**Projeto de Bases de Dados**

**Parte 3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Grupo:** 50 | | | |
| **Número** | **Nome** | **Contribuição** | **Esforço (em Horas)** |
| 93695 | Catarina Sofia dos Santos Sousa | 33% | 5h |
| 93743 | Nelson Alexandre Geada Trindade | 34% | 5h |
| 93754 | Rodrigo Rodrigues Major | 33% | 5h |

**Sala:** 1-29

**Turno:** 4ºFeira – 15h00

**Docente:** Carlota De Oliveira Lopes Dias

**Arquitetura da Aplicação Flask**

Para a aplicação funcionar da maneira como foi concebida, tivemos de criar comandos que alterassem as tabelas anteriormente criadas. Uma das alterações foi o “update cascade”, que faz com que ao alterar alguma informação numa tabela, se esta for foreign key de outra, então esta informação vai ser alterada nesta segunda tabela. Na outra alteração, utilizamos o “drop cascade”, onde ao eliminar um registo, se ele existir como foreign key noutra tabela, então o registo contendo informações desta linha também será eliminada na segunda tabela. (Ver ficheiro alter.sql)

Para além disto tivemos de dar update a alguns valores da tabela análise, para que, no caso das análises de glicémia, estas possam ser vistas no browser, uma vez que anteriormente nenhum registo continha o nome “glicemia”.

A aplicação foi desenvolvida utilizando Python e uma library chamada Flask. Esta biblioteca permite, com o auxílio do python, desenvolver e manipular páginas html para apresentar os dados/queries indo ao servidor da nossa base de Dados e executando os pedidos. O código esta a ser executado no webserver do tecnico, onde pode testar através deste link

<https://web2.ist.utl.pt/~ist193743/BD_P3/app.cgi/>

Ao entrar no link, temos uma página inicial (index.html) onde estão todos os links de todo o trabalho desenvolvido, distribuído pelas suas alíneas.

Tanto a alínea a), como b), utilizam os mesmos estilo de paginas de html. Cada Tabela tem 3 html’s, um para mostrar o seu conteúdo (x.html), outro para modificar o conteúdo de uma linha (xUpdate.html) e ainda um para inserir uma linha na tabela (xInsert.html) (x = Nome\_Tabela). Para eliminar um registo, não é necessário nenhum html. A aplicação apenas direciona o utilizador para um URL no qual o código que está a ser executado pelo python vai tratar de remover a linha e redirecionando-o de volta à página que mostra o seu conteúdo.

Na alínea c) fizemos três páginas de html. A primeira é um formulário (registoVenda.html) para ser preenchido pelo utilizador e é constituído por uma parte para quem tenha prescrição e outra para fazer uma venda unitária. Ao fazer o pedido de prescrição, este será direcionado para uma nova página no qual mostra o conteúdo da tabela prescricao\_venda (prescricao\_venda.html). Ao fazer o pedido de venda única, este será direcionado para uma nova página no qual mostra o conteúdo da tabela venda\_farmácia (venda\_farmacia.html).

Na alínea d) é apenas constituída por duas páginas de html. A primeira é um formulário (perguntad.html) para ser preenchido pelo utilizador com o mês no qual deseja ver e também o ano (default 2020), para além de especificar o número de cédula do médico em questão. Esta página direciona para uma outra página html (perguntadTable.html) que mostra o resultado da query criada para este caso, com as variaveis no formulário inseridos. Aconselhamos uma vez que não temos muitos dados para este exemplo a verificar valores com num\_cedula =1, mês=7, ano=2020.

Na alínea e) é apenas constituída por uma página de html. Esta página (perguntae.html) é uma página que apenas mostra uma tabela com o resultado da query criada para esta pergunta:

With temp as (

  SELECT num\_concelho, num\_doente, COUNT(a.nome) as counter

  FROM analise a

  INNER JOIN instituicao i

  ON a.inst = i.nome

  WHERE a.nome='glicemia'

  GROUP BY num\_concelho, num\_doente

  ORDER BY num\_concelho ASC

  )

  SELECT \*

  FROM (SELECT num\_concelho, num\_doente as maxi\_id, counter as maxi

      FROM temp sub

        NATURAL JOIN

        (SELECT num\_concelho,MAX(sub.counter) as counter

          FROM temp sub GROUP BY num\_concelho) sub2) al

    NATURAL JOIN (SELECT num\_concelho, num\_doente as minimo\_id, counter as mini

      FROM temp sub

        NATURAL JOIN

        (SELECT num\_concelho,MIN(sub.counter) as counter

            FROM temp sub GROUP BY num\_concelho) sub2) al2

  ORDER BY num\_concelho ASC, maxi DESC, mini DESC;