



Aula dia 21 de  
novembro



## Conectando com banco de dados

- Utilizar a biblioteca [PyGreSQL](#) (Exemplo da instalação a seguir)
- Analizar e replicar o [exemplo do arquivo indicado](#).

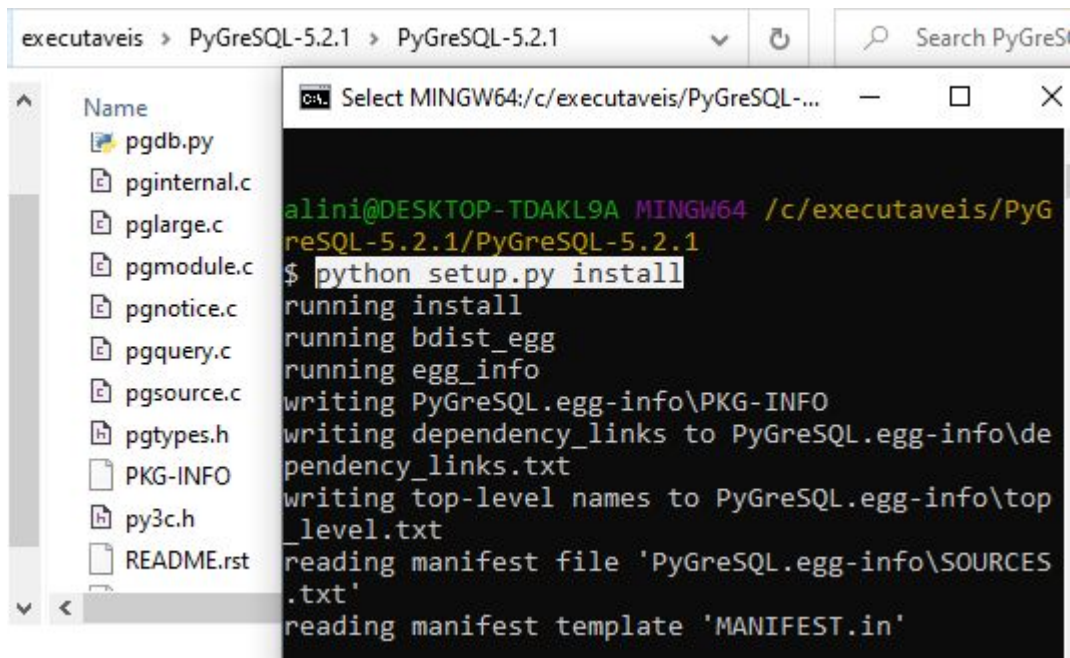
# Instalando a biblioteca

Página do PyGreSQL na pypi

comando-> `pip install pyGreSQL`

OU

Baixar arquivo zip, descompactar e de dentro da pasta, executar o comando `python setup.py install`



The image shows a Windows file explorer window on the left, displaying the contents of the 'PyGreSQL-5.2.1' directory. The files listed are: pgdb.py, pginternal.c, pglarge.c, pgmodule.c, pgnotice.c, pgquery.c, pgsource.c, pgtypes.h, PKG-INFO, py3c.h, and README.rst. On the right, a terminal window titled 'Select MINGW64:/c/executaveis/PyGreSQL-...' shows the execution of the command 'python setup.py install'. The terminal output indicates that the installation is running, including steps for running bdist\_egg, egg\_info, writing PKG-INFO, writing dependency\_links, writing top-level names, reading the manifest file, and reading the manifest template.

```
executaveis > PyGreSQL-5.2.1 > PyGreSQL-5.2.1
Name
pgdb.py
pginternal.c
pglarge.c
pgmodule.c
pgnotice.c
pgquery.c
pgsource.c
pgtypes.h
PKG-INFO
py3c.h
README.rst

alini@DESKTOP-TDAKL9A MINGW64 /c/executaveis/PyGreSQL-5.2.1
$ python setup.py install
running install
running bdist_egg
running egg_info
writing PyGreSQL.egg-info\PKG-INFO
writing dependency_links to PyGreSQL.egg-info\dependency_links.txt
writing top-level names to PyGreSQL.egg-info\top_level.txt
reading manifest file 'PyGreSQL.egg-info\SOURCES.txt'
reading manifest template 'MANIFEST.in'
```



# Modelar nosso banco



- Quantas tabelas precisamos criar?

Volte para as especificações e verifique qual dado vamos precisar disponibilizar e faça uma comparação com os dados que o arquivo possui.



# Especificações

- Receberemos um arquivo de extensão .csv com os dados dos inscritos.
- Precisamos armazenar os dados em um banco de dados relacional
- Precisamos disponibilizar API's para que nosso sistema se comunique com o mundo externo.

API's disponibilizadas:

- **Listar todos os participantes**
- Deletar um participante
- Alterar dados de um participante
- Adicionar um novo participante
- Detalhar um participante



# Modelar nosso banco

- De acordo com as especificações, apenas vamos construir uma tabela chamada pessoa (**make it simple**).
- Esta tabela irá possuir os campos: Nome, endereço, cpf, estado, turma, período, módulo
- Vã para o pgAdmin e crie um **banco de dados** específico para nosso projeto, chamado **mvcad-cursos** e nele, crie uma **tabela** chamada **pessoa**, com todos os campos necessários.
- Comando: **xxxxxxx**



# Conexão com o banco

- Estabelecer conexão com banco ->

```
db = DB(dbname='mvcad-cursos', host='localhost', port=5432, user='postgres', passwd='postgres')
```

- Buscar todas as tabelas -> `db.get_tables()`

- Inserir dados na tabela -> `db.insert('aluno', id=uuid4(), name='Alini')`

- Fazer select do banco -> 

```
query = db.query('select * from aluno')
resultato = query.getresult()
```



# Criando métodos que manipulam dados do banco

- Criar função que salva uma pessoa
- Criar função que busca uma pessoa
- Criar função que retorna todas as pessoas salvas
- Criar função que deleta uma pessoa





# Ligar as funções criadas com o arquivo CSV importado

- Criar função que lê todos os dados do arquivo csv e envia para a função criada anteriormente que salva uma pessoa no banco de dados. (Lembrando que você pode encontrar o projeto da última [aula aqui](#).)
- Esta função deve estar dentro do arquivo `leitor.csv`, percorrer todos os dados da `lista_pessoas` e chamar a função criada anteriormente para criar uma pessoa no banco.





# Recapitulando:

- Conhecemos como criar uma conexão entre o banco de dados e a aplicação python
- Criamos funções que enviam os dados para o banco
- Salvamos os dados do nosso arquivo .csv no banco de dados.



# E agora?

Analisar o arquivo CSV que contém os dados de matrícula e responder:

- Os dados encontrados na planilha são referentes a que? Como é o formato do arquivo?

Resposta: Planilha contém dados dos alunos inscritos no curso, possui um cabeçalho e muitas linhas com os registros dos alunos

- Para armazenar estes dados, precisamos criar quantas tabelas?

Resposta: Pessoa, Endereço e Curso

- O que vamos precisar conhecer para conseguir colocar esse arquivo pra dentro da nossa aplicação python?

Resposta: ~~Manipulação Strings, Listas, Dicionários, Manipulação de arquivos, Conectar python com banco de dados,~~ **API**

