Compra Livros

Aqui está a explicação detalhada da função ComprarLivros(), que tem como objetivo **registrar a compra de livros e atualizar o estoque**. Vamos analisar cada parte do código para facilitar o entendimento:

---

1. Limpar a tela

```
os.system("cls")
```

- os.system("cls"): Limpa a tela do terminal (comum em sistemas Windows).

---

2. Solicitar o ID do livro

```
id_livro = input("Insira o ID do livro a ser comprado: ")
if not id_livro:
    ComprarLivros()
if id_livro.lower() == "sair":
    Menu()
```

- Solicita ao usuário o **ID do livro** que será comprado.
- Se o campo estiver vazio (if not id_livro), a função se chama novamente (ComprarLivros()).
- Se o usuário digitar "sair", a função retorna ao Menu().

---

3. Verificar se o livro existe

```
cursor.execute(f"SELECT * FROM livro WHERE id = {id_livro}")
resultado = cursor.fetchall()
if len(resultado) == 0:
    print(f"\033[31mNão há livro registrado com o id {id_livro}\033[m")
    input()
    Menu()
```

- Executa uma consulta SQL para verificar se o livro com o ID fornecido existe na tabela livro.
- Se não houver resultados (len(resultado) == 0), exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

4. Recuperar informações do livro

```
for linha in resultado:
    titulo = linha[1]
    valor_compra = linha[6]
```

- Itera sobre o resultado da consulta (mesmo que haja apenas uma linha) e armazena:
    - **Título do livro**: titulo = linha[1].
    - **Valor de compra**: valor_compra = linha[6].

---

5. Solicitar a quantidade de cópias

```
print(f"Insira quantas cópias do livro {titulo} serão compradas.")
try:
    quantidade = int(input())
except ValueError:
    print("\033[31mDado inválido.\033[m")
    input()
    Menu()
```

- Solicita ao usuário a **quantidade de cópias** do livro que serão compradas.
- Se o usuário digitar um valor não numérico, exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

6. Solicitar a data da compra

```
data_compra = input("Registre a data da compra. (AAAA/MM/DD): ")
if len(data_compra) != 10:
    print("\033[31m data precisa ter 10 caracteres.\033[m")
    input()
    Menu()
```

- Solicita a **data da compra** no formato AAAA/MM/DD.
- Se a data não tiver exatamente 10 caracteres, exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

7. Calcular o custo total

```
custo = valor_compra * quantidade
```

- Calcula o **custo total** da compra multiplicando o valor de compra do livro pela quantidade.

---

8. Atualizar o estoque e registrar a compra

```
try:
  cursor.execute("SELECT * from estoque")
  resultado3 = cursor.fetchall()
  for linha in resultado3:
    quantidadeEstoque = linha[1]
    atualizarQuantidade = quantidadeEstoque + quantidade
  cursor.execute(f"INSERT INTO compra (id_livro, quantidade, data_compra, custo) VALUES ('{id_livro}','{quantidade}','{data_compra}','{custo}')")
  conexao.commit()
  cursor.execute(f"UPDATE estoque SET quantidade = {atualizarQuantidade} WHERE id_livro = {id_livro}")
  conexao.commit()
  print("\033[32mCompra registrada com sucesso.\033[m")
  input()
except:
  print("\033[31mNão foi possível registrar a compra. verifique os dados e tente novamente.\033[m")
  input()
```

- **Passo 1**: Recupera a quantidade atual do livro no estoque (quantidadeEstoque) e calcula a nova quantidade (atualizarQuantidade = quantidadeEstoque + quantidade).
- **Passo 2**: Insere os dados da compra na tabela compra, incluindo:
  - ID do livro (id_livro).
  - Quantidade comprada (quantidade).
  - Data da compra (data_compra).
  - Custo total (custo).
- **Passo 3**: Atualiza a quantidade do livro no estoque (UPDATE estoque).
- Se tudo der certo, exibe uma mensagem de sucesso em verde (\033[32m).

- Se ocorrer algum erro (por exemplo, dados inválidos ou falha no banco de dados), exibe uma mensagem de erro em vermelho (\033[31m).

---

9. Retorno ao menu

`Menu()`

- Retorna ao Menu() principal do sistema.

Resumo da funcionalidade

A função ComprarLivros():

1. Solicita o ID do livro e verifica se ele existe.
2. Recupera o título e o valor de compra do livro.
3. Solicita a quantidade de cópias e a data da compra.
4. Calcula o custo total da compra.
5. Atualiza o estoque e registra a compra no banco de dados.
6. Exibe mensagens de sucesso ou erro.
7. Retorna ao menu principal.

Essa função é útil para gerenciar as compras de livros, atualizando o estoque e mantendo um registro das transações.

Lista Compras

Aqui está a explicação detalhada da função ListarCompras(), que tem como objetivo **listar as compras de livros registradas no sistema**. Vamos analisar cada parte do código para facilitar o entendimento:

---

1. Limpar a tela

```
os.system("cls")
```

- os.system("cls"): Limpa a tela do terminal (comum em sistemas Windows).

---

2. Consulta ao banco de dados

```
cursor.execute("SELECT c.id, l.id, l.titulo, c.quantidade, c.data_compra, c.custo FROM compra as c JOIN livro as l ON c.id_livro = l.id")
```

```
resultado = cursor.fetchall()
```

- cursor.execute(...): Executa um comando SQL que seleciona dados das tabelas compra e livro.
    - **Tabela compra (c)**: Recupera o id, quantidade, data_compra e custo da compra.
    - **Tabela livro (l)**: Recupera o id e titulo do livro.
    - A junção (JOIN) é feita pelo id_livro da compra e o id do livro (c.id_livro = l.id).
- cursor.fetchall(): Recupera todos os resultados da consulta e armazena na variável resultado.

---

3. Verificação de registros

```
if len(resultado) == 0:
    print("\033[31mNão há registro.\033[m")
    input()
    Menu()
```

- Verifica se a lista de resultados (resultado) está vazia.
- Se não houver registros, exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

4. Exibição das compras

```
print("="*50)
```

```
print("REGISTRO DE COMPRAS")
print("="*50)
for linha in resultado:
    id = linha[0]
    id_livro = linha[1]
    titulo = linha[2]
    quantidade = linha[3]
    data_compra = linha[4]
    custo = linha[5]
    print(f"ID da compra: {id}")
    print(f"ID do livro: {id_livro}")
    print(f"Livro: {titulo}")
    print(f"Quantidade comprada: {quantidade}")
    print(f"Data da compra: {data_compra}")
    print(f"Custo total: {custo}R$\n")
    print("-"*50)
```

- Exibe um cabeçalho formatado com = para destacar o título "REGISTRO DE COMPRAS".
- Itera sobre cada linha (linha) da lista resultado e exibe os seguintes dados de cada compra:
    - **ID da compra**: Identificador único da compra.
    - **ID do livro**: Identificador único do livro comprado.
    - **Livro**: Título do livro comprado.
    - **Quantidade comprada**: Quantidade de cópias compradas.
    - **Data da compra**: Data em que a compra foi realizada.
    - **Custo total**: Valor total da compra.
- Após exibir os dados de uma compra, insere uma linha de separação (-).

---

5. Mensagem de conclusão

```
print("\033[32mListagem concluída com sucesso.\033[m")
```

- Exibe uma mensagem de sucesso em verde (\033[32m) indicando que a listagem foi concluída.

---

6. Retorno ao menu

input()

Menu()

- Aguarda o usuário pressionar Enter (input()).
- Retorna ao Menu() principal do sistema.

Resumo da funcionalidade

A função ListarCompras():

1. Limpa a tela.
2. Consulta os dados das compras, combinando informações das tabelas compra e livro.
3. Verifica se há registros. Se não houver, exibe uma mensagem de erro e retorna ao menu.
4. Exibe os dados de todas as compras registradas em um formato organizado.
5. Mostra uma mensagem de sucesso ao concluir a listagem.
6. Retorna ao menu principal.

Essa função é útil para visualizar o histórico de compras de livros, permitindo ao usuário conferir detalhes como o título do livro, a quantidade comprada, a data da compra e o custo total.

Vender Livros

Aqui está a explicação detalhada da função VenderLivros(), que tem como objetivo **registrar a venda de livros e atualizar o estoque**. Vamos analisar cada parte do código para facilitar o entendimento:

---

1. Limpar a tela

```
os.system("cls")
```

- os.system("cls"): Limpa a tela do terminal (comum em sistemas Windows).

---

2. Verificar disponibilidade de livros

```
cursor.execute("SELECT * FROM livro")
disponibilidade = cursor.fetchall()
if len(disponibilidade) == 0:
    print("\033[31mNão há livros cadastrados para a venda.\033[m")
    input()
    Menu()
```

- Consulta a tabela livro para verificar se há livros cadastrados.
- Se não houver livros (len(disponibilidade) == 0), exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

3. Solicitar o ID do livro

```
id_livro = input("Insira o ID do livro a ser vendido: ")
if id_livro.lower() == "sair":
    Menu()
if not id_livro:
    VenderLivros()
```

- Solicita ao usuário o **ID do livro** que será vendido.
- Se o usuário digitar "sair", a função retorna ao Menu().
- Se o campo estiver vazio (if not id_livro), a função se chama novamente (VenderLivros()).

---

4. Verificar se o livro existe

```
cursor.execute(f"SELECT * FROM livro WHERE id = {id_livro}")
resultado = cursor.fetchall()
if len(resultado) == 0:
    print(f"\033[31mNão há livro registrado com o id {id_livro}\033[m")
    input()
    VenderLivros()
```

- Executa uma consulta SQL para verificar se o livro com o ID fornecido existe na tabela livro.
- Se não houver resultados (len(resultado) == 0), exibe uma mensagem de erro em vermelho (\033[31m) e chama novamente a função VenderLivros().

---

5. Recuperar informações do livro

```
for linha in resultado:
    id = linha[0]
    titulo = linha[1]
    valor_compra = linha[6]
    valor_venda = linha[7]
```

- Itera sobre o resultado da consulta (mesmo que haja apenas uma linha) e armazena:
    - **ID do livro**: id = linha[0].
    - **Título do livro**: titulo = linha[1].
    - **Valor de compra**: valor_compra = linha[6].
    - **Valor de venda**: valor_venda = linha[7].

---

6. Verificar estoque do livro

```
cursor.execute(f"SELECT quantidade FROM estoque WHERE id_livro = {id}")
estoqueBolado = cursor.fetchall()
for linha in estoqueBolado:
    quantidadeEstocada = linha[0]
if quantidadeEstocada == 0:
    print(f"\033[31mNão há cópias do livro {titulo} disponíveis a serem vendidas.\033[m")
```

```
    input()
    Menu()
```

- Consulta a tabela estoque para verificar a quantidade disponível do livro (quantidadeEstocada).
- Se não houver cópias disponíveis (quantidadeEstocada == 0), exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

7. Solicitar o ID do cliente

```
id_cliente = input(f"Insira o ID do cliente que irá comprar o livro {titulo}: ")
if id_cliente.lower() == "sair":
    Menu()
cursor.execute(f"SELECT * FROM cliente WHERE id = {id_cliente}")
resultado2 = cursor.fetchall()
for linha in resultado2:
    nome = linha[1]
if len(resultado2) == 0:
    print(f"\033[31mNão há cliente registrado com o id {id_livro}\033[m")
    input()
    VenderLivros()
```

- Solicita ao usuário o **ID do cliente** que está comprando o livro.
- Se o usuário digitar "sair", a função retorna ao Menu().
- Verifica se o cliente existe na tabela cliente.
    - Se não houver cliente (len(resultado2) == 0), exibe uma mensagem de erro em vermelho (\033[31m) e chama novamente a função VenderLivros().

---

8. Solicitar a quantidade de cópias

```
print(f"Insira quantas cópias do livro {titulo} o senhor(a) {nome} irá comprar:")
try:
    quantidade = int(input())
except ValueError:
    print("\033[31mDado inválido.\033[m")
    input()
```

```
    VenderLivros()
if quantidade > quantidadeEstocada:
    quantidade = quantidadeEstocada
```

- Solicita ao usuário a **quantidade de cópias** do livro que serão vendidas.
- Se o usuário digitar um valor não numérico, exibe uma mensagem de erro em vermelho (\033[31m) e chama novamente a função VenderLivros().
- Se a quantidade solicitada for maior que a quantidade em estoque, ajusta a quantidade para o valor disponível.

---

9. Solicitar a data da venda

```
data_venda = input("Registre a data da venda. (AAAA/MM/DD): ")
if len(data_venda) != 10:
    print("\033[31m data precisa ter 10 caracteres.\033[m")
    input()
    VenderLivros()
```

- Solicita a **data da venda** no formato AAAA/MM/DD.
- Se a data não tiver exatamente 10 caracteres, exibe uma mensagem de erro em vermelho (\033[31m) e chama novamente a função VenderLivros().

---

10. Calcular o lucro

```
lucro = (valor_venda - valor_compra) * quantidade
```

- Calcula o **lucro** da venda subtraindo o valor de compra do valor de venda e multiplicando pela quantidade vendida.

---

11. Atualizar o estoque e registrar a venda

```
novaQuantidade = quantidadeEstocada - quantidade
try:
    cursor.execute(f"INSERT INTO venda (data_venda, quantidade, id_cliente, id_livro, lucro) VALUES ('{data_venda}','{quantidade}','{id_cliente}','{id_livro}','{lucro}')")
    conexao.commit()
    cursor.execute(f"UPDATE estoque SET quantidade = {novaQuantidade} WHERE id_livro = {id_livro}")
    conexao.commit()
```

```
    print("\033[32mVenda registrada com sucesso.\033[m")
    input()
except:
    print("\033[32mNão foi possível registrar a venda. verifique os dados e tente novamente.\033[m")
    input()
```

- **Passo 1**: Calcula a nova quantidade em estoque (novaQuantidade = quantidadeEstocada - quantidade).
- **Passo 2**: Insere os dados da venda na tabela venda, incluindo:
    - Data da venda (data_venda).
    - Quantidade vendida (quantidade).
    - ID do cliente (id_cliente).
    - ID do livro (id_livro).
    - Lucro (lucro).
- **Passo 3**: Atualiza a quantidade do livro no estoque (UPDATE estoque).
- Se tudo der certo, exibe uma mensagem de sucesso em verde (\033[32m).
- Se ocorrer algum erro, exibe uma mensagem de erro em vermelho (\033[31m).

---

12. Retorno ao menu

```
Menu()
```

- Retorna ao Menu() principal do sistema.

Resumo da funcionalidade

A função VenderLivros():

1. Verifica se há livros cadastrados.
2. Solicita o ID do livro e verifica se ele existe.
3. Recupera o título, valor de compra e valor de venda do livro.
4. Verifica se há cópias disponíveis em estoque.
5. Solicita o ID do cliente e verifica se ele existe.
6. Solicita a quantidade de cópias e ajusta se necessário.
7. Solicita a data da venda.
8. Calcula o lucro da venda.

9. Atualiza o estoque e registra a venda no banco de dados.
10. Exibe mensagens de sucesso ou erro.
11. Retorna ao menu principal.

Essa função é útil para gerenciar as vendas de livros, atualizando o estoque e mantendo um registro das transações, incluindo o lucro obtido.

Listar Vendas

Aqui está a explicação detalhada da função ListarVendas(), que tem como objetivo **listar todas as vendas registradas no sistema**. Vamos analisar cada parte do código para facilitar o entendimento:

---

1. Limpar a tela

```
os.system("cls")
```

- os.system("cls"): Limpa a tela do terminal (comum em sistemas Windows).

---

2. Consulta ao banco de dados

```
cursor.execute("SELECT v.id, l.id, c.id, l.titulo, c.nome, v.quantidade, v.lucro FROM venda AS v JOIN livro AS l ON v.id_livro = l.id JOIN cliente AS c ON v.id_cliente = c.id")

resultado = cursor.fetchall()
```

- Executa uma consulta SQL que seleciona dados das tabelas venda, livro e cliente:
  - **Tabela venda (v)**: Recupera o id, quantidade e lucro da venda.
  - **Tabela livro (l)**: Recupera o id e titulo do livro.
  - **Tabela cliente (c)**: Recupera o id e nome do cliente.
  - A junção (JOIN) é feita pelo id_livro da venda e o id do livro (v.id_livro = l.id), e pelo id_cliente da venda e o id do cliente (v.id_cliente = c.id).
- cursor.fetchall(): Recupera todos os resultados da consulta e armazena na variável resultado.

---

3. Verificação de registros

```
if len(resultado) == 0:
    print("\033[31mNão há registro.\033[m")
    input()
    Menu()
```

- Verifica se a lista de resultados (resultado) está vazia.
- Se não houver registros, exibe uma mensagem de erro em vermelho (\033[31m) e retorna ao Menu().

---

4. Exibição das vendas

```
print("="*50)
print("LISTA DE VENDAS")
print("="*50)
for linha in resultado:
    v_id = linha[0]
    l_id = linha[1]
    c_id = linha[2]
    l_titulo = linha[3]
    c_nome = linha[4]
    v_quantidade = linha[5]
    v_lucro = linha[6]
    print(f"ID da venda: {v_id}")
    print(f"ID do livro: {l_id}")
    print(f"ID do cliente: {c_id}")
    print(f"Livro: {l_titulo}")
    print(f"Cliente: {c_nome}")
    print(f"Cópias vendidas: {v_quantidade}")
    print(f"Lucro: {v_lucro}R$\n")
    print("-"*50)
```

- Exibe um cabeçalho formatado com = para destacar o título "LISTA DE VENDAS".
- Itera sobre cada linha (linha) da lista resultado e exibe os seguintes dados de cada venda:
    - **ID da venda**: Identificador único da venda.
    - **ID do livro**: Identificador único do livro vendido.
    - **ID do cliente**: Identificador único do cliente que comprou o livro.
    - **Livro**: Título do livro vendido.
    - **Cliente**: Nome do cliente que comprou o livro.
    - **Cópias vendidas**: Quantidade de cópias vendidas.
    - **Lucro**: Lucro obtido com a venda.
- Após exibir os dados de uma venda, insere uma linha de separação (-).

---

5. Mensagem de conclusão

```
print("\033[32mListagem concluída com sucesso.\033[m")
```

- Exibe uma mensagem de sucesso em verde (\033[32m) indicando que a listagem foi concluída.

---

6. Retorno ao menu

```
input()
```

```
Menu()
```

- Aguarda o usuário pressionar Enter (input()).
- Retorna ao Menu() principal do sistema.

Resumo da funcionalidade

A função ListarVendas():

1. Limpa a tela.
2. Consulta os dados das vendas, combinando informações das tabelas venda, livro e cliente.
3. Verifica se há registros. Se não houver, exibe uma mensagem de erro e retorna ao menu.
4. Exibe os dados de todas as vendas registradas em um formato organizado.
5. Mostra uma mensagem de sucesso ao concluir a listagem.
6. Retorna ao menu principal.

Essa função é útil para visualizar o histórico de vendas, permitindo ao usuário conferir detalhes como o livro vendido, o cliente que realizou a compra, a quantidade de cópias vendidas e o lucro obtido.