Ficha de trabalho nº 1	
Disciplina	**Modelação de dados em Engenharia**
Ano Lectivo	2020/2021
Objectivo	Modelar um problema num contexto de engenharia Electrotécnica
Aulas	5 aulas x 3 horas + 12 horas extra
Data de Entrega	**2021/04/26**

Objectivos concretos:

1. Modelação de dados
    a. Compreender e interiorizar os conceitos de modelo e o papel da modelação no contexto de engenharia Eletrotécnica.
    b. Representação de modelos (linguagens e ferramentas).
    c. Tipo de modelos com enfase nos tipos conceptuais/lógicos/físicos para a modelação de dados.
    d. O Modelo de Entidades e relacionamentos (DER).
    e. Modelar e implementar a solução para o problema apresentado em baixo.

2. DBMS (Database Management System)
    a. Passagem dum esquema lógico para um esquema físico
    b. SQL (para as componentes DDL[1] e DML[2])
    c. JDBC[3]
    d. ORACLE

3. Ilustrar a utilização do modelo num contexto prático de aplicação
    a. Ilustrar com JAVA/C++/C#, ou outro ambiente onde o aluno se sinta mais confortável, a utilização dum modelo de dados num contexto prático de aplicação (esta componente vale no máximo 10% da avaliação deste trabalho).
    b. Relação com outras disciplinas (Integração de Sistemas, Supervisão Inteligente, Robótica).

4. Responder à ficha de trabalho (no Moodle) ao longo e até ao final do trabalho, para ser entregue na data de entrega do trabalho.

[1] Data Definition Language
[2] Data Manipulation Language
[3] JAVA Database Connectivity (ODBC: Open Database Connectivity) ou abordagem "Web Application"

Descrição do trabalho a efectuar

Este trabalho destina-se a proporcionar aos alunos um contacto com os conceitos básicos acerca do projeto/modelação de bases de dados, tendo em vista a sua posterior utilização em atividades enquadradas no seio da Engenharia Eletrotécnica e de Computadores.

Concretamente, o estudo e a implementação do problema proposto neste trabalho servirão para fazer uma abordagem aos seguintes conceitos, que constituem as fases de um projeto de criação de uma base de dados:

- Estudo do problema apresentado.
- Modelação dum diagrama de Entidades e Relacionamentos (*DER*) que satisfaça os requisitos do problema apresentado.
- Transformação do modelo num esquema relacional e implementação numa base-de-dados, utilizando o *SQL*.
- Implementação de questões (*SQL queries*), vistas (*SQL views*) e funções (*SQL functions/procedures*) de forma a implementar as funcionalidades pretendidas no problema.
- Implementação de um conjunto de interfaces amigáveis para a visualização e manipulação dos dados armazenados na BD (*JDBC*).

Descrição geral do problema (a parte a azul é a mais importante)

Atualmente, as pessoas têm manifestado maior preocupação em manterem estilos de vida saudáveis. Um dos fatores que permitiram esta tendência foi a crescente disponibilidade de dispositivos (bandas, relógios e outros) que, tipicamente associados aos smartphones, permitem monitorizar e proporicionar informação relativao às condições físicas e bem-estar de cada pessoa.

Mercê desta realidade, a empresa Smart-Banda (um novo unicórnio Português), procedeu à criação dum novo produto/plataforma, designado de myFitness, com o objectivo de monitorizar e oferecer recomendações e alertas relativos ao bem estar dos associados. O produto integra também uma App com o mesmo nome (myFitness).

Considere que foi contratado como engenheiro para participar no desenvolvimento da base-de-dados (BD) para o lado backend da plataforma myFitness (considere que a BD para a App está a ser desenvolvida por uma equipa diferente). A BD da plataforma é importante, pois mercê da quantidade massiva de dados recolhidos e utilizando métodos "Machine Learning", consegue-se detetar padrões e situações importantes, que resultam em notificações e alertas enviados para os respetivos associados.

Para que a plataforma assegure a privacidade dos associados, o único elemento partilhado entre o back-end e cada utilizador é um código alfanumérico (UUID) partilhado entre o back-end e cada App (cada interação entre a App e o backend utiliza este código como identificação).

Em termos de requisitos para a BD, é necessário identificar cada associado (doravante identificado como utilizador), por exemplo, o UUID, a data de inscrição e o tipo de serviço ("gratuito", "premium") e se o utilizador está ativo (valor booleano). Cada utilizador possui um perfil que descreve as suas características físicas (peso, altura, ano/mês de nascimento, etc.). A cada utilizador corresponde também um conjunto de objetivos diários, por exemplo, o nº de passos, a quantidade de calorias, etc. A plataforma (incluindo a App) considera que os utilizadores façam um conjunto de tipos de exercícios (corrida, caminhada, bicicleta, etc.) e em

que cada exercício afeta uma ou mais variáveis (passos, calorias, distância, ……..). Portanto, quando um utilizador vai iniciar uma atividade física, escolhe então o tipo de exercício, sendo que no final deste, deve-se ter como resultado, o nome do exercício, os instante inicial/final e os valores das variáveis associadas com o exercício. Decorrente do seu perfil e das suas atividades físicas, cada utilizador possui a informação agregada, designada de valores diários, para cada data (data, passos, calorias, peso, distância, etc).

Considere que a Smart-Banda recorreu á ajuda de peritos em medicina e desposto para se poder obter estas especificações.

Para captar associados, foi atribuído um valor mensal de 0.99 Euros durante o período de lançamento da plataforma. Os movimentos monetários ficam registados como pagamentos (uuid, data, valor, etc).

Requisitos funcionais para o problema apresentado

O desenvolvimento do projeto para o cenário apresentado requer a construção dum modelo para a base-de-dados, que consiga satisfazer um conjunto considerável de requisitos. No entanto, dado o tempo disponível para este trabalho, é apenas necessário que o modelo consiga satisfazer os requisitos funcionais definidos na tabela 1.

Tabela 1 – Requisitos funcionais

Req. Func.	Descrição
RF 1	Implementar as operações CRUD[4] da BD em termos de utilizadores, perfis, objetivos diários e tipos de exercícios.
RF 2	Proceder ao registo de exercícios físicos efetuados por cada utilizador.
RF 3	Proceder ao registo de notificações, sempre que após um exercício, algum parâmetro possua um valor abaixo ou acima duma gama recomendada (por exemplo, o ritmo cardíaco).
RF 4	Visualizar os valores diários para um utilizador entre duas datas.
RF 5	Visualizar as notificações dum utilizador nos últimos 7 dias.
RF 6	Visualizar utilizadores com ritmo cardíaco abaixo ou acima de certos limiares, considerando a sua idade.
RF 7	Visualizar cada utilizador e o total pago desde uma data anterior (fornecida) até à data actual.
RF 8	Visualizar os clientes não ativos. Para cada um deles, visualizar as datas de inicio/fim do período de activo e eventualmente, o total de valor pago.
RF 9	*Proponha um requisito relevante e ainda por identificar que implique a especificação de uma ou mais entidades. Implemente.*
RF 10	*Proponha um requisito relevante ainda por identificar e que requeira uma query simples para o satisfazer. Implemente.*
RF 11	*Proponha um requisito relevante ainda por identificar e que requeira uma query com funções de agregação (sum, max, min, etc) para o satisfazer. Implemente.*

[4] CRUD - Create, Read, Update, Delete

RF 12	*Proponha um requisito relevante ainda por identificar e que requeira o desenvolvimento de functions / procedures para o satisfazer. Implemente.*

Para que se consiga avaliar adequadamente a qualidade do modelo de dados, é necessário que a BD para o myFitness possua dados em quantidade razoável. Por exemplo, calcular a média dum valor com base numa lista que contenha apenas um (ou poucos valores) não é satisfatório.

A avaliação do trabalho terá em consideração os aspetos atrás referidos, os requisitos da tabela 1, tanto ao nível da criação do modelo como da sua implementação no *SGBD Oracle XE*. A utilização do modelo numa aplicação em C++/Java/Web, apenas para ilustrar o acesso à base-de-dados num padrão de interação servidor-cliente, tem um peso máximo de 10% da nota.

Nota:
Ter em atenção que o projeto admite soluções alternativas, não existindo alguma que seja 100% correta à partida (e ainda não se tem experiencia para isso). A melhor abordagem é ir começando com uma solução inicial plausível e experimentar o modelo com dados de testes. Conforme se vai testando cada requisito funcional, se algum não puder ser satifisfeito, então reformula-se o modelo. Recomenda-se falar com os docentes sobre a estratégia para alterar o modelo, sem se perder os dados já existentes na BD.

De facto, num ambiente de produção (sistema em funcionamento em contexto real), o modelo (e respetivo esquema BD) é frequentemente alterado (ex: alter table …, create view, …) de forma a conseguir acompanhar a evolução do negócio/serviço/…

Modelação

No seu papel de analista do sistema, e face às necessidades da própria organização, já tomou a decisão que a melhor ferramenta para albergar os dados será o DBMS *Oracle*. Neste contexto, o seu trabalho consiste em três fases distintas, cada uma delas com um objectivo e um papel específico:

- Analista – Fazer a análise do sistema e construir um modelo abstracto (DER) dos dados
- Administrador de Base de Dados – Implementar o modelo abstracto em SQL, através de ferramentas adequadas, por exemplo o *Oracle Data Modeler*. Garantir que as bases-de-dados por si geridas estão "sempre de boa saúde", ou seja, servem o propósito para que foram projectadas e estão sempre correctas tendo em conta o seu esquema relacional. O administrador também tem o papel de gerir as permissões de acesso às tabelas e recursos existentes na base-de-dados.
- Programador – Enquadrar num contexto de aplicação a BD obtida. Neste caso, irá recorrer a linguagem Java, ou outra da sua preferência, e à utilização da conectividade JDBC/outra. O programador/utilizador que utiliza a base-de-dados fica assim sujeito ao modelo e regras de integridade definidas no esquema relacional (vale 10% da nota neste trabalho).

Na sua futura vida profissional poderá ser confrontado com as três tarefas em separado (em empresas de grande dimensão) ou as três em conjunto (nas empresas de pequena e média dimensão - mais típicas em Portugal). O trabalho de organização dos dados de uma empresa, bem como a melhoria da troca de informação, é muito importante para o aumento dos níveis de produtividade de qualquer organização.

Como analista do sistema deverá questionar as várias pessoas chave da organização, de modo a que a estrutura de informação seja a mais adequada, começando, portanto, pela obtenção de um modelo abstrato que albergue as características importantes do problema considerado. Num cenário em que se pretende desenvolver uma base-de-dados ou um esquema relacional, procede-se à respetiva modelação através das seguintes fases de projecto:

1. Identificação das entidades presentes no problema (nível conceptual).
2. Identificação dos relacionamentos existentes entre essas entidades (nível conceptual).
3. Determinação dos atributos de cada entidade e as correspondentes propriedades desses atributos (nível logico).
4. Obtenção do esquema relacional (nível físico)

Implementação

Após obtido um *DER* para o problema proposto, passa-se para a fase de transformação desse diagrama num esquema relacional, que neste caso é implementado num *DBMS*.

Posteriormente, deverá construir as questões (*SQL queries*), vistas (*SQL views*) e funções (*SQL functions/procedures*) de forma a implementar as funcionalidades pretendidas no problema. Nesta fase é necessário que a base de dados obtida possua dados que possam ser utilizados para implementar e testar as *queries/procedures/…*; caso esses dados não existam, cria-se então um bom conjunto de dados, que corresponda o mais possível com a "realidade" do problema.

A fase final consiste de construir a interface com o utilizador, neste caso usando Java ou outra linguagem que considere mais adequada ou preferível, e que o aluno tenha já um bom domínio. Tal como o nome indica, e passando a redundância, a interface serve apenas como interface de visualização de resultados. Ou seja, os requisitos funcionais e não funcionais de utilização da BD deverão ser implementados na camada de base-de-dados. Limite o tempo dedicado à componente interface a um máximo de 10% do tempo disponível.

Durante a execução deste trabalho utilizar-se-ão as seguintes ferramentas:

- *Data Modeler* – Permite definir DERs e posterior conversão em esquemas relacionais de forma automática.
- *Oracle SQL developer* – Interagir com a BD através de *Queries.*
- *OracleXE* – Sistema de Gestão de bases de dados da Oracle.
- *JDBC/ODBC/…* – Componente de interface (*driver*) que estabelece a conectividade entre um programa e um DBMS, que neste caso é necessária para a ilustração do problema desenvolvido num contexto de aplicação. Em princípio já vem instalada no sistema operativo.

- Adicionalmente, a máquina deverá ter o JAVA SDK instalado. Em algumas situações em que ocorreram problemas na instalação das ferramentas, estas deveram-se ao facto do sistema operativo corresponder a uma das versões "Windows *Home* edition".

Elementos de Reflexão

Esta parte consiste em considerações mais reflexivas que poderiam servir como "inspiração" para a escrita dum relatório (no entanto, não é necessária entrega de relatório para este trabalho). No final do trabalho, as seguintes questões poderiam ser respondidas:

- O que se entende por modelação.
- Noção de modelo.
- Conceito de abstração (ilustrando com pormenores específicos do problema apresentado)
- A utilidade da modelação no contexto da Engenharia.
- O carácter ambíguo, redundante e incompleto da linguagem natural como formalismo de modelação.
- As características que um formalismo de modelação (ou linguagem) deve possuir para que possa representar modelos de forma adequada.

Este trabalho segue o seguinte plano:

0ª aula (15/03/2021):
Instalação do *Software* necessário ao desenvolvimento do trabalho.
Interação com a base-de-dados Oracle XE. "Queries" SQL, exemplo de aplicação.

1ª aula (22/03/2021):
Leitura e análise do trabalho prático. Criação das primeiras tabelas (utilizadores, exercícios, …). Especificação de regras de integridade com SQL.

2ª aula (29/03/2021):
Criar/importar o esquema da DB resultante da semana anterior, convertendo num DER.
Análise dos requisitos e modelação DER. Implementação da BD (forward eengineering).

3ª aula (05/04/2021):
Implementação das funcionalidades requeridas no trabalho prático, CRUD e *Queries* para consulta de informação. Funções de agregação.

4ª aula (12/04/2021):
PL/SQL: procedimentos, funções e *Triggers*.

5ª aula (19/04/2021):
Utilização de ODBC/JDBC. Construção de Interfaces gráficas (ilustrativas) para acesso à base-de-dados.

Entrega (no MOODLE)

Antes de mais, é importante que cada grupo consiga assegurar-se que os docentes das práticas vão conseguir visualizar o modelo no "*Data Modeler*" e executar os ficheiros "DDL.sql", "DML.sql", e "data.sql" dentro do *SQLdeveloper*. Por exemplo, se entregar o modelo DER com o nome DER_XPTO, é necessário enviar um ficheiro DER_XPTO e uma diretoria com o mesmo nome, ou seja, a diretoria DER_XPTO.

Para efeitos de verificação, aconselha-se que cada aluno/grupo construa um "zip/rar" do modelo e o extraia para outra directoria, verificando se se consegue abrir o modelo.

No ficheiro zip/rar, deve-se entregar tudo o que foi feito num ficheiro rar/zip, que deverá nomeadamente conter os seguintes elementos:

- Ficheiro **"DDL.sql"** com o DDL referente ao DER.
- Ficheiro "**queries.sql"** com as *queries* referentes aos requisitos funcionais (excepto RF 9).
- Ficheiro **"DML.sql"** com as funções, procedimentos, views, sequences, etc, feitas ao longo do trabalho.
- Ficheiro **"data.sql"** com os dados de teste.
- Ficheiro "**tudo.sql**", com o conteúdo dos ficheiros **ddl**, **dml**, e **data**, de forma a que os docentes consigam reproduzir e experimentar o modelo (façam o teste antes de entregar).
- "Video Capture" a demonstrar o teste de cada requisito funcional. Ou seja, deve-se conseguir visualizar no SQLDeveloper o teste de cada query para cada requisito e os respetivos resultados. O tempo máximo do vídeo é de 10 minutos, ou 15 se incluir demonstração adicional de JAVA / C++/C# (e neste caso, só deve aparecer após a parte demonstrada no SQLDeveloper).
- **Projeto** do programa Java/C++/Visual Basic, etc, caso tenha sido feito.

Este trabalho requer o preenchimento obrigatório dum questionário no Moodle, de um vídeo e de um ponto de entrega para o projeto completo, também no Moodle. Os docentes vão poder selecionar aleatoriamente (ou sempre que necessário) alguns trabalhos para apresentação presencial. A falha nalgum destes pontos, implica avaliação do trabalho somente depois dos restantes.
Entregas feitas através de email não são consideradas.

Docentes:
Teórica-Prática:
Luis Camarinha-Matos, cam@uninova.pt
João Rosas, jrosas@uninova.pt

Prática:
João Rosas, jrosas@uninova.pt
Ana Inês Oliveira, aio@uninova.pt
Filipa Ferrada, faf@uninova.pt
André Rocha, andre.rocha@uninova.pt