APP RELATORIO GERENCIAL

Este script é um aplicativo de interface gráfica desenvolvido com Tkinter, que se conecta a um banco de dados Oracle, busca dados com base em filtros fornecidos pelo usuário e exibe os resultados em uma tabela. Além disso, oferece a opção de exportar os dados filtrados para um arquivo Excel. Vou explicar as principais partes do código detalhadamente:

1. Importações

- `cx_Oracle`: Usado para estabelecer a conexão com o banco de dados Oracle.
- `tkinter` e `ttk`: Bibliotecas para criar a interface gráfica com widgets, como botões, tabelas e caixas de entrada.
- `filedialog`, `messagebox`: Módulos do Tkinter para abrir a janela de seleção de arquivos e exibir mensagens de erro ou sucesso.
- `pandas`: Importado, mas não utilizado no código fornecido (talvez para manipulação de dados em algum momento futuro).
- `datetime`: Usado para manipular e formatar datas.

2. Configuração da Conexão com o Banco de Dados

- A função `cx_Oracle.makedsn()` cria um identificador de rede para se conectar ao banco, utilizando o endereço IP e a porta do servidor (`192.168.50.7`, `2115`), além do `service_name`.
- A conexão é estabelecida através de `cx_Oracle.connect()`, onde são passados o usuário e a senha.
- Se a conexão falhar, uma mensagem de erro é exibida usando `messagebox.showerror()`.

3. Função `buscar_dados()`

Esta função é chamada quando o botão "Filtrar" é pressionado. Ela busca os dados no banco de dados de acordo com os filtros aplicados pelo usuário:

- **Filtros de data**: O script verifica se as datas de "Data Inclusão" e "Data" estão preenchidas corretamente e no formato adequado (`DD/MM/YYYY`). Se não estiverem, uma mensagem de erro é exibida.

- **Filtros adicionais**: O status selecionado (por exemplo, "Excluídos") e o filtro de "Processo" também são levados em consideração ao montar a consulta SQL.
- **Execução da consulta**: A query SQL é construída dinamicamente conforme os filtros, utilizando parâmetros para prevenir SQL injection. Os filtros de data e status são incluídos na cláusula WHERE.
- **Exibição dos dados**: Após a execução da consulta, os dados retornados são exibidos na tabela (Treeview) na interface gráfica. A função também atualiza o número de registros encontrados.

4. Função `exportar_excel()`

Esta função permite ao usuário exportar os dados filtrados para um arquivo Excel:

- **Seleção do arquivo**: O usuário escolhe onde salvar o arquivo Excel através da janela de diálogo `filedialog.asksaveasfilename()`.
- **Formatação dos dados**: Antes de exportar, o campo "Processo" é convertido para string com 20 caracteres, preenchendo à esquerda com zeros (zfill).
- **Criação do arquivo Excel**: O `openpyxl` é utilizado para criar um novo arquivo Excel. As colunas e dados são inseridos na planilha. As colunas têm larguras ajustadas para 37, e o conteúdo de todas as células é centralizado.
- **Estilo da tabela**: A função cria uma tabela no Excel com listras alternadas nas linhas (usando `TableStyleInfo`), o que melhora a leitura.
- **Salvamento e mensagem de sucesso**: Após a criação e formatação do arquivo, ele é salvo, e uma mensagem de sucesso é exibida.

5. Interface Gráfica com Tkinter

A interface gráfica permite que o usuário forneça filtros e visualize os dados:

- **Filtros**: A interface possui campos para inserção de datas de início e fim, processo, e status. O filtro de status é um combo box com opções como "Todos", "Excluídos", etc.
- **Tabela (Treeview)**: Os resultados da consulta são exibidos em uma tabela com 15 colunas, cada uma representando um campo retornado pela consulta SQL. A tabela possui barras de rolagem horizontal e vertical.

- **Botões**: O script possui botões para acionar a busca de dados (Filtrar) e para exportar os dados para Excel (Exportar para Excel).
- **Total de Registros**: O número de registros retornados pela consulta é mostrado na interface.

6. Fluxo de Execução

- O script começa criando a interface gráfica, onde o usuário pode fornecer filtros.
- Ao clicar em "Filtrar", os dados são buscados no banco de dados conforme os filtros definidos.
- Os dados são exibidos na tabela, e o número total de registros encontrados é mostrado.
- O usuário pode então exportar os dados filtrados para um arquivo Excel clicando no botão "Exportar para Excel".

Conclusão

Este script proporciona uma forma eficiente de buscar dados no banco de dados Oracle com base em filtros e exibi-los em uma interface gráfica. Além disso, a funcionalidade de exportação para Excel permite que o usuário salve os dados em um formato útil para análise ou relatório posterior. O uso de `cx_Oracle` para conexão com o banco, `openpyxl` para criação do arquivo Excel e `Tkinter` para a interface gráfica torna o aplicativo robusto e fácil de usar.