
jcr_feicao.setGeometry(jcr_geom)           # Define a geometria (ponto)
jcr_feicao.setAttribute(0, "Jacareí")      # Define o nome (campo na posição 0)
jcr_feicao.setAttribute(1, 240275)         # Define a população (campo na posição 1)
provider.addFeature(jcr_feicao)            # Adiciona a feição ao provedor de dados

# Cria uma feição para Santa Branca e define sua geometria e atributos
stb_feicao = QgsFeature(camada.fields())
stb_feicao.setGeometry(stb_geom)
stb_feicao.setAttribute(0, "Santa Branca")
stb_feicao.setAttribute(1, 13975)
provider.addFeature(stb_feicao)

# Adiciona a camada criada ao painel do QGIS (interface gráfica)
QgsProject.instance().addMapLayer(camada)
```

A seguir tem-se a camada criada e tabela de atributos da camada.



Criando camada com linhas

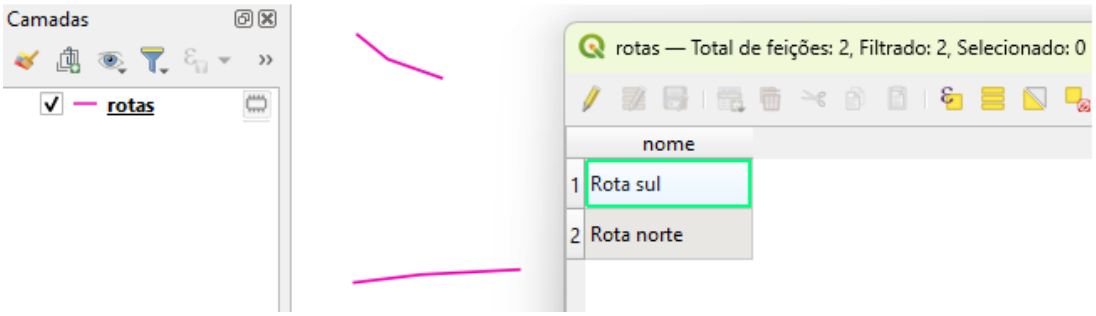
Agora criaremos uma camada com feições do tipo **linha** (`LineString`) representando rotas entre localidades.

Exemplo 4:

```
# Define o nome da camada
nome_camada = "rotas"

# Cria uma camada vetorial temporária do tipo linha (LineString),
# com sistema de referência WGS 84 (EPSG:4326)
camada = QgsVectorLayer("LineString?crs=EPSG:4326", nome_camada, "memory")

# Obtém o provedor de dados da camada, responsável por manipular feições e atributos
provider = camada.dataProvider()

# Adiciona apenas o campo "nome" (tipo texto) à camada
provider.addAttributes([QgsField("nome", QVariant.String)])
camada.updateFields()

# Define os pontos geográficos das rotas
gua = QgsPointXY(-46.03542, -23.41555)  # Guararema
stb = QgsPointXY(-45.88441, -23.39755)  # Santa Branca
prb = QgsPointXY(-45.66306, -23.38624)  # Paraibuna

mv  = QgsPointXY(-46.02808, -22.85695)  # Monte Verde
sfx = QgsPointXY(-45.95976, -22.91206)  # São Francisco Xavier
ml  = QgsPointXY(-45.83838, -22.95449)  # Monteiro Lobato

# Cria a feição "Rota sul" com geometria formada pelos pontos:
# Guararema → Santa Branca → Paraibuna
rota_sul = QgsFeature(camada.fields())
rota_sul.setGeometry(QgsGeometry.fromPolylineXY([gua, stb, prb]))
rota_sul.setAttribute("nome", "Rota sul")
provider.addFeature(rota_sul)

# Cria a feição "Rota norte" com geometria formada pelos pontos:
# Monte Verde → São Francisco Xavier → Monteiro Lobato
rota_norte = QgsFeature(camada.fields())
rota_norte.setGeometry(QgsGeometry.fromPolylineXY([mv, sfx, ml]))
rota_norte.setAttribute("nome", "Rota norte")
```

```
provider.addFeature(rota_norte)

# Adiciona a camada criada ao painel do QGIS (interface gráfica)
QgsProject.instance().addMapLayer(camada)
```

A seguir tem-se a camada criada e tabela de atributos da camada.



Criando camada com polígonos

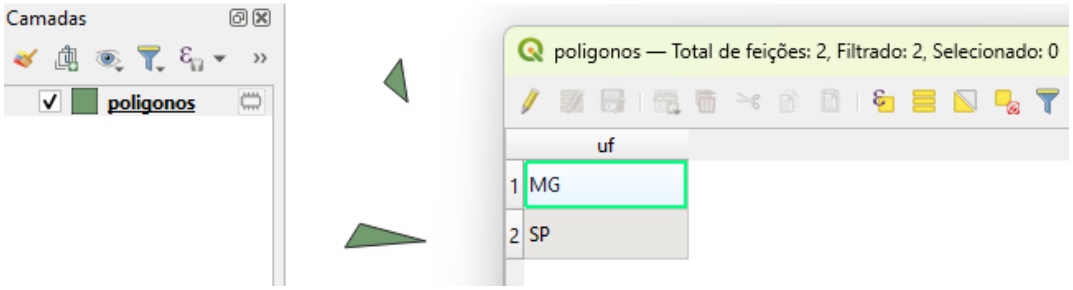
Neste trecho, construiremos uma camada com geometrias do tipo **polígono** (Polygon), útil para representar áreas como estados, zonas de proteção ou áreas urbanas.

Exemplo 5:

```
# Define o nome da camada
nome_camada = "poligonos"

# Cria uma camada temporária do tipo polígono com o sistema de referência EPSG:4326 (WGS 84)
camada = QgsVectorLayer("Polygon?crs=EPSG:4326", nome_camada, "memory")

# Obtém o provedor de dados da camada
provider = camada.dataProvider()

# Adiciona o campo "uf" (tipo texto)
provider.addAttributes([QgsField("uf", QVariant.String)])
camada.updateFields()

# === Feição 1: Estado de MG ===
gon = QgsPointXY(-45.85587, -22.65843)  # Gonçalves
par = QgsPointXY(-45.77874, -22.54922)  # Paraisópolis
spm = QgsPointXY(-45.74707, -22.74892)  # Sapucaí-Mirim

# Cria a geometria da feição MG (polígono fechado)
poligono_mg = QgsFeature(camada.fields())
poligono_mg.setGeometry(QgsGeometry.fromPolygonXY([[gon, par, spm, gon]]))
poligono_mg.setAttribute("uf", "MG")
provider.addFeature(poligono_mg)

# === Feição 2: Estado de SP ===
jcr = QgsPointXY(-45.96642, -23.30555)  # Jacareí
gua = QgsPointXY(-46.03542, -23.41555)  # Guararema
```

```
prb = QgsPointXY(-45.66306, -23.38624)  # Paraibuna

# Cria a geometria da feição SP (polígono fechado)
poligono_sp = QgsFeature(camada.fields())
poligono_sp.setGeometry(QgsGeometry.fromPolygonXY([[jcr, gua, prb, jcr]]))
poligono_sp.setAttribute("uf", "SP")
provider.addFeature(poligono_sp)

# Adiciona a camada ao projeto QGIS
QgsProject.instance().addMapLayer(camada)
```

A seguir tem-se a camada criada e tabela de atributos da camada.



Exercícios

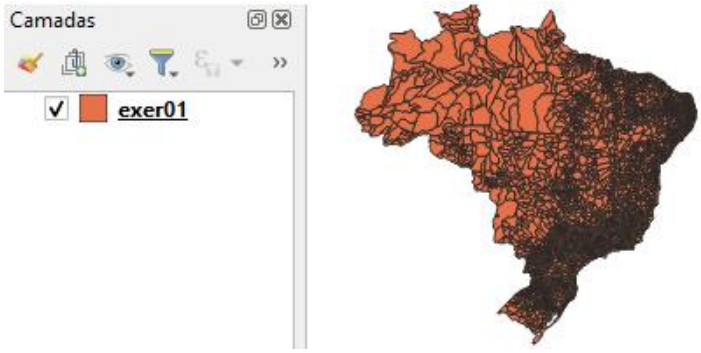
Veja os vídeos se tiver dúvidas nos exercícios:

Exercícios 1 e 2: https://youtu.be/ERuNQdeVDuA

Exercícios 3 a 5: https://youtu.be/QqLfnuxBIGk

Exercício 1: Altere o código do Exemplo 1 para carregar o arquivo municipio.gpkg. A camada deverá ser nomeada “exer01”.

Resposta:



Dicas:

- Atualize o caminho do arquivo para o local em que se encontra o arquivo municipio.gpkg no seu computador;
- Altere o nome da camada para exer01.

Observação: Certifique-se de que o caminho do arquivo não contenha espaços, acentos ou caracteres especiais (como ç, ã, é, entre outros).

Exercício 2: Modifique o código do Exemplo 3 para adicionar uma feição representando a cidade de Jambeiro. A camada deverá ter o nome “exer02”.

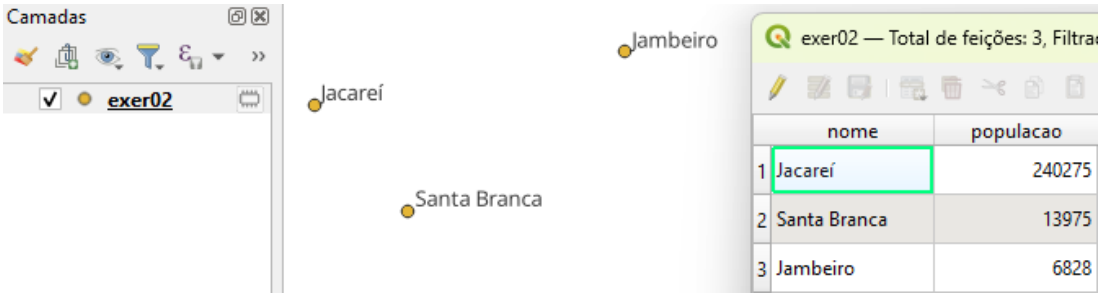
Dados da cidade:

- Coordenadas: longitude -45.69457, latitude -23.25871;
- População: 6.828 habitantes.

Dicas:

- Crie um ponto utilizando `QgsPointXY` com as coordenadas fornecidas;
- Converta o ponto para geometria usando `QgsGeometry.fromPointXY`;
- Crie uma feição com a geometria e os atributos correspondentes e adicione-a à camada.

Resposta:



Exercício 3: Modifique o código do Exercício 2 para incluir um novo campo chamado area (do tipo double). A camada deverá ser nomeada como “exer03”.

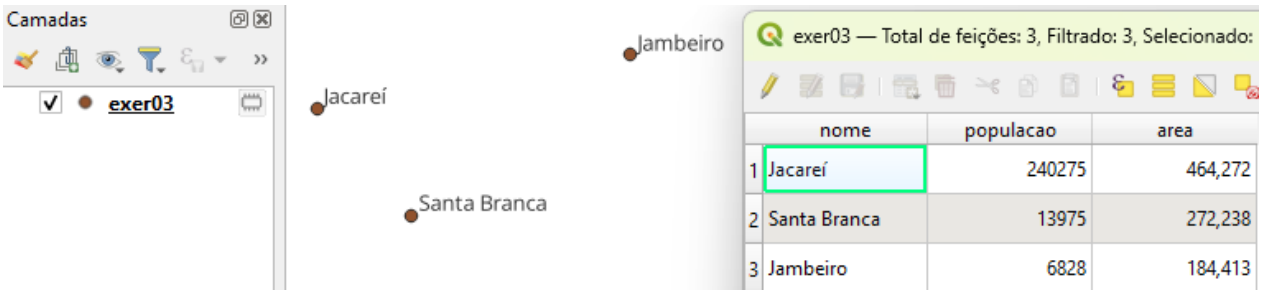
Áreas dos municípios (em km²) (fonte: https://www.ibge.gov.br/geociencias/organizacao-do-territorio/estrutura-territorial/15761-areas-dos-municipios.html):

- Jacareí: 464.272
- Jambeiro: 184.413
- Santa Branca: 272.238

Dicas:

- Adicione o campo `"area"` à camada com o tipo `QVariant.Double`;
- Utilize o método `setAttribute` para definir a área de cada município;
- Lembre-se de que o separador decimal deve ser o ponto (e não vírgula). Exemplo correto: `464.272`.

Resposta:



Exercício 4: Utilizar uma estrutura de repetição para criar feições automaticamente a partir de uma lista de dados contendo informações de várias cidades.

Lista de cidades:

Cada elemento da lista contém: `[nome, longitude, latitude, população, área]`

```
cidades = [
    ["Guararema", -46.03542, -23.41555, 31236, 270.816],
    ["Jacareí", -45.96642, -23.30555, 240275, 464.272],
    ["Jambeiro", -45.69457, -23.25871, 6828, 184.413],
    ["Paraibuna", -45.673288, -23.373758, 17667, 809.576],
    ["Salesópolis", -45.847091, -23.531665, 15202, 424.997],
    ["Santa Branca", -45.88441, -23.39755, 13975, 272.238]
]
```

Adicione no código do Exercício 3 a capacidade de criar as feições a partir dos dados fornecidos na lista `cidades`.

Dicas:

- Utilize um laço `for` para percorrer a lista `cidades`:

```
for cidade in cidades:
    # Define o ponto com coordenadas geográficas
    ponto = QgsPointXY(cidade[1], cidade[2])
```

- Dentro do laço, execute os seguintes passos:
  - Converta o ponto para uma geometria com `QgsGeometry.fromPointXY`;
  - Crie uma feição (`QgsFeature`);
  - Defina a geometria e os atributos: nome, população e área;
  - Adicione a feição ao provedor de dados.

Nota: Acesse os valores da lista pelo índice:

`cidade[0]` (nome)

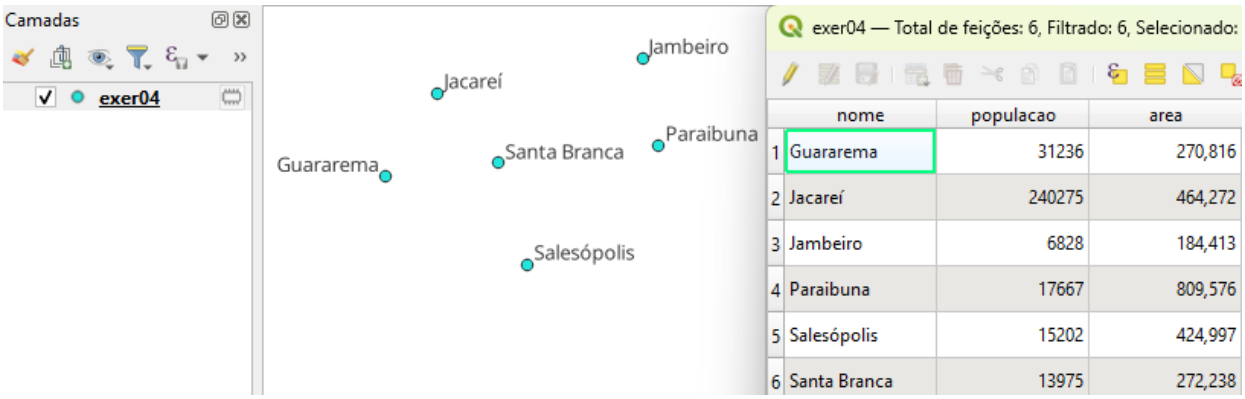
`cidade[1]` (longitude)

`cidade[2]` (latitude)

`cidade[3]` (população)

`cidade[4]` (área).

Resposta:



	nome	populacao	area
1	Guararema	31236	270,816
2	Jacareí	240275	464,272
3	Jambeiro	6828	184,413
4	Paraibuna	17667	809,576
5	Salesópolis	15202	424,997
6	Santa Branca	13975	272,238

Exercício 5: Ampliar a lista do Exercício 4 com os dados das cidades de Lagoinha e São Luiz do Paraitinga.

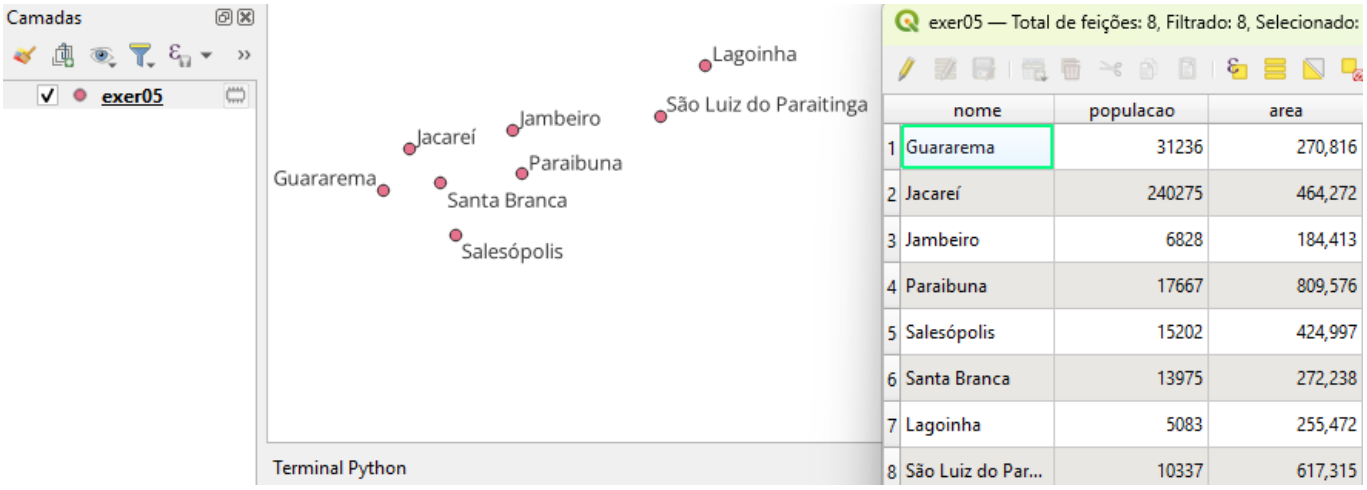
Dados das cidades a serem adicionadas:

- Lagoinha
  - Coordenadas: longitude -45.19012, latitude -23.09025
  - População: 5.083 habitantes
  - Área: 255.472 km²
- São Luiz do Paraitinga
  - Coordenadas: longitude -45.31049, latitude -23.22217
  - População: 10.337 habitantes
  - Área: 617.315 km²

Dica:

- Adicione essas duas cidades como novos elementos ao final da lista `cidades` definida no exercício anterior.

Resposta:



	nome	populacao	area
1	Guararema	31236	270,816
2	Jacareí	240275	464,272
3	Jambeiro	6828	184,413
4	Paraibuna	17667	809,576
5	Salesópolis	15202	424,997
6	Santa Branca	13975	272,238
7	Lagoinha	5083	255,472
8	São Luiz do Par...	10337	617,315