TÉCNICO LISBOA

Projeto de Bases de Dados, Parte 3

Nome	Número de Aluno	Percentagem Relativa de Contribuição	Esforço
Pedro Alegre Caldeira	83539	33,33%	13 horas
Francisco Lopes Pereira de Carvalho	84050	33,33%	13 horas
João Miguel Fernandes Pina	85080	33,33%	13 horas

Grupo 10 - turno de quarta-feira às 8:00

Docente do laboratório: André Vasconcelos

TÉCNICO LISBOA

Comandos da Criação da Base de Dados

```sql
drop table if exists reposicao cascade;
drop table if exists planograma cascade;
drop table if exists evento_reposicao cascade;
drop table if exists prateleira cascade;
drop table if exists corredor cascade;
drop table if exists fornece_sec cascade;
drop table if exists produto cascade;
drop table if exists fornecedor cascade;
drop table if exists constituida cascade;
drop table if exists super_categoria cascade;
drop table if exists categoria_simples cascade;
drop table if exists categoria cascade;


create table categoria(
  nome varchar(64) not null,

  primary key(nome)
);

create table categoria_simples(
  nome varchar(64) not null,

  primary key(nome),
  foreign key(nome)
    references categoria(nome)
    ON DELETE CASCADE
);

create table super_categoria(
  nome varchar(64) not null,

  primary key(nome),
  foreign key(nome)
    references categoria(nome)
    ON DELETE CASCADE
);

create table constituida(
  super_categoria varchar(64) not null,
  categoria varchar(64) not null,

  primary key(super_categoria, categoria),
  foreign key(super_categoria)
    references super_categoria(nome)
    ON DELETE CASCADE,
  foreign key(categoria)
    references categoria(nome)
    ON DELETE CASCADE,
  check(super_categoria != categoria)
);

create table fornecedor(
  nif numeric(9,0) not null,
  nome varchar(64) not null,
  primary key(nif),
  unique(nif,nome)
);

create table produto(
  ean numeric(13,0) not null,
  design varchar(64) not null,
  categoria varchar(64) not null,
  forn_primario numeric(9,0) not null,
  data date not null,

  primary key(ean),
  foreign key(categoria)
    references categoria(nome),
  foreign key(forn_primario)
    references fornecedor(nif)
);

```

```
74  create table fornece_sec(
75    nif numeric(9,0) not null,
76    ean numeric(13,0) not null,
77
78    primary key(nif,ean),
79    foreign key(nif)
80      references fornecedor(nif)
81      ON DELETE CASCADE,
82    foreign key(ean)
83      references produto(ean)
84      ON DELETE CASCADE
85  );
86
87  create table corredor(
88    nro integer not null,
89    largura float not null,
90    primary key(nro)
91  );
92
93  create table prateleira(
94    nro integer not null,
95    lado char(1) not null,
96    altura char(1) not null,
97
98    primary key(nro,lado,altura),
99    foreign key(nro)
100     references corredor(nro)
101     ON DELETE CASCADE,
102   check(lado in ('E', 'D')),
103   check(altura in ('C', 'M', 'S'))
104 );
105
106 create table planograma(
107   ean numeric(13,0) not null,
108   nro integer not null,
109   lado char(1) not null,
110   altura char(1) not null,
111   face integer not null,
112   unidades integer not null,
113   loc integer not null,
114
115   primary key(ean,nro,lado,altura),
116   foreign key(ean)
117     references produto(ean)
118     ON DELETE CASCADE,
119   foreign key(nro,lado,altura)
120     references prateleira(nro,lado,altura)
121     ON DELETE CASCADE
122 );
123
124 create table evento_reposicao(
125   operador varchar(64) not null,
126   instante timestamp not null,
127   primary key(operador,instante),
128   check(instante <= CURRENT_TIMESTAMP)
129 );
130
131 create table reposicao(
132   ean numeric(13,0) not null,
133   nro integer not null,
134   lado char(1) not null,
135   altura char(1) not null,
136   operador varchar(64) not null,
137   instante timestamp not null,
138   unidades integer not null,
139
140   primary key(ean,nro,lado,altura,operador,instante),
141   foreign key(ean,nro,lado,altura)
142     references planograma(ean,nro,lado,altura)
143     ON DELETE CASCADE,
144   foreign key(operador,instante)
145     references evento_reposicao(operador,instante));
146
```

TÉCNICO LISBOA

Queries em SQL

```sql
--a)Qual o nome do fornecedor que forneceu o maior número de categorias?
  --Note que pode ser mais do que um fornecedor.

-----------------------------------------------

SELECT nome
FROM fornecedor
WHERE nif = (
  SELECT forn
  FROM (
    SELECT forn
    FROM (
      (SELECT forn_primario as forn, categoria FROM produto)
      UNION (   --tabela com nif e categoria de todos os fornecedores
      SELECT nif as forn, categoria FROM (
          (SELECT ean, categoria FROM produto) as ps natural join fornece_sec) as fs )
    ) as forncat
    GROUP BY forn
    HAVING COUNT(categoria) >= all(
      SELECT nrcat as n
      FROM (
        SELECT COUNT(categoria) as nrcat
        FROM (
          (SELECT forn_primario as forn, categoria FROM produto)
            UNION
          (SELECT nif as forn, categoria FROM (
            (SELECT ean,categoria FROM produto) as ps natural join fornece_sec) as
            fs )
        ) as forncat
      GROUP BY forn) as lel)
  )as a);

--b)Quais os fornecedores primários (nome e nif) que forneceram produtos de todas a
categorias simples

SELECT nif, nome
FROM fornecedor
WHERE nif in (
  ( SELECT forn_primario --aqui começa uma divisao de producto pela tabela de todas
  as cats simples
    FROM produto as newprod
    WHERE categoria in (SELECT nome FROM categoria_simples as cs)
    GROUP BY forn_primario
    HAVING COUNT(*) = (
      SELECT COUNT(*)
      FROM categoria_simples as num_categoria
    )--aqui acaba uma divisao
  )
);


--c)Quais os produtos (ean) que nunca foram repostos?

SELECT ean
FROM produto
WHERE ean NOT IN (SELECT ean FROM reposicao);


--d) Quais os produtos (ean) com um número de fornecedores secundários superior a 10?

SELECT ean
FROM (SELECT ean
      FROM fornece_sec as count_aux
      GROUP BY ean
      HAVING count(ean) > 10) as count_secundarios;

--e)Quais os produtos (ean) que foram repostos sempre pelo mesmo operador?

SELECT ean
FROM (SELECT ean
      FROM reposicao as count_aux
      GROUP BY ean
      HAVING count(operador) = 1) as single_operadores;
```

TÉCNICO LISBOA

Arquitetura da Aplicação PHP

A aplicação é composta por 2 elementos: o índex (*index.html*), a partir do qual podemos aceder a qualquer uma das funcionalidades implementadas na aplicação; os ficheiros PHP que estão divididos em dois tipos: os ficheiros *"form"* cujo principal objetivo é criar a interface para cada uma das funcionalidades e enviar os dados (inseridos nas caixas de texto pelo utilizador) para os ficheiros onde são executadas as *queries* e preparados os resultados a devolver.

Em cada uma destas páginas que servem de formulário para o utilizador, no caso de faltar o preenchimento de algum dos elementos de carácter obrigatório, o utilizador é notificado com um alerta a pedir que preencha o campo em falta e é impedido de submeter os dados até que o mesmo seja preenchido.

Cada página PHP tem a possibilidade de voltar ao menu anterior de forma a facilitar a navegação do utilizador através das diversas páginas pelas quais é composta a aplicação.

É de salientar que todas as funcionalidades tal como estão implementadas são realizadas de forma atómica (com recurso às funções *start transaction* e *commit)*, é dizer, no caso de existir um erro na atualização de alguma das tabelas necessárias à execução correta da funcionalidade é revertido todo o processo com recurso à função *rollback.*

É também importante referir que a parte core de cada uma das funcionalidades encontra-se dentro de funções try-catch de forma a que se existir algum erro no processo de atualização das tabelas ou algum erro na inserção de dados por parte do utilizador seja possível devolver uma mensagem SQL mais específica sobre o que possa ter corrido mal e permitir também o uso da tal função *rollback*.

Todas as funcionalidades devolvem uma mensagem ao ser concluídas, ora de sucesso ora de erro. Quanto às de erro, existe parte delas que devolvem o erro que se dá na base de dados ao executar a *query* uma vez que precisamos dessa especificidade do SQL para perceber exactamente qual o erro que se deu. Aquelas que vêm de erros que o utilizador possa ter feito no preenchimento dos formulários são explicadas no ecrã sem a necessidade da mensagem devolvida pelo SQL.

Alínea a):

A estrutura da alínea a) é composta pelos ficheiros *formcategorias.php*, *addcategoria.php* e *removecategoria.php*. No *formcategorias* existem duas secções: uma para a opção de adicionar categorias e outra para a opção de remover categorias. Cada uma destas caixas liga o *formcategorias.php* ao *removecategoria.php* e ao *addcategoria.php* respetivamente. De forma a facilitar a adição e remoção de categorias, são apresentadas as tabelas com todos os tipos de categorias: categorias, categorias-simples e super-categorias.

Quanto ao *addcategoria.php* verificamos se todos os requisitos se cumprem e se tal se verificar adicionamos a mesma à tabela de **Categorias** e à tabela de **Categorias Simples**. O utilizador tem também a opção de especificar qual a super-categoria desta nova categoria, se esse campo for preenchido (não obrigatório) é verificada a tabela de **super-categorias** para verificar que esta existe já na tabela das **super-categorias** (caso base em que é simplesmente ligada à super-categoria já existente ou no caso de já existir, mas ser uma categoria simples a mesma passa à tabela das **super-categorias** e é apagada da tabela de **categorias-simples**. Finalmente são ambas então adicionadas numa única linha da tabela **Constituida**.

Já no *removecategoria.php* verificamos primeiro se a categoria existe, se se verificar, verificamos se é sub-categoria de alguma super-categoria cuja única sub-categoria era a categoria que queremos remover, caso em que passamos a super-categoria da mesma a sub-categoria uma vez que deixa de ter categorias subordinadas.

Alínea b):

A estrutura da alínea b) é composta pelos ficheiros *formprodutos.php*, *addproduto.php* e *removeproduto.php*. No *formprodutos* temos caixas de texto para preencher com cada uma das categorias dos produtos que queremos adicionar e com as características dos seus fornecedores primários e secundários.

De forma a cobrir o caso em que um determinado produto tem mais do que um Fornecedor Secundário, temos um botão que abre mais caixas de texto para o preenchimento das características dos Fornecedores adicionais. Nesta página são também apresentadas as tabelas relativas aos **Produtos**, **Categorias** e **Fornecedores** existentes no sistema.

No caso de *addproduto.php* começamos por verificar se os fornecedores determinados pelo utilizador já se encontram inseridos nas tabelas. Em caso negativo adicionamos os mesmos às tabelas de **Fornecedores** e **fornece_sec**. Verificamos os demais parâmetros e no caso de serem todos válidos (como por exemplo o *ean* ser um numero e demais obrigações) adicionamos o produto à tabela **produto**. O caso mais complexo tratado pelo código é aquele em que são adicionados vários fornecedores secundários, uma vez que há que guardar os vários fornecedores em vetores de *NIF e nome*.

Quanto ao *removeproduto.php* verificamos se o produto a remover é um produto válido e em caso positivo removemos o mesmo mostrando uma mensagem de sucesso e atualizando as tabelas, caso contrário nenhuma tabela é alterada e é apresentada uma mensagem de erro.

Alíneas c) /e):

As alíneas c) e a e) têm um funcionamento bastante semelhante. A estrutura da alínea c) é composta pelos ficheiros *formlistrep.php* e *listrep.php*. Relativamente à estrutura da alínea e), a mesma é composta pelos ficheiros *formlistsubcats.php* e *listsubcats.php*.

No caso do ficheiro *listsubcats.php*, recorremos a uma função auxiliar recursiva (*findsubcats($db, $supercategoria))* de forma a obter todas as sub-categorias de uma super-categoria. A função vai imprimindo no ecrã as sub-categorias que encontra para cada categoria chega ao fim quando alcança uma categoria que não tem mais sub-categorias.

No caso de *listrep.php* é realizada uma consulta simples de forma a obter todos os eventos de reposição do produto dado e os mesmos são listados em forma de tabela.

Em cada um dos ficheiros *form* temos caixas de texto para o utilizador preencher com o EAN do produto (no caso da alínea c)) ou com o nome da super-categoria (no caso da alínea e)).

De forma a facilitar o uso das operações de listagens, são apresentadas, no caso da alínea c), as tabelas de Produtos e de Eventos de Reposição e, no caso da alínea e), as tabelas de **Categorias**, **Super-Categorias** e **Categorias Simples.**

Alínea d):

A estrutura da alínea d) é composta pelos ficheiros *formdesign.php* e *updatedesign.php*. No *formdesign* temos duas caixas de texto que o utilizador deve preencher com o EAN do produto cuja designação deseja alterar e com a nova designação pretendida. Novamente, de forma a facilitar o uso desta operação, é apresentada a tabela dos **Produtos** que estão presentes no nosso sistema.

Quanto ao *updatedesign.php* é feita uma consulta de *UPDATE*, no caso de sucesso da mesma é mostrada uma mensagem que mostra o *ean* do produto e a nova designação do mesmo. Caso contrário é mostrada uma mensagem de erro explicando ao utilizador o porquê de o processo não ter sido concluído com sucesso.