PROJETO(PART 2)

November 15, 2017

Mariana Galrinho 81669
Paulo Eusébio 81607
Renato Henriques 81588

Grupo 15 - SIBD
Instituto Superior Técnico

1. Criação da Base de Dados:

```
create table Patient (
   number int unsigned,
   name varchar(255),
   birthday DATE,
   address varchar(255),
   primary key(number) );

create table Doctor(
   number int unsigned,
   doctor_id int unsigned,
   primary key(doctor_id),
   foreign key(number) references Patient(number));

create table Device(
   serialnum varchar(255),
   manufacturer varchar(255),
   model varchar(255),
   primary key(serialnum, manufacturer));

create table Sensor(
   snum varchar(255),
   manuf varchar(255),
   units varchar(255),
   primary key(snum, manuf),
   foreign key(snum, manuf) references
       Device(serialnum, manufacturer));

create table Reading(
   snum varchar(255),
   manuf varchar(255),
   datetime DATETIME,
   value numeric(10,2),
   primary key(snum, manuf, datetime),
   foreign key(snum, manuf) references
       Sensor(snum, manuf));

create table Period(
   start DATETIME,
   end DATETIME,
   primary key(start, end));

create table Wears(
   start DATETIME,
   end DATETIME,
   patient int unsigned,
   snum varchar(255),
   manuf varchar(255),
   primary key(start, end, patient),
   foreign key(start, end) references
       Period(start, end),
   foreign key(patient) references Patient(number),
   foreign key(snum, manuf) references
       Device(serialnum, manufacturer));

create table Request(
   number int unsigned,
   patient_id int unsigned,
   doctor_id int unsigned,
   date DATE,
   primary key(number),
   foreign key(patient_id) references
       Patient(number),
   foreign key(doctor_id) references
       Doctor(doctor_id));

create table Study(
   request_number int unsigned,
   description varchar(255),
   date DATE,
   doctor_id int unsigned,
   manufacturer varchar(255),
   serial_number varchar(255),
   primary key(request_number, description),
   foreign key(request_number) references
       Request(number),
   foreign key(doctor_id) references
       Doctor(doctor_id),
   foreign key(serial_number, manufacturer)
       references Device(serialnum, manufacturer));

create table Series(
   series_id int unsigned,
   name varchar(255),
   base_url varchar(255),
   request_number int unsigned,
   description varchar(255),
   primary key(series_id),
   foreign key(request_number, description)
       references Study(request_number,
       description));

create table Element(
   series_id int unsigned,
   elem_index int unsigned,
   primary key(series_id, elem_index),
   foreign key(series_id) references
       Series(series_id));

create table Region(
   series_id int unsigned,
```

```
elem_index int unsigned,
x1 FLOAT(4, 3),
x2 FLOAT(4, 3),
y1 FLOAT(4, 3),
y2 FLOAT(4, 3),
primary key(series_id, elem_index, x1, x2, y1,
    y2),
foreign key(series_id, elem_index) references
    Element(series_id, elem_index));
```

2. REGISTOS DA BASE DE DADOS:

```
insert into Patient values(1,'Ruben', '1995-02-25', 'Av. do Tecnico');
insert into Patient values(6,'Francisco', '1989-12-19', 'Av. da Liberdade');
insert into Patient values(2,'Andre', '1912-04-15', 'Rua oliveirinha');
insert into Patient values(18012,'Mariana', '1999-01-23', 'Valverde');
insert into Patient values(27233,'Fedra', '1987-11-25', 'Cova da Moura');
insert into Patient values(9256926,'Ruben', '1991-03-30', 'Benfica');
insert into Patient values(13,'Diogo', '1996-10-20', 'Sintra');
insert into Patient values(187529,'Rita', '1987-01-25', 'Lourel');
insert into Patient values(7465,'Joaquim', '1998-09-15', 'Beja');
insert into Patient values(7549,'Miguel', '1992-01-20', 'Braga');

insert into Doctor values(6, 8246527);
insert into Doctor values(27233, 76592659);
insert into Doctor values(7465, 1996);
insert into Doctor values(9256926, 4477);
insert into Doctor values(2, 12345);
insert into Doctor values(1, 5555);

insert into Device values('3000','Medtronic','GlucoseReader');
insert into Device values('3333','Medtronic','GlucoseReader');
insert into Device values('2000','Siemens','BloodPressureReader');
insert into Device values('4552','Samsung','PulseMeter');
insert into Device values('1234','LG','CholesterolMeter');
insert into Device values('3345','LG','CholesterolMeter');
insert into Device values('20','Siemens','X-ray1');
insert into Device values('31','Medtronic','scanner');
insert into Device values('443','Medtronic','BloodReader');
insert into Device values('31','Samsung','Echo123');
insert into Device values('443','Samsung','Echo123');

insert into Sensor values('3000','Medtronic','glucose in mmol/L');
insert into Sensor values('3333','Medtronic','glucose in mmol/L');
insert into Sensor values('2000','Siemens','LDL cholesterol in mg/dL');
insert into Sensor values('4552','Samsung','LDL cholesterol in mg/dL');
insert into Sensor values('1234','LG','LDL cholesterol in mg/dL');
insert into Sensor values('3345','LG','LDL cholesterol in mg/dL');

insert into Reading values('3000','Medtronic','2017-02-16 19:03:00',5);
insert into Reading values('3000','Medtronic','2017-02-17 11:03:00',6);
insert into Reading values('3000','Medtronic','2017-02-20 11:03:00',250);
insert into Reading values('3333','Medtronic','2002-05-11 12:01:10',10);
insert into Reading values('2000','Siemens','2017-09-16 11:03:10',202);
insert into Reading values('2000','Siemens','2013-02-16 11:23:01',1);
insert into Reading values('4552','Samsung','2010-02-16 12:13:06',60);
```

```
insert into Reading values('3345','LG','2017-11-10 18:24:00',250);
insert into Reading values('3345','LG','2017-10-10 21:22:22',201);
insert into Reading values('3345','LG','2017-10-11 22:23:02',100);
insert into Reading values('1234','LG','2017-09-02 12:00:22',300);
insert into Reading values('1234','LG','2017-09-22 12:00:22',340);

insert into Period values('2012-02-23 11:02:00','2013-02-23 09:10:00');
insert into Period values('2013-12-21 09:10:00','2014-01-19 11:01:00');
insert into Period values('2011-11-26 10:08:00','2015-11-23 07:10:00');
insert into Period values('2009-03-11 10:11:00','2010-03-21 08:11:00');
insert into Period values('2016-02-25 12:02:00','2017-05-19 15:15:00');
insert into Period values('2015-10-03 12:30:00','2017-09-05 09:00:00');
insert into Period values('2016-09-16 08:30:00','2016-12-01 10:05:00');
insert into Period values('2001-01-30 14:00:00','2003-02-16 19:03:00');
insert into Period values('2013-02-15 11:22:00','2999-12-31 00:00:00');
insert into Period values('2017-02-02 14:00:00','2999-12-31 00:00:00');
insert into Period values('2017-03-11 10:11:00','2017-09-21 08:11:00');

insert into Wears values('2013-02-15 11:22:00','2999-12-31 00:00:00',9256926,'2000','Siemens');
insert into Wears values('2016-02-25 12:02:00','2017-05-19 15:15:00',6,'3000','Medtronic');
insert into Wears values('2001-01-30 14:00:00','2003-02-16 19:03:00',2,'3333','Medtronic');
insert into Wears values('2009-03-11 10:11:00','2010-03-21 08:11:00',7465,'4552','Samsung');
insert into Wears values('2017-03-11 10:11:00','2017-09-21 08:11:00',27233,'1234','LG');
insert into Wears values('2017-02-02 14:00:00','2999-12-31 00:00:00',1,'3345','LG');

insert into Request values(874, 2, 12345,'2002-02-11');
insert into Request values(86351, 9256926, 4477, '2014-10-23');
insert into Request values(126, 6, 8246527,'2016-02-11');
insert into Request values(9769, 7465, 1996, '2009-10-23');
insert into Request values(111,1,5555,'2016-05-05');
insert into Request values(112,1,5555,'2017-08-05');
insert into Request values(128,6,8246527,'2016-02-23');
insert into Request values(1000,7465,1996,'2010-10-01');

insert into Study values(86351,'X-ray left foot','2014-10-24',8246527,'Siemens','20');
insert into Study values(126,'MRI scan', '2016-02-15',1996,'Medtronic','31');
insert into Study values(874,'Blood analysis','2002-03-01',1996,'Medtronic','443');
insert into Study values(9769,'Echography right arm','2009-10-27',76592659,'Samsung','31');
insert into Study values(1000,'Echography right arm','2010-10-27',76592659,'Samsung','443');
insert into Study values(111,'Blood analysis', '2016-05-20',1996,'Medtronic','443');
insert into Study values(112,'MRI scan','2017-08-22',8246527,'Medtronic','31');
insert into Study values(128,'Blood analysis','2016-02-27',76592659,'Medtronic','443');

insert into Series values(1000,'X-ray left foot','www.clinic.com/86351xray',86351,'X-ray left
    foot');
insert into Series values(2000,'MRI shot','www.clinic.com/126mri',126,'MRI scan');
insert into Series values(3000,'Blood parameters','www.clinic.com/874blood',874,'Blood
    analysis');
insert into Series values(4000,'Right arm
    echography','www.clinic.com/9769echo',9769,'Echography right arm');
insert into Series values(5000,'Blood parameters','www.clinic.com/111blood',111,'Blood
    analysis');
insert into Series values(6000,'MRI shot','www.clinic.com/112mri',112,'MRI scan');
insert into Series values(7000,'Blood parameters','www.clinic.com/128blood',128,'Blood
    analysis');
insert into Series values(4001,'Right arm
    echography','www.clinic.com/1000echo',1000,'Echography right arm');
```

```
insert into Element values(1000,1);
insert into Element values(1000,2);
insert into Element values(1000,3);
insert into Element values(2000,1);
insert into Element values(3000,1);
insert into Element values(3000,2);
insert into Element values(4000,1);
insert into Element values(4000,2);
insert into Element values(5000,1);
insert into Element values(5000,2);
insert into Element values(6000,1);
insert into Element values(6000,2);
insert into Element values(7000,1);
insert into Element values(7000,2);
insert into Element values(7000,3);
insert into Element values(4001,2);

insert into Region values(1000,1,0.5,0.5,0.7,0.7);
insert into Region values(1000,2,0.1,0.5,0.2,0.7);
insert into Region values(1000,3,0.5,0.6,0.7,0.7);
insert into Region values(2000,1,0.5,0.5,0.7,0.7);
insert into Region values(3000,1,0.5,0.5,0.7,0.7);
insert into Region values(3000,1,0.1,0.5,0.3,0.7);
insert into Region values(4000,1,0.5,0.1,0.7,0.3);
insert into Region values(4000,2,0.1,0.5,0.2,0.9);
insert into Region values(5000,1,0.2,0.1,0.3,0.3);
insert into Region values(5000,2,0.4,0.5,0.5,0.6);
insert into Region values(6000,1,0.2,0.5,0.5,0.7);
insert into Region values(6000,2,0.7,0.2,0.9,0.3);
insert into Region values(7000,1,0.2,0.4,0.4,0.5);
insert into Region values(7000,2,0.1,0.5,0.3,0.6);
insert into Region values(7000,3,0.1,0.4,0.4,0.6);
insert into Region values(4001,2,0.2,0.5,0.4,0.7);
```

3. Query para obter o nome dos pacientes com maior número de Readings com units "LDL cholesterol in mg/dL" acima de 200 nos últimos 90 dias:

```
select name
from Patient as p, Wears as w natural join Sensor as s natural join Reading as r
where p.number=w.patient and r.datetime between w.start and w.end and s.units="LDL cholesterol
    in mg/dL" and r.value>200 and datediff(current_date,datetime)<=90
group by number
having count(datetime) >=all (select count(datetime)
                  from Patient as p, Wears as w natural join Sensor as s natural join Reading
                  as r
                  where p.number=w.patient and r.datetime between w.start and w.end and
                      s.units="LDL cholesterol in mg/dL" and
                  r.value>200 and datediff(current_date,datetime)<=90
                  group by number);
```

A query apresentada encontra todos os pacientes que obtiveram valores superiores a 200 para "LDL cholesterol in mg/dL" no espaço temporal em que a diferença entre a data de realização da query com a data da reading é inferior a 90 (ou seja, ultimos 90 dias), e conta o número de vezes que cada um desses pacientes

obteve readings dentro desses parâmetros. É também necessário garantir que só se pode associar uma reading a um paciente se esta for efetuada no intervalo de tempo em que o paciente está a usar o respetivo device.

O paciente com o maior número de reading é encontrado através da nested query que retorna a contagem que é maior ou igual a todas as restantes (ou seja, a maior!). De notar que, caso haja vários pacientes com o mesmo número máximo de leituras, todos eles serão apresentados pela query.

4. QUERY PARA OBTER O NOME DOS PACIENTES QUE FORAM SUBMETIDOS A STUDIES COM TODOS OS DEVICES DA 'MEDTRONIC' DURANTE O ANO PASSADO:

```
select name
from Patient as p
where not exists ( select serial_number
   from Study as d where manufacturer='Medtronic'
   and serial_number not in ( select serial_number
       from Study as s, Request as r, Patient as p2
       where s.request_number=r.number and p2.number=r.patient_id and p.number=p2.number and
         manufacturer='Medtronic' and YEAR(current_date)-YEAR(s.date)=1));
```

A query pretendida é equivalente à seguinte formulação: pretende-se obter o nome dos pacientes para os quais não existe um device utilizado em exames e cujo manufacturer é Medtronic que não esteja no set de devices utilizados por esse paciente nos seus exames no ano anterior ao presente, do manufacturer Medtronic. Como cada Device é identificado por uma chave conjunta composta pelo serial number e manufacturer, para obter os diferentes devices do manufacturer Medtronic, recorre-se à selecção do serial number.

5. TRIGGERS:

i) Assegurar que o médico que prescreve o exame não o pode realizar

```
delimiter $$

drop trigger if exists check_doctor_insert;

create trigger check_doctor_insert before insert on Study
for each row
begin
   if new.doctor_id in (
   select doctor_id
   from Request
   where Request.number=new.request_number) then
      call error_doctor_not_allowed();
   end if;
end$$

delimiter ;


delimiter $$

drop trigger if exists check_doctor_update;

create trigger check_doctor_update before update on Study
for each row
begin
   if new.doctor_id in (
   select doctor_id
```

```
   from Request
   where Request.number=new.request_number) then
      call error_doctor_not_allowed();
   end if;
end$$

delimiter ;
```

Este trigger simplesmente verifica se, ao inserir um registo na tabela study, o médico que executou o exame não é o mesmo que o receitou (casos em que o request number é igual ao study number). Caso seja é invocada uma função que não existe, o que interrompe e evita a inserção. O primeiro trigger atua na inserção de novos registos e o segundo na atualização de registos existentes.

ii) Assegurar que um device não é associado a um paciente em períodos que se sobrepõe

```
delimiter $$

drop trigger if exists check_overlapping_periods_insert;

create trigger check_overlapping_periods_insert before insert on Wears
for each row
begin
   if exists (
   select start,end
   from Wears
   where Wears.snum=new.snum and Wears.manuf=new.manuf
   and timediff(new.start,Wears.end)<0 and timediff(new.end,Wears.start)>0) then
      signal sqlstate '45000' set message_text = 'Overlapping Periods';
   end if;
end$$

delimiter ;


delimiter $$

drop trigger if exists check_overlapping_periods_update;

create trigger check_overlapping_periods_update before update on Wears
for each row
begin
   if exists (
   select start,end
   from Wears
   where Wears.snum=new.snum and Wears.manuf=new.manuf
    and Wears.end <> old.end and Wears.start <> old.start
   and timediff(new.start,Wears.end)<0 and timediff(new.end,Wears.start)>0) then
      signal sqlstate '45000' set message_text = 'Overlapping Periods';
   end if;
end$$

delimiter ;
```

Teve-se em conta que um device não pode ser usado por mais de um paciente no mesmo período de tempo e que um paciente pode usar vários devices no mesmo período. Os triggers acima garantem que o utilizador não insere várias vezes o mesmo device em períodos sobrepostos, tal como garantem que se um device estiver a ser utilizado por um certo paciente este não pode ser atribuído a um outro paciente durante o mesmo período

de tempo.

Para testar se há sobreposição dos períodos basta ver se new.start<wears.end e se new.end>wears.start já que estas condições cobrem todos os casos de sobreposição e nunca são ambas verificadas em casos em que não ocorra sobreposição. Se existir alguma data de início e de fim para um certo Device que se sobreponha com o período de tempo da nova inserção, então é feita uma chamada a erro impedindo que ocorra a inserção (ou atualização).

Utilizou-se a função *timediff()* invés da função *datediff()*, uma vez que, desta forma, é possível obter precisão na comparação de períodos de tempo ao nível dos segundos.

Para além disso, no trigger que verifica o update teve-se também em atenção que se quer comparar o novo período de tempo do Device com todos os períodos já existentes na tabela de Wears excepto o período que se quer atualizar, sendo necessário descartar este último. Para isso, selecciona-se as start e end dates que são diferentes de old.

6. Function que para a área duma dada região de um elemento A que pertence a um exame, indica se há ou não interseção com a área de uma dada região B:

Para construir a função de teste em relação à sobreposição de regiões considerou-se que as coordenadas do ponto 1 (x1, y1) não têm que ser necessariamente inferiores às coordenadas do ponto 2 (x2, y2), quer para a região A, que para a B. Deste modo, foi necessário executar um teste que verifica se alguma das coordenadas do ponto 2 é inferior à do ponto 1, tendo-se criado a função CheckCoordinates para esse fim. Assim, caso tal aconteça, é efetuada a troca das coordenadas, de forma a ter-se sempre (x1,y1) <= (x2,y2) quando se realiza o teste da sobreposição.

Como há muitas maneiras das duas regiões estarem sobrepostas, para simplificar o código, decidiu-se averiguar apenas os casos em que as regiões não se sobrepõe. Caso as regiões a testar não estejam nos quatro casos de não sobreposição, então implica que têm que ser sobrepostas, sendo também isto verificado na função checkCoordinates. Posto isto, na função region_overlaps_elements basta apenas ver existe alguma região do elemento A que se sobreponha com B, o que acontece quando checkCoordinates retorna True.

```
delimiter $$

drop function if exists checkCoordinates;

create function checkCoordinates(Ax1 FLOAT(4, 3), Ax2 FLOAT(4, 3), Ay1 FLOAT(4,3), Ay2
    FLOAT(4,3), Bx1 FLOAT(4, 3), Bx2 FLOAT(4, 3), By1 FLOAT(4,3), By2 FLOAT(4,3))
returns BOOLEAN
begin
   declare Aux FLOAT(4,3);

   -- Conditions used to guarantee that x1 < x2 and y1 < y2
   if Ax1 > Ax2 then set Aux = Ax1; set Ax1 = Ax2; set Ax2 = Aux; end if;

   if Bx1 > Bx2 then set Aux = Bx1; set Bx1 = Bx2; set Bx2 = Aux; end if;

   if By1 > By2 then set Aux = By1; set By1 = By2; set By2 = Aux; end if;

   if Ay1 > Ay2 then set Aux = Ay1; set Ay1 = Ay2; set Ay2 = Aux; end if;

   /* checking if the coordinates overlap */
   /* we check the cases that they don't overlap instead of the ones they do, because there are
       less conditions. */
    /*returns false if they don't overlap and true otherwise*/
   if Bx1 >= Ax2 or Bx2 <= Ax1 or By1 >= Ay2 or By2 <= Ay1 then
      return FALSE;
   else
      return TRUE;
```

```
   end if;
end $$


drop function if exists region_overlaps_element;

create function region_overlaps_element(idA int unsigned, indexA int unsigned, x1B FLOAT(4,
    3), y1B FLOAT(4, 3), x2B FLOAT(4, 3), y2B FLOAT(4, 3))
returns BOOLEAN
begin
   -- returns true if there is overlapping or false if there isn't
   return exists (select r.series_id, r.elem_index
      from Region as r
      where r.series_id = idA and r.elem_index = indexA and checkCoordinates(r.x1, r.x2, r.y1,
          r.y2, x1B, x2B, y1B, y2B));
   -- checkCoordinates is a functions that tests if x1 < x2 and y1 < y2 and if the coordinates
       overlap
end$$

delimiter ;
```