Instituto Superior Técnico

Sistemas de Informação e Bases de Dados

# PROJECTO

PARTE II


*Autores:*

| | |
|---|---|
| Rúben Tadeia | 75268 |
| Carla Marreiros | 75682 |
| Bruno Pereirinha | 79297 |


| *Grupo:* | *Turno:* | *Ano Letivo:* |
|---|---|---|
| 15 | 2.ªf, 11h-12h30 | 2016-2017 |

16 Novembro de 2016

# 1. For the relational model above, write the SQL instructions to create the database in our database server (i.e. MySQL on db.ist.utl.pt). You should choose the most appropriate SQL data types for each column

**Script Utilizado:**

```
/* USE ist175682
USE ist179297*/
USE ist175268;

SET FOREIGN_KEY_CHECKS = 0; /* Disable foreign key checking.*/
drop table if exists patient;
drop table if exists doctor;
drop table if exists appointment;
drop table if exists request;
drop table if exists equipment;
drop table if exists study;
drop table if exists series;
drop table if exists element;
drop table if exists region;
SET FOREIGN_KEY_CHECKS = 1; /* Enable foreign key checking.*/

create table IF NOT EXISTS patient(
        patient_id  integer NOT NULL AUTO_INCREMENT,
    name  varchar(50) NOT NULL,
    birthday  date NOT NULL,
    address  varchar(255) NOT NULL,
    primary key(patient_id)
    );

create table IF NOT EXISTS doctor(
        doctor_id  integer NOT NULL AUTO_INCREMENT,
    name  varchar(50) NOT NULL,
    specialty  varchar(20) NOT NULL,
    primary key(doctor_id)
    );

create table IF NOT EXISTS appointment(
        patient_id  integer NOT NULL,
        doctor_id  integer NOT NULL,
    date  date NOT NULL,
    office  varchar(100) NOT NULL,
    primary key(patient_id, doctor_id, date),
    foreign key(patient_id) references patient(patient_id),
    foreign key(doctor_id) references doctor(doctor_id)
    );


create table IF NOT EXISTS request(
        request_number  integer NOT NULL AUTO_INCREMENT,
```

```
    patient_id  integer NOT NULL ,
        doctor_id  integer NOT NULL,
    date  date NOT NULL,
    primary key(request_number),
    UNIQUE KEY request_number (request_number),
    foreign key(patient_id , doctor_id , date) references appointment(patient_id , doctor_id , date)
    );

create table IF NOT EXISTS equipment(
        manufacturer  varchar(50) NOT NULL,
    serial_number  varchar(50) NOT NULL,
        model  varchar(50) NOT NULL,
    primary key(manufacturer, serial_number, model)
    );

create table IF NOT EXISTS study(
        request_number  integer NOT NULL,
    description  varchar(255) NOT NULL,
        date  date NOT NULL,
        doctor_id  integer NOT NULL,
        manufacturer  varchar(50) NOT NULL,
        serial_number  varchar(50) NOT NULL,
    primary key(request_number, description),
    foreign key(request_number) references request(request_number),
    foreign key(doctor_id) references doctor(doctor_id),
    foreign key(manufacturer , serial_number) references equipment(manufacturer , serial_number)
    );

create table IF NOT EXISTS series(
        series_id  integer NOT NULL,
    name  varchar(100) NOT NULL,
        base_url  varchar(255) NOT NULL,
        request_number  integer NOT NULL,
        description  varchar(255) NOT NULL,
    primary key(series_id),
    UNIQUE KEY series_id (series_id),
    foreign key(request_number , description) references study(request_number , description)
    );

create table IF NOT EXISTS element(
        series_id  integer NOT NULL,
    elem_index  integer NOT NULL,
    primary key(series_id, elem_index),
    foreign key(series_id) references series(series_id)
    );

create table IF NOT EXISTS region(
        series_id  integer NOT NULL,
    elem_index  integer NOT NULL,
    x1  decimal(4,3) NOT NULL,
    y1  decimal(4,3) NOT NULL,
    x2  decimal(4,3) NOT NULL,
```

```
y2  decimal(4,3) NOT NULL,
primary key(series_id, elem_index, x1, y1 ,x2 , y2),
foreign key(series_id , elem_index) references element(series_id , elem_index)
);
```

**Explicação:**

- Foi utilizado um método de criação de tabelas em todo, semelhante àquele encontrado no primeiro laboratório;
- Existem algumas variáveis (varchar) que contêm um número de caracteres inferiores a 255, pois o grupo, achou que seria uma maneira de que o programa fosse eficiente a nível de memória;
- Existem alguma restrições extras, como por exemplo os valores não puderem ser nulos, existir um incremento automárico nos id's e também a destruição das tabelas no início do script (usamos o foreign key check a zero para não ser necessário eliminar as tabelas por ordem).

## 2. Write a query to retrieve the name of the doctor that performed more studies containing as “X-ray” in the description, and that were performed in the last 7 seven days with an equipment from manufacturer “Philips”

**Script Utilizado:**

```
select name
from doctor as d natural join study as s
where s.description = 'X-ray' AND s.manufacturer = 'Philips' AND (datediff(current_date, s.date) <= 7)
group by name
having count(request_number) >= all (select count(request_number) from study WHERE (datediff(current_date, study.date) <= 7) group by doctor_id);
```

**Script com dados aleatórios para efeitos de teste (o mesmo script para as alíneas seguintes):**

```
USE ist175268;

/* Limpeza das tabelas */
SET FOREIGN_KEY_CHECKS = 0; /* Disable foreign key checking.*/
TRUNCATE TABLE region;
TRUNCATE TABLE element;
TRUNCATE TABLE series;
TRUNCATE TABLE study;
TRUNCATE TABLE equipment;
TRUNCATE TABLE request;
TRUNCATE TABLE appointment;
TRUNCATE TABLE patient;
TRUNCATE TABLE doctor;
SET FOREIGN_KEY_CHECKS = 1; /* Enable foreign key checking. */

/* patient */
insert into patient values (5, 'John', '1994-04-14', 'Rua das Lagartixas, segunda osga à direita');
insert into patient values (6, 'Carlos', '1990-12-25' ,'Sem abrigo okay?');
insert into patient values (2, 'Carla',  '1999-02-28' ,'@@@ Rebentar com o sistema %&/()');
insert into patient values (7, 'Emma',  '2004-05-17' ,'Tipica casa em Samora Correia');
insert into patient values (4, 'Roberta',  '1900-03-20' ,'Casa');
insert into patient values (3, 'Francesca',  '1993-05-12' ,'Italia');
insert into patient values (1, 'Joaquim',  '2000-07-18' ,'Portugal');
insert into patient (name, birthday, address) values ('Matilde',  '2018-12-31' ,'Madeira');

/* doctor */
insert into doctor values (1, 'António', 'Clinical specialist');
insert into doctor values (2,'Joana', 'Clinical specialist');
insert into doctor values (3, 'Maria', 'Technical specialist');
insert into doctor values (4, 'Raul', 'Technical specialist');
insert into doctor values (5, 'Fábio', 'Technical specialist');

/* appointment */
```

```
insert into appointment values (4, 1, '2016-11-01', 'Escritorio 4');
insert into appointment values (5, 2, '2016-11-01', 'Foi numa tasca');
insert into appointment values (5, 3, '2016-11-02',  'Nao me lembro, tive um acidente nesse dia');
insert into appointment values (7, 3, '2016-11-02', 'Escritorio na Australia');
insert into appointment values (6, 4, '2016-11-01',  'Escritorio de Las Vegas');
insert into appointment values (2, 3, '2016-11-03',  'Polinesia Francesa');
insert into appointment values (1, 4, '2016-11-04',  'Polinesia Francesa');
```

**/* request */**

```
insert into request values (1 , 4, 1, '2016-11-01');
insert into request values (2 , 5, 2, '2016-11-01');
insert into request values (3 , 6, 4, '2016-11-01');
insert into request values (4 , 5, 3, '2016-11-02');
insert into request values (5 , 7, 3, '2016-11-02');
insert into request values (6 , 2, 3, '2016-11-03');
insert into request values (7 , 1, 4, '2016-11-04');
```

**/* equipment */**

```
insert into equipment values ( 'Philips', '1118-1590-5341-6933-7029-3016' , 'Model 5');
insert into equipment values ( 'Philips', '8MEH-RVNUH-PDU6U-E3VSR-VBTV3-VEMBR-ACED' , 'Model 4' );
insert into equipment values ( 'Philips', 'AXBN-VB3R-79H5-5MHD' , 'Model 3');
insert into equipment values ( 'Roger', 'YM38-Q84W-75CA-YYQD' , 'Unique Model' );
insert into equipment values ( 'Droid' , '2EC72-368A4-5E4E9-D54A1' , 'Best Model' );
insert into equipment values ( 'Philips', 'FR4S-8542-UYQW-USH9' , 'Last Model');
insert into equipment values ( 'R2D2', 'Ahah-hehe-ihih-hoho-uhuh' , 'Space Unit');
```

**/* study */**

```
insert into study values ( 1, 'Electrocardiograma', '2016-11-07' , 2 , 'Philips', '1118-1590-5341-6933-7029-3016');
insert into study values ( 2, 'X-ray', '2016-11-07' , 1, 'Philips', '8MEH-RVNUH-PDU6U-E3VSR-VBTV3-VEMBR-ACED');
insert into study values ( 3, 'X-ray' , '2016-11-09', 3, 'Philips', 'AXBN-VB3R-79H5-5MHD');
insert into study values ( 4, 'Electrocardiograma', '2016-11-09', 4 , 'Roger', 'YM38-Q84W-75CA-YYQD');
insert into study values ( 5, 'X-ray' , '2016-11-09', 5 , 'Droid' , '2EC72-368A4-5E4E9-D54A1');
insert into study values ( 6, 'X-ray' , '2016-11-10', 2, 'Philips', 'FR4S-8542-UYQW-USH9');
insert into study values ( 7, 'X-ray' , '2016-11-11', 3, 'Philips', 'FR4S-8542-UYQW-USH9');
```

**/* series */**

```
insert into series values (1, 'Series A' , '/home/ruben/Documents/' , 1 , 'Electrocardiograma');
insert into series values (2, 'Series B' , '/home/ruben/Models/' , 2 , 'X-ray');
insert into series values (3, 'Series C' , '/home/ruben/Downloads/' , 3 , 'X-ray');
insert into series values (4, 'Series D' , '/home/ruben/Music/' , 4 , 'Electrocardiograma');
insert into series values (5, 'Series E' , '/home/ruben/Pictures/' , 5 , 'X-ray');
insert into series values (6, 'Series F' , '/home/ruben/Desktop/' , 6 , 'X-ray');
insert into series values (7, 'Series G' , '/home/ruben/IST' , 7 , 'X-ray');
```

**/* element */**

```
insert into element values (1, 1);
insert into element values (1, 2);
```

```
insert into element values (1, 3);
insert into element values (2, 1);
insert into element values (2, 2);
insert into element values (3, 3);
insert into element values (4, 4);
insert into element values (5, 5);
insert into element values (6, 6);
insert into element values (7, 7);
```

**/* region */**

```
insert into region values (1, 1 , 0 , 0 , 0.2 , 0.2);
insert into region values (2, 2 , 0.25 , 0.25 ,  0.4 , 0.4);
insert into region values (3, 3 , 0.5, 0.5 , 0.9, 0.9);
insert into region values (4, 4 , 0.4 , 0.4 , 0.5 , 0.5);
insert into region values (5, 5 , 0.425 ,0.425 , 0.495 , 0.495);
```

**/* Teste da função*/**

```
insert into region values (1, 1 , 0.150 , 0.155 ,  0.160 , 0.165);
insert into region values (1, 2 , 0.200 , 0.200 ,  0.249 , 0.249);
insert into region values (1, 3 , 0.350 , 0.350 ,  0.399 , 0.399);
insert into region values (2, 1 , 0.300 , 0.300 ,  0.500 , 0.500);
```

**/* Rebenta na primeira parte do trigger */**

```
insert into appointment values (8, 1, '2016-11-06',  'Explosion in trigger part 1');
insert into request values (8 , 8, 1, '2016-11-06');

insert into study values ( 8, 'X-ray' , '2016-11-06', 1 , 'R2D2' , 'Ahah-hehe-ihih-hoho-uhuh');
```

**/*Resposta : ERROR 1644 (45000): The same doctor cannot perform any study that he/she requested! */**

**/* Rebenta na segunda parte do trigger */**

```
insert into appointment values (8, 1, '2016-11-05',  'Explosion in trigger part 2');
insert into request values (9 , 8, 1, '2016-11-05');

insert into study values ( 9 , 'X-ray' , '2016-10-31', 2, 'R2D2', 'Ahah-hehe-ihih-hoho-uhuh');
```

**/*Resposta : ERROR 1644 (45000): The date of a study must be posterior to the date of the appointment that requested the study! */**

**Explicação:**

Correndo a query anteriormente feita no dia 16 de Novembrode 2016, obtemos:

```
+-------+
| name  |
+-------+
| Maria |
+-------+

1 row in set (0.00 sec)
```

## 3. Write a trigger to prevent the same doctor from performing any study that he/she requested. Additionally, the date of a study must be posterior to the date of the appointment that requested the study

**Script Utilizado:**

```
USE ist175268;

delimiter $$
create trigger protect_study before insert on study
for each row
begin
/* declaration of variables */
declare studyRN integer;
declare requestRN integer;
declare studyDI integer;
declare requestDI integer;
declare dateDifference integer;
declare appointmentDT date;
declare studyDT date;

SET studyRN = new.request_number;
SET studyDI = new.doctor_id;
SET requestDI = (select request.doctor_id from request where request.request_number = studyRN);
SET studyDT = new.date;

/* Assumindo que o appointment tem a mesma data que o resquest */
SET appointmentDT = (select distinct appointment.date from appointment, request , study where
appointment.date = request.date
        AND request.request_number = studyRN);
SET dateDifference = (select datediff(studyDT , appointmentDT));
if ( studyDI = requestDI)
then
/* Na Versao 5.7 funcionaria assim, como no servidor do tecnico nao funciona, usamos a linha
nao comentada
signal sqlstate '45000' set message_text = 'The same doctor cannot perform any study that
he/she requested!';*/
call the_same_doctor_cannot_perform_any_study_he_or_she_requested();
elseif ( dateDifference <= 0 )
then
/* Na Versao 5.7 funcionaria assim, como no servidor do tecnico nao funciona, usamos a linha
nao comentada
signal sqlstate '45000' set message_text = 'The date of a study must be posterior to the date of
the appointment that requested the study!';*/
call study_date_isnt_posterior_to_date_of_the_appointment();
end if;
end$$
delimiter ;
```

**Explicação:**

mysql> insert into study values ( 8, 'X-ray' , '2016-11-06', 1 , 'R2D2' , 'Ahah-hehe-ihih-hoho-uhuh');
**ERROR 1305 (42000): PROCEDURE ist175268.the_same_doctor_cannot_perform_any_study_he_or_she_requested does not exist**

mysql> insert into study values ( 9 , 'X-ray' , '2016-10-31', 2, 'R2D2', 'Ahah-hehe-ihih-hoho-uhuh');
**ERROR 1305 (42000): PROCEDURE ist175268.study_date_isnt_posterior_to_date_of_the_appointment does not exist**

- Quando se tentou inserir estes dados, estes foram os erros que apareceram, é portanto aconcelhava a criar o trigger primeiro e de seguida correr o script de inserção de dados (fornecino neste projecto) para obtenção dos mesmo resultados

## 4. Write a function that, given two series A and B, returns true if any region of the elements of A overlaps with any region of the elements of B, and false otherwise

**Script Utilizado:**

```
USE ist175268;

delimiter $$
create function dettectIntersection(series_id_A integer, series_id_B integer)
returns varchar(5)
begin
declare intersection varchar(5);

/* inicialization of the variable*/
SET intersection = 'False';
if
(
	select distinct true
	from (select series_id, x1, y1, x2, y2 from region where series_id = series_id_A) as s1,
	(select series_id, x1, y1, x2, y2 from region where series_id = series_id_B) as s2
	WHERE
	(
		((s1.x1 >= LEAST(s2.x1, s2.x2) AND s1.x1 <= GREATEST(s2.x1, s2.x2))
		OR
		(s1.x2 >= LEAST(s2.x1, s2.x2) AND s1.x2 <= GREATEST(s2.x1, s2.x2)))
		AND
		((s1.y1 >= LEAST(s2.y1, s2.y2) AND s1.y1 <= GREATEST(s2.y1, s2.y2))
		OR
		(s1.y2 >= LEAST(s2.y1, s2.y2) AND s1.y2 <= GREATEST(s2.y1, s2.y2)))
	OR
	(
		((s2.x1 >= LEAST(s1.x1, s1.x2) AND s2.x1 <= GREATEST(s1.x1, s1.x2))
		OR
		(s2.x2 >= LEAST(s1.x1, s1.x2) AND s2.x2 <= GREATEST(s1.x1, s1.x2)))
		AND
		((s2.y1 >= LEAST(s1.y1, s1.y2) AND s2.y1 <= GREATEST(s1.y1, s1.y2))
		OR
		(s2.y2 >= LEAST(s1.y1, s1.y2) AND s2.y2 <= GREATEST(s1.y1, s1.y2)))
	)
)
)
THEN SET intersection = 'True';
end if;
return intersection;
end$$
delimiter ;
```

**Explicação:**

**mysql> select dettectIntersection(1,2);**

**insert into region values (1, 3 , 0.350 , 0.350 ,  0.399 , 0.399);**

```
+---------------------------+
| dettectIntersection(1,2) |
+---------------------------+
|           True            |
+---------------------------+
  1 row in set (0.00 sec)
```

**/*insert into region values (1, 3 , 0.350 , 0.350 ,  0.399 , 0.399);*/**

```
+---------------------------+
| dettectIntersection(1,2) |
+---------------------------+
|           False           |
+---------------------------+
  1 row in set (0.00 sec)
```

Em cima temos os resultados da função criada, no primeiro caso, aquando da inserção dos dados na base de dados, não temos a linha a negrito comentada, e no segundo caso, comentado a linha a negrito.

**Pequena justificação de como funciona a função:**

- Temos uma variável que é definida como "intersection" que toma o valor False caso não exista intersecção e toma o valor True caso exista;
- A pergunta "existe intersecção?" é respondida na query que utiliza o select. Esta query junta numa só tabela, as duas séries A e B a comparar (dadas como input pelo utilizador) e analisa os seus pontos;
- Dentro do select temos um where que utiliza 2 funções (não dadas em aula) que verificam se os pontos da série 1 então entre os pontos da série 2 (ou seja, se um ponto da série 1 é maior que o menor dos dois pontos da série 2 e menor que o maior dos dois pontos da série 2), tanto para a coordenada do x como para a coordenada do y;
- Com este WHERE, está associado um OR, que serve para cobrir a excepção de um quadrado estar dentro do outro.