FIAP GRADUAÇÃO

TBD

Database Application Development
Conexão Oracle

PROFA. PATRICIA ANGELINI profpatricia.angelini@fiap.com.br

Conexão com Banco Oracle

Conexão com Banco de Dados

Porque conectar um código Python com um Banco de Dados?

1. **Armazenamento de Dados**: Bancos de dados são projetados para armazenar dados de forma estruturada e persistente. Conectar o Python a um banco de dados permite que você armazene informações importantes de forma organizada e segura.
2. **Recuperação de Dados**: Você pode usar consultas SQL (Structured Query Language) para recuperar dados específicos de um banco de dados. Isso é essencial para extrair informações relevantes quando necessário.
3. **Manipulação de Dados**: Com o Python conectado a um banco de dados, você pode executar operações de criação, atualização e exclusão de registros. Isso é crucial para manter seus dados atualizados e corrigir erros quando necessário.
4. **Escalabilidade**: Bancos de dados são escaláveis e podem lidar com grandes volumes de dados. Isso permite que seu aplicativo ou sistema cresça à medida que mais informações são armazenadas.
5. **Consistência e Integridade**: Bancos de dados oferecem mecanismos para garantir a consistência e integridade dos dados, por meio de restrições, chaves primárias, chaves estrangeiras e outras regras. Isso ajuda a manter a qualidade dos dados.

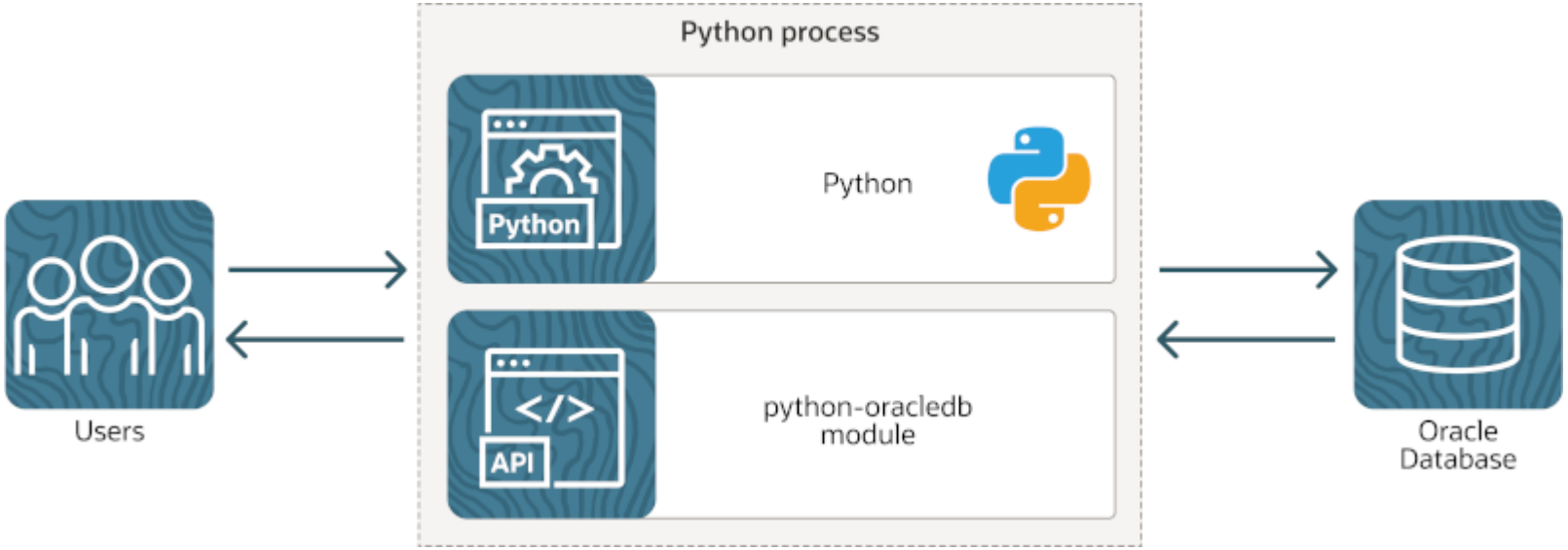
Conexão com Banco de Dados

Porque conectar um código Python com um Banco de Dados

1. **Segurança**: Bancos de dados geralmente têm recursos de segurança incorporados para proteger os dados contra acesso não autorizado. Isso é vital para proteger informações confidenciais.
2. **Concorrência**: Quando várias partes do seu aplicativo precisam acessar os mesmos dados simultaneamente, um banco de dados pode gerenciar isso de forma eficaz, garantindo que os dados não sejam corrompidos ou perdidos.
3. **Backup e Recuperação**: Bancos de dados oferecem recursos para fazer backup dos dados regularmente e recuperá-los em caso de falha do sistema. Isso ajuda a evitar a perda irreparável de informações valiosas.
4. **Análise de Dados**: Conectar o Python a um banco de dados permite que você execute análises de dados avançadas e gere insights valiosos a partir dos dados armazenados.
5. **Integração com Aplicações**: Muitas aplicações e sistemas dependem da integração com bancos de dados para funcionar corretamente. Isso inclui aplicativos da web, sistemas de gerenciamento de conteúdo, sistemas de CRM (Customer Relationship Management), entre outros.
6. **Histórico e Auditoria**: Bancos de dados podem registrar todas as operações realizadas nos dados, criando um histórico que pode ser útil para auditorias, rastreamento de alterações e conformidade regulatória.

Python & Oracle

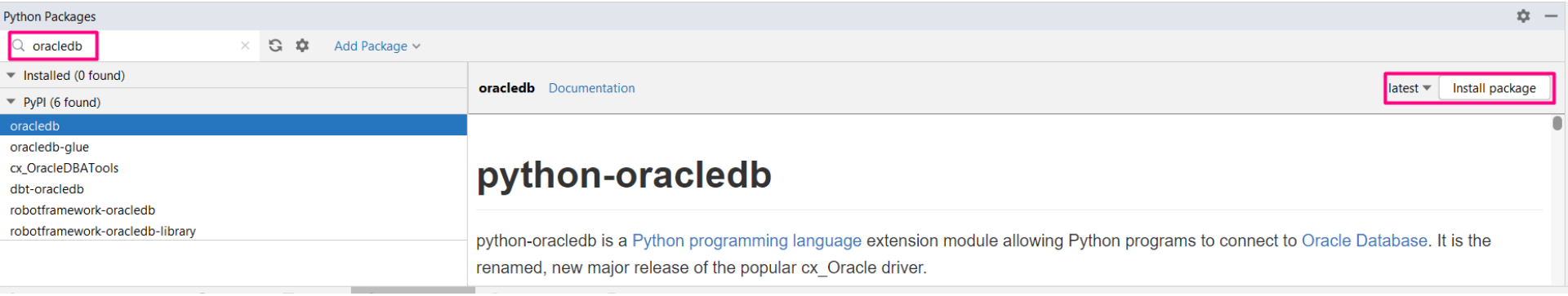
Python e Oracle



- Para o Python falar com um banco de dados você vai precisar de uma biblioteca

- O Oracle Database usa um protocolo de comunicação proprietário e específico para interagir com aplicativos. Uma biblioteca é projetada para entender e implementar esse protocolo, permitindo que o Python se comunique de forma eficaz com o Oracle.
- A biblioteca mais atualizada para conexão com o Oracle é o **oracledb**

Instalando a biblioteca oracledb



- O primeiro passo é garantir que a biblioteca esteja instalada. Dependendo da IDE utilizada você pode pesquisar se ela já está instalada e se não estiver pode usar o Install Package

```
PS D:\Projetos\ConexaoOracle> pip install oracledb
Collecting oracledb
  WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after connection broken by 'ReadTimeoutError("HTTPSConnec
tionPool(host='files.pythonhosted.org', port=443): Read timed out. (read timeout=15)")': /packages/50/74/0fc5d8a94dc8eb5466a9ce7c34058e210b672c6183
5d075fe670307dd622/oracledb-1.4.0-cp311-cp311-win_amd64.whl
  Downloading oracledb-1.4.0-cp311-cp311-win_amd64.whl (1.3 MB)
     ---------------------------------------- 1.3/1.3 MB 1.1 MB/s eta 0:00:00
```

- Ou dentro do Terminal pode usar o comando **pip install oracledb**

Estabelecendo a conexão

Estabelecendo a conexão com o banco de dados

```python
import oracledb
try:
    # Conectar ao banco de dados
    # ================================================================================================================ #
    # CUIDADO!!! ==>>> COM 3 TENTATIVAS ERRADAS SUA CONTA FICA EM LOCK (Solicitar ao Help Desk via Whats para desbloquear)
    # ================================================================================================================ #

    conn = oracledb.connect(user="seu usuario", password="sua senha", dsn="oracle.fiap.com.br:1521/orcl")
```

- Importe a biblioteca em seu script Python **import oracledb**
- Você deve criar uma conexão com o banco de dados Oracle usando as informações apropriadas, como o nome do servidor, nome de usuário, senha e o serviço do banco de dados

conn = oracledb.connect(user="seu usuario", password="sua senha", dsn="host:port:SID")

Executando comandos Oracle no Python

Funções com Parâmetros

```
with conn.cursor() as c_consulta:
    c_consulta.execute("select * from T_PY_ALUNO")
    ldados = list(c_consulta.fetchall())
```

- Um cursor é usado para executar comandos SQL e recuperar resultados. Você pode criar um cursor da seguinte forma: **cursor = connection.cursor()**;
- O cursor dever ficar aberto durante toda execução de comandos e recuperação de resultados. Por isso podemos usar o statement **with** como vemos aqui: **with conn.cursor() as c_cursor:**
- E podemos usar o cursor para executar comandos e consultas no banco de dados: **c_cursor.execute("sua consulta")**
- Você pode recuperar os resultados das consultas usando métodos como **fetchone()** ou **fetchall()**
- É importante fechar a conexão quando você terminar de trabalhar com o banco de dados **conn.close()** (só é necessário se você não usar o with)

FIAP

EXERCÍCIOS

EXERCÍCIO DIRIGIDO

Escreva um programa que leia a tabela T_PY_ALUNO e

1. Recupere todos os dados
2. Insira um novo aluno
3. Recupere um aluno
4. Altere os dados de um aluno
5. Exclua um aluno

REFERÊNCIAS

- OLIVEIRA, Jayr Figueiredo de; MANZANO, José Augusto N. G. **Algoritmos: Lógica para Desenvolvimento de Programação de Computadores.** 23ª Edição. São Paulo: Érica, 2010.
- MIZRAHI, Victorine Viviane. **Treinamento em Linguagem C**. 2ª Edição. São Paulo: Pearson, 2008.