



Cento Universitário UNA

Análise de Dados e Big Data

Graduação – TI e Engenharias

Práticas de Laboratório

Diego Augusto de Faria Barros, Tadeu Moreira Perona, Thiérs Hofman do Bom Conselho,
Wesley Dias Maciel

2020/02



Centro Universitário UNA
Graduação – TI e Engenharias
Análise de Dados e Big Data
Prática de Laboratório
Diego Augusto de Faria Barros, Tadeu Moreira Perona, Thiérs Hofman do Bom
Conselho, Wesley Dias Maciel
2020/02

pyodbc



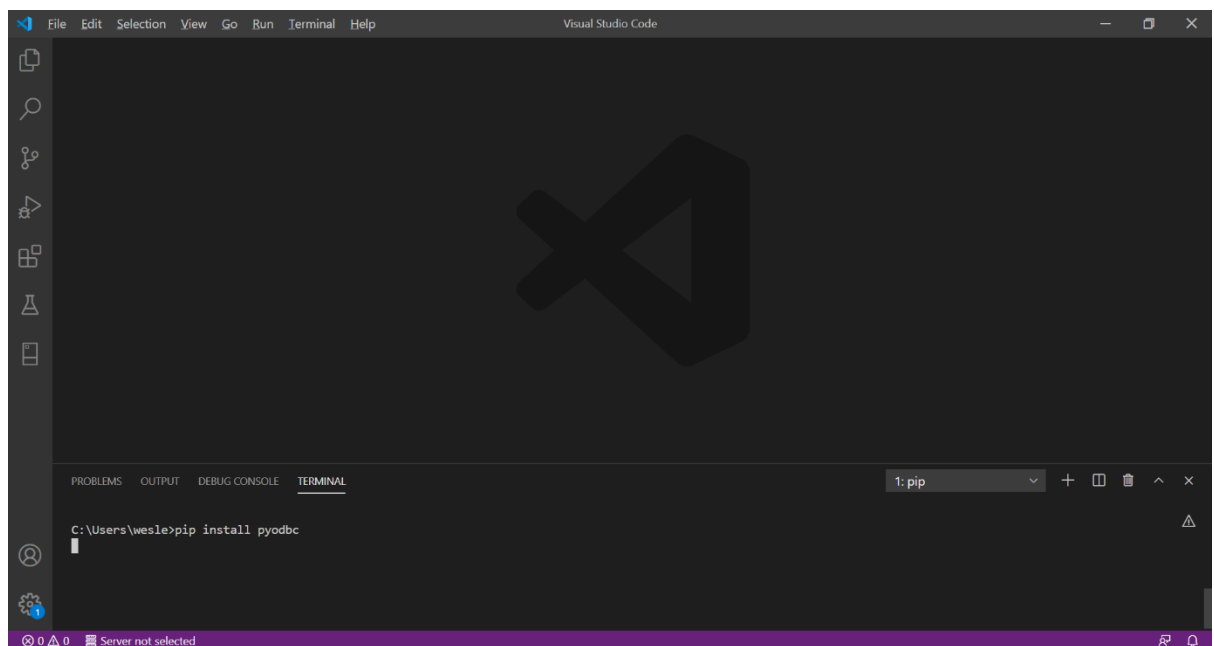
Prática 03

pyodbc

A biblioteca “pyodbc” permite estabelecer comunicação com o sistema gerenciador de banco de dados (SGBD) a partir da linguagem Python.

- 1) Em um terminal do Visual Studio Code, baixe a biblioteca “pyodbc”, executando o comando abaixo.

`pip install pyodbc`

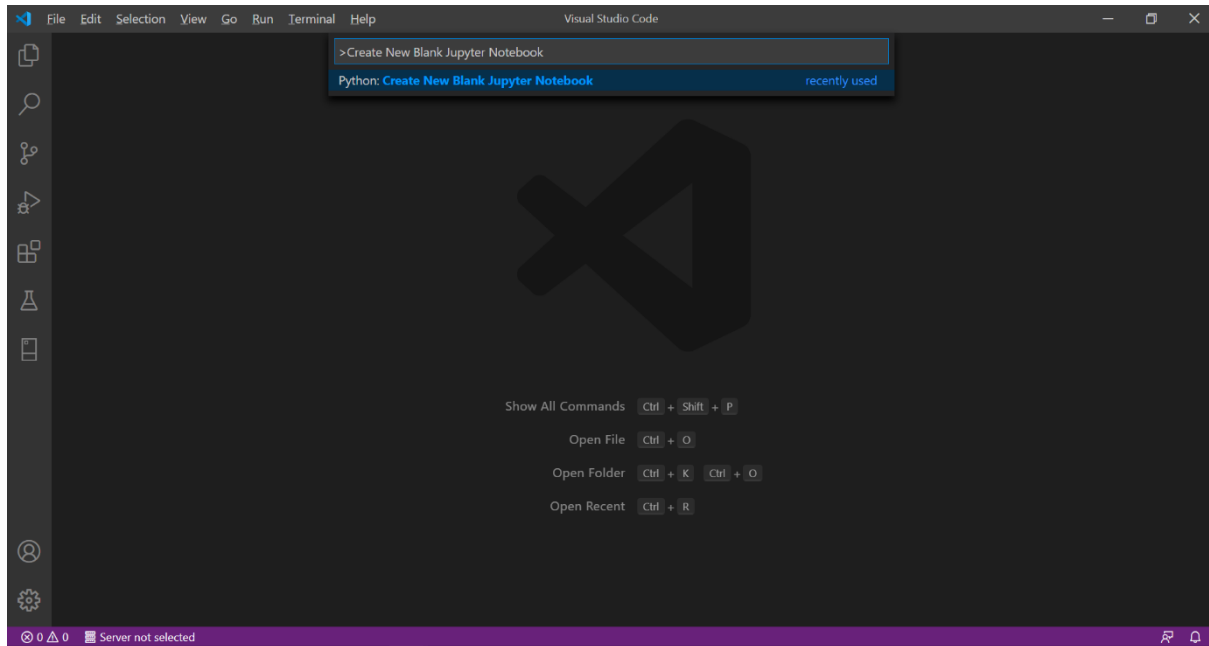


- 2) No Visual Studio Code, crie um novo notebook. Clique em View > Command Palette ou digite Ctrl + Shift + P. Na caixa de entrada, informe:

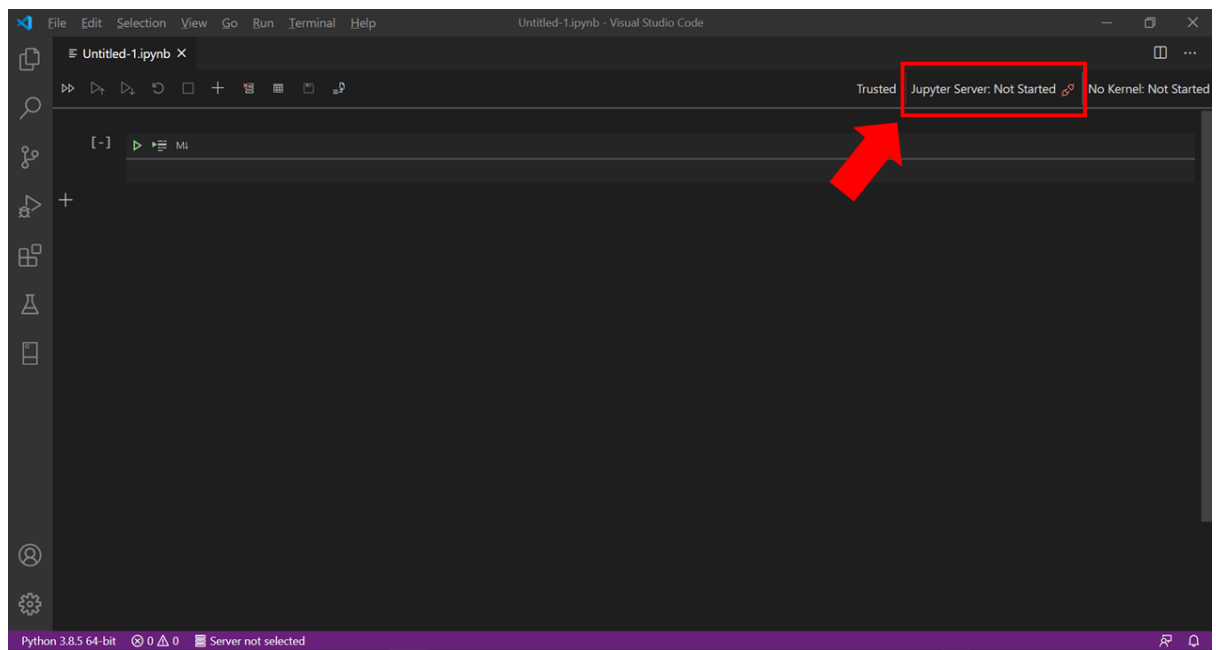
`>Create New Blank Jupyter Notebook`



Centro Universitário UNA
Graduação – TI e Engenharias
Análise de Dados e Big Data
Prática de Laboratório
Diego Augusto de Faria Barros, Tadeu Moreira Perona, Thiérs Hofman do Bom Conselho, Wesley Dias Maciel
2020/02

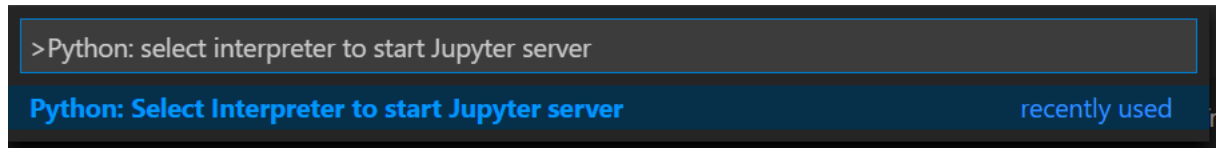


3) Observe se o servidor do Jupyter foi iniciado.

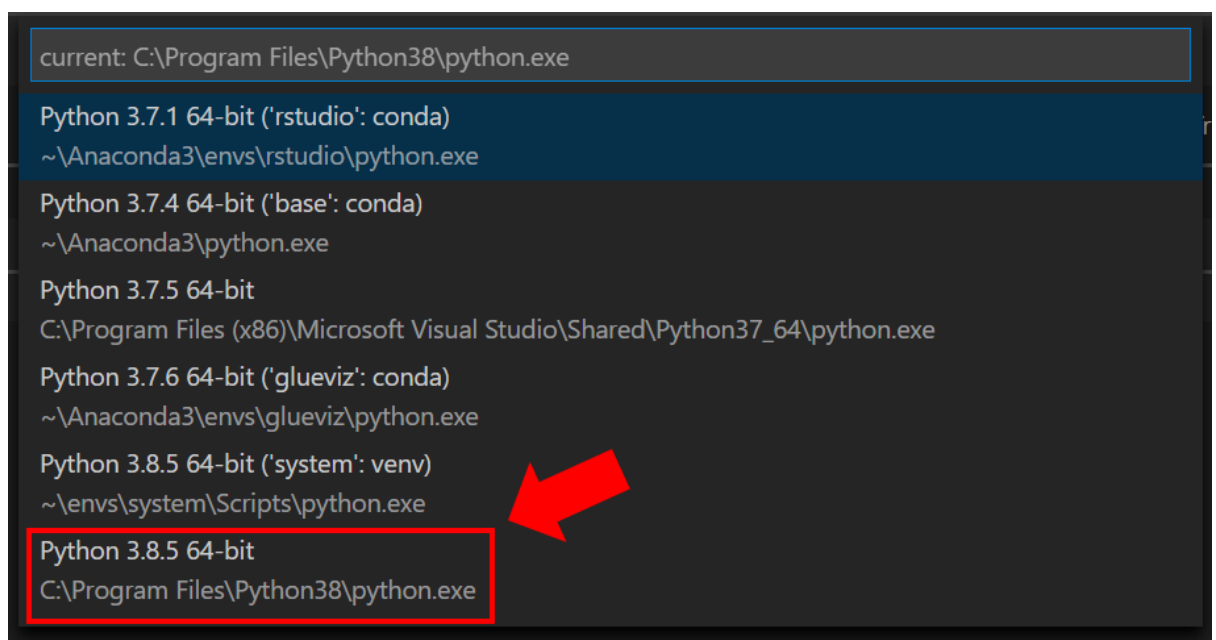


Caso o servidor do Jupyter não tenha sido iniciado, clique em View > Command Palette ou digite Ctrl + Shift + P e Informe:

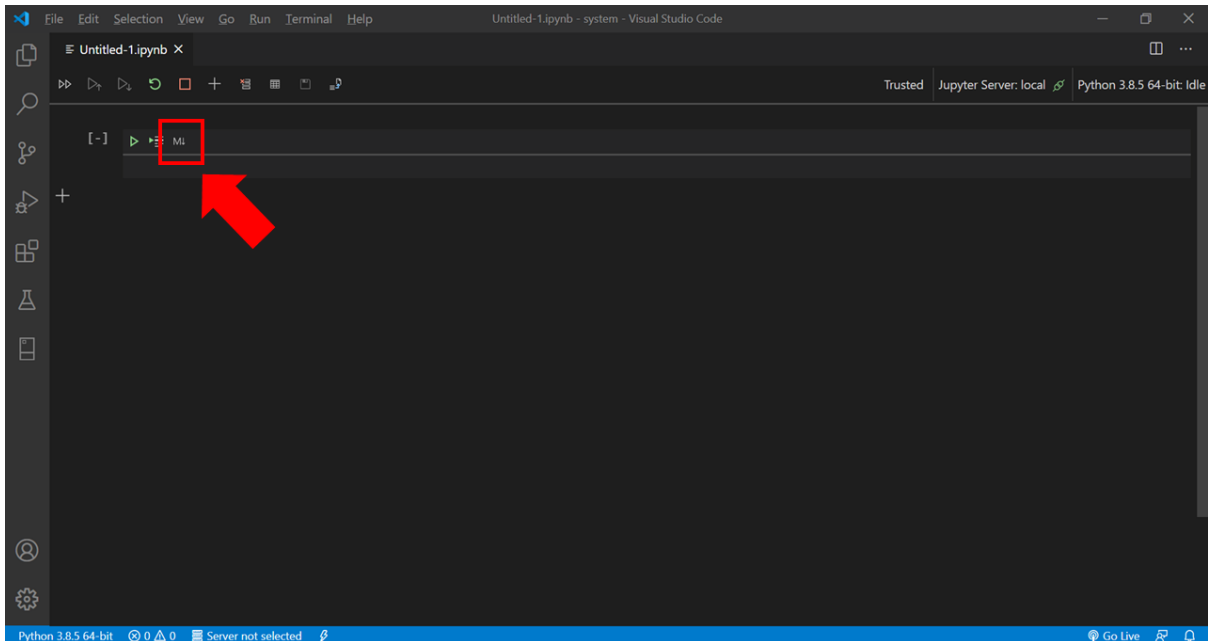
>Python: select interpreter to start Jupyter server



Depois, selecione o interpretador Python que você instalou em sua máquina.

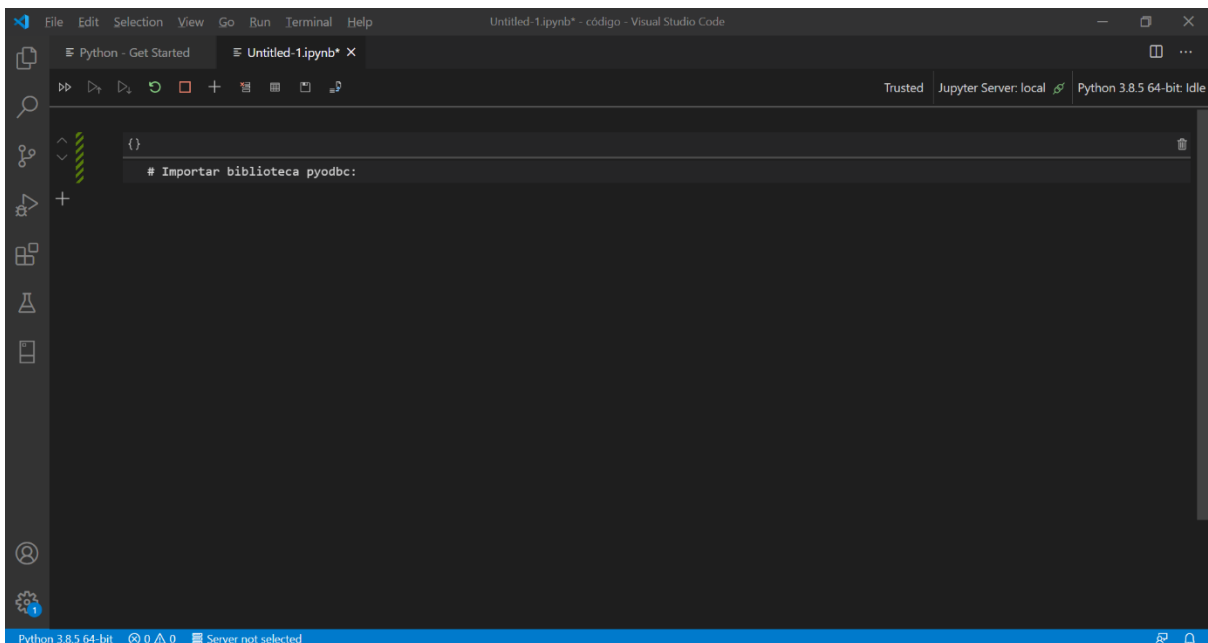


- 4) Clique no ícone da linguagem Markdown para criar uma nova célula para texto.

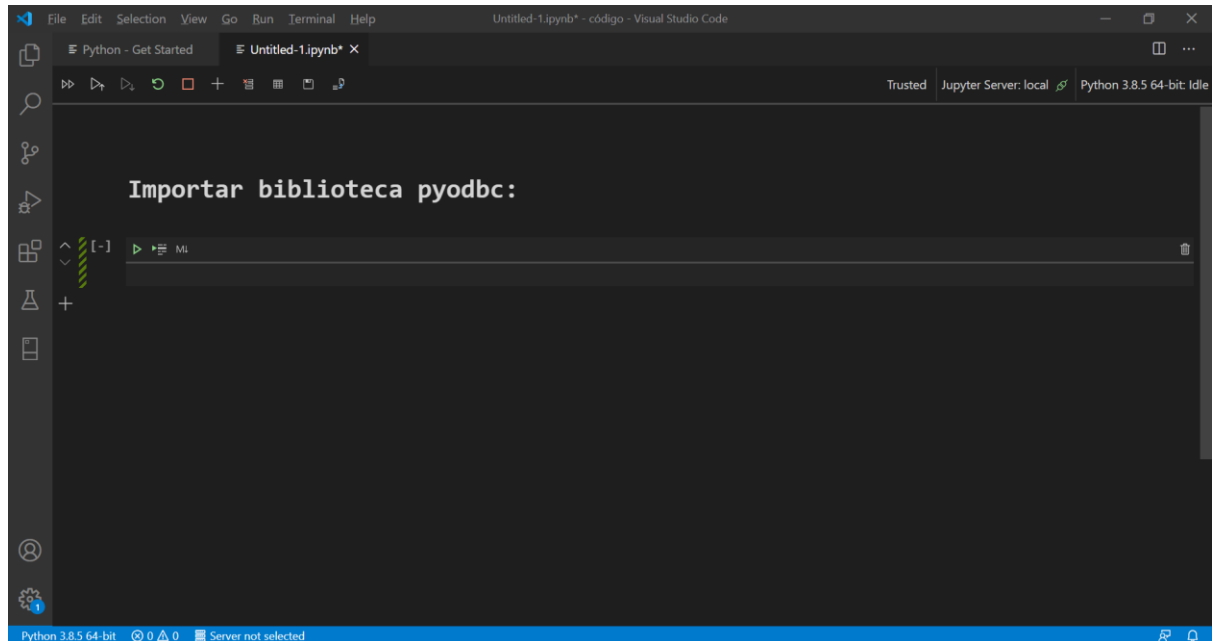


5) Na nova célula, crie o título:

Importar biblioteca pyodbc:

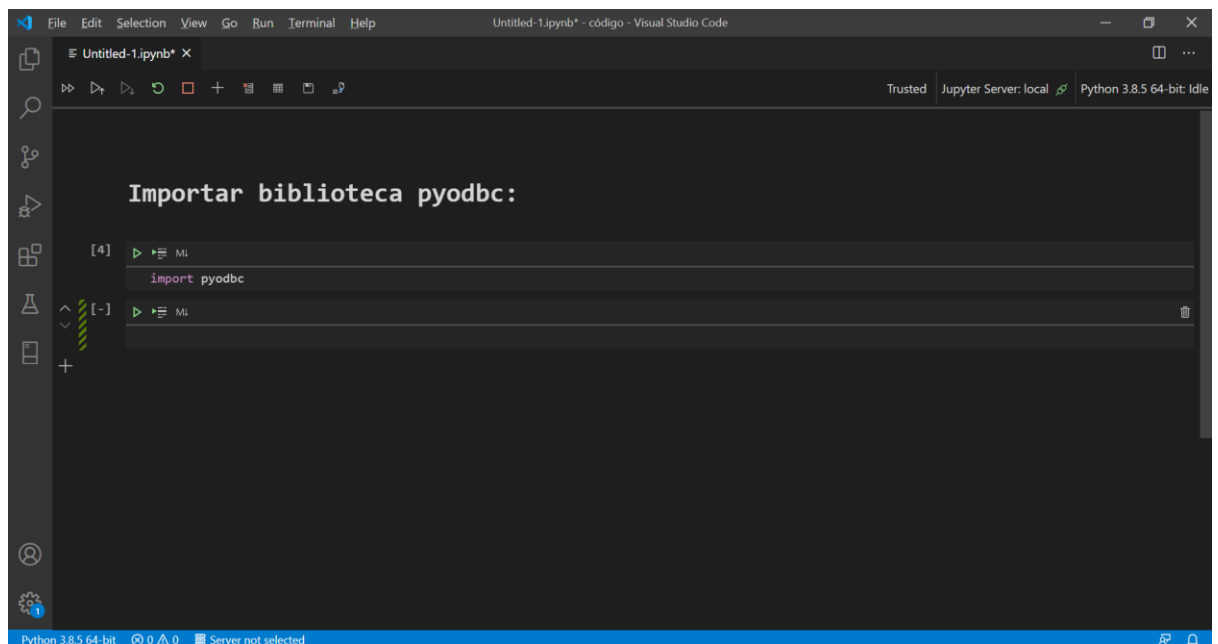


6) Em seguida, clique em Shift + Enter, para executar a código Markdown e já criar uma nova célula de código.



- 7) Na nova célula, informe o código abaixo. Em seguida, clique em Shift + Enter, para executar a importação da biblioteca “pyodbc” e já criar uma nova célula de código.

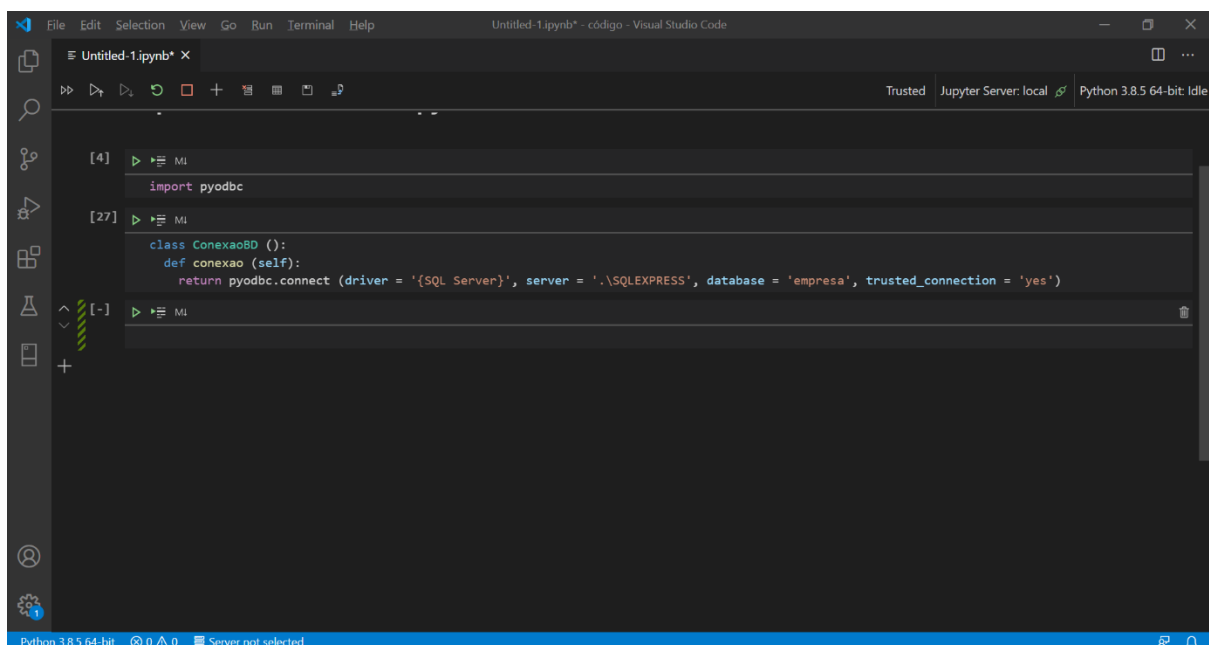
`import pyodbc`





- 8) Na nova célula, defina a classe abaixo para gerar conexões com o banco de dados. Em seguida, clique em Shift + Enter, para executar o código.

```
class ConexaoBD ():  
  
    def conexao (self):  
  
        return pyodbc.connect (driver = '{SQL Server}', server = '.\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')
```



OBS: a função “connect” da biblioteca “pyodbc” possui um parâmetro “server”. O parâmetro “server” recebe o nome do servidor do banco de dados. Por simplificação, nesse exemplo, o nome passado nesse parâmetro foi ‘.\SQLEXPRESS’. Entretanto, você pode passar o nome do servidor diretamente. Exemplo: ‘DESKTOP-AD759B5\SQLEXPRESS’. Nesse caso, usando:

```
class ConexaoBD ():  
  
    def conexao (self):  
  
        return pyodbc.connect (driver = '{SQL Server}', server = 'DESKTOP-AD759B5\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')
```



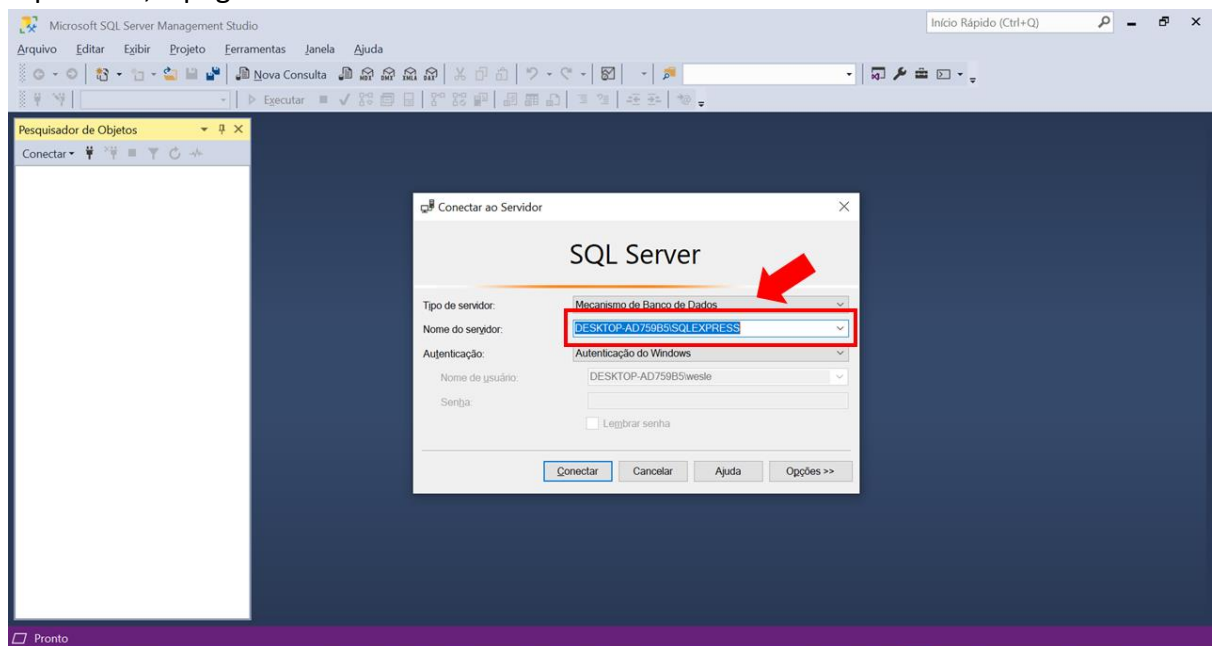

```
[4] ▶ ➤ MI
import pyodbc

[28] ▶ ➤ MI
class ConexaoBD():
    def conexao(self):
        return pyodbc.connect(driver = '{SQL Server}', server = 'DESKTOP-AD759B5\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')

[~] ▶ ➤ MI
```

Há algumas formas de você descobrir o nome do servidor:

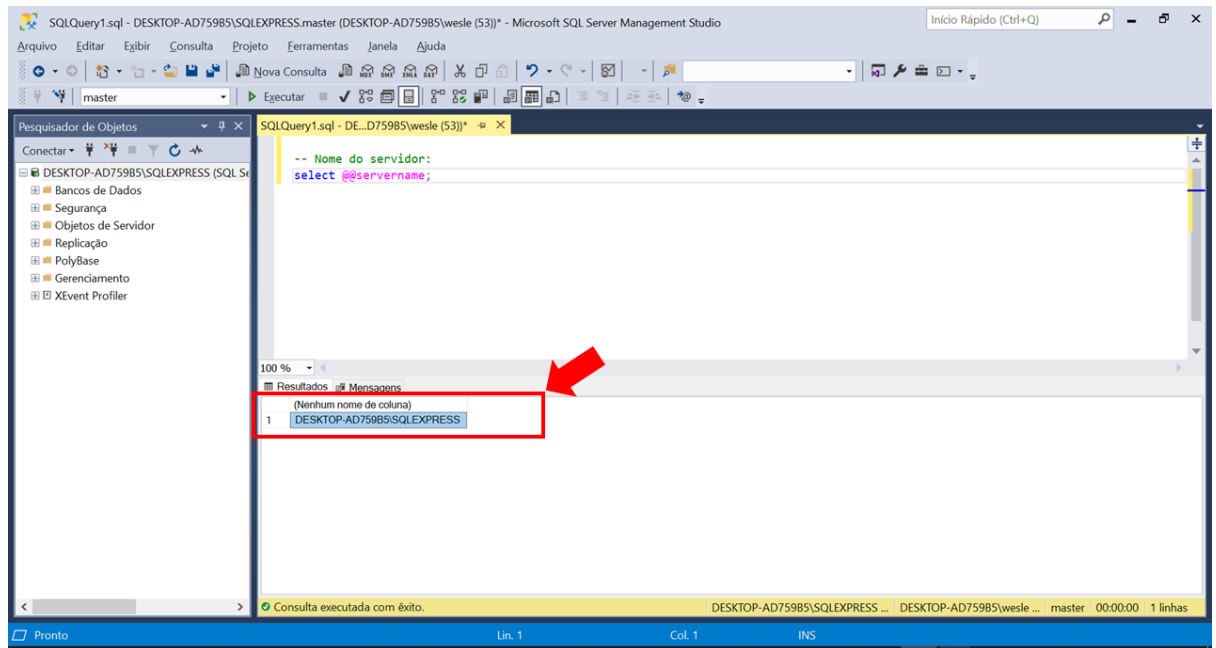
- 1) A primeira, é pegar esse nome na tela de abertura do SSMS:



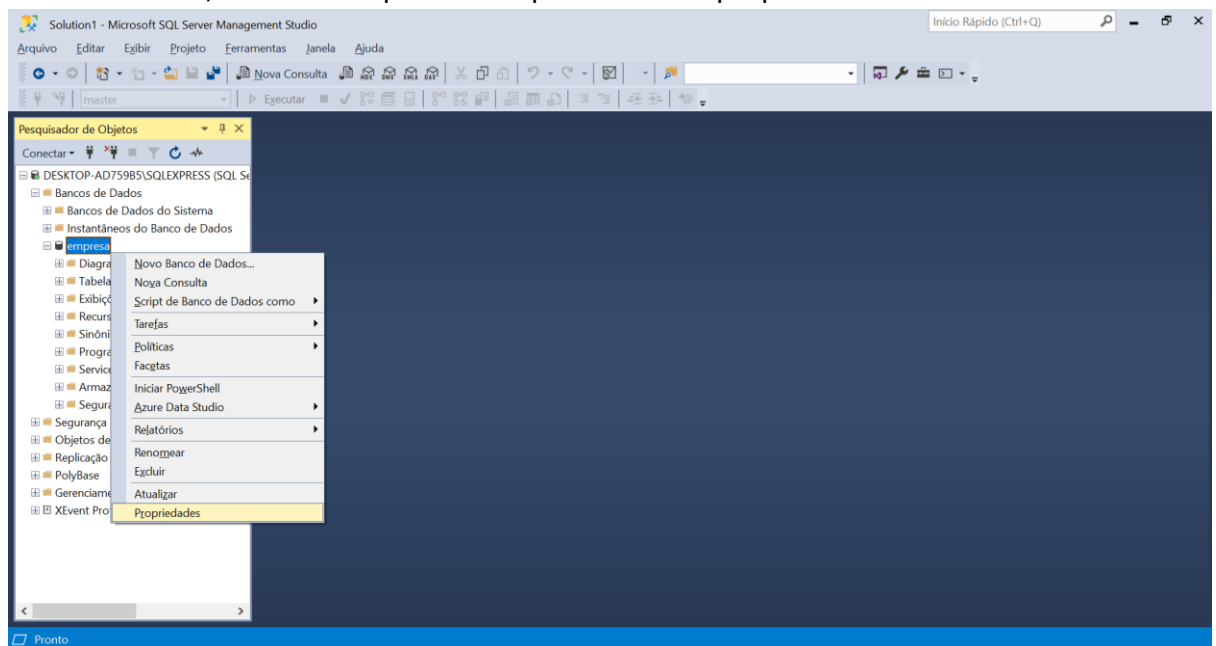
- 2) Uma segunda forma, é executar o comando abaixo no SSMS:



```
-- Nome do servidor:  
select @@servername;
```



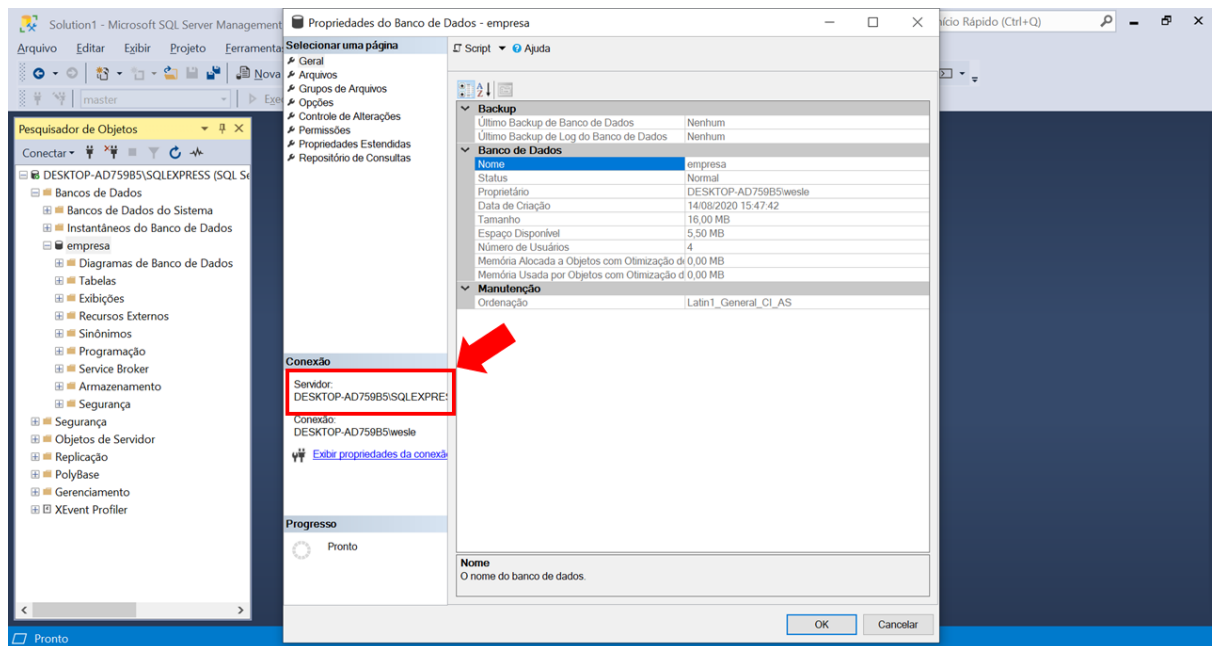
- 3) Você também pode clicar com o botão direito do mouse sobre o nome da base de dados no SSMS, no caso “empresa” e depois clicar em propriedades.



Então, ler o nome do servidor na nova janela aberta.



Centro Universitário UNA
Graduação – TI e Engenharias
Análise de Dados e Big Data
Prática de Laboratório
Diego Augusto de Faria Barros, Tadeu Moreira Perona, Thiérs Hofman do Bom Conselho, Wesley Dias Maciel
2020/02



Você também pode informar `server = 'localhost\SQLEXPRESS'`. Exemplo:

```
class ConexaoBD ():
```

```
    def conexao (self):
```

```
        return pyodbc.connect (driver = '{SQL Server}', server = 'localhost\SQLEXPRESS', database  
                                = 'empresa', trusted_connection = 'yes')
```



```
[4] In [4]: import pyodbc

[44] In [44]: class ConexaoBD():
def conexao(self):
return pyodbc.connect(driver = '{SQL Server}', server = 'localhost\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')

[-] In [-]:
```

- 9) No Visual Studio Code, instancie um objeto da classe “ConexaoBD” e crie uma referência para uma conexão com o banco de dados, usando o código abaixo. Em seguida, clique em Shift + Enter, para executar o código e criar uma nova célula.

`bd = ConexaoBD ()`

`c = bd.conexao ()`

```
[4] In [4]: import pyodbc

[30] In [30]: class ConexaoBD():
def conexao(self):
return pyodbc.connect(driver = '{SQL Server}', server = '.\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')

[32] In [32]: bd = ConexaoBD ()
c = bd.conexao ()

[-] In [-]:
```



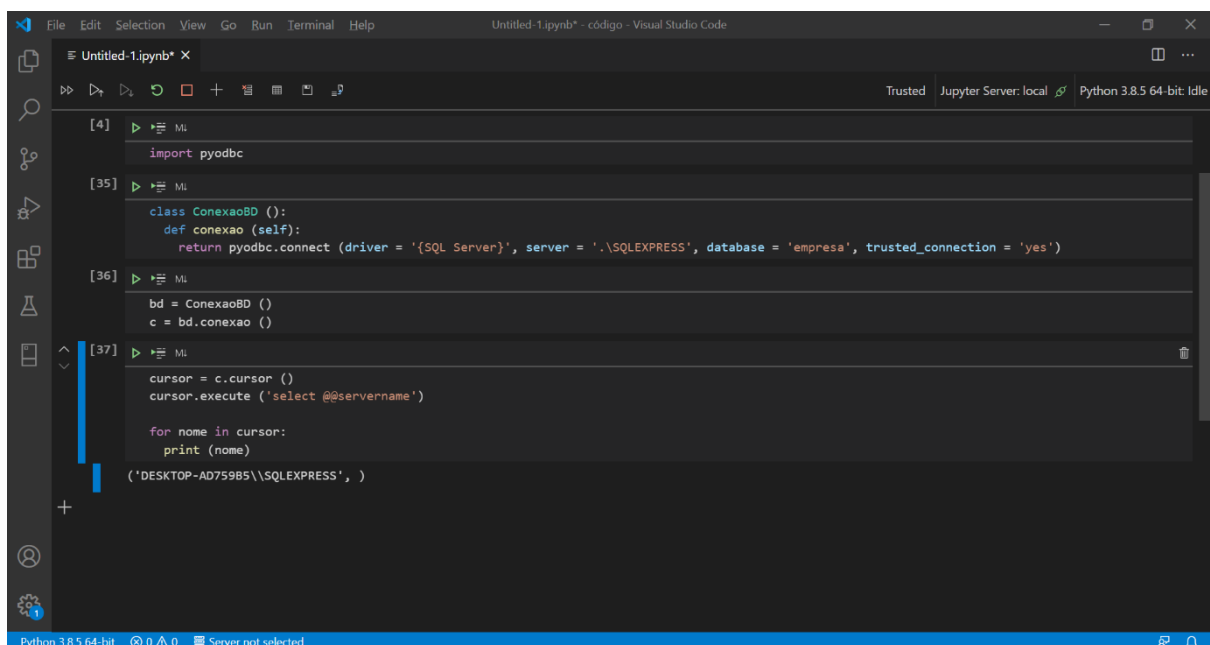
10) Consulte o nome do servidor, empregando o código abaixo:

```
cursor = c.cursor ()
```

```
cursor.execute ('select @@servername')
```

```
for nome in cursor:
```

```
    print (nome)
```



```
[4] In [4]: In: MI
import pyodbc

[35] In [35]: In: MI
class ConexaoBD ():
    def conexao (self):
        return pyodbc.connect (driver = '{SQL Server}', server = '.\\SQLEXPRESS', database = 'empresa', trusted_connection = 'yes')

[36] In [36]: In: MI
bd = ConexaoBD ()
c = bd.conexao ()

[37] In [37]: In: MI
cursor = c.cursor ()
cursor.execute ('select @@servername')

for nome in cursor:
    print (nome)

('DESKTOP-AD759B5\\SQLEXPRESS', )
```

OBS: um cursor é um objeto que permite percorrer os registros em um banco de dados.