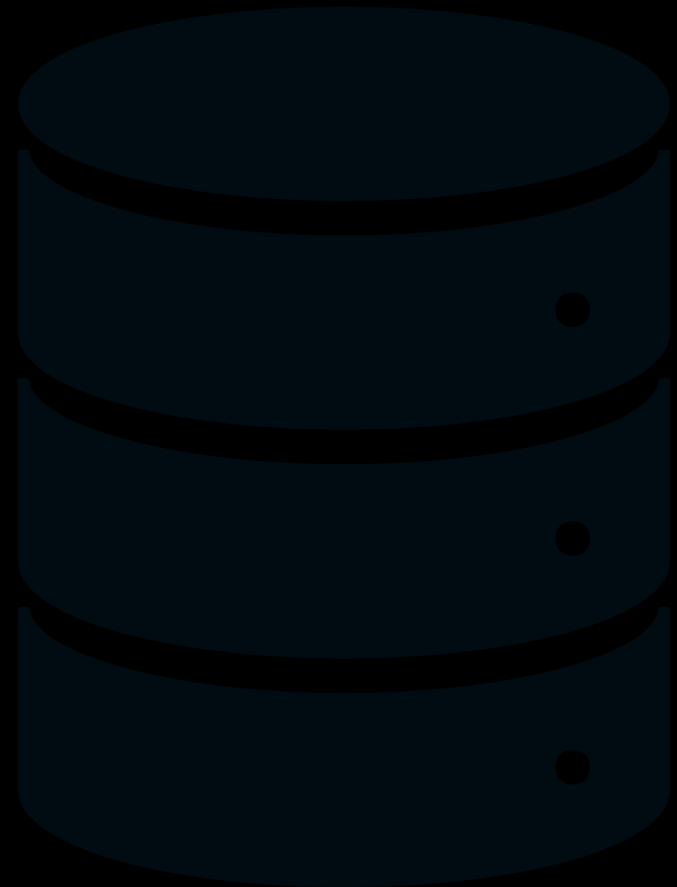


FIAP

# Tutorial para conectar GitHub ao Oracle SQL Developer

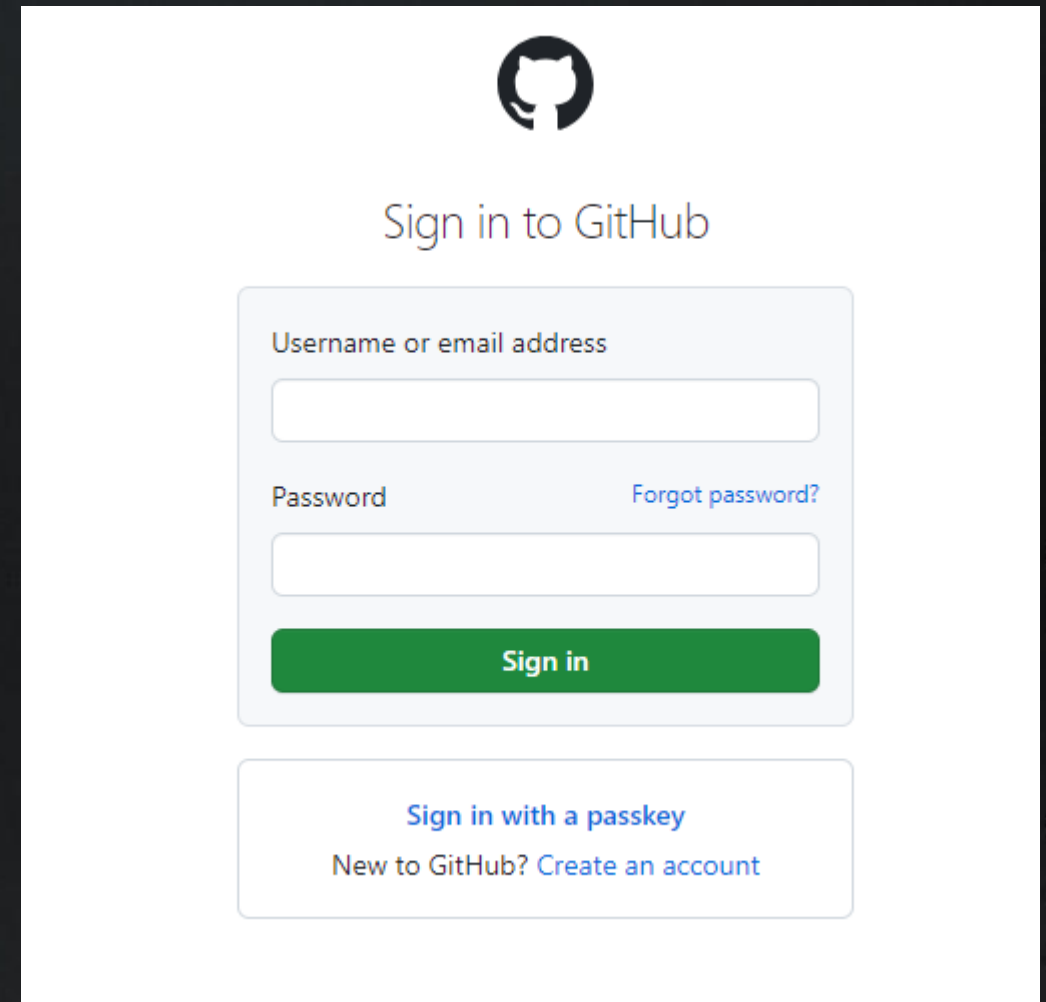
Pré requisitos:

- Ter uma conta o GitHub
- Criar um diretório local
- Criar um repositório Público no GitHub
- Ter Oracle SQL Developer Instalado localmente
- Gerar um token válido para conexão de API no GitHub



Para que o Oracle Developer possa conectar no GitHub e ter permissão de realizar commit e push é necessário criar um token, então nosso primeiro passo é gerar o token no Git.

Logue com seu usuário e senha no site: <https://github.com/login>



The image shows the GitHub login page. At the top is the GitHub logo (Octocat). Below it is the text "Sign in to GitHub". The main form has two input fields: "Username or email address" and "Password". To the right of the password field is a link "Forgot password?". Below the password field is a green "Sign in" button. At the bottom of the form is a link "Sign in with a passkey". Below that is a link "New to GitHub? Create an account".

Sign in to GitHub

Username or email address

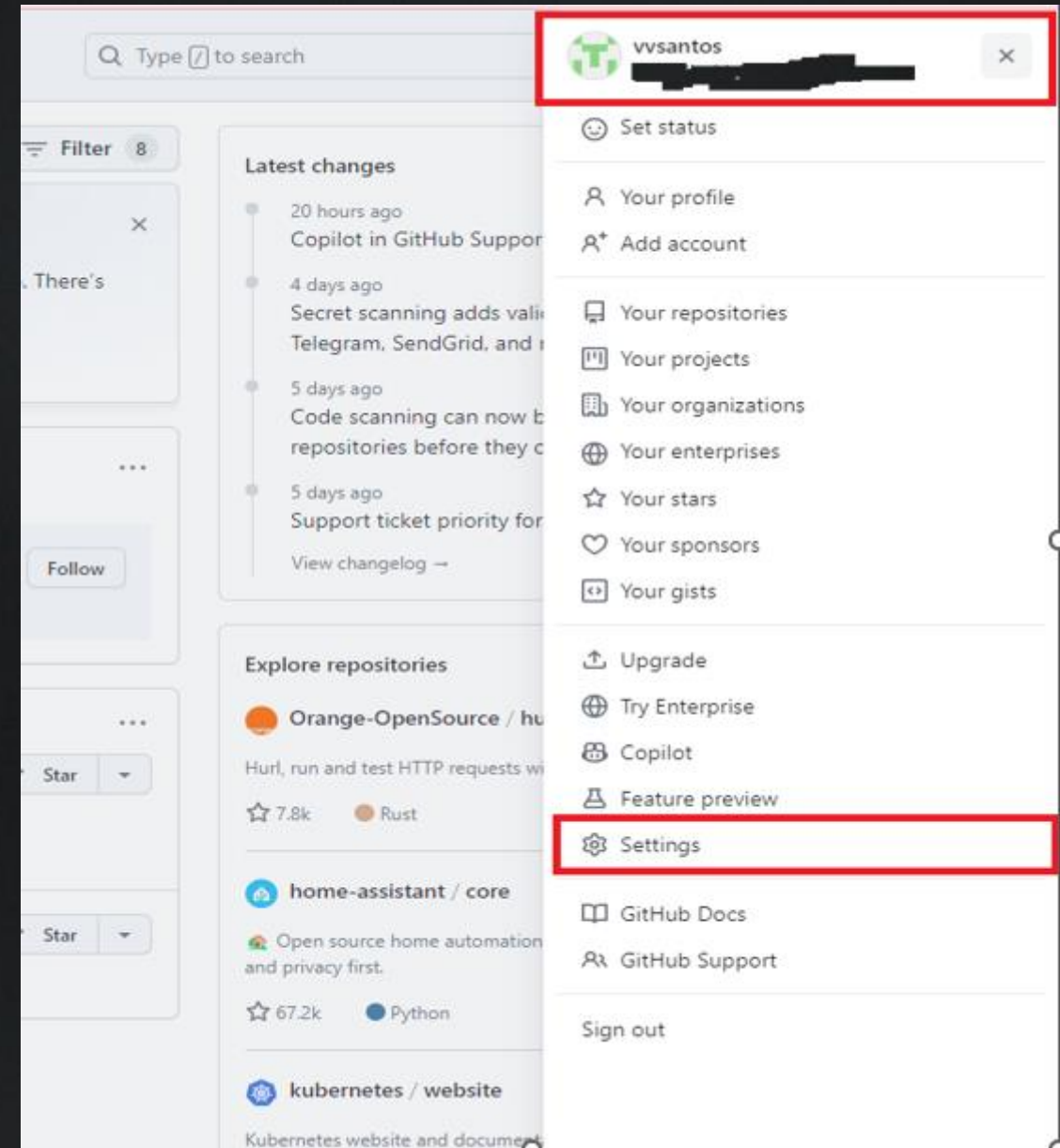
Password [Forgot password?](#)

[Sign in](#)

[Sign in with a passkey](#)

New to GitHub? [Create an account](#)

Clique no ícone do seu nome no canto superior direito e role a barra de rolagem até o final da página e clique em **Settings**



Do lado esquerdo da janela seguinte desça a barra de rolagem e cliquem em:

**<> Developer Settings**

**Moderation**

Code, planning, and automation

Repositories

Codespaces

Packages

Copilot

Pages

Saved replies

Security

Code security and analysis

Integrations

Applications

Scheduled reminders

Archives

Security log

Sponsorship log

**<> Developer settings**

Don't specify

URL

Social accounts

Link to social profile

Link to social profile

Link to social profile

Link to social profile

Company

You can @mention your company's GitHub organization to link it.

Location

Brazil

☒ Display current local time

Other users will see the time difference from their local time.

Time zone

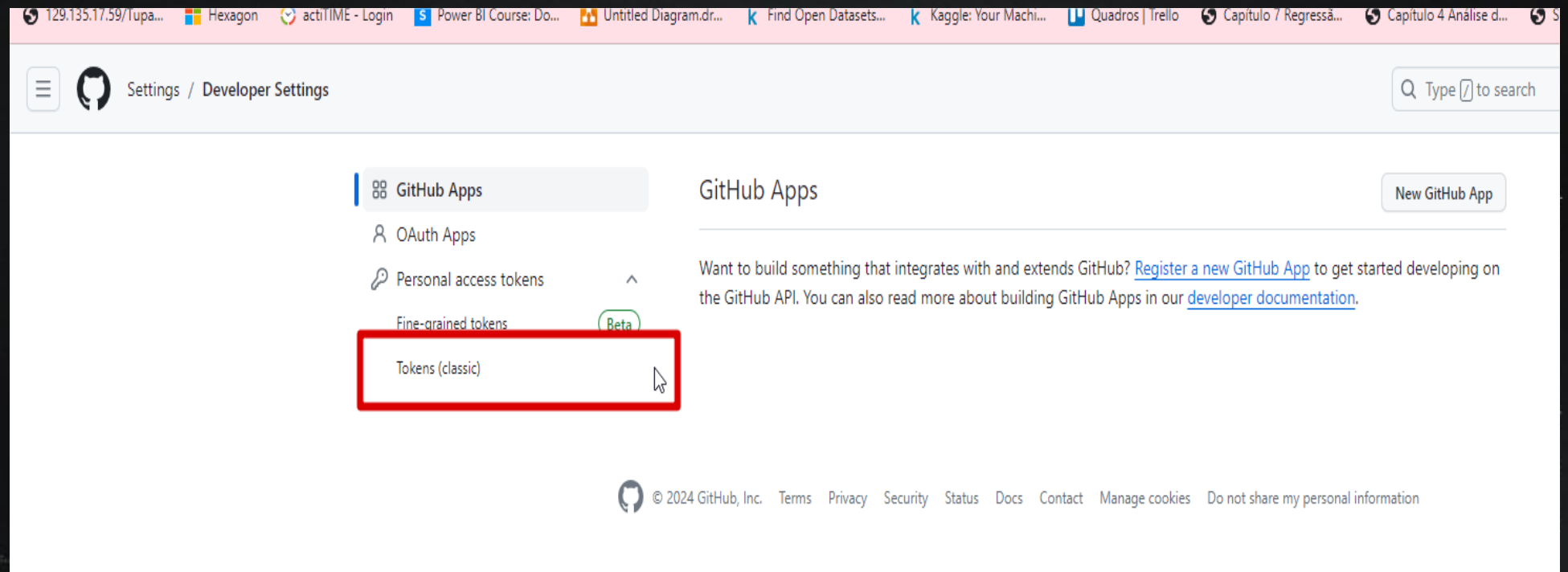
(GMT-03:00) Brasilia

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information.

Update profile

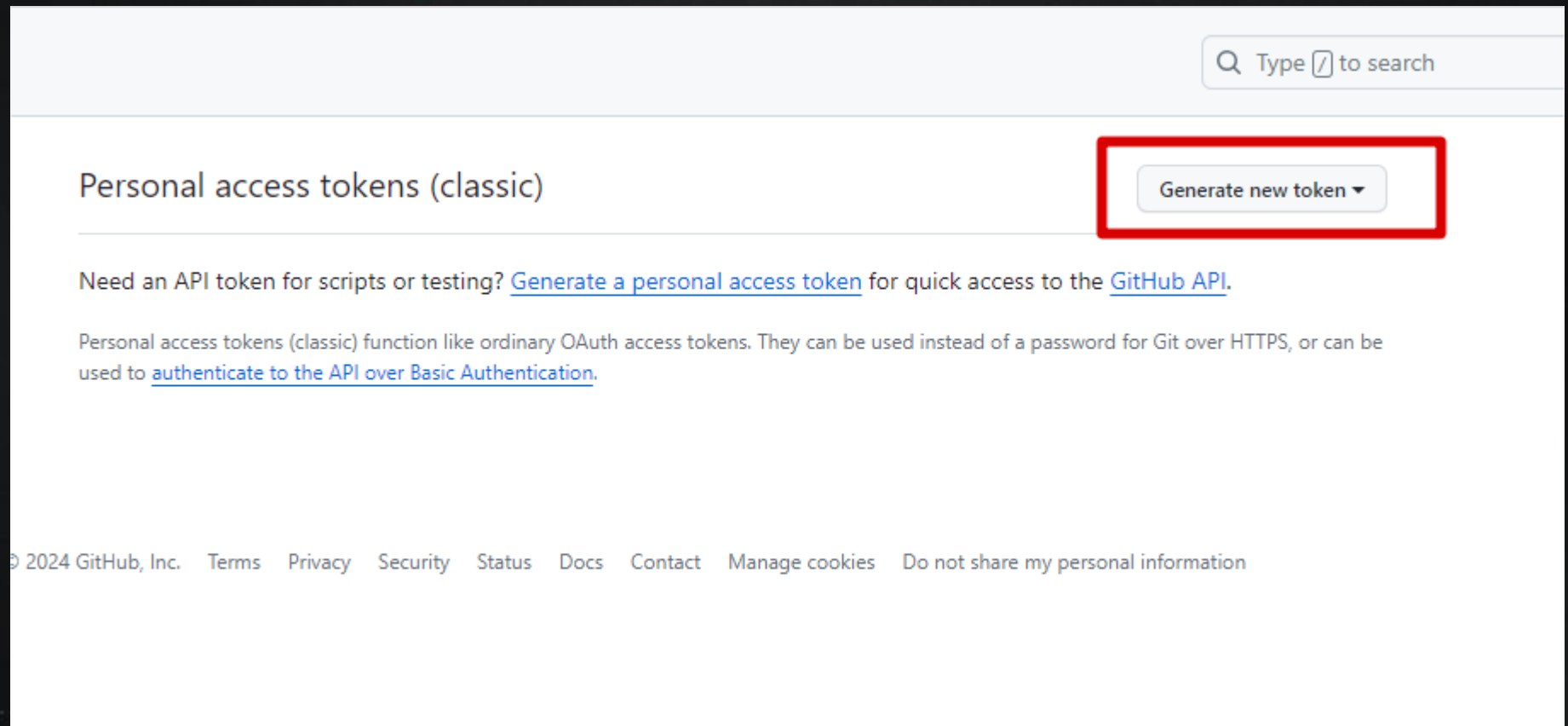
Na janela seguinte clique em:

Token (Classic)

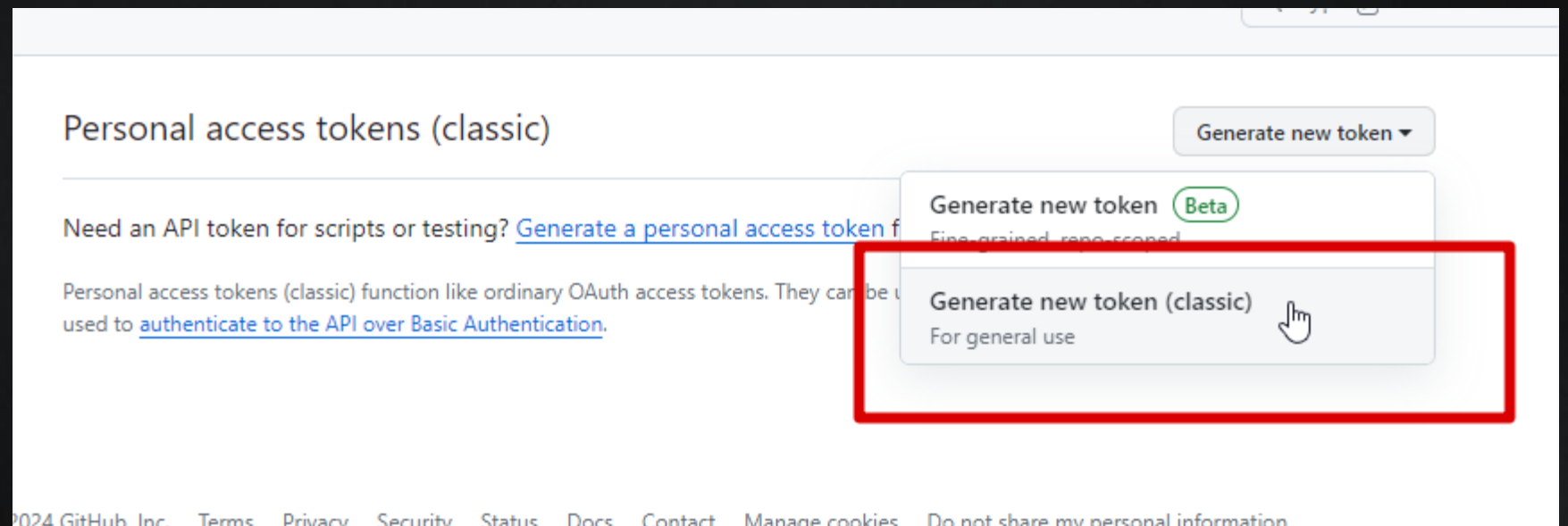


Na aba seguinte clique em:

Generate new token



Escolha a opção de token Clássico conforme a imagem





Escolha um nome para o token, mude a data de expiração para 90 dias ou mais e marque todas as opções de **repo** para o scope

## New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

**Note**

oracle

**Expiration \***

90 days The token will expire on Sat, May 11 2024

**Select scopes**

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

|   |   |
|---|---|
| <input checked="" type="checkbox"/> <b>repo</b>     | Full control of private repositories                                |
| <input checked="" type="checkbox"/> repo:status     | Access commit status  |
| <input checked="" type="checkbox"/> repo_deployment | Access deployment status  |
| <input checked="" type="checkbox"/> public_repo     | Access public repositories  |
| <input checked="" type="checkbox"/> repo:invite     | Access repository invitations                                       |
| <input checked="" type="checkbox"/> security_events | Read and write security events                                      |
| <input type="checkbox"/> workflow                   | Update GitHub Action workflows                                      |
| <input type="checkbox"/> write:packages             | Upload packages to GitHub Package Registry                          |
| <input type="checkbox"/> read:packages              | Download packages from GitHub Package Registry                      |
| <input type="checkbox"/> delete:packages            | Delete packages from GitHub Package Registry                        |
| <input type="checkbox"/> admin:org                  | Full control of orgs and teams, read and write org projects         |
| <input type="checkbox"/> write:org                  | Read and write org and team membership, read and write org projects |

Realizado os passos anteriores  
basta clicar em **Generate token**

|   |   |
|---|---|
| <input type="checkbox"/> read:audit_log         | Read access of audit log                                      |
| <input type="checkbox"/> codespace              | Full control of codespaces                                    |
| <input type="checkbox"/> codespace:secrets      | Ability to create, read, update, and delete codespace secrets |
| <input type="checkbox"/> copilot                | Full control of GitHub Copilot settings and seat assignments  |
| <input type="checkbox"/> manage_billing:copilot | View and edit Copilot Business seat assignments               |
| <input type="checkbox"/> project                | Full control of projects                                      |
| <input type="checkbox"/> read:project           | Read access of projects                                       |
| <input type="checkbox"/> admin:gpg_key          | Full control of public user GPG keys                          |
| <input type="checkbox"/> write:gpg_key          | Write public user GPG keys                                    |
| <input type="checkbox"/> read:gpg_key           | Read public user GPG keys                                     |
| <input type="checkbox"/> admin:ssh_signing_key  | Full control of public user SSH signing keys                  |
| <input type="checkbox"/> write:ssh_signing_key  | Write public user SSH signing keys                            |
| <input type="checkbox"/> read:ssh_signing_key   | Read public user SSH signing keys                             |

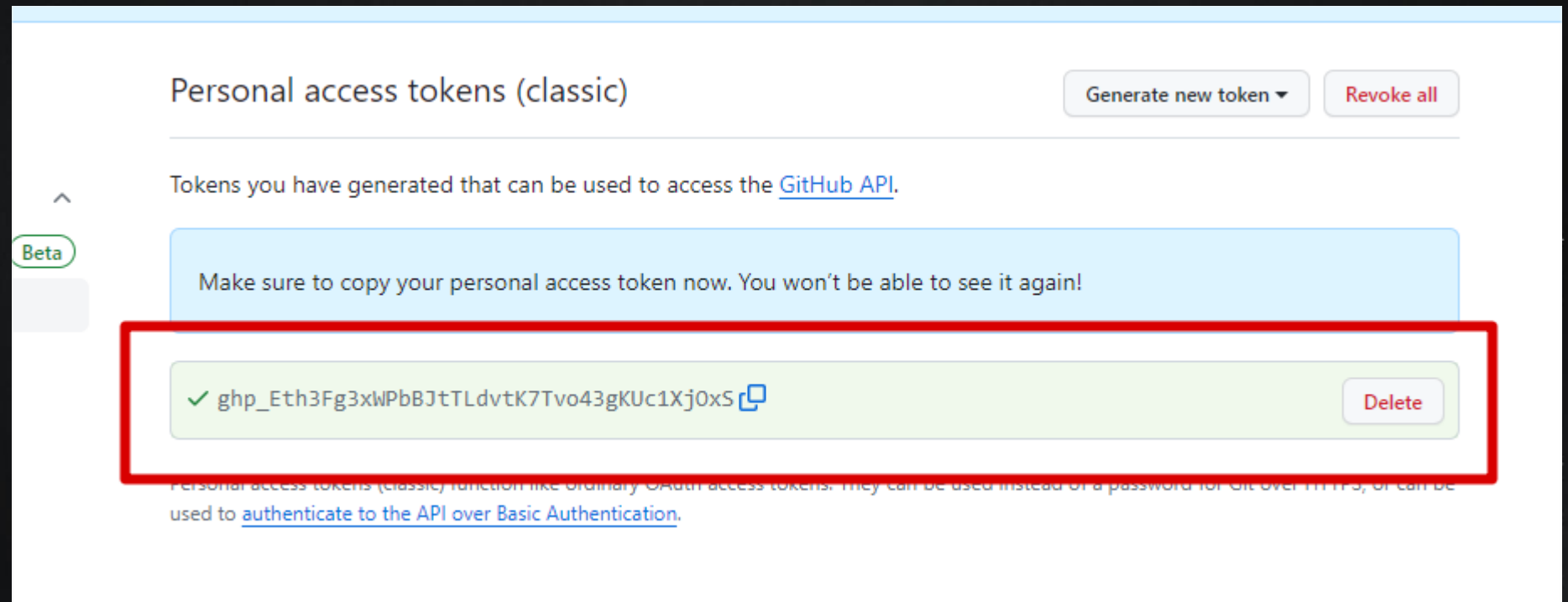
Generate token

Cancel



© 2024 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

Na janela que irá abrir basta copiar a chave gerada e guardar ela, que iremos utilizar em outras vezes.



Volte para página inicial do GitHub  
e crie um novo repositório como  
Público.


## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*


Owner \*

Repository name \*

 vvsantos

 / 

plsql

 plsql is available.

☒  Public

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

☐ Add a README file

.gitignore template: None

License: None

 You are creating a public repository in your personal account.

Create repository

Copie o endereço do repositório criado, pois vamos utilizar ele para clonar as informações no Oracle SQL Developer

The screenshot shows the GitHub repository page for 'plsqli' (Public). At the top, there are buttons for 'Pin', 'Unwatch' (1), 'Fork' (0), and 'Star' (0). Below these are two main cards: 'Set up GitHub Copilot' and 'Add collaborators to this repository'. A red rectangular box highlights the 'Quick setup — if you've done this kind of thing before' section. This section contains a 'Set up in Desktop' button, an 'or' separator, and buttons for 'HTTPS' and 'SSH'. The 'HTTPS' button is selected, showing the repository URL: `https://github.com/vvsantos/plsqli.git`. Below the buttons, it says 'Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).' Below the red box, there is a section titled '...or create a new repository on the command line' with a code block containing the following commands:

```
echo "# plsqli" >> README.md
git init
git add README.md
git commit -m "first commit"
```

Abra o Oracle SQL Developer e crie uma nova conexão com o banco de dados.

New / Select Database Connection

Connection Name Connection Details

Name oracle

Database Type Oracle

User Info Proxy User

Authentication Type Default

Username \*\*\*\*\*

Password \*\*\*\*\*

Role default

Save Password

Connection Type Basic

Details Advanced

Hostname \*\*\*\*\*

Port 1521

☒ SID ord

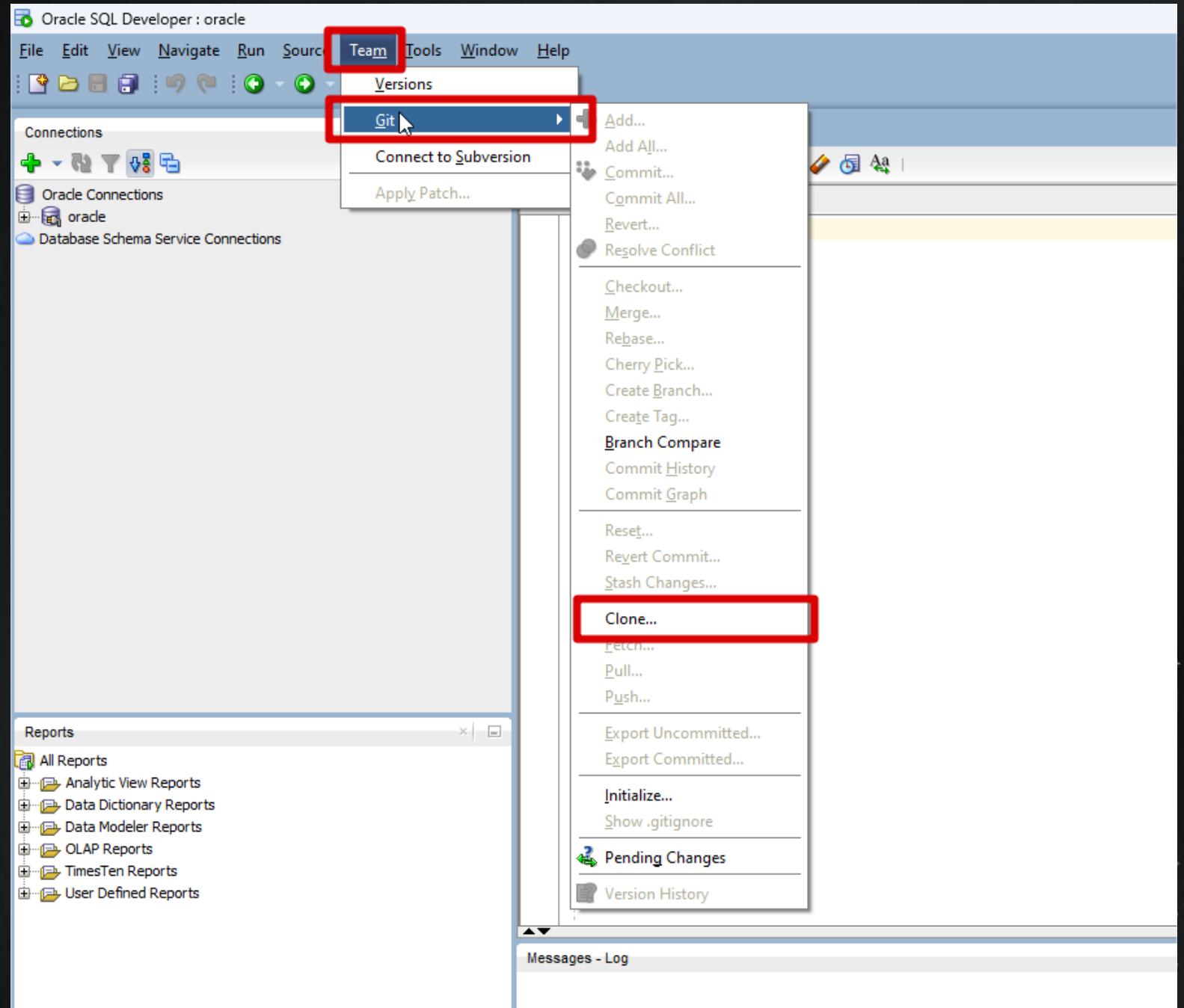
☐ Service name

Status : Success

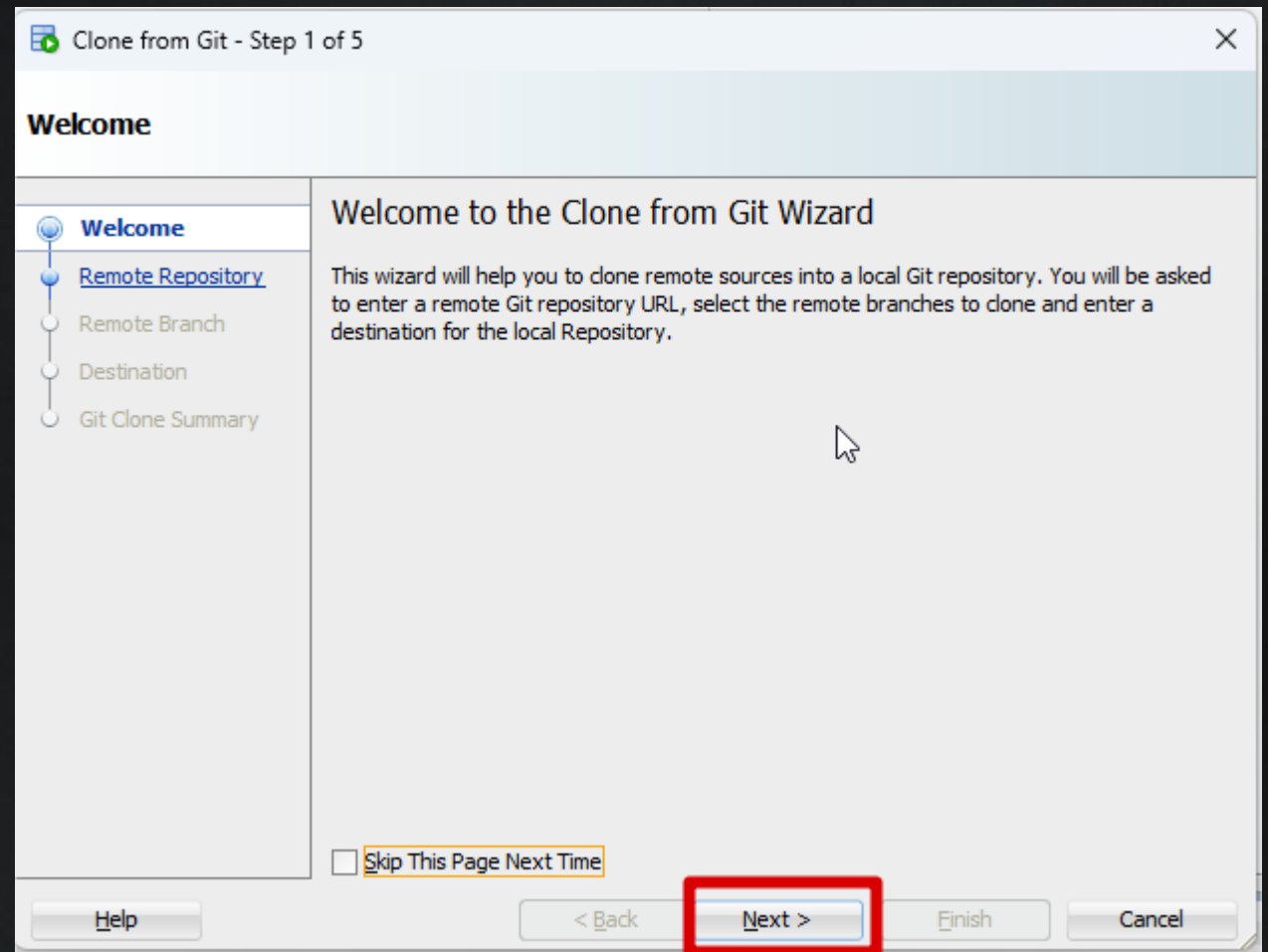
Help Save Clear Test Connect Cancel

Dentro do SQL Developer clique em:

Team > Git > Conect to Git ou Clone



Na janela a seguir basta clicar em  
**Next**





Informe os dados de conexão:

- **Repository URL:** o repositório criado anteriormente
- **User Name:** coloque seu usuário do GitHub
- **Password:** coloque o token gerado (**Não utilizar senha pessoal, irá dar erro de permissão**)
- Clique em **Next**

Clone from Git - Step 2 of 5

### Remote Repository

Select or enter the remote name and repository url. Enter a user name and password or SSH key information if the remote Git Repository does not support anonymous access.

Remote Name:

Repository URL:  [Proxy Settings...](#)

User Name:

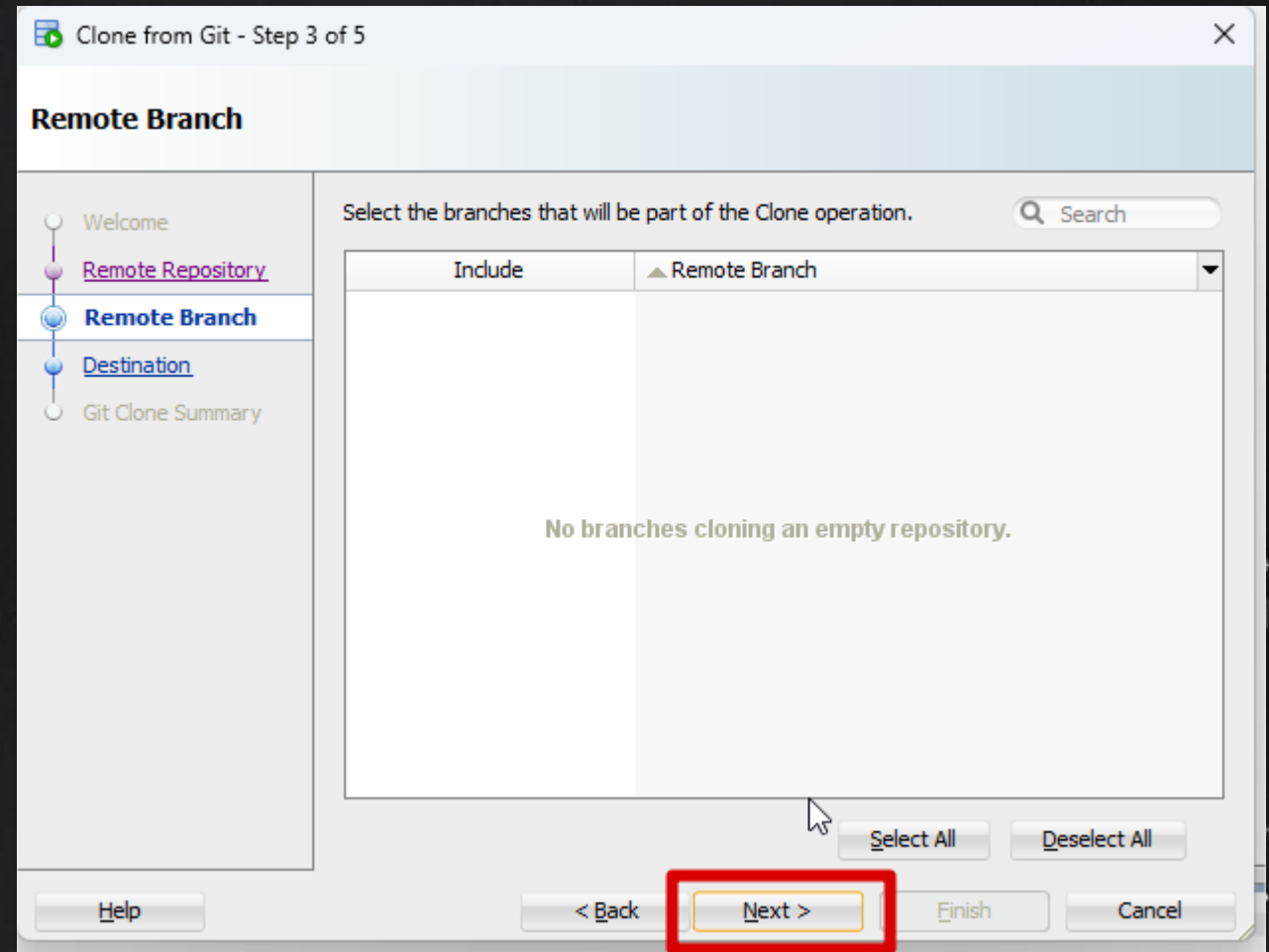
☒ Password:

☐ Private Key File:  [Browse...](#)

Passphrase:

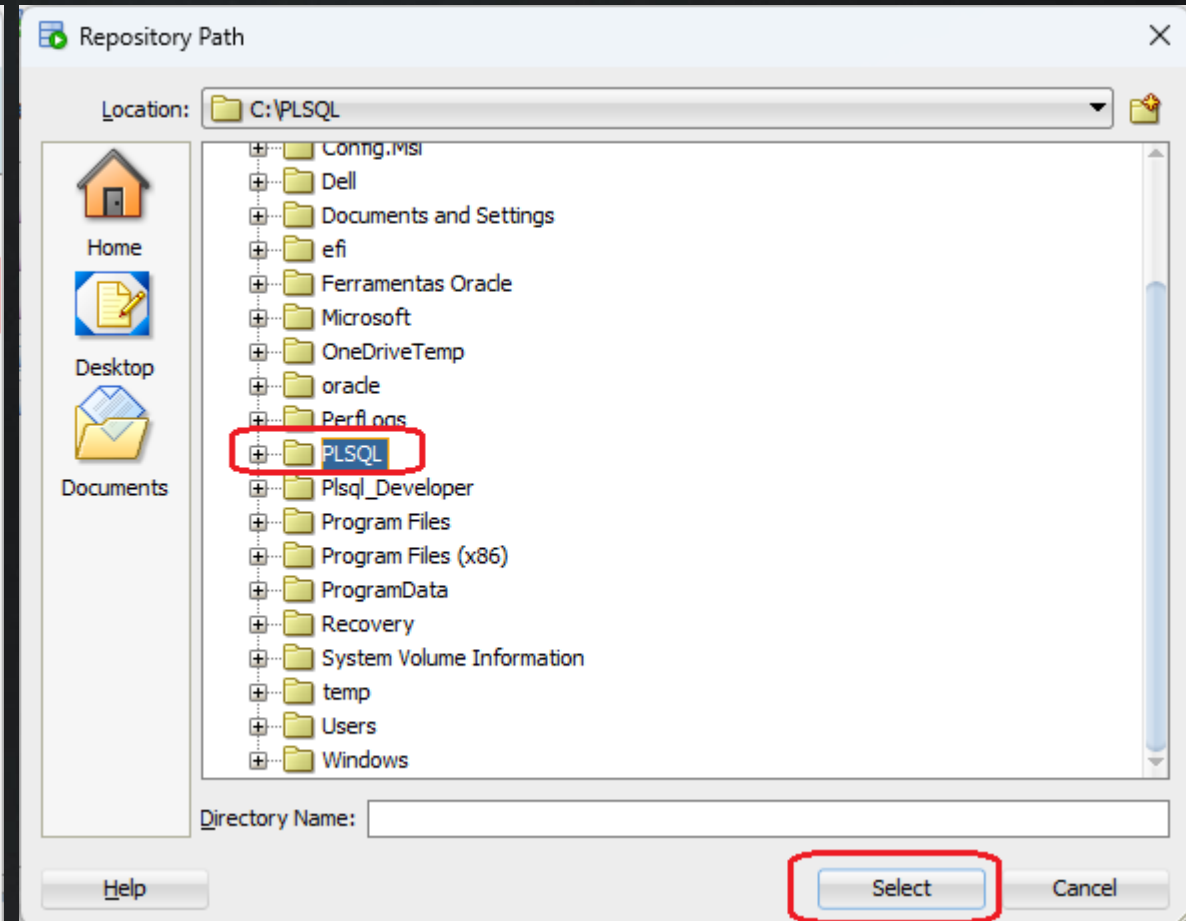
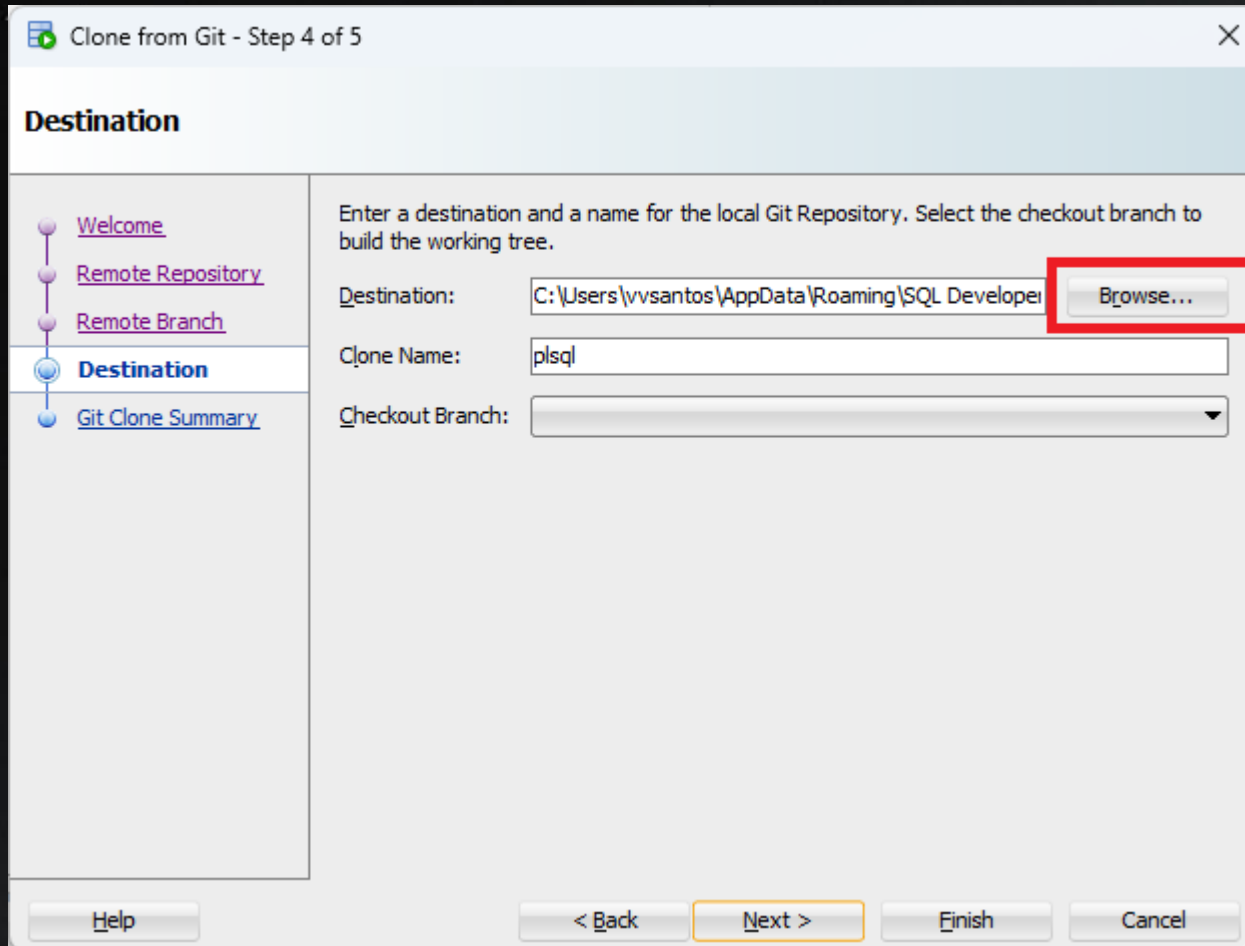
[Help](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

Na janela a seguir basta clicar em  
**Next**



Na janela seguinte clique em **Browse** e escolha uma pasta criada localmente na máquina.

Clique em **select**



Na janela a seguir basta clicar em  
**Next**

Clone from Git - Step 4 of 5

**Destination**

Welcome  
Remote Repository  
Remote Branch  
**Destination**  
Git Clone Summary

Enter a destination and a name for the local Git Repository. Select the checkout branch to build the working tree.

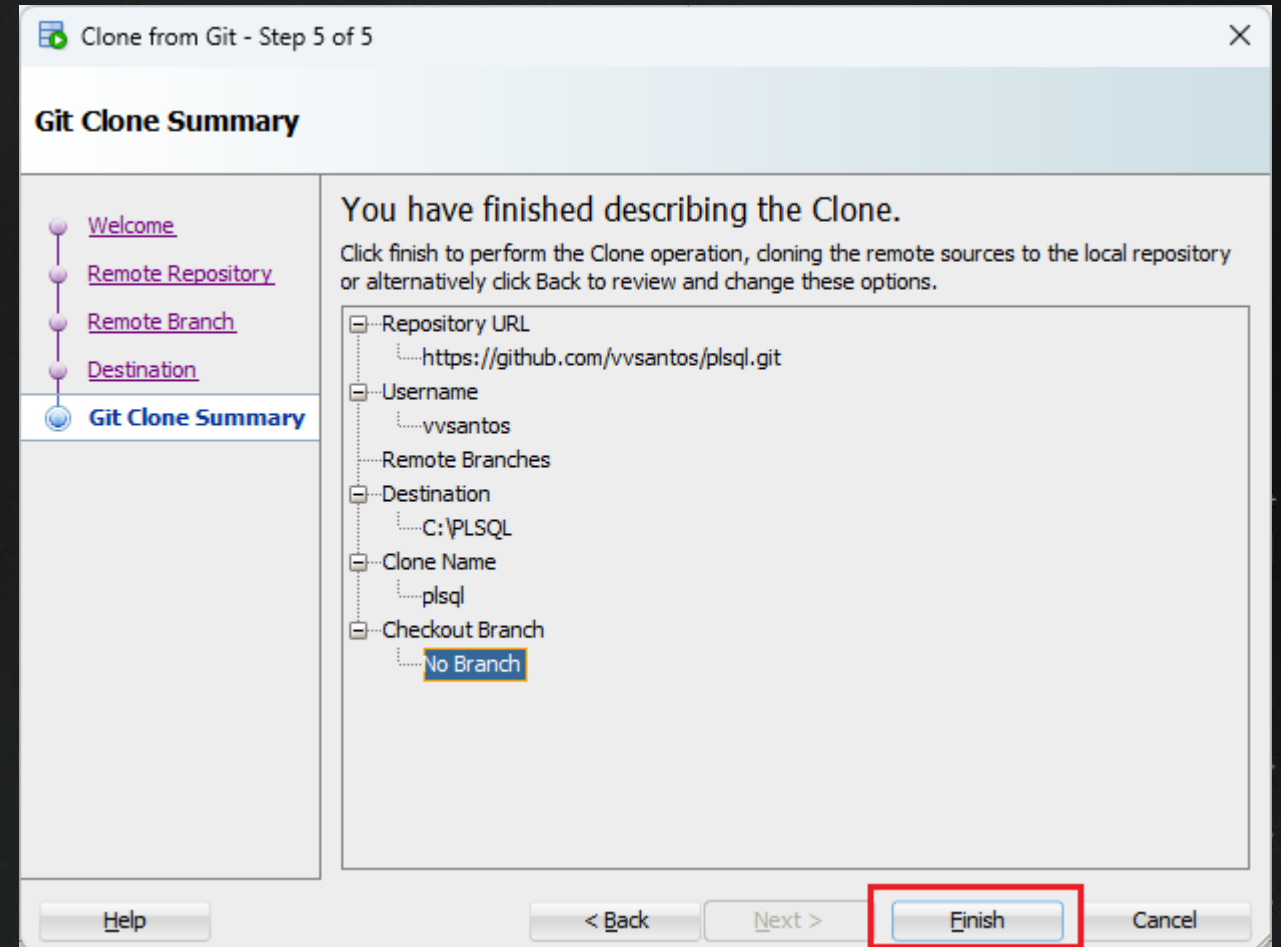
Destination: C:\PLSQL Browse...

Clone Name: plsql

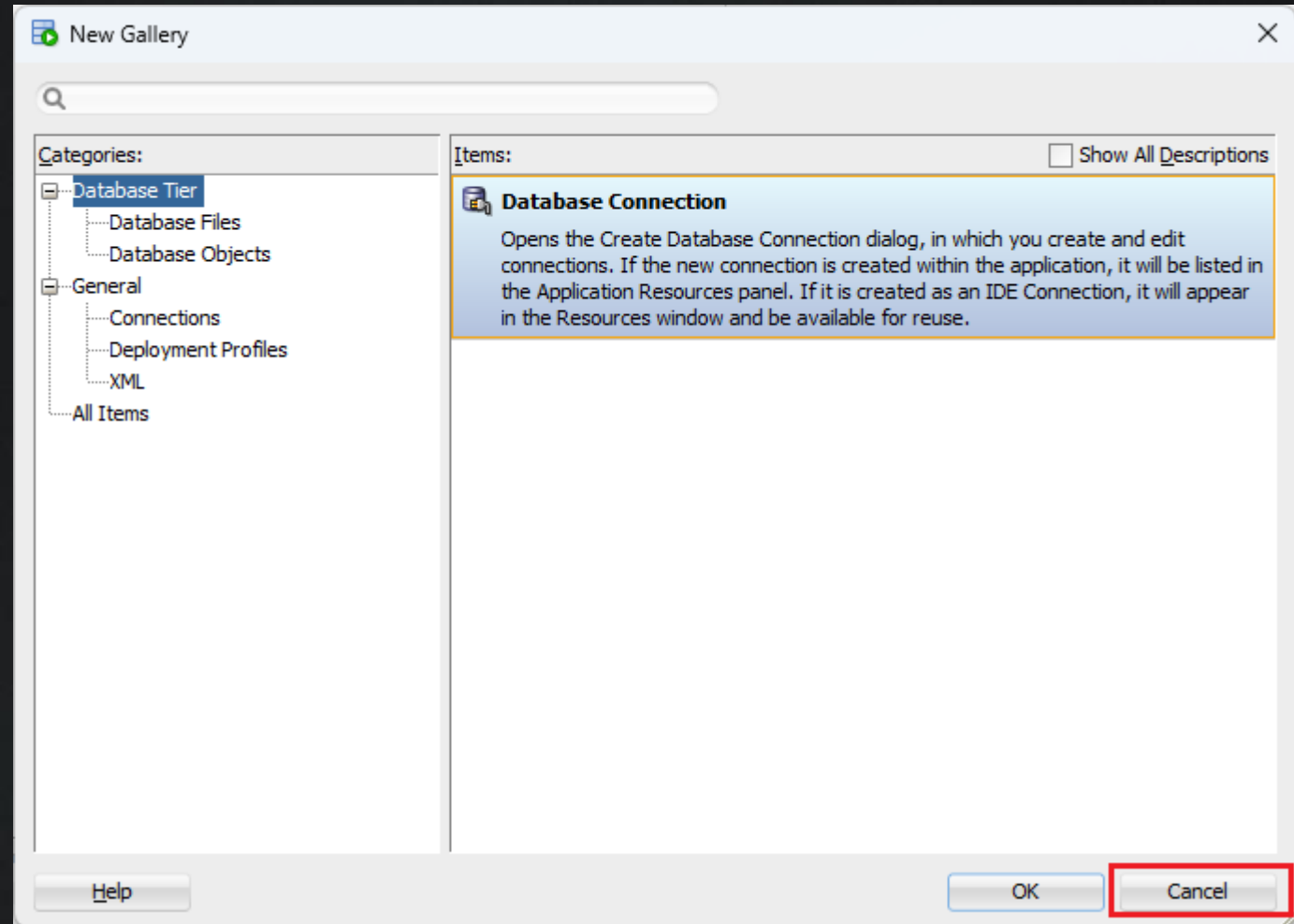
Checkout Branch:

Help < Back **Next >** Finish Cancel

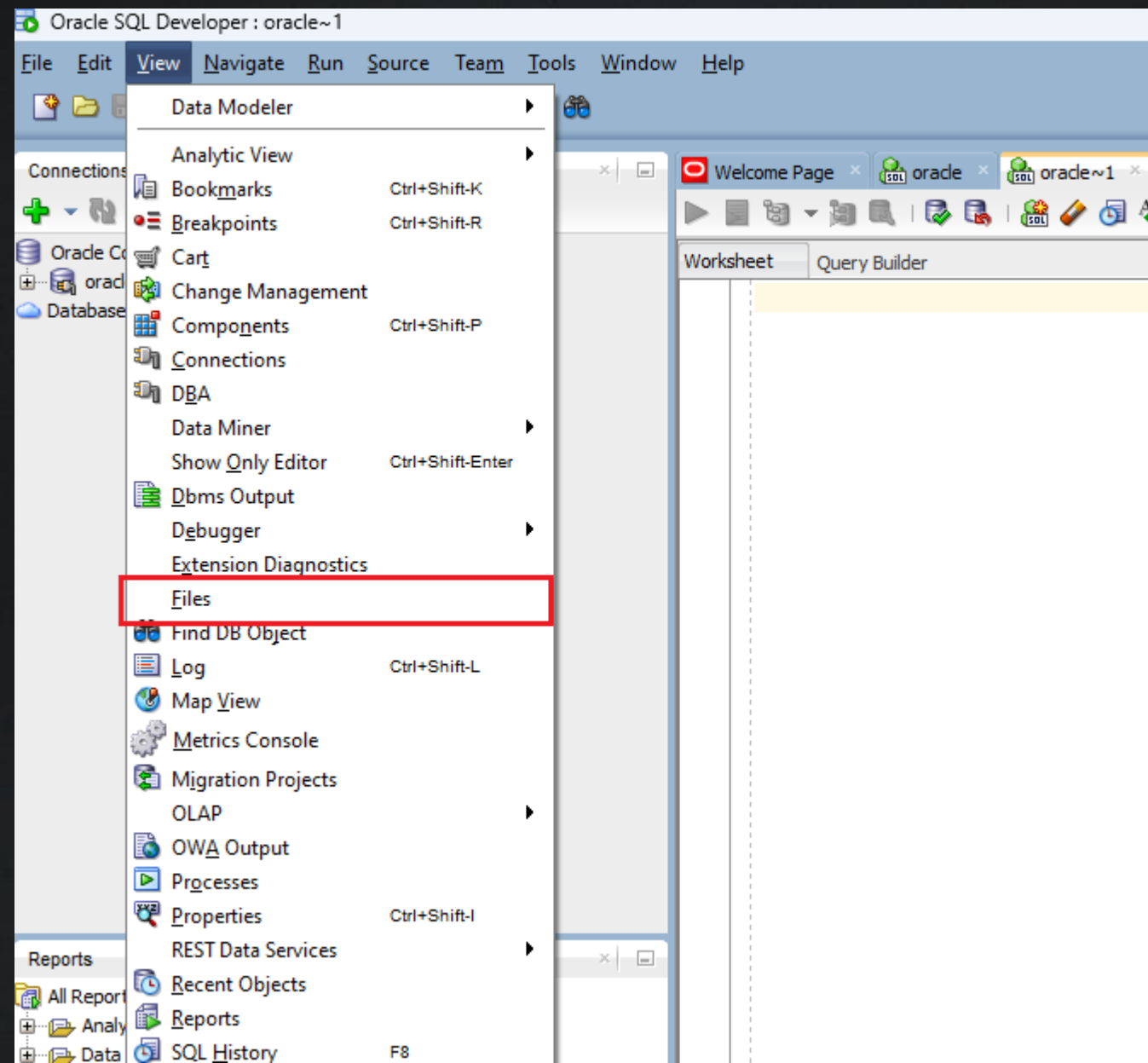
Na janela a seguir basta clicar em  
**Finish**



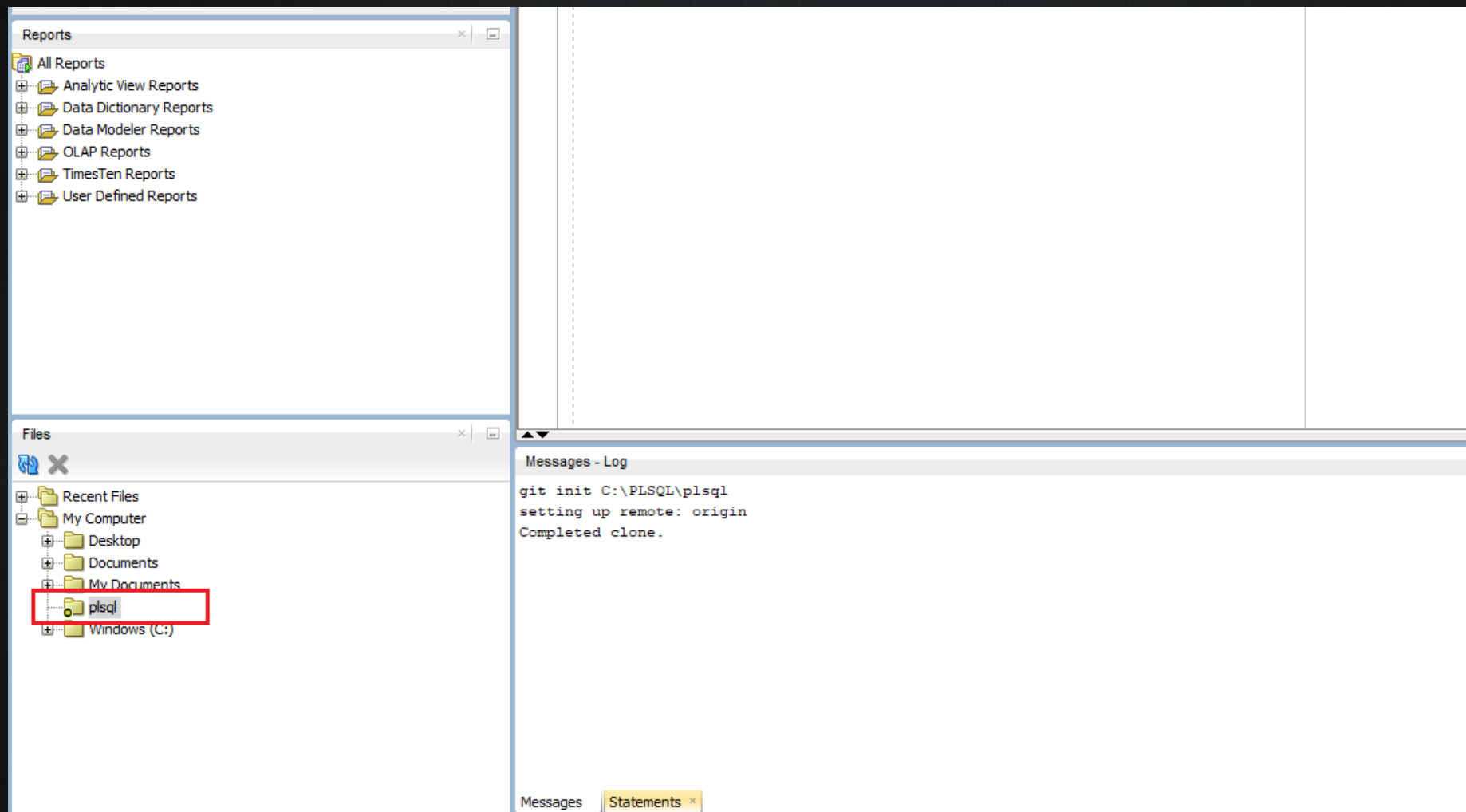
Na janela a seguir basta clicar em  
**Cancel**



Na janela a seguir basta clicar em  
**View > Files**

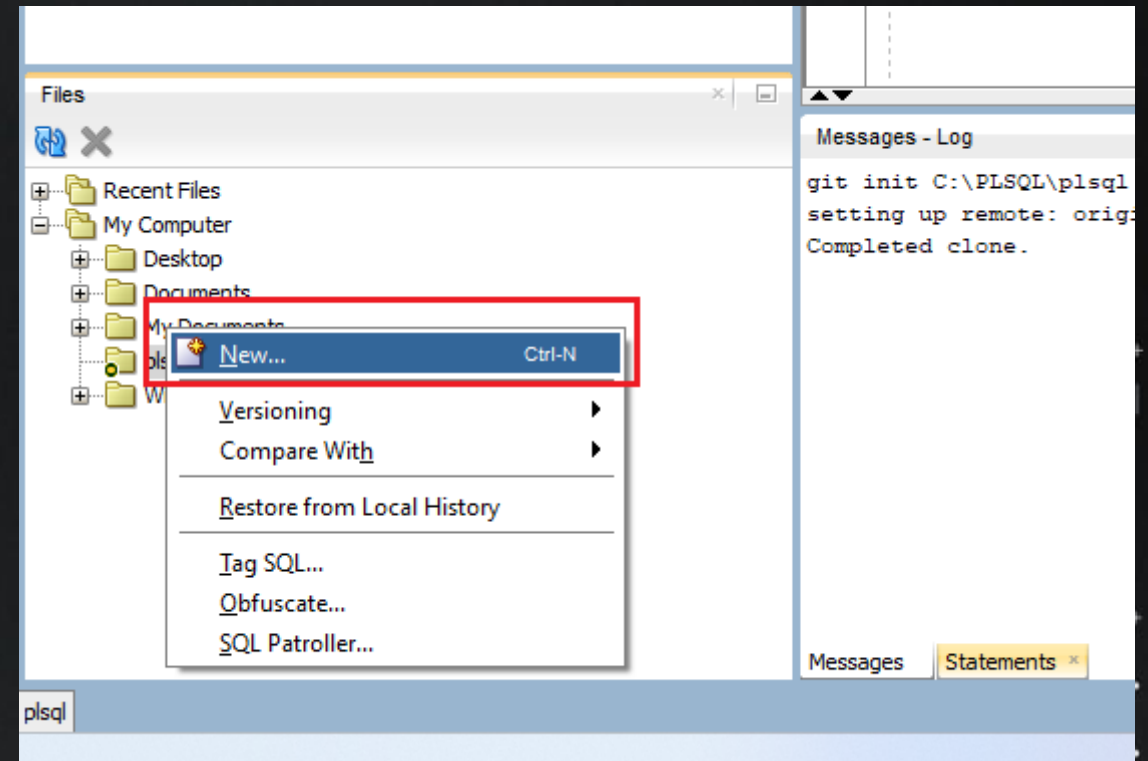


O diretório clonado do GitHub irá aparecer no canto inferior esquerdo.

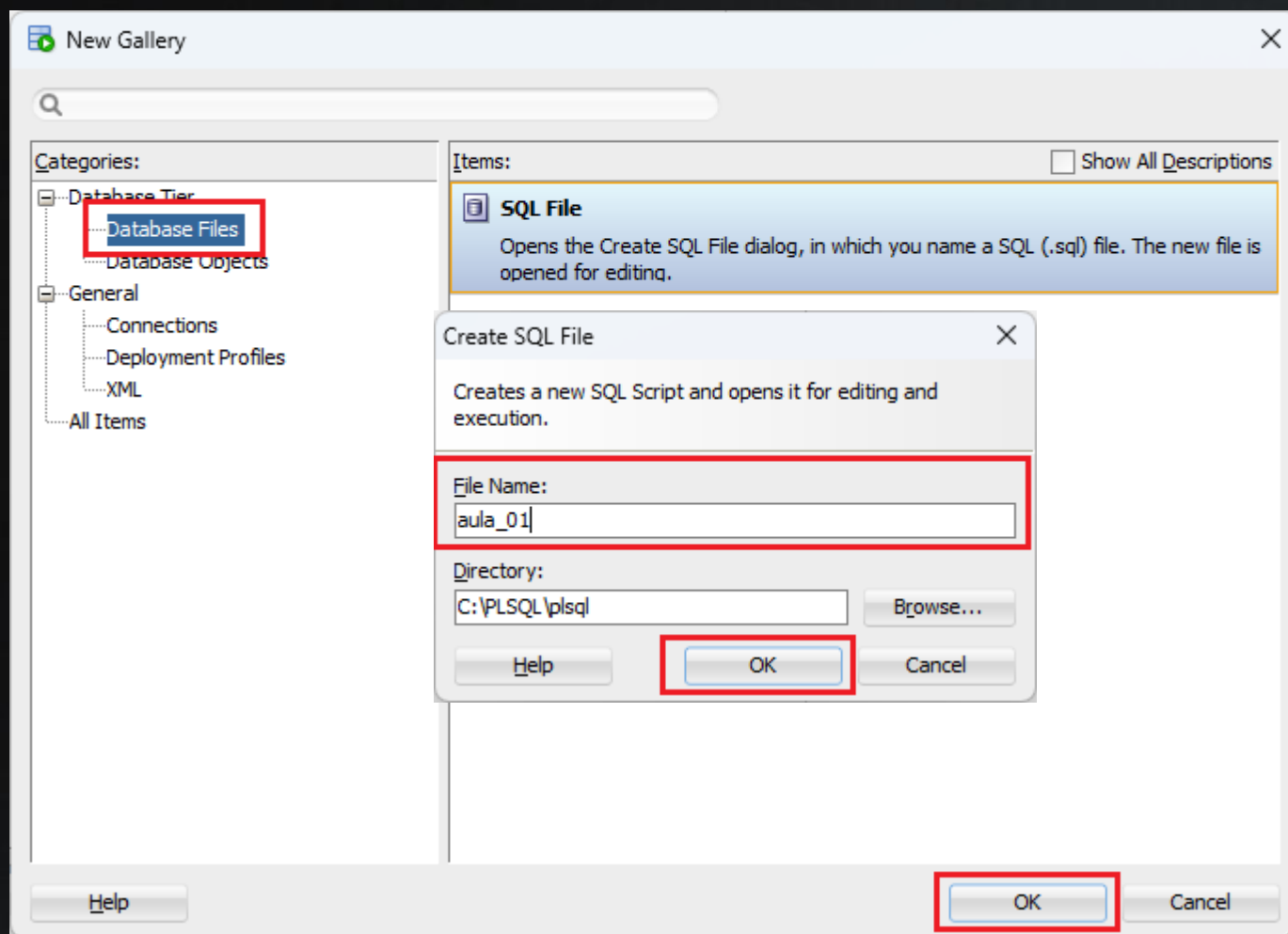




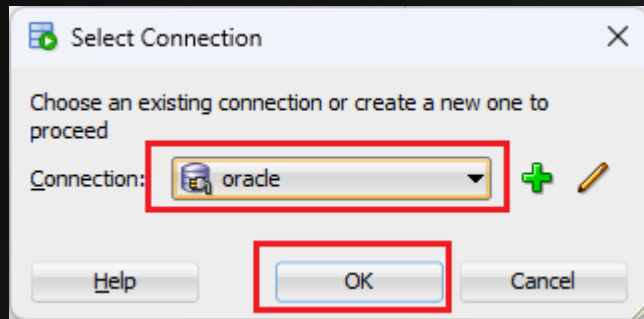
Crie um novo arquivo de conexão  
ao banco de dados para executar  
os comandos sql ou criação de  
objetos de banco de dados



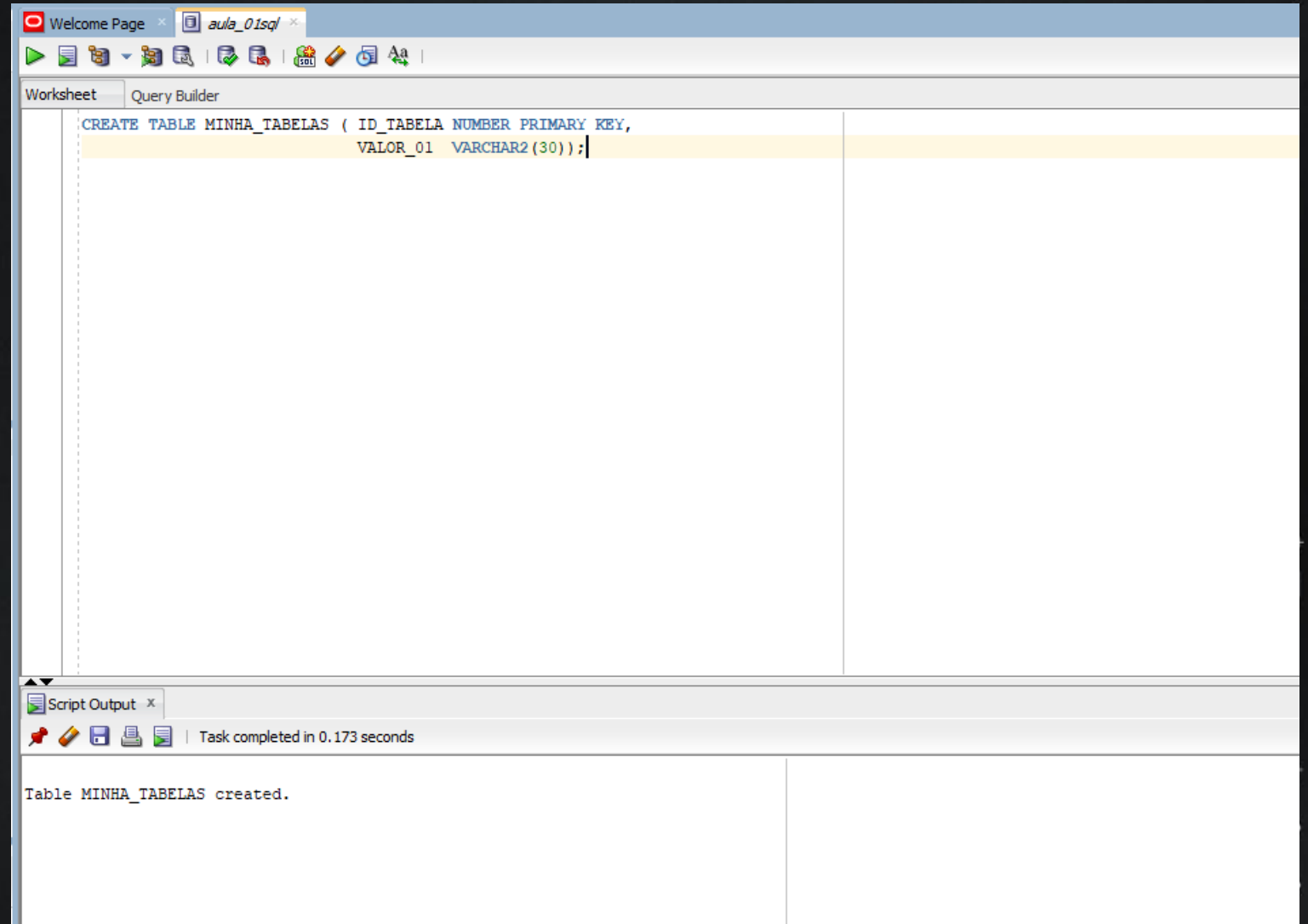
Selecione **Database Files** na pop up escolha o nome do arquivo e clique em **OK**.



Selecione a conexão com o banco criada anteriormente e clique em  
OK



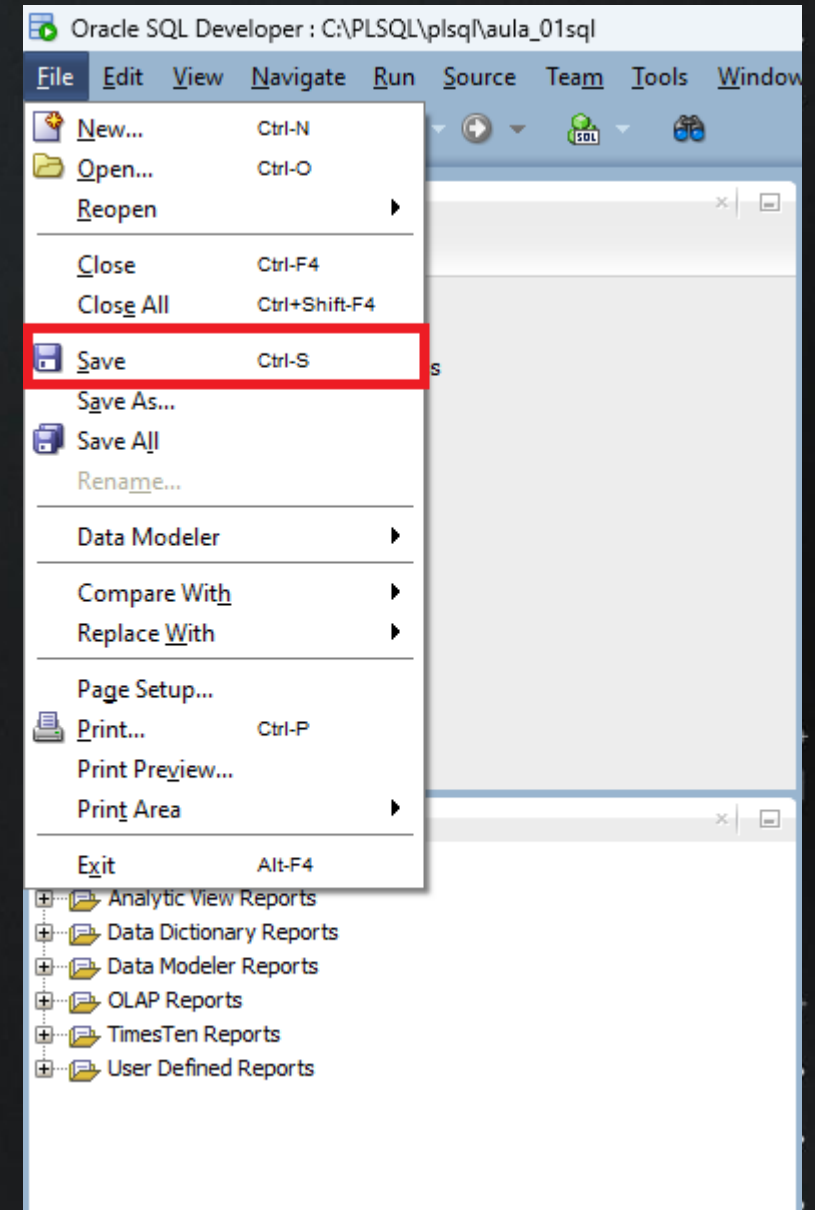
Crie uma query ou objeto qualquer só para fazermos nosso primeiro commit no Git.



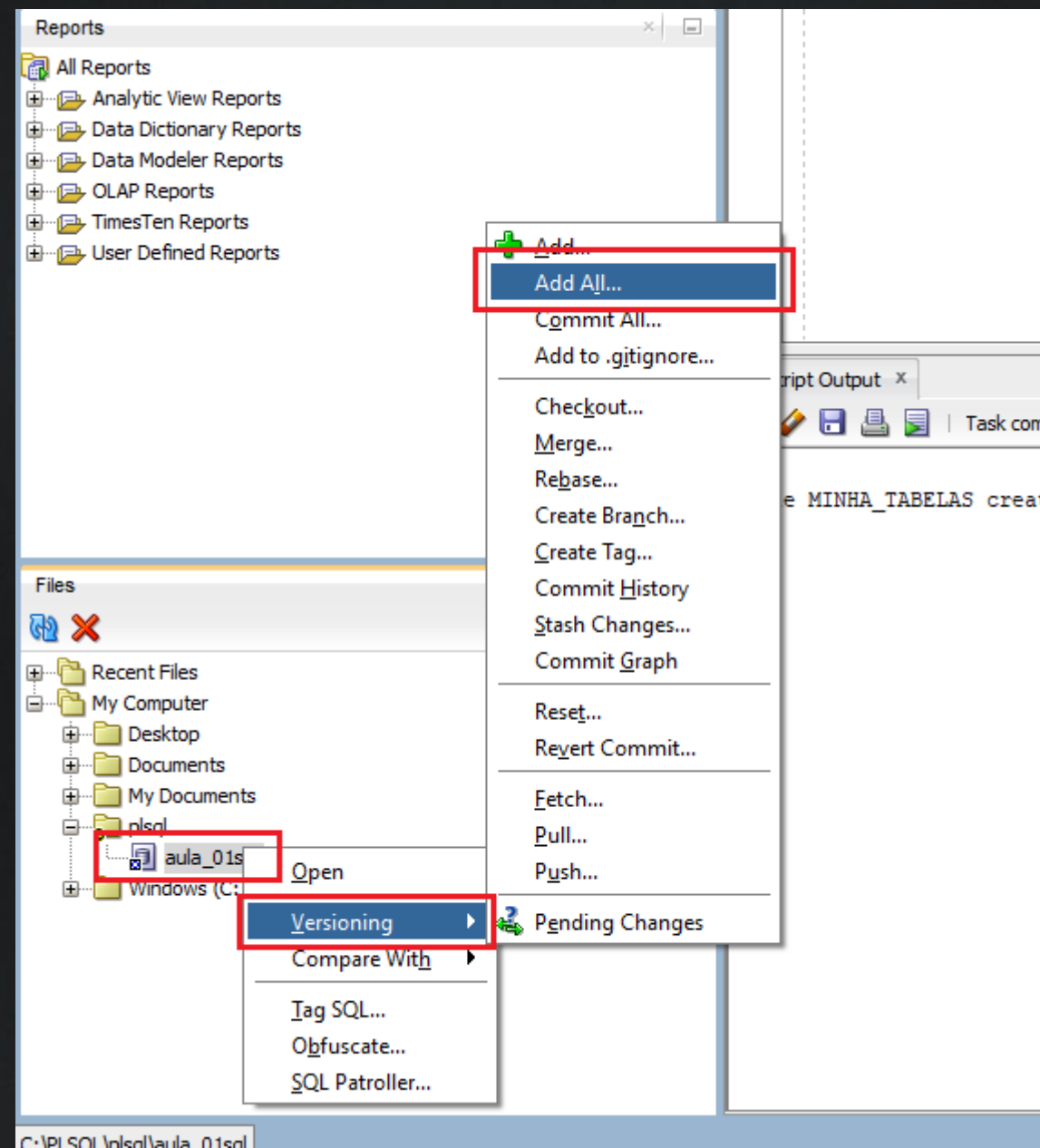
Salve o arquivo criado em:

File > Save

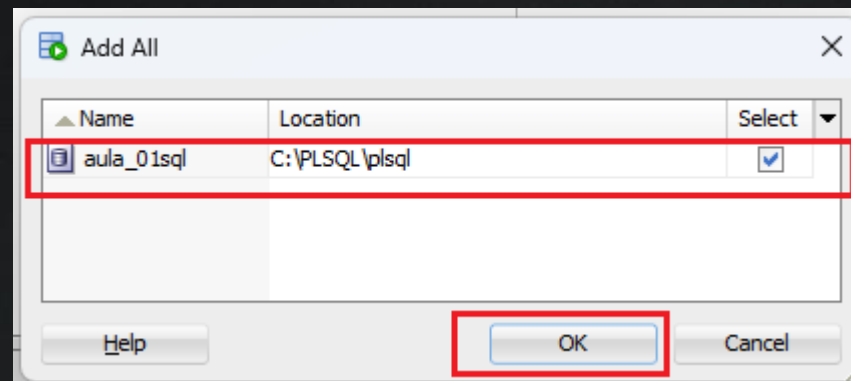
Agora podemos enviar nosso código para o GitHub



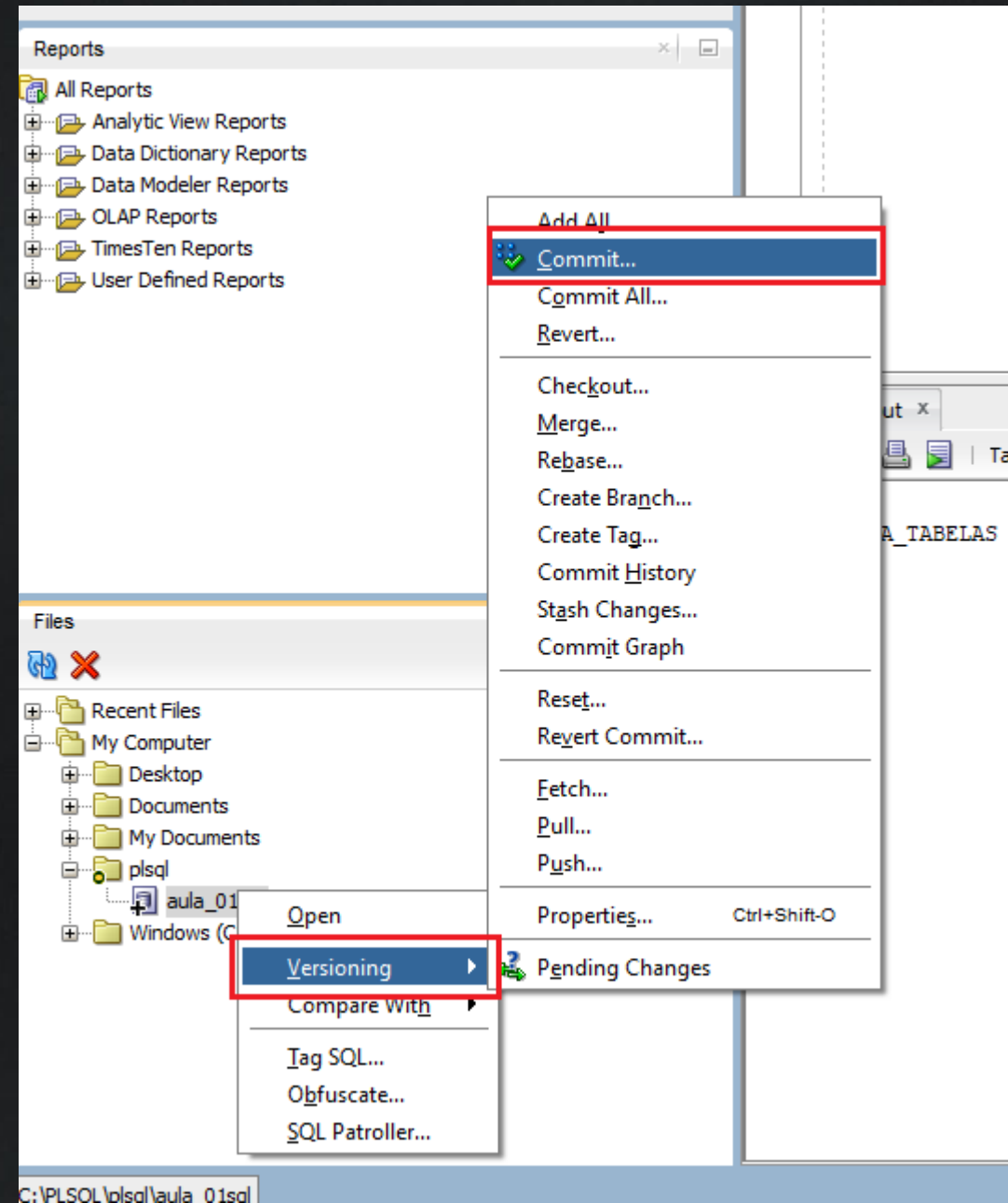
No diretório mapeado clique com o botão direito no arquivo salvo  
**Versioning > Add all**



Selecione o arquivo que quer enviar  
para o Git e clique em **OK**



Clique novamente com o botão direito em cima do arquivo salvo e agora clique em **Versioning > Commit**





Agora basta selecionar o arquivo  
arquivo e clicar em **OK**

Commit

| Name       | Location       | Select                              |
|------------|----------------|-------------------------------------|
| aula_01sql | C:\PLSQL\plsql | <input checked="" type="checkbox"/> |

☐ Commit non-staged files.

Author: vvsantos <vvsantos@SAO-XS1714.ingrnet.com> Committer: vvsantos <vvsantos@SAO-XS1714.ingrnet.com>

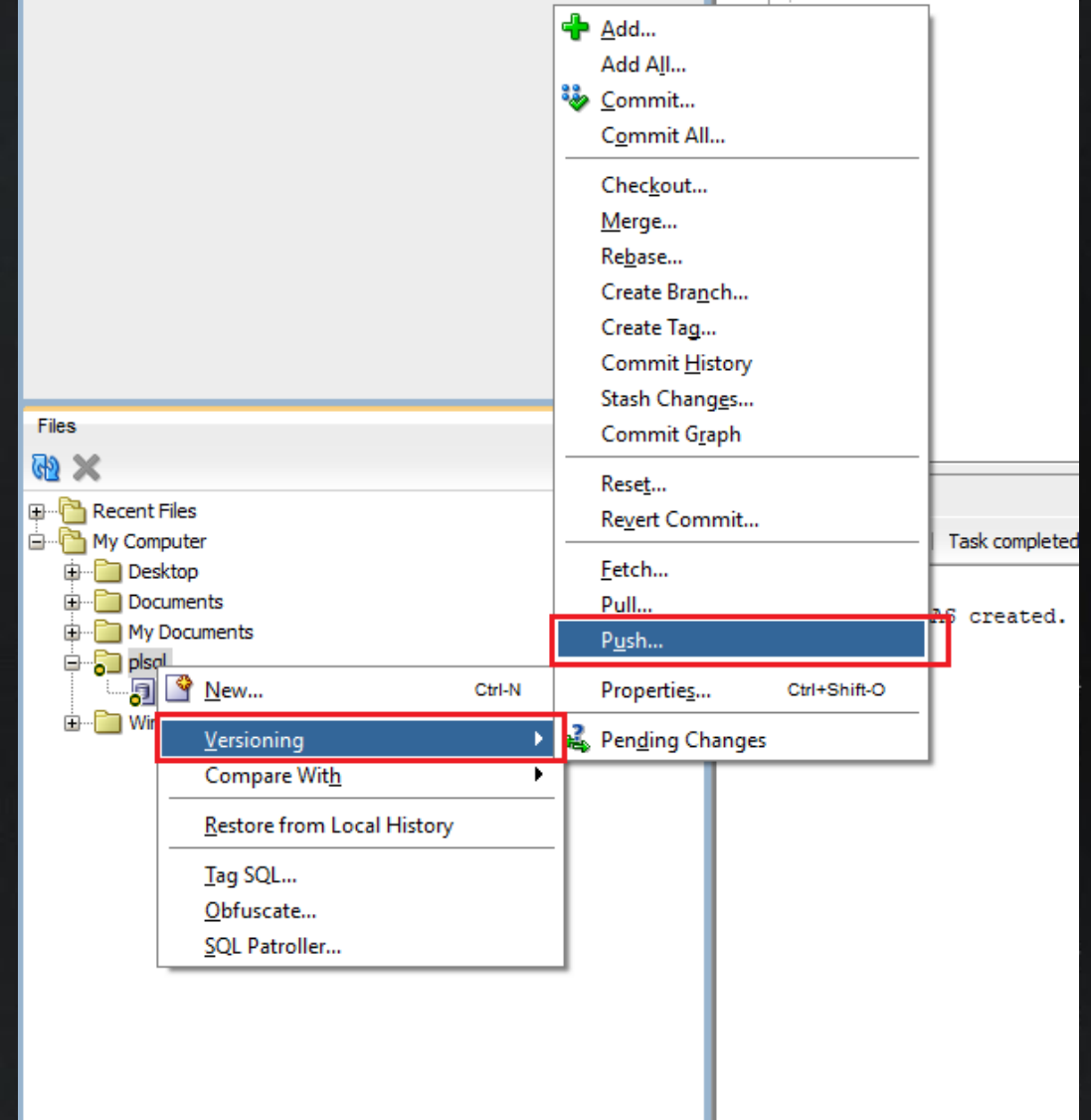
Comments:  
Commit primeira tabela

Template or Previous Comments:  
<Select>

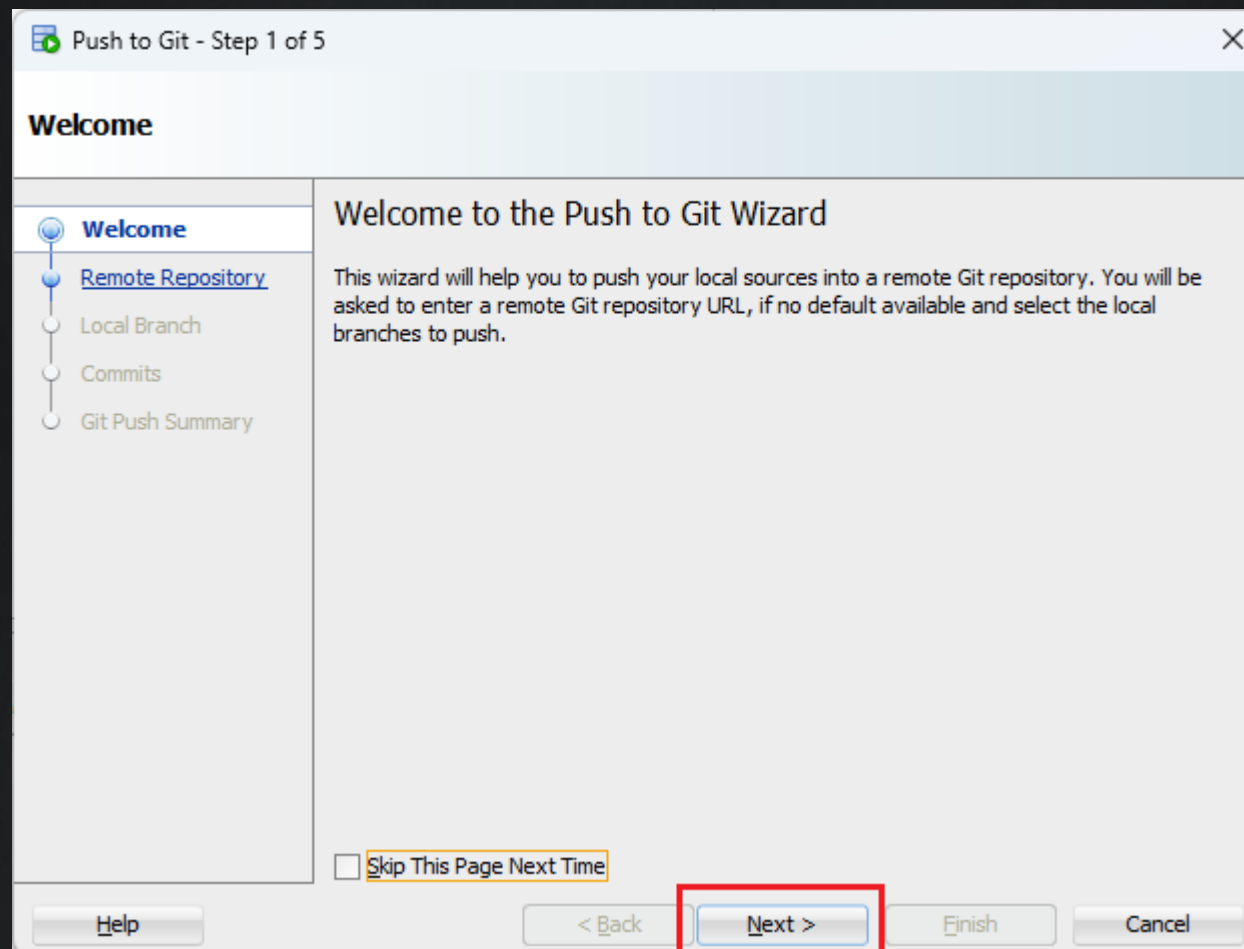
This list is configured with [comment templates](#)

Help OK Cancel

Agora clique em **Versioning > Push**.



Clique em **Next**



Passe as informações de login e clique em **Next**.

Obs.: Colocar o token gerado e não a senha.

Push to Git - Step 2 of 5

### Remote Repository

Select or enter the remote repository url. Enter a user name and password or SSH key information if the remote Git Repository does not support anonymous access.

Repository URL:

[Proxy Settings...](#)

User Name:

☒ Password:

☐ Private Key File:

Passphrase:

Clique em **Next**

Push to Git - Step 3 of 5

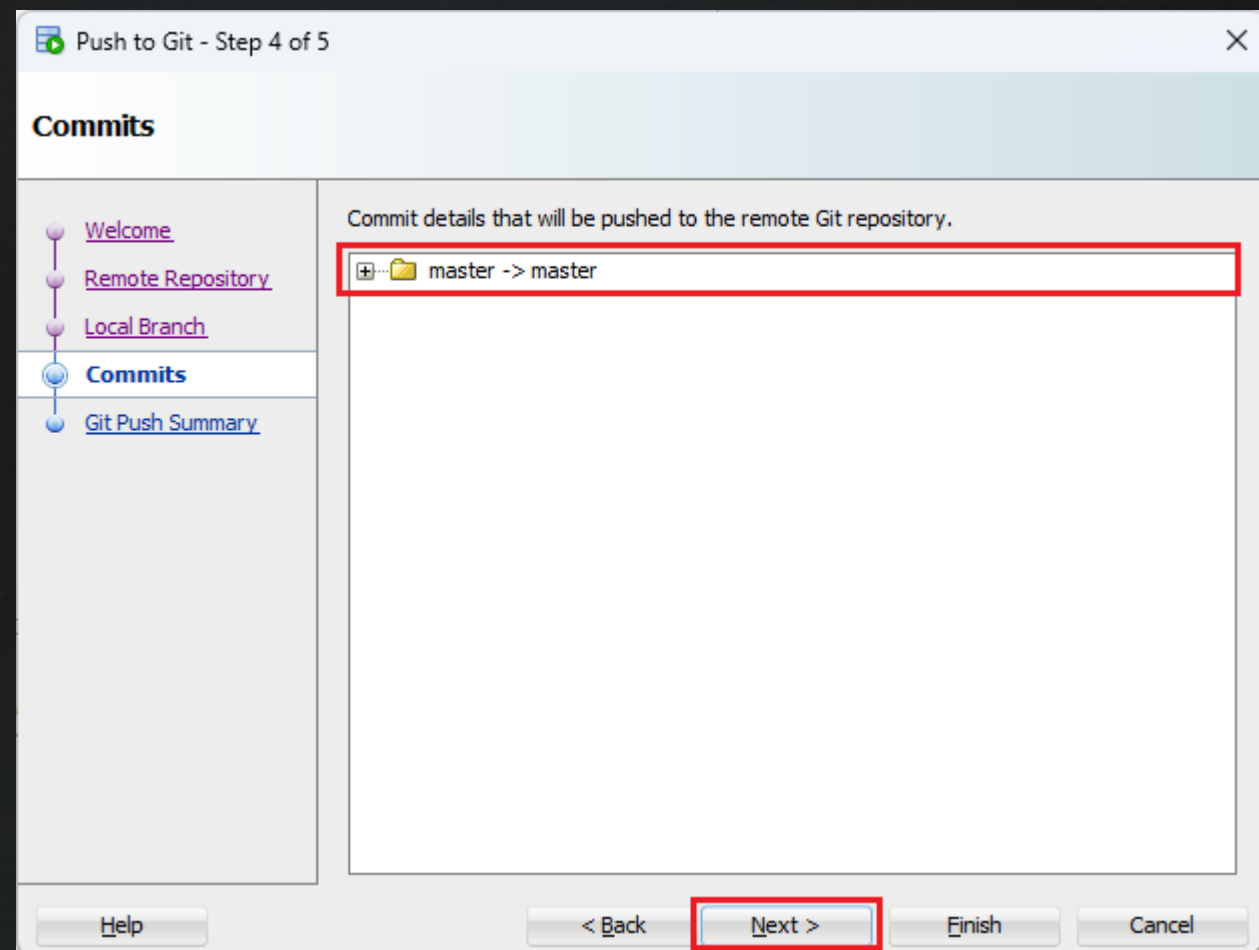
### Local Branch

Select the branches that will be part of the Push operation.

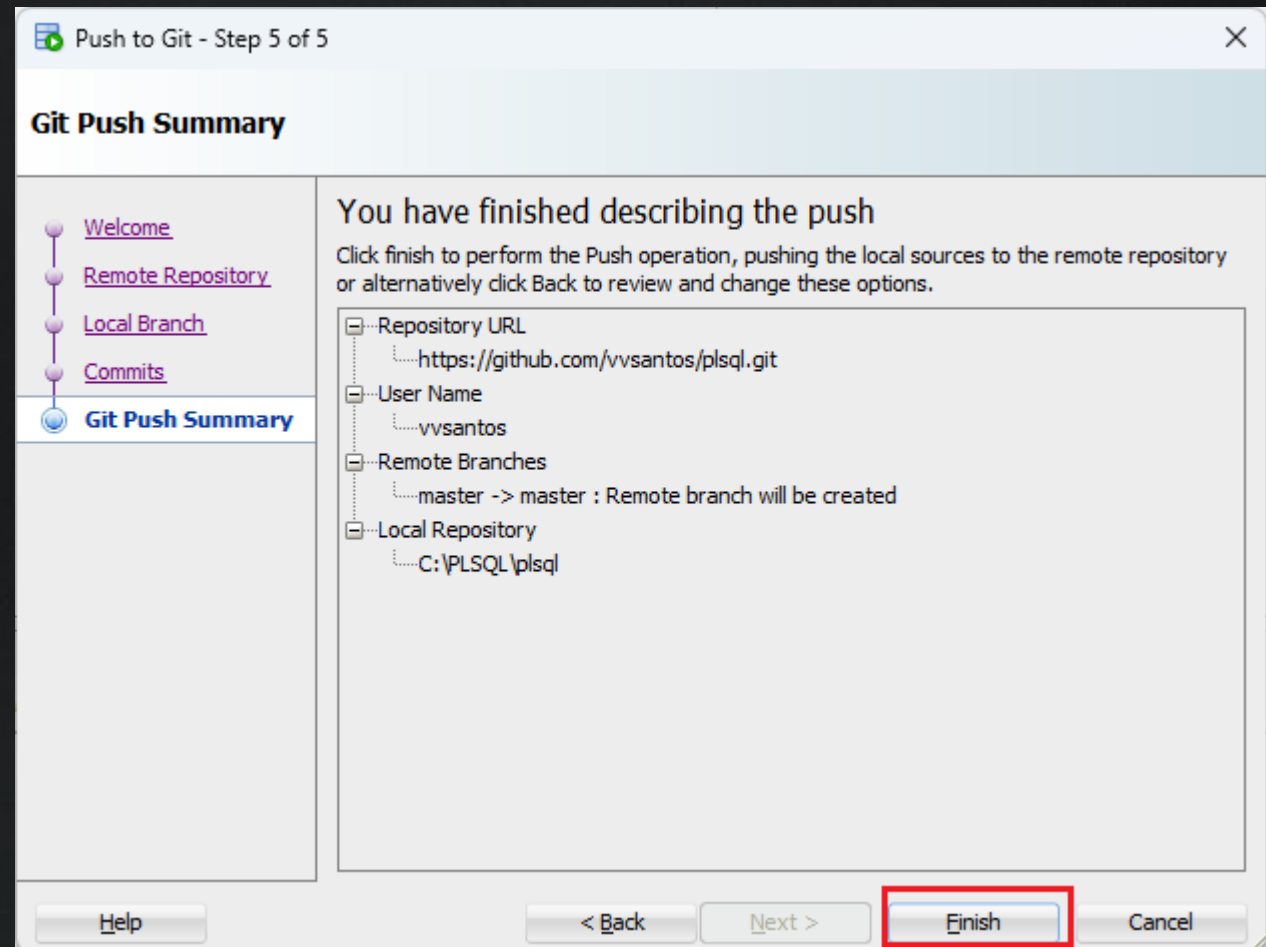
| Include                             | From   | To     | Status |
|-------------------------------------|--------|--------|--------|
| <input checked="" type="checkbox"/> | master | master | Create |

< Back   **Next >**   Finish   Cancel

Clique em **Next**



Clique em **Finish**



Pronto, nossos arquivos criados em aula e exercícios podem ser sincronizados com o GitHub.

The screenshot shows a GitHub repository named 'plsql' by user 'vvsantos'. The repository is public and has 1 branch and 0 tags. A commit titled 'Commit primeira tabela' is highlighted with a red box, showing a file named 'aula\_01sql'. The repository currently has no README, and a large section prompts the user to 'Add a README'. On the right sidebar, there are sections for 'About' (no description), 'Activity' (0 stars, 1 watching, 0 forks), 'Releases' (no releases published), and 'Packages' (no packages published).

plsql Public

Pin Unwatch 1 Fork 0 Star 0

master 1 Branch 0 Tags

Go to file Add file Code

vvsantos and vvsantos Commit primeira tabela 832f5cc · 7 minutes ago 1 Commits

aula\_01sql Commit primeira tabela 7 minutes ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

About

No description, website, or topics provided.

Activity

0 stars

1 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

profvergilio.santos@fiap.com.br