Projeto de Base de Dados

Parte 3

Prof. André Vasconcelos

Grupo 33 **Turno L03** **Segunda às 08:00**

	Aluno	Contribuição (%)	Esforço(horas)
86426	Gabriel Figueira	33	20
86461	Lívio Costa	33	20
86499	Pedro Carvalho	33	20

Criação da Base de Dados

```
drop table Solicita cascade;
drop table Audita cascade;
drop table Coordenador cascade;
drop table Acciona cascade;
drop table Alocado cascade;
drop table Transporta cascade;
drop table MeioSocorro cascade;
drop table MeioApoio cascade;
drop table MeioCombate cascade;
drop table Meio cascade;
drop table EntidadeMeio cascade;
drop table EventoEmergencia cascade;
drop table ProcessoSocorro cascade;
drop table Vigia cascade;
drop table Local cascade;
drop table SegmentoVideo cascade;
drop table Video cascade;
drop table Camara cascade;

create table Camara(numCamara integer not null,
   constraint pk_Camara primary key(numCamara));

create table Video(dataHoraInicio timestamp not null,
   dataHoraFim timestamp not null,
   numCamara integer not null,
   constraint pk_Video primary key(dataHoraInicio, numCamara),
   constraint fk_Video_Camara foreign key(numCamara) references
Camara(numCamara) on delete cascade);

create table SegmentoVideo(numSegmento integer not null,
   duracao interval not null,
   dataHoraInicio timestamp not null,
   numCamara integer not null,
   constraint pk_SegmentoVideo primary key(numSegmento, dataHoraInicio,
numCamara),
   constraint fk_SegmentoVideo_Video foreign key(dataHoraInicio, numCamara)
references Video(dataHoraInicio, numCamara) on delete cascade);

create table Local(moradaLocal varchar(255) not null,
   constraint pk_Local primary key(moradaLocal));

create table Vigia(moradaLocal varchar(255) not null,
   numCamara integer not null,
   constraint pk_Vigia primary key(moradaLocal, numCamara),
   constraint fk_Vigia_Local foreign key(moradaLocal) references Local(moradaLocal)
on delete cascade,
```

```
    constraint fk_Vigia_Camara foreign key(numCamara) references
Camara(numCamara) on delete cascade);

create table ProcessoSocorro(numProcessoSocorro integer not null,
    constraint pk_ProcessoSocorro primary key(numProcessoSocorro));

create table EventoEmergencia(numTelefone varchar(15) not null,
    instanteChamada timestamp not null,
    nomePessoa varchar(80) not null,
    moradaLocal varchar(255) not null,
    numProcessoSocorro integer not null,
    constraint pk_EventoEmergencia primary key(numTelefone, instanteChamada),
    constraint fk_EventoEmergencia_Local foreign key(moradaLocal) references
Local(moradaLocal) on delete cascade,
    constraint fk_EventoEmergencia_ProcessoSocorro foreign key(numProcessoSocorro)
references ProcessoSocorro(numProcessoSocorro) on delete cascade,
    unique(numTelefone, nomePessoa));

create table EntidadeMeio(nomeEntidade varchar(200) not null,
    constraint pk_EntidadeMeio primary key(nomeEntidade));

create table Meio(numMeio integer not null,
    nomeMeio varchar(200) not null,
    nomeEntidade varchar(200) not null,
    constraint pk_Meio primary key(numMeio, nomeEntidade),
    constraint fk_Meio_EntidadeMeio foreign key(nomeEntidade) references
EntidadeMeio(nomeEntidade) on delete cascade);

create table MeioCombate(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    constraint pk_MeioCombate primary key(numMeio, nomeEntidade),
    constraint fk_MeioCombate_Meio foreign key(numMeio, nomeEntidade) references
Meio(numMeio, nomeEntidade) on delete cascade);

create table MeioApoio(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    constraint pk_MeioApoio primary key(numMeio, nomeEntidade),
    constraint fk_MeioApoio_Meio foreign key(numMeio, nomeEntidade) references
Meio(numMeio, nomeEntidade) on delete cascade);

create table MeioSocorro(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    constraint pk_MeioSocorro primary key(numMeio, nomeEntidade),
    constraint fk_MeioSocorro_Meio foreign key(numMeio, nomeEntidade) references
Meio(numMeio, nomeEntidade) on delete cascade);

create table Transporta(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    numVitimas integer not null,
```

```
    numProcessoSocorro integer not null,
    constraint pk_Transporta primary key(numMeio, nomeEntidade,
numProcessoSocorro),
    constraint fk_Transporta_MeioSocorro foreign key(numMeio, nomeEntidade)
references MeioSocorro(numMeio, nomeEntidade) on delete cascade on update
cascade,
    constraint fk_Transporta_ProcessoSocorro foreign key(numProcessoSocorro)
references ProcessoSocorro(numProcessoSocorro) on delete cascade);

create table Alocado(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    numHoras integer not null,
    numProcessoSocorro integer not null,
    constraint pk_Alocado primary key(numMeio, nomeEntidade, numProcessoSocorro),
    constraint fk_Alocado_MeioApoio foreign key(numMeio, nomeEntidade) references
MeioApoio(numMeio, nomeEntidade) on delete cascade on update cascade,
    constraint fk_Alocado_ProcessoSocorro foreign key(numProcessoSocorro)
references ProcessoSocorro(numProcessoSocorro) on delete cascade);

create table Acciona(numMeio integer not null,
    nomeEntidade varchar(200) not null,
    numProcessoSocorro integer not null,
    constraint pk_Acciona primary key(numMeio, nomeEntidade, numProcessoSocorro),
    constraint fk_Acciona_Meio foreign key(numMeio, nomeEntidade) references
Meio(numMeio, nomeEntidade) on delete cascade,
    constraint fk_Acciona_ProcessoSocorro foreign key(numProcessoSocorro)
references ProcessoSocorro(numProcessoSocorro) on delete cascade);

create table Coordenador(idCoordenador integer not null,
    constraint pk_Coordenador primary key(idCoordenador));

create table Audita(idCoordenador integer not null,
    numMeio integer not null,
    nomeEntidade varchar(200) not null,
    numProcessoSocorro integer not null,
    dataHoraInicio timestamp not null,
    dataHoraFim timestamp not null,
    dataAuditoria date not null,
    texto text not null,
    constraint pk_Audita primary key(idCoordenador ,numMeio, nomeEntidade,
numProcessoSocorro),
    constraint fk_Audita_Coordenador foreign key(idCoordenador) references
Coordenador(idCoordenador) on delete cascade,
    constraint fk_Audita_Acciona foreign key(numMeio, nomeEntidade,
numProcessoSocorro) references Acciona(numMeio, nomeEntidade,
numProcessoSocorro) on delete cascade,
    check (dataHoraInicio < dataHoraFim),
    check (dataAuditoria <= now()));
```

```
create table Solicita(idCoordenador integer not null,
   dataHoraInicioVideo timestamp not null,
   numCamara integer not null,
   dataHoraInicio timestamp not null,
   dataHoraFim timestamp not null,
   constraint pk_Solicita primary key(idCoordenador, dataHoraInicioVideo,
numCamara),
   constraint fk_Solicita_Coordenador foreign key(idCoordenador) references
Coordenador(idCoordenador) on delete cascade,
   constraint fk_Solicita_Video foreign key(dataHoraInicioVideo, numCamara)
references Video(dataHoraInicio, numCamara) on delete cascade);
```

Restrições de Integridade

A restrição de integridade indicada no modelo relacional no Processo de Socorro "todo o processo de socorro está associado a um ou mais EventoEmergencia" não foi possível ser representada na criação da tabela ProcessoSocorro visto que teriamos que fazer uso de triggers.

Queries SQL

1.
```
select numProcessoSocorro
   from Acciona
   group by numProcessoSocorro
   having count(1) >= all (select count(1)
                           from Acciona
                           group by numProcessoSocorro);
```

2.
```
select nomeEntidade
   from EventoEmergencia
   natural join Acciona
   where instanteChamada >= '2018-06-21'
   and instanteChamada <= '2018-09-21'
   group by nomeEntidade
   having count(1) >= all (select count(1)
                           from EventoEmergencia
                           natural join Acciona
                           where instanteChamada >= '2018-06-21'
                           and instanteChamada <= '2018-09-21'
                           group by nomeEntidade);
```

3.
```
select distinct numProcessoSocorro
   from (select numProcessoSocorro, count(1) as numAcc
         from Acciona
         natural join EventoEmergencia
         where moradaLocal = 'Oliveira do Hospital'
         and date_part('year', instanteChamada) = 2018
         group by numProcessoSocorro) as Acc_EE
   natural join (select numProcessoSocorro, count(1) as numAud
                 from Audita
                 natural join EventoEmergencia
                 where moradaLocal = 'Oliveira do Hospital'
```

```
        and date_part('year', instanteChamada) = 2018
        group by numProcessoSocorro) as Aud_EE
        where Acc_EE.numAcc > Aud_EE.numAud;
```

4.
```
select count(1) as totalSegmentos
from Vigia
natural join Video
natural join SegmentoVideo
where duracao > '00:01:00'
and moradaLocal = 'Monchique'
and dataHoraInicio >= '2018-08-01 00:00:00'
and dataHoraFim < '2018-09-01 00:00:00';
```

5.
```
select *
from MeioCombate
where (numMeio,nomeEntidade) not in (select numMeio, nomeEntidade
                                     from MeioApoio
                                     natural join Acciona);
```

6.
```
select nomeEntidade
from MeioCombate m_c1
where not exists(select distinct numProcessoSocorro
                 from Acciona
                 except
                 select distinct numProcessoSocorro
                 from (Acciona natural join MeioCombate) m_c2
                 where m_c2.numMeio=m_c1.numMeio and
                 m_c2.nomeEntidade=m_c1.nomeEntidade);
```

Explicação da aplicação

A arquitetura da aplicação PHP tem a seguinte organização:

- Toda a informação respectiva a uma dada tabela pode ser visualizada clicando na opção de menu respectiva. Por exemplo, para visualizar/listar toda a informação sobre os Locais, clica-se em “Locais”(primeira opção do menu).
- A operação de inserção é possível clicando na sub-opção de menu “Inserir” respectiva à opção de menu.
- No caso especial da inserção de um novo Processo de Socorro (como um Processo tem de existir sempre associado a pelo menos um Evento de Emergência) é requerida, pelo formulário, a associação a um Evento de Emergência. Consequentemente, no caso da eliminação (ou edição) do último Evento de Emergência a que um Processo de Socorro está associado, é feita a eliminação do Processo de Socorro, mecanismo que é feito ao nível da aplicação php.
- Para a edição e/ou remoção de registos é necessário aceder à opção de menu pretendida e na tabela onde está presente toda a informação, existe uma opção de edição/remoção.

- Caso efectue uma edição, aparecerá uma listagem com toda a informação útil para a operação, bem como um formulário para efectuar a edição do registo, formulário este que se encontrará inicialmente preenchido.
- Ao remover um local, serão apagados todos os eventos de emergência cuja morada do local corresponde a este e consequentemente, todos os processos de socorro que deixam de ter uma associação com um evento de emergência.
- Para efectuar uma associação de um processo de socorro existe uma sub-opção "Associar Processo" no "Evento de Emergência" e "Meios". Associar um processo de socorro a um evento de emergência apenas faz uma troca do processo de socorro no evento de emergência. Associar um processo de socorro a um meio corresponde a inserir um accionamento, ou seja, dizer a que meio está associado o processo de socorro.
- No final de cada operação que é realizada, seja inserção, edição ou remoção, será mostrada uma mensagem. Esta pode ser de sucesso ou de erro.

Relações entre os diversos ficheiros da aplicação PHP:

- O ficheiro index.php tem um menu com todas as opções disponíveis (este código html é depois replicado em cada ficheiro php para ter sempre o menu disponível). Os ficheiros correspondentes ao menu, menu.php e submenu.php (igual ao menu.php mas para ficheiros contidos em pastas), estão na pasta "menu".
- A ligação à base de dados é efectuada pelo ficheiro connect.php.
- As queries relacionadas com as associações e listagens pedidas no enunciado estão no ficheiros list.php.
- Para executar as queries é preciso identificar que tabela é que está a ser afetada, passando uma string, entre ficheiros, que representa o nome da tabela.
- As queries bem como todo o mecanismo para fazer inserções/remoções/edições na base de dados estão contidos em ficheiros que estão em pastas correspondentes à sua natureza.
- A inserção é feita, passando os dados que são preenchidos no formulário, entre o ficheiro insert.php (que contém o formulário) e o ficheiro runInsertion.php. Estes ficheiros estão presentes na pasta "InsertQueries".
- A edição tem o mesmo mecanismo que a inserção mas ao efectuar uma edição os dados correspondentes a chaves da tabela na base de dados são passados entre o ficheiro list.php e o ficheiro update.php(presente na pasta "UpdateQueries"), daqui em diante o processo é igual à inserção o que muda é só o ficheiro que trata do update na base de dados que é o runUpdate.php (presente também na pasta "UpdateQueries").
- A remoção é feita passando chaves de tabelas entre os ficheiros list.php e o runRemoval.php, que se encontra presente na pasta "RemoveQueries".