

Bases de Dados

Universidade de Aveiro

Pedro Martins, Pedro Santos



Bases de Dados

Departamento de Eletrónica Telecomunicações e
Informática

Universidade de Aveiro

Pedro Martins, Pedro Santos
pbmartins@ua.pt, pedroamaralsantos@ua.pt

03/05/2015

Resumo

As bases de dados são utilizadas em inúmeras situações, sendo indispensáveis não só no relacionamento dos seus elementos, mas essencialmente no arquivo de informações. Dada a sua aplicabilidade/utilidade e importância em guardar informações, as bases de dados informáticas, então, estão por toda a parte, desde os gigantescos bancos de dados do Facebook até a uma base dados de uma pequena biblioteca. Desde modo, foi proposta a criação de um banco de dados de uma biblioteca e o desenvolvimento de uma aplicação de interação com a mesma.

A base de dados foi dividida em três tabelas, cada uma dedicada exclusivamente a utilizadores, livros e requisições, e desenvolvida utilizando SQLite. As tabelas de utilizadores e livros estão diretamente relacionadas com a das requisições, sendo dois dos campos desta última tabela `Book_id` e `User_id`.

A aplicação, construída em Python, utilizando o módulo `sqlite3`, permite a interação de alto nível com a base dados, podendo aceder, adicionar, editar e remover quaisquer dados. A aplicação permite listar utilizadores e/ou as suas requisições, criar um utilizador, livro ou mesmo requisição, editar os dados dos mesmos, alargar prazos de requisições, listar livros em dívida, entre outras. Tudo isto através de uma simples interface na linha de comandos.

A construção em SQLite e a interação usando *queries* em Python torna-se bastante trivial usando as ferramentas mencionadas.

Agradecimentos

É deixada uma palavra de agradecimento ao professor João Paula Barraca pelas dúvidas esclarecidas.

Conteúdo

1	Introdução	1
2	Metodologia	2
2.1	Base de Dados	2
2.2	Aplicação de Interação com a Base de Dados	4
3	Análise	6
3.1	Base de dados SQLite	6
3.2	Aplicação em Python	6
3.2.1	Opção 1	7
3.2.2	Opção 2	9
3.2.3	Opção 3	10
3.2.4	Opção 4	10
3.2.5	Opção 5	12
3.2.6	Opção 6	15
3.2.7	Opção 7	17
4	Conclusões	19

Lista de Tabelas

3.1	Importação de módulos e conexão à base de dados	7
3.2	Função <code>increment_month</code> , para incrementar a data num mês.	15

Capítulo 1

Introdução

As bases de dados são sistemas organizacionais que permitem o fácil acesso e gerenciamento de grandes (ou pequenas) quantidades de dados. Hoje em dia, as bases de dados informáticas estão por todo o lado, desde os grandes servidores carregados de contas do Facebook até aos dados duma pequena mercearia.

Foi-nos proposto a criação de uma pequena base dados de uma biblioteca e de uma aplicação que pudesses interagir com a mesma, de maneira a podermos criar, alterar e ver os dados nela guardados. Para tal, também nos foi proposto o desenvolvimento do banco de dados em SQLite, uma Relational Database Management System (RDMS) que usa Structural Query Language (SQL) para executar os *queries*. Para a construção da aplicação será utilizado Python (versão 2.7) em parceria com o módulo `sqlite3`, que permite conectar um programa a uma base de dados SQLite3, e executar *queries* para aceder e alterar a informação pretendida.

Antes do desenvolvimento do programa, será apresentada a metodologia seguida, e, por fim, a análise do funcionamento do programa Python com vários testes funcionais.

Capítulo 2

Metodologia

Foi proposto a construção de um programa de software em Python que servisse para a comunicação com uma pequena base de dados de uma biblioteca. Primeiramente, analisar-se-á como será construída a base de dados e depois como será o desenvolvimento da aplicação de comunicação com a mesma. Na secção Capítulo 3, serão apresentados testes funcionais da aplicação.

2.1 Base de Dados

Uma base de dados é uma organização de dados, de modo a que estes sejam facilmente acessíveis e modificáveis. Em sistemas de computação, as bases de dados são classificadas mediante o seu sistema organizacional, sendo as mais comuns as relacionais, isto é, com os dados organizados de forma tabular, de tal modo a que possa ser reorganizado e acessível de diferentes maneiras, utilizando, por exemplo, SQL. Existem também outros tipos, tais como bases de dados orientadas a objetos, em que a informação está organizada por classes e subclasses.[1]

Os sistemas de gestão de bases de dados são um *software* informático que interagem com o utilizador e com a própria base dados. Os Database Management System (DBMS) foram desenhados para permitir o acesso, a criação, a atualização e a gestão da informação numa base de dados, por meio de *queries*. Para além disso, também são eles que guardam o "esquema", ou seja, a estrutura lógica interna da base de dados. [3]

Existem vários tipos de DBMS, entre os quais RDMS (o que se adapta à maioria dos casos), In-Memory Database Management System (IMDBMS) (melhor performance e rapidez de resposta em relação às demais) e, ainda, Cloud-Based Data Management System (CBDMS) (em que o fornecedor do serviço *cloud* é responsável pela manutenção do DBMS).

De entre os mais conhecidos destacam-se o MySQL, Microsoft SQL Server, Postgres and Oracle. Por norma, uma base de dados não pode comunicar

simultaneamente usando diferentes DBMS, no entanto, pode utilizar uma linguagem como SQL em parceria com Open Database Connectivity (ODBC) ou Java Database Connectivity (JDBC), de maneira a que a mesma aplicação de *software* possa funcionar utilizando múltiplos DBMS.

De entre aqueles que usam SQL, existem também diferentes tipos. Existem sistemas baseados em servidores (comunicação via *socket*), como o MySQL e o Microsoft SQL Server, bem como sistemas baseados em ficheiros, nomeadamente o SQLite. O SQLite é uma biblioteca baseada na linguagem C e, em contraste com outros sistemas de gestão de banco de dados, o SQLite não é um mecanismo que tem como base "cliente-servidor". Ao contrário do que acontece em MySQL, por exemplo, ele é o próprio servidor. Para além disso, o SQLite lê e escreve diretamente no disco, ou seja, toda uma base de dados está contida num único ficheiro.

Para a construção da base de dados será utilizado um sistema baseado em ficheiros, o SQLite3. Sendo um sistema baseado em SQL, a interação com a base de dados é feita através de *queries*, comandos textuais, os quais são sempre terminados em ";".

A base de dados da biblioteca a considerar será dividida em 3 tabelas, **books** (livros), **users** (utilizadores) e **requisitions** (requisições), ambas interligadas entre elas.

A tabela **books** terá 4 campos:

- Uma chave primária de identificação (INTEGER PRIMARY KEY);
- O título do livro (TEXT);
- O autor (TEXT);
- O estado de requisição (TEXT).

Por outro lado, a tabela **users**, isto é, a tabela relativa aos utilizadores, terá:

- Uma chave primária de identificação (INTEGER PRIMARY KEY);
- Nome (TEXT);
- Morada (TEXT);
- Contacto (INTEGER).

Por fim, a tabela das requisições (**requisitions**), que estará relacionada com as anteriores:

- Uma chave primária de identificação (INTEGER PRIMARY KEY);
- O identificador do livro (INTEGER);
- O identificador do utilizador (INTEGER);

- Data de requisição (DATE);
- Data limite de entrega (DATE);
- O estado de requisição (TEXT).

A base de dados seguirá o esquema apresentado na Figura 2.1.

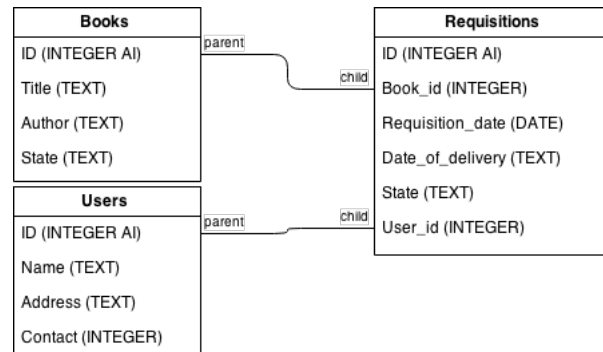


Figura 2.1: Esquema da base de dados.

Para a criação da base de dados, das tabelas e de alguns elementos nas respetivas tabelas (que apenas servem para teste do programa), existe na pasta Code, em anexo, um ficheiro de texto `BibDatabase.txt`, com todos os *queries* utilizados no desenvolvimento do ficheiro de base de dados, assim como o próprio ficheiro de base de dados, `BibDatabase.db`.

2.2 Aplicação de Interação com a Base de Dados

Para comunicar e gerir a base de dados, será desenvolvida uma aplicação em Python (versão 2.7), utilizando, entre outros, o pacote `sqlite3`. O programa consistirá em estabelecer uma comunicação com a base dados logo no seu arranque e, a partir daí, mostrar um menu, com diversas opções, que depois de executada uma das funções, é exibido até que o programa seja encerrado:

- Pesquisar por um livro pelo nome, autor e/ou estado de requisição;
- Listar todos os utilizadores;
- Listar requisições de um certo utilizador;
- Criar, apagar ou editar elementos da base de dados, sejam eles utilizadores, livros ou requisições;
- Entregar livros, notificar um livro perdido (cancelar requisição) e alargar prazo de entrega de um livro;

- Listar livros por entregar;
- Sair do programa.

Para a comunicação entre a base de dados e o programa, será necessário a importação do módulo `sqlite3` para a aplicação Python. Depois, deve-se executar a conexão à base dados e, sempre que se execute um *query* que altere dados nas tabelas, isto é, que não seja simplesmente de pesquisa, deve executar-se um *commit*, para que as tabelas sejam atualizadas depois da instrução. Por fim, depois de selecionada a opção de abandonar a aplicação, deve ser encerrada a conexão entre eles.

Capítulo 3

Análise

3.1 Base de dados SQLite

A base de dados, como referido anteriormente, seguirá o esquema apresentado na Figura 2.1, e todos os comandos necessários à construção das tabelas na própria base de dados estão em anexo. Foram criadas três tabelas (utilizadores, livros e requisições), assim como os devidos campos de informação de cada uma, e foram adicionados alguns utilizadores, livros e requisições que serviram como exemplo para a análise do funcionamento do programa.

3.2 Aplicação em Python

A aplicação desenvolvida em Python consiste num menu de opções que, depois de executada uma das opções do menu, é exibido até que se selecione a opção 0, "Terminar o programa". Depois de introduzir a opção que se pretende, o programa pesquisa num dicionário a função da opção escolhida e executa-a.

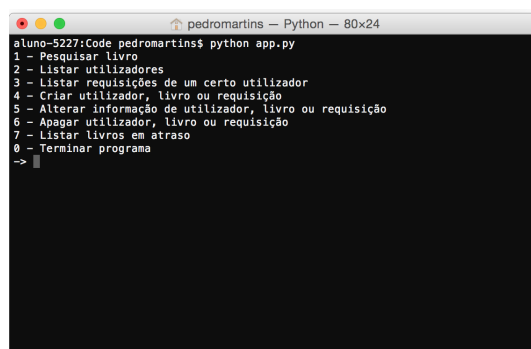
A screenshot of a terminal window titled 'pedromartins - Python - 80x24'. The terminal shows a command prompt 'aluno-5227:Code pedromartins\$ python app.py' followed by a menu of options: 1 - Pesquisar livro, 2 - Listar utilizadores, 3 - Listar requisições de um certo utilizador, 4 - Criar utilizador, livro ou requisição, 5 - Alterar informação de utilizador, livro ou requisição, 6 - Apagar utilizador, livro ou requisição, 7 - Listar livros em atraso, and 0 - Terminar programa. A cursor is visible on the line '0 - Terminar programa'.

Figura 3.1: Menu principal da aplicação.

Todo o código desta aplicação e o respetivo ficheiro que serve para a sua execução está em anexo na pasta Code e tem como nome `app.py`.

Depois de importados todos os módulos, nomeadamente o `sqlite3` (para conexão à base de dados), `calendar` e `datetime` (para atualização das datas das requisições), é estabelecida a ligação entre o programa e a base de dados, e definido que se pode utilizar *strings* com a codificação UTF-8 na mesma.

```
# encoding: utf-8
import sys
import sqlite3 as sql
import calendar
from datetime import date, timedelta

# Connect to database
db = sql.connect("BibDatabase.db")

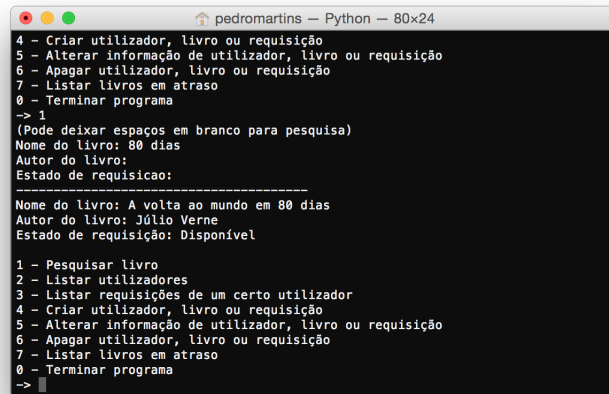
# Correct utf-8 characters input on database
db.text_factory = str
```

Tabela 3.1: Importação de módulos e conexão à base de dados

3.2.1 Opção 1

A opção 1 (Pesquisar livros) do menu principal da aplicação serve para pesquisar livros a partir do nome do livro, autor do livro e/ou estado de requisição. Sendo assim, aparecem os devidos campos para preencher, contudo, pode-se deixar espaços em branco. Se se deixar todos os espaços em branco irão ser listados todos os livros que existem na base de dados. A pesquisa é parcial, isto é, se se colocar apenas parte do título do livro, a aplicação irá retornar todos os livros que contenham a parte pesquisada. Se nenhum livro for encontrado, irá ser impressa no terminal a mensagem "Livro não encontrado". A função `option1` vai guardar num *array* as informações que foram digitadas nos vários campos e em seguida vai procurar na tabela `Books` correspondência aos campos preenchidos. Por último, vai imprimir os resultados da pesquisa.

Seguem-se vários exemplos de pesquisas:



```
pedromartins — Python — 80x24
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 1
(Pode deixar espaços em branco para pesquisa)
Nome do livro: 80 dias
Autor do livro:
Estado de requisicao:
-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisicao: Disponível
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
->
```

Figura 3.2: Pesquisa parcial de livros que contenham no titulo "80 dias".



```
pedromartins — Python — 80x24
0 - Terminar programa
-> 1
(Pode deixar espaços em branco para pesquisa)
Nome do livro:
Autor do livro:
Estado de requisicao:
-----
Nome do livro: O fim da Inocência
Autor do livro: Francisco Salgueiro
Estado de requisicao: Requisitado
-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisicao: Disponível
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
```

Figura 3.3: Pesquisa com todos os campos em branco.

```
pedromartins — Python — 80x24
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 1
(Pode deixar espaços em branco para pesquisa)
Nome do livro: 80 dias
Autor do livro: Verne
Estado de requisição: Requisitado

Livro não encontrado

1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
->
```

Figura 3.4: Pesquisa em que nenhum livro não é encontrado.

3.2.2 Opção 2

A opção 2 (Listar utilizadores) do menu principal serve para visualizar o nome, morada e contacto de todos os utilizadores da biblioteca que estão inseridos na base de dados. A função `option2` recolhe todos os dados dos utilizadores da base de dados e guarda-os num *tuple*. Por fim, as informações são imprimidas no ecrã.

```
aluno-5227:Code pedromartins$ python app.py
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 2

-----
Nome: Pedro Emanuel Amaral Santos
Morada: Viseu
Contacto: 911242532

-----
Nome: Joana Bernardino Gomes
Morada: Aveiro
Contacto: 911415324

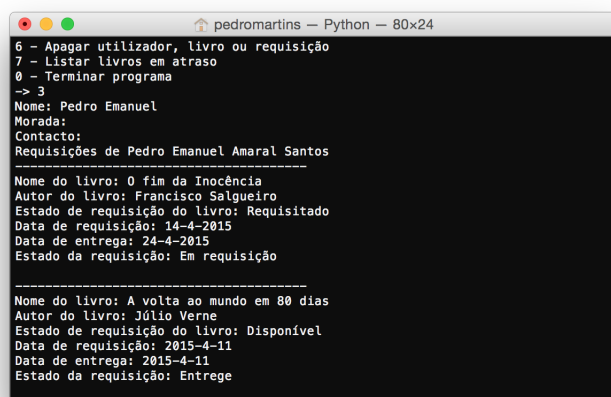
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
```

Figura 3.5: Listagem de todos os utilizadores inseridos na base de dados.

3.2.3 Opção 3

Na opção 3 (Listar requisições de um certo utilizador) , pesquisando pelas informações de um utilizador, o programa mostra todas as suas requisições. Tal como noutras opções, também é utilizada a função `get_user_data` para ler e validar as informações do teclado. A função `option3` irá, através do que foi inserido nos campos, fazer uma pesquisa na tabela `Users` (usando a função `search_user`, que devolve um *tuple* com todas as informações dos utilizadores encontrados), e, caso descubra pelo menos um utilizador, irá procurar pelas requisições às quais o campo `User_id` é igual ao ID encontrado anteriormente. Caso esse utilizador não tenha nenhuma requisição, será impressa uma mensagem informando tal.

Caso o *tuple* resultante dessa pesquisa tenha uma dimensão superior a 0, o programa irá executar outra *query*, desta feita à tabela `Books`, para imprimir as informações dos livros em questão.



```
pedromartins - Python - 80x24
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 3
Nome: Pedro Emanuel
Morada:
Contacto:
Requisições de Pedro Emanuel Amaral Santos
-----
Nome do livro: O fim da Inocência
Autor do livro: Francisco Salgueiro
Estado de requisição do livro: Requisitado
Data de requisição: 14-4-2015
Data de entrega: 24-4-2015
Estado da requisição: Em requisição

-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisição do livro: Disponível
Data de requisição: 2015-4-11
Data de entrega: 2015-4-11
Estado da requisição: Entregue
```

Figura 3.6: Listagem das requisições de um utilizador.

3.2.4 Opção 4

A opção 4 (Criar utilizador, livro ou requisição) tem um submenu com 3 opções: utilizador, livro ou requisição. Cada uma destas opções adiciona respetivamente um utilizador, um livro ou cria uma requisição na base de dados da biblioteca, depois de corretamente preenchidos todos os campos pedidos.

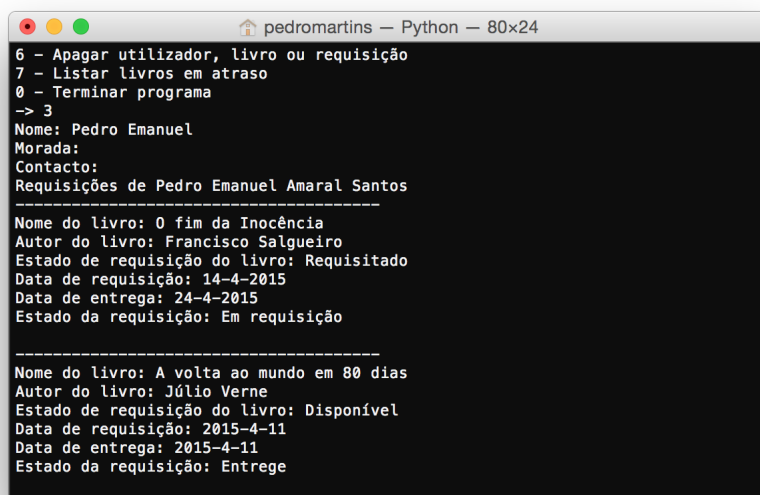
No caso de se pretender adicionar um livro, é necessário colocar o nome do livro e o respetivo autor. Caso se pretenda adicionar um utilizador, é necessário inserir o nome, morada e contacto. Por último, se for uma

requisição, tem de se inserir o título do livro, o nome do requisitante e o seu respectivo contacto.

Já em relação ao seu funcionamento interno, quando se selecciona a opção 1 do submenu, o programa verifica se não foram deixados campos por preencher e ainda se o utilizador que tentamos inserir ainda não existe. Por fim, insere o utilizador na tabela **Users** da base de dados.

A opção 2 do submenu é semelhante à anterior: verifica também se não foram deixados campos por preencher e se não existe nenhum livro já com as informações inseridas. Se as informações foram validadas, o livro é adicionado com sucesso na tabela **Books** da base de dados.

Por último, relativamente à opção de criar uma nova requisição, depois de preenchidos os campos relativamente ao livro, se este existir e caso sejam encontrados vários resultados na pesquisa, é necessário dizer o **Book_id** do livro que se quer requisitar. Em seguida, preenchem-se os campos relativamente ao utilizador e o programa verifica, também, se este existe e se todos os campos foram preenchidos. Por fim, é verificado se o livro já está requisitado ou se foi perdido anteriormente. Caso uma das situações referidas aconteça, não fica concluída a requisição. Para além disso, é guardada automaticamente a data do dia em que é feita a requisição, assim como a data limite de entrega, utilizando a função **increment_month**, que incrementa 32 dias num objeto do tipo **date** (isto é, o limite de entrega é passado 1 mês do dia da requisição). Toda esta informação é guardada na tabela **Requisitions**.



```
pedromartins — Python — 80x24
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 3
Nome: Pedro Emanuel
Morada:
Contacto:
Requisições de Pedro Emanuel Amaral Santos
-----
Nome do livro: O fim da Inocência
Autor do livro: Francisco Salgueiro
Estado de requisição do livro: Requisitado
Data de requisição: 14-4-2015
Data de entrega: 24-4-2015
Estado da requisição: Em requisição
-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisição do livro: Disponível
Data de requisição: 2015-4-11
Data de entrega: 2015-4-11
Estado da requisição: Entregue
```

Figura 3.7: Listagem de todos os livros requisitados por um utilizador.

3.2.5 Opção 5

A opção 5 (Alterar informação de utilizador, livro ou requisição) do menu principal da aplicação serve para editar informações já existentes na base de dados da biblioteca. Quando selecionada a opção, aparece um submenu com 3 opções: escolher se queremos editar informações de um utilizador, de um livro ou de uma requisição.

Ao escolher a opção 1, isto é, alterar informações de um utilizador, pesquisamos pelos dados do mesmo (não sendo necessário preencher todos os campos) e os resultados da pesquisa serão impressos no ecrã. Caso exista apenas um resultado na pesquisa, o programa pede imediatamente as novas informações ao utilizador, caso contrário, pede o identificador (ID) do utilizador que se quer editar. Aquando da inserção de novos dados, caso se deixe algum campo em branco, as informações desse campo manter-se-ão iguais às originais. Tudo isto pode verificar-se na Figura 3.8.



```
pedromartins — Python — 83x24
0 - Terminar programa
-> 5
1 - Utilizador
2 - Livro
3 - Requisição
-> 1
Pesquisar utilizador a editar:
Nome: Pedro
Morada:
Contacto:
Foram encontrados os seguintes resultados:
-----
Identificador na BD: 1
Nome: Pedro Emanuel Amaral Santos
Morada: Viseu
Contacto: 911242532

Novos dados do utilizador:
(Caso deixe campos em branco, os valores antigos desse campo não se vão alterar)
Nome: Pedro Santos
Morada:
Contacto:
Informações do utilizador atualizadas com sucesso!
1 - Pesquisar livro
```

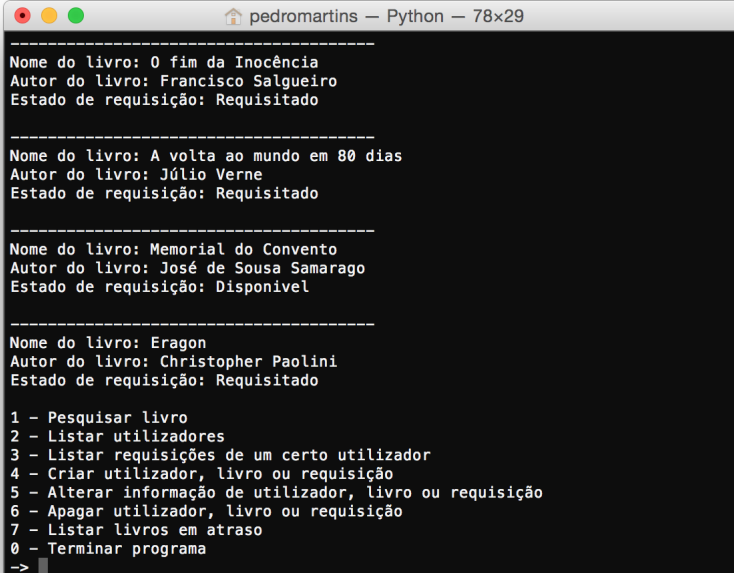
Figura 3.8: Alteração das informações de um utilizador.

Escolhendo a opção 2 do submenu, dedicada à edição de livros, o processo é em tudo semelhante ao da opção anterior, mudando, apenas, as informações que se podem alterar, excetuando o estado de requisição de um livro, que somente é alterado consoante a criação, cancelamento (livro perdido) ou encerramento (entrega do livro) de requisições.

Por outro lado, quando é escolhida a opção das requisições (opção 3), depois de se pesquisar pela requisição que se procura alterar, podem acontecer duas situações: se a requisição não estiver em curso (cancelada ou encer-

rada), não é possível editar as informações da mesma e o programa imprime no terminal "A requisição escolhida já não pode ser alterada!"; caso a requisição ainda esteja em curso, aparece outro submenu com 3 opções: entregar um livro, cancelar uma requisição, isto é, declarar o livro como perdido, ou alargar o prazo de entrega do livro. As entregas de livro apenas "fecham" a requisição e alteram o estado do livro para "Disponível"; as outras duas opções serão explicadas com o exemplo abaixo.

Segue-se um exemplo em que nos focaremos no livro Eragon, requisitado pela Joana Gomes no dia 28 de Abril de 2015. A Figura 3.9 mostra a listagem de alguns livros da base de dados (podemos verificar que o livro em questão já se encontra requisitado).



```
pedromartins — Python — 78x29
-----
Nome do livro: O fim da Inocência
Autor do livro: Francisco Salgueiro
Estado de requisição: Requisitado
-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisição: Requisitado
-----
Nome do livro: Memorial do Convento
Autor do livro: José de Sousa Samarago
Estado de requisição: Disponível
-----
Nome do livro: Eragon
Autor do livro: Christopher Paolini
Estado de requisição: Requisitado

1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
->
```

Figura 3.9: Listagem de alguns livros da base de dados.

Imagine-se, que a Joana quer alargar o prazo de entrega da requisição do livro Eragon. Deste modo, executa-se a opção 5 do menu e logo de seguida escolhe-se editar uma requisição. Como se vê na Figura 3.10, existem múltiplos resultados da pesquisa, mas deverá ser selecionado a requisição que está em curso, visto que é impossível editar uma requisição já encerrada (livro entregue), como se verificar na Figura 3.11.

```
pedromartins - Python - 78x29
Pesquisar requisição a editar:
Nome do livro: Eragon
Livro encontrado!
Nome do requisitante: Joana
Contacto do requisitante: 911415324
Foram encontrados os seguintes resultados:
-----
Identificador na BD: 4
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Requisitado
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-5-2015
Estado da requisicao: Entregue
-----
Identificador na BD: 6
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Requisitado
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-5-2015
Estado da requisicao: Em curso
ID da requisição a editar -> █
```

Figura 3.10: Resultados da pesquisa na edição de requisições.

```
pedromartins - Python - 78x29
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-5-2015
Estado da requisicao: Entregue
-----
Identificador na BD: 6
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Requisitado
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-5-2015
Estado da requisicao: Em curso
ID da requisição a editar -> 4
Entregue
A requisição escolhida já não pode ser alterada!
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> █
```

Figura 3.11: Escolha de requisição já encerrada.

Como podemos ver na Figura 3.12, a data limite de entrega foi incrementada num mês. A execução desta função tem como base a função `increment_month`, a qual aceita um objeto do tipo `date` como argumento, do módulo `datetime`, e incrementa a data em 32 dias, isto é, um mês. Segue-se o código da função `increment_month`, criado por Matthew Schinckel:

```
def increment_month(date_input):
    # Go to first of this month, and add 32 days to get to the next month
    next_month = date_input.replace(day=1) + timedelta(32)
    # Get the day of month that corresponds
    day = min(date_input.day, calendar.monthrange(next_month.year, next_month.month)[1])
    return next_month.replace(day=day)
```

Tabela 3.2: Função `increment_month`, para incrementar a data num mês.

Na mesma imagem, podemos também verificar que escolhendo a opção 2, cancelamos a requisição e damos o livro como perdido. Sendo assim, nunca mais o mesmo pode ser requisitado.

```
pedromartins — Python — 78x29
Data limite de entrega: 28-5-2015
Estado da requisicao: Entregue

-----
Identificador na BD: 6
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Requisitado
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-6-2015
Estado da requisicao: Em curso

ID da requisicao a editar -> 6
1 - Entregar livro
2 - Cancelar requisicao (em caso de livro perdido)
3 - Aumentar o prazo
-> 2
Requisicao cancelada com sucesso!
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisicoes de um certo utilizador
4 - Criar utilizador, livro ou requisicao
5 - Alterar informacao de utilizador, livro ou requisicao
6 - Apagar utilizador, livro ou requisicao
7 - Listar livros em atraso
0 - Terminar programa
->
```

Figura 3.12: Escolha de requisição já encerrada.

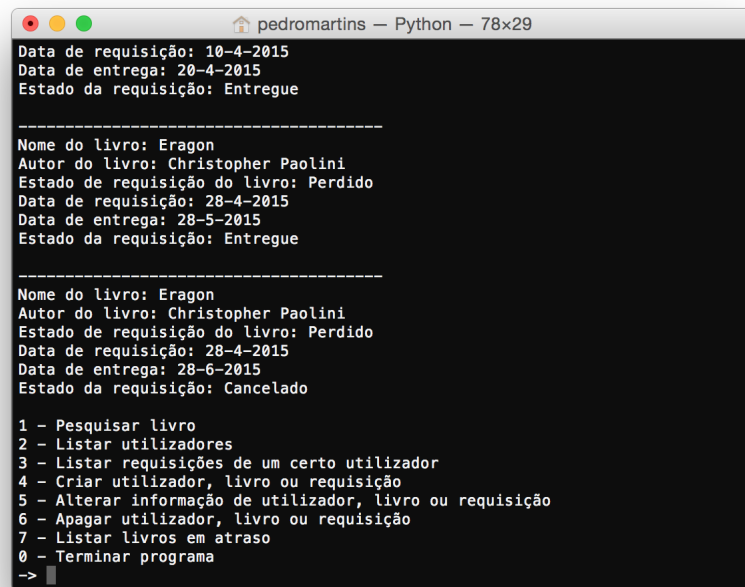
Escolhendo a opção 1, o requisitante entregaria o livro à biblioteca e o mesmo voltaria a estar disponível para outra requisição.

3.2.6 Opção 6

A opção 6 do menu é simplesmente a remoção de elementos da base de dados, sejam eles utilizadores, livros ou requisições, sendo estes maioritariamente usados para apagar elementos com erros de introdução. Tal como na opção 5, também aparece um submenu com as 3 diferentes opções (utilizadores, livros ou requisições). Basicamente, todas elas funcionam inserindo dados completos ou parciais para pesquisa e, em caso de vários resultados na mesma, escolhe-se qual o elemento a apagar. No entanto, a opção das requisições tem uma particularidade: ao eliminar essa requisição, caso o livro

tenha sido dado como perdido noutra requisição não igual à que se pretende eliminar, o estado do livro manter-se-á "Perdido". Caso contrário, o livro ficará novamente disponível para ser requisitado.

Na Figura 3.13, podemos verificar que, listando as requisições da Joana, existem duas requisições de Eragon, uma em que o livro foi entregue e outra que foi cancelada, logo, em ambas, é notificado que o livro está perdido.



```
pedromartins - Python - 78x29
Data de requisição: 10-4-2015
Data de entrega: 20-4-2015
Estado da requisição: Entregue

-----
Nome do livro: Eragon
Autor do livro: Christopher Paolini
Estado de requisição do livro: Perdido
Data de requisição: 28-4-2015
Data de entrega: 28-5-2015
Estado da requisição: Entregue

-----
Nome do livro: Eragon
Autor do livro: Christopher Paolini
Estado de requisição do livro: Perdido
Data de requisição: 28-4-2015
Data de entrega: 28-6-2015
Estado da requisição: Cancelado

1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
->
```

Figura 3.13: Listagem de requisições.

Se a requisição que foi cancelada for removida da base de dados, tal como referido anteriormente, o estado do livro será na mesma "Perdido", como se pode verificar nas Figura 3.14 (em que se pesquisa pela requisição a remover) e Figura 3.15 (listagem das requisições já depois da referida ter sido removida).

```
pedromartins — Python — 78x29
-> 3
Pesquisar requisição a remover:
Nome do livro: Eragon
Livro encontrado!
Nome do requisitante: Joana
Contacto do requisitante: 911415324
Foram encontrados os seguintes resultados:
-----
Identificador na BD: 4
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Perdido
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-5-2015
Estado da requisicao: Entregue
-----
Identificador na BD: 6
Nome do livro: Eragon
Autor: Christopher Paolini
Estado de requisicao: Perdido
Nome do requisitante: Joana Bernardino Gomes
Contacto: Aveiro
Data de requisicao: 28-4-2015
Data limite de entrega: 28-6-2015
Estado da requisicao: Cancelado
```

Figura 3.14: Remoção de uma requisição da base de dados.

```
pedromartins — Python — 78x29
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 3
Nome: Joana
Requisições de Joana Bernardino Gomes
-----
Nome do livro: A volta ao mundo em 80 dias
Autor do livro: Júlio Verne
Estado de requisição do livro: Requisitado
Data de requisição: 10-4-2015
Data de entrega: 20-4-2015
Estado da requisição: Entregue
-----
Nome do livro: Eragon
Autor do livro: Christopher Paolini
Estado de requisição do livro: Perdido
Data de requisição: 28-4-2015
Data de entrega: 28-5-2015
Estado da requisição: Entregue
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
```

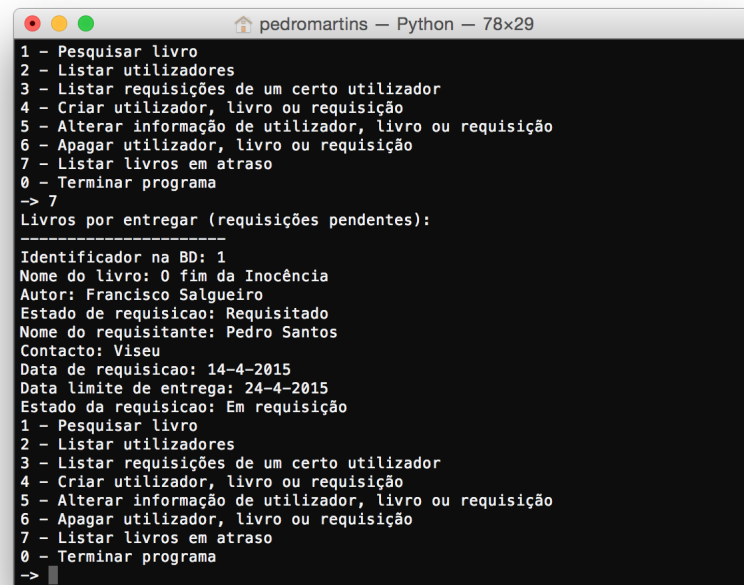
Figura 3.15: Listagem de requisições, depois de removida uma requisição.

3.2.7 Opção 7

A opção 7 do menu tem como objetivo listar as requisições ativas para lá da data de entrega, isto é, os livros que já deviam ter sido entregues, mas ainda estão na posse do requisitante. A função base do seu funcionamento é muito simples: consiste apenas em percorrer os elementos da tabela das requisições e comparar as datas limite de entrega (extraíndo a *string* da data presente base de dados e convertendo-a num objeto do tipo *date*) e verificar

se a mesma ultrapassa o presente dia (utilizando a função `date.today()` que constrói um objeto do tipo `date`) e, caso a requisição tenha como estado "Em curso", imprime as informações da requisição em questão.

A Figura 3.16 mostra um exemplo do seu funcionamento.



```
pedromartins — Python — 78x29
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 7
Livros por entregar (requisições pendentes):
-----
Identificador na BD: 1
Nome do livro: O fim da Inocência
Autor: Francisco Salgueiro
Estado de requisicao: Requisitado
Nome do requisitante: Pedro Santos
Contacto: Viseu
Data de requisicao: 14-4-2015
Data limite de entrega: 24-4-2015
Estado da requisicao: Em requisição
1 - Pesquisar livro
2 - Listar utilizadores
3 - Listar requisições de um certo utilizador
4 - Criar utilizador, livro ou requisição
5 - Alterar informação de utilizador, livro ou requisição
6 - Apagar utilizador, livro ou requisição
7 - Listar livros em atraso
0 - Terminar programa
-> 
```

Figura 3.16: Listagem de requisições em dívida.

Capítulo 4

Conclusões

Em suma, como se pode verificar neste trabalho, a interação de uma aplicação em Python com uma base de dados SQL, neste caso, usando SQLite3, torna-se bastante simples, importando o módulo Python `sqlite3`. Tudo se resume a fazer a ligação a uma base de dados usando `db = sqlite3.connect(ficheiro da DB)` e escrevendo *queries* que retornam *tuples* com os resultados do que procuramos. Sendo assim, desenvolveu-se uma aplicação simplificada para gerir uma pequena biblioteca através da linha de comandos com uma interface *user friendly*.

Acrónimos

DBMS Database Management System

RDMS Relational Database Management System

IMDBMS In-Memory Database Management System

CBDMS Cloud-Based Data Management System

SQL Structural Query Language

ODBC Open Database Connectivity

JDBC Java Database Connectivity

Bibliografia

- [1] SearchSQLServer, *What is a database management system (dbms)?*, [Online; acedido em Maio 2015]. endereço: <http://searchsqlserver.techtarget.com/definition/database-management-system>.
- [2] Wikipedia, *Sql*, [Online; acedido em Maio 2015]. endereço: <http://en.wikipedia.org/wiki/SQL>.
- [3] —, *Database*, [Online; acedido em Maio 2015]. endereço: <http://en.wikipedia.org/wiki/Database>.