Desenvolvimento do Projeto de Banco de Dados da Disciplina MAC0350

Etapa 3. Publicada em 08/06/2020. Data de entrega em 30/06/2020

Nesta Etapa 3 do Desenvolvimento do Projeto de Banco de Dados da Disciplina MAC0350 teremos dois desafios. São eles:

1) A evolução dos projetos conceitual, lógico e físico do banco de dados apresentados na Etapa 2, de modo a representar o conceito de Pessoa. Esse conceito é uma classe que generaliza as entidades Usuário e Paciente. Uma instância da classe Pessoa pode ser especializada em Usuário e/ou Paciente. Nem todas as instâncias da classe Pessoa estão representadas nas classes filhas Usuário e Paciente. Neste primeiro desafio, refaça os projetos conceitual, lógico e físico do banco de dados apresentado na Etapa 2 adequando as instâncias necessárias.

2) Desenvolvimento de uma interface gráfica para inserção, atualização, eliminação e consulta das entidades Usuário, Perfil, Serviço e Exame. Neste desafio 2, vamos desenvolver uma aplicação web que gerencie o modelo de banco de dados evoluído para Etapa 3. Essa aplicação web deverá ter o seu código fonte entregue via e-Disciplinas para a reprodução e correção da Etapa3. Para isso, vamos utilizar o framework Django, um MVC que utiliza a linguagem Python para o desenvolvimento de aplicações web. Mais concretamente vamos apresentar, na décima nona aula (09/06/2020) virtual de MAC0350 exemplos e orientações de como realizar consultas SQL no SGBDR PostgreSQL com o framework Python Django. Adicionamente, para documentar detalhes e orientações de como utilizar Python com o Django apresentamos o que se segue.

2.1) Dependências Python(pip):
django
django-extensions(opcional)
psycopg2-binary

2.2) Dependências de sistema:
postgresql
graphviz(opcional)

2.3)Instruções de uso
O sistema a ser implementado utiliza apenas uma fração das funcionalidades do framework MVC Django para integração de um SGBDR com um sistema web. Para maiores detalhes e especificidades da arquitetura do Django, recomendamos a leitura do tutorial disponível em: https://docs.djangoproject.com/en/3.0/intro/tutorial01/

Primeiramente, é necessário instalar as dependências listadas acima. As dependências de pacotes Python podem ser instaladas via pip, possivelmente em um ambiente virtual, ou via gerenciador de pacote de sua distribuição linux, assim como as demais dependências de sistema. Caso já não possua uma instalação do Postgresql disponível, recomendamos consultar instruções específicas para instalá-lo em seu sistema operacional. Quem implementou a parte 2 do projeto utilizando o pgadmin da disciplina, poderá utilizar seu banco de dados acessando-o diretamente em data.ime.usp.br na porta 53231, desde que esteja utilizando a vpn da usp.

Para executar esse exemplo:

- ajuste os parâmetros de conexão com o banco de dados em mysite/settings.py;

- dentro do diretório raiz do projeto, execute o comando: python manage.py runserver. Caso não ocorram erros, você verá uma mensagem terminada em:

Django version 3.0.7, using settings 'mysite.settings'

Starting development server at http://127.0.0.1:8000/

Quit the server with CONTROL-C.

- Em seguida, abra seu navegador e vá até o endereço localhost:8000/example. Você verá uma mensagem indicando que o sistema está executando.

- Os dois exemplos de queries estão disponíveis em localhost:8000/example/query1 e localhost:8000/example/query2.

O Django oferece uma interface para administrar as tabelas do banco de dados disponível em localhost:8000/admin. No entanto, para utilizar essa funcionalidade é preciso fazer o mapeamento objeto-relacional das tabelas as quais deseja-se gerenciar por meio dessa interface. Isso é feito no arquivo example/models.py desse projeto. Note que, os dois exemplos apresentados utilizam apenas consultas diretas ao banco de dados e, por simplicidade, exibem diretamente seus resultados sem mapeá-los para objetos python.

As interfaces de administração do Django requerem um superusuário e senha, os quais podem ser configurados por meio do comando python manage.py createsuperuser. Nesse exemplo, o usuário foi definido como "admin" e senha "admin". O código dos dois exemplos de consulta está disponível no arquivo example/views.py, enquanto que suas respectivas páginas web são declaradas em example/urls.py e renderizadas com os templates disponíveis em example/templates/example/

Para quem não tem familiaridade com o Django e quer utilizar esse exemplo como base da Etapa 3 do projeto de mac0350, recomendamos fortemente que sigam o tutorial citado acima e compare-o com o código desse exemplo.

2.4) Opcionalmente, após gerar o mapeamento objeto-relacional, é possível gerar um diagrama UML do referido modelo físico. Isso pode ser feito executando-se o seguinte comando em um terminal dentro do diretório raiz do projeto:

```
python manage.py graph_models example | dot -T pdf > model.pdf
```

Esse comando requer a instalação das dependências adicionais django-extensions e graphviz.

2.5) ATENÇÃO!
Antes de executar esse exemplo, FAÇA UM BACKUP DO SEU BANCO DE DADOS ! Ao implementar o mapeamento objeto-relacional, o Django irá criar tabelas adicionais em seu banco de dados ao executar o comando python manage.py migrate.