TÉCNICO LISBOA

Projecto de Sistemas de Informação e Bases de Dados

1st Semester 2017/2018

Project Assignment - Part II - Report

Database Creation, Querying & Functions

Group 20:

– João Alves, Nº 78181.
– Marco Montez. Nº 78508
– Tomás Cordovil, Nº79021.

Departamento de Engenharia Electrotécnica e de Computadores

Contents

1 Introduction

In this second part of the project assignment our goal is to create a database, inspired on the schema elaborated on the first part of the assignment, using the schema provided in the project statement. We will also implement some relevant queries for the health care centre, further on this report.
This report includes the data types chosen for each column, the SQL script to populate the database, the requested queries, the requested triggers as well as the **region_overlaps_element()** function.

2 Table Creation

2.1 Code

```
DROP TABLE IF EXISTS Region, Element, Series, Study, Request, Wears, Period, Reading, Sensor
    , Device, Doctor, Patient;

--Table creation

create table Patient(
   number          integer NOT NULL,
   name            varchar(255) NOT NULL,
   birthday        date NOT NULL,
   address         varchar(255) NOT NULL,
   primary key(number)
);

create table Doctor(
   number          integer NOT NULL,
   doctor_id       integer NOT NULL,
   primary key(doctor_id),
   foreign key(number) references Patient(number)
);

create table Device(
   serialnum       integer NOT NULL,
   manufacturer    varchar(255) NOT NULL,
   model           varchar(255) NOT NULL,
   primary key(serialnum, manufacturer)
);

create table Sensor(
   snum           integer NOT NULL,
   manuf          varchar(255) NOT NULL,
   units          varchar(255) NOT NULL,
   primary key(snum, manuf),
   foreign key(snum, manuf) references Device(serialnum, manufacturer)
);

create table Reading(
   snum           integer NOT NULL,
   manuf          varchar(255) NOT NULL,
   datetime       timestamp NOT NULL,
   value          numeric(15,1) NOT NULL,
   primary key(snum, manuf, datetime),
   foreign key(snum, manuf) references Sensor(snum, manuf)
);

create table Period(
   start           timestamp NOT NULL,
   end             timestamp NOT NULL,
   primary key(start, end)
);

create table Wears(
   start          timestamp NOT NULL,
   end            timestamp NOT NULL,
   patient        integer NOT NULL,
   snum           integer NOT NULL,
   manuf          varchar(255) NOT NULL,
   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          integer NOT NULL,
    patient_id      integer NOT NULL,
    doctor_id       integer NOT NULL,
    date            timestamp NOT NULL,
    primary key(number),
    foreign key(patient_id) references Patient(number),
    foreign key(doctor_id) references Doctor(doctor_id)
);

create table Study (
    request_number   integer NOT NULL,
    description      varchar(255) NOT NULL,
    date             timestamp NOT NULL,
    doctor_id        integer NOT NULL,
    manufacturer     varchar(255) NOT NULL,
    serial_number    integer NOT NULL,
    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        integer NOT NULL,
    name            varchar(255) NOT NULL,
    base_url         varchar(255) NOT NULL,
    request_number   integer NOT NULL,
    description      varchar(255) NOT NULL,
    primary key(series_id),
    foreign key(request_number, description) references Study(request_number, description)
);

create table 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 Region(
    series_id        integer NOT NULL,
    elem_index       integer NOT NULL,
    x1               numeric(2,2) NOT NULL,
    y1               numeric(2,2) NOT NULL,
    x2               numeric(2,2) NOT NULL,
    y2               numeric(2,2) NOT NULL,
    primary key(series_id, elem_index, x1, y1, x2, y2),
    foreign key(series_id, elem_index) references Element(series_id, elem_index)
);
```

2.2 Comments

The table creation can be found in the *creation.sql*.
The tables were created as requested in the project statement. We had some problems with some constraints but they were resolved in the lab classes. We've added the *NOT NULL* constraint to all table entries because we want the tables to have significant values and not accept incomplete entries.

3 Table Population

3.1 Code

```
--Now that every table is created let's fill them


--This populates Patient table

insert into Patient values(28484755, 'Angela Martins', '1930-04-19', 'Rua do Alecrim, 16');
insert into Patient values(84853295, 'Domiciano Cavem', '1932-12-21', 'Rua do Carvalho, 69');
insert into Patient values(87478283, 'Mario Coluna', '1935-08-06', 'Rua do Eterno Capitao, 36');
insert into Patient values(48529837, 'Costa Pereira', '1929-12-22', 'Rua do Eucalipto, 12');
insert into Patient values(53892894, 'Fernanda Cruz', '1940-10-12', 'Rua do Sobreiro, 17');
insert into Patient values(52935753, 'Germano Figueiredo', '1932-12-23', 'Rua do Jivago, 21');
insert into Patient values(89587353, 'Jose Aguas', '1930-11-09', 'Rua da Cabeca de Ouro, 3');
insert into Patient values(98678233, 'Eusebio Ferreira', '1942-01-25', 'Rua da Pantera Negra, 72');
insert into Patient values(39040532, 'Ines Casa De Agua', '1895-03-31', 'Rua de Salem, 66');
insert into Patient values(39495830, 'Joaquim Hyde', '1831-07-25', 'Rua do Parque, 7');
insert into Patient values(38958323, 'Jack Ripper', '1947-01-23', 'Rua de Londres, 69');
insert into Patient values(48958394, 'Frederico Krueger', '1953-11-13', 'Rua de Elm, 13');

--This populates Doctor table

insert into Doctor values(48958394, 8489);
insert into Doctor values(38958323, 6234);
insert into Doctor values(39495830, 5235);
insert into Doctor values(39040532, 6948);
insert into Doctor values(53892894, 4343);

--This Populates Device

insert into Device values(48394, 'Samsung', 'A');
insert into Device values(48993, 'Samsung', 'C');
insert into Device values(73464, 'Samsung', 'B');
insert into Device values(77443, 'Samsung', 'D');
insert into Device values(88543, 'Samsung', 'A');
insert into Device values(23567, 'Samsung', 'E');
insert into Device values(67663, 'Samsung', 'F');
insert into Device values(45632, 'Medtronic', 'A');
insert into Device values(64467, 'Medtronic', 'B');
insert into Device values(63467, 'Medtronic', 'A');
insert into Device values(85633, 'Medtronic', 'B');
insert into Device values(96455, 'Medtronic', 'E');
insert into Device values(86864, 'Medtronic', 'C');
insert into Device values(83246, 'Medtronic', 'A');
insert into Device values(48394, 'Siemens', 'Z');
insert into Device values(63255, 'Siemens', 'G');
insert into Device values(89238, 'Siemens', 'B');
insert into Device values(77823, 'Siemens', 'A');
insert into Device values(03857, 'Siemens', 'B');
insert into Device values(34374, 'Siemens', 'R');
insert into Device values(98898, 'Siemens', 'Y');
insert into Device values(37847, 'Novartis', 'A');
insert into Device values(12231, 'Novartis', 'A');
insert into Device values(54646, 'Novartis', 'A');
insert into Device values(84842, 'Novartis', 'A');
insert into Device values(88886, 'Novartis', 'A');
insert into Device values(84563, 'Novartis', 'A');
```

```
insert into Device values(26583, 'Novartis', 'A');
insert into Device values(83457, 'Novartis', 'A');
insert into Device values(72422, 'Novartis', 'A');

--This populates Sensor table

insert into Sensor values(48394, 'Samsung', 'LDL cholesterol in mg/dL');
insert into Sensor values(48993, 'Samsung', 'Amount of blood in L');
insert into Sensor values(73464, 'Samsung', 'Cranium Pressure in Pa');
insert into Sensor values(45632, 'Medtronic', 'LDL cholesterol in mg/dL');
insert into Sensor values(64467, 'Medtronic', 'Body temperature in Celsius');
insert into Sensor values(63467, 'Medtronic', 'Body mass in kG');
insert into Sensor values(37847, 'Novartis', 'Creatine in mg/dL');
insert into Sensor values(12231, 'Novartis', 'Amount of urine in L');
insert into Sensor values(54646, 'Novartis', 'Heartrate in bpm');

--This populates Reading table

insert into Reading values(48394, 'Samsung', '2014-05-05 12:45:01', '202');
insert into Reading values(48394, 'Samsung', '2017-01-04 15:04:24', '193');
insert into Reading values(48394, 'Samsung', '2017-04-12 18:35:56', '174');
insert into Reading values(48394, 'Samsung', '2016-10-01 01:05:23', '186');
insert into Reading values(48993, 'Samsung', '2013-12-15 13:07:05', '1.4');
insert into Reading values(48993, 'Samsung', '2012-05-25 16:08:04', '2.0');
insert into Reading values(48993, 'Samsung', '2010-06-01 12:12:12', '1.9');
insert into Reading values(48993, 'Samsung', '2017-01-06 01:56:02', '4.0');
insert into Reading values(73464, 'Samsung', '2015-03-05 02:01:09', '2.6');
insert into Reading values(73464, 'Samsung', '2012-01-01 01:06:34', '1.0');
insert into Reading values(73464, 'Samsung', '2014-05-30 04:01:01', '3.1');
insert into Reading values(73464, 'Samsung', '2010-02-14 15:10:56', '4.2');
insert into Reading values(45632, 'Medtronic', '2017-06-25 18:47:56', '186');
insert into Reading values(45632, 'Medtronic', '2016-07-01 19:15:00', '178');
insert into Reading values(45632, 'Medtronic', '2015-09-17 14:15:16', '198');
insert into Reading values(45632, 'Medtronic', '2017-09-01 17:20:13', '215');
insert into Reading values(64467, 'Medtronic', '2017-07-17 13:25:16', '38.3');
insert into Reading values(64467, 'Medtronic', '2017-07-18 17:26:10', '38.7');
insert into Reading values(64467, 'Medtronic', '2017-07-19 16:10:13', '38.2');
insert into Reading values(64467, 'Medtronic', '2017-08-01 13:20:13', '36.5');
insert into Reading values(63467, 'Medtronic', '2013-06-12 09:27:43', '76');
insert into Reading values(63467, 'Medtronic', '2014-11-21 18:30:13', '78');
insert into Reading values(63467, 'Medtronic', '2015-01-30 14:27:53', '73');
insert into Reading values(63467, 'Medtronic', '2017-09-21 10:20:13', '70');
insert into Reading values(37847, 'Novartis', '2017-03-02 17:40:13', '1.2');
insert into Reading values(37847, 'Novartis', '2017-06-04 18:10:13', '2.1');
insert into Reading values(37847, 'Novartis', '2016-05-12 10:10:13', '3.1');
insert into Reading values(37847, 'Novartis', '2017-09-15 09:10:13', '3.3');
insert into Reading values(12231, 'Novartis', '2015-01-16 20:40:33', '4.3');
insert into Reading values(12231, 'Novartis', '2015-04-01 17:50:43', '4.2');
insert into Reading values(12231, 'Novartis', '2016-01-13 17:20:13', '1.2');
insert into Reading values(12231, 'Novartis', '2017-09-21 13:24:23', '1.0');
insert into Reading values(54646, 'Novartis', '2014-08-03 11:20:13', '79');
insert into Reading values(54646, 'Novartis', '2017-02-02 11:10:03', '110');
insert into Reading values(54646, 'Novartis', '2017-08-05 15:30:43', '67');
insert into Reading values(54646, 'Novartis', '2017-10-30 16:30:13', '95');

--This populates Time table

insert into Period values ('2010-09-20 11:50:22', '2013-01-20 10:40:53');
insert into Period values ('2011-01-20 00:30:13', '2017-10-13 20:50:23');
insert into Period values ('2015-02-12 21:10:01', '2017-02-15 09:15:12');
insert into Period values ('2011-08-23 17:34:31', '2017-10-25 10:15:13');
insert into Period values ('2011-08-23 17:34:31', '2017-10-30 16:30:15');
insert into Period values ('2014-04-05 12:16:23', '2014-10-18 17:26:11');
insert into Period values ('2017-10-01 10:21:10', '2017-10-01 12:48:27');
```

```
insert into Period values ('2016-08-15 17:23:09', '2017-02-16 22:42:11');
insert into Period values ('2017-05-22 11:40:43', '2017-09-01 18:15:14');
insert into Period values ('2011-02-21 14:20:15', '2017-07-13 10:50:33');
insert into Period values ('2017-05-22 11:40:43', '2017-10-31 18:15:14');
insert into Period values ('2010-04-05 12:16:23', '2010-10-18 17:26:11');
insert into Period values ('2011-01-20 00:30:13', '2014-11-13 20:50:23');
insert into Period values ('2011-01-20 00:30:13', '2017-10-13 20:50:24');

--This populates Wears table

insert into Wears values ('2010-09-20 11:50:22', '2013-01-20 10:40:53', 28484755, 48394, '
    Samsung');
insert into Wears values ('2011-01-20 00:30:13', '2017-10-13 20:50:23', 28484755, 45632, '
    Medtronic');
insert into Wears values ('2015-02-12 21:10:01', '2017-02-15 09:15:12', 28484755, 48394, '
    Siemens');
insert into Wears values ('2010-04-05 12:16:23', '2010-10-18 17:26:11', 84853295, 45632, '
    Medtronic');
insert into Wears values ('2017-10-01 10:21:10', '2017-10-01 12:48:27', 84853295, 48394, '
    Samsung');
insert into Wears values ('2011-01-20 00:30:13', '2014-11-13 20:50:23', 87478283, 48993, '
    Samsung');
insert into Wears values ('2011-02-21 14:20:15', '2017-07-13 10:50:33', 48529837, 54646, '
    Novartis');
insert into Wears values ('2016-08-15 17:23:09', '2017-02-16 22:42:11', 48529837, 12231, '
    Novartis');
insert into Wears values ('2011-08-23 17:34:31', '2017-10-25 10:15:13', 53892894, 37847, '
    Novartis');
insert into Wears values ('2014-04-05 12:16:23', '2014-10-18 17:26:11', 53892894, 63467, '
    Medtronic');
insert into Wears values ('2011-02-21 14:20:15', '2017-07-13 10:50:33', 52935753, 73464, '
    Samsung');
insert into Wears values ('2015-02-12 21:10:01', '2017-02-15 09:15:12', 98678233, 64467, '
    Medtronic');
insert into Wears values ('2011-01-20 00:30:13', '2014-11-13 20:50:23', 98678233, 98898, '
    Siemens');
insert into Wears values ('2011-01-20 00:30:13', '2017-10-13 20:50:23', 48529837, 48394, '
    Samsung');
UPDATE Wears SET snum=48394, manuf='Samsung' WHERE start ='2011-02-21 14:20:15' AND end ='
    2017-07-13 10:50:33' AND patient=52935753;

--This populates Request table

insert into Request values(9838, 28484755, 8489, '2010-01-14 12:26:52');
insert into Request values(7653, 87478283, 6234, '2010-09-17 09:01:34');
insert into Request values(5436, 48529837, 8489, '2011-10-16 14:19:20');
insert into Request values(3263, 53892894, 5235, '2012-05-25 14:33:07');
insert into Request values(8753, 52935753, 5235, '2012-07-12 16:56:59');
insert into Request values(0978, 89587353, 6234, '2012-12-12 12:12:12');
insert into Request values(1423, 98678233, 6948, '2013-07-25 17:43:19');
insert into Request values(9876, 39040532, 4343, '2013-12-21 08:24:00');
insert into Request values(4362, 39495830, 6234, '2014-05-30 18:50:25');
insert into Request values(9394, 38958323, 4343, '2014-11-15 10:20:52');
insert into Request values(0987, 48958394, 8489, '2016-02-10 14:01:01');
insert into Request values(4658, 98678233, 8489, '2017-06-13 18:51:54');
insert into Request values(8457, 38958323, 8489, '2017-02-23 11:40:34');
insert into Request values(7563, 38958323, 8489, '2015-02-23 11:40:34');




--This populates Study table (for some reason, it doesn't work without specifying the
    columns)
insert into Study values(7563, 'Blood Scan', '2016-10-17 09:01:34', 5235, 'Medtronic',
```

```
    45632);
insert into Study values(7563, 'CAT Scan', '2016-10-17 10:01:34', 5235, 'Medtronic', 64467);
insert into Study values(7563, 'Torso Scan', '2016-10-17 11:01:34', 5235, 'Medtronic',
    63467);
insert into Study values(7563, 'Toe nail Scan', '2016-10-17 12:01:34', 5235, 'Medtronic',
    85633);
insert into Study values(7563, 'ECG', '2016-10-17 13:01:34', 5235, 'Medtronic', 96455);
insert into Study values(7563, 'X-Ray', '2016-10-17 14:01:34', 5235, 'Medtronic', 86864);
insert into Study values(7563, 'Left Arm Scan', '2016-10-17 15:01:34', 5235, 'Medtronic',
    83246);
insert into Study values(9838, 'ECG', '2010-02-14 12:26:52', 6234, 'Novartis', 72422);
insert into Study values(9838, 'Torso CAT Scan', '2010-03-01 18:30:00', 6234, 'Samsung',
    88543);
insert into Study values(7653, 'Blood Scan', '2010-10-17 09:01:34', 5235, 'Medtronic',
    83246);
insert into Study values(5436, 'Left Ankle X-Ray', '2011-11-16 14:19:20', 6234, 'Siemens',
    98898);
insert into Study values(5436, 'Right Ankle X-Ray', '2012-01-26 23:11:21', 6234, 'Siemens',
    98898);
insert into Study values(3263, 'Head CAT Scan', '2012-06-25 14:33:07', 8489, 'Samsung',
    88543);
insert into Study values(8753, 'Blood Scan', '2012-08-12 16:56:59', 8489, 'Medtronic',
    83246);
insert into Study values(0978, 'ECG', '2013-01-12 12:12:12', 5235, 'Novartis', 72422);
insert into Study values(0978, 'Head CAT Scan', '2013-03-14 15:24:21', 5235, 'Samsung',
    88543);
insert into Study values(1423, 'Torso X-Ray', '2013-08-25 17:43:19', 8489, 'Siemens', 98898)
    ;
insert into Study values(1423, 'Blood Scan', '2013-12-02 15:34:12', 8489, 'Medtronic',
    83246);
insert into Study values(1423, 'ECG', '2013-12-29 17:11:21', 8489, 'Novartis', 72422);
insert into Study values(9876, 'Left leg X-Ray', '2014-01-21 08:24:00', 6948, 'Siemens',
    98898);
insert into Study values(4362, 'Right leg X-Ray', '2015-06-30 18:50:25', 6948, 'Siemens',
    98898);
insert into Study values(9394, 'Blood Scan', '2014-12-15 10:20:52', 6234, 'Medtronic',
    83246);
insert into Study values(9394, 'Head CAT Scan', '2015-04-31 19:20:31', 6234, 'Samsung',
    88543);
insert into Study values(0987, 'ECG', '2016-03-10 14:01:01', 4343, 'Novartis', 72422);
insert into Study values(4658, 'Wrist X-Ray', '2017-07-10 14:01:01', 4343, 'Medtronic',
    86864);
insert into Study values(8457, 'Left Leg X-Ray', '2017-03-23 11:40:31', 4343, 'Medtronic',
    86864);
insert into Study values (9838, 'ECG', '2010-02-14 12:26:52', 8489, 'Novartis', 72422);
UPDATE Study SET description='Left Leg X-Ray', date='2017-03-23 11:40:31', doctor_id=8489,
    manufacturer='Medtronic', serial_number=86864 WHERE request_number=8457;

--This populates Series table

insert into Series values(6201, 'A', 'results.com/a6201', 9838, 'ECG');
insert into Series values(1402, 'B', 'results.com/b1402', 9838, 'Torso CAT Scan');
insert into Series values(5203, 'C', 'results.com/c5203', 7653, 'Blood Scan');
insert into Series values(3404, 'A', 'results.com/a3404', 5436, 'Left Ankle X-Ray');
insert into Series values(2505, 'B', 'results.com/b2505', 5436, 'Right Ankle X-Ray');
insert into Series values(2506, 'C', 'results.com/c2506', 3263, 'Head CAT Scan');
insert into Series values(3207, 'A', 'results.com/a3207', 8753, 'Blood Scan');
insert into Series values(3708, 'B', 'results.com/b3708', 0978, 'ECG');
insert into Series values(3509, 'C', 'results.com/c3509', 0978, 'Head CAT Scan');
insert into Series values(7210, 'A', 'results.com/a7210', 1423, 'Torso X-Ray');
insert into Series values(1211, 'B', 'results.com/b1211', 1423, 'Blood Scan');
insert into Series values(5212, 'C', 'results.com/c5212', 1423, 'ECG');
insert into Series values(7213, 'A', 'results.com/a7213', 9876, 'Left leg X-Ray');
insert into Series values(3214, 'B', 'results.com/b3214', 4362, 'Right leg X-Ray');
```

```
insert into Series values(8215, 'C', 'results.com/c8215', 9394, 'Blood Scan');
insert into Series values(5216, 'B', 'results.com/b5216', 9394, 'Head CAT Scan');
insert into Series values(8217, 'C', 'results.com/c8217', 0987, 'ECG');
insert into Series values(5618, 'A', 'results.com/a5618', 4658, 'Wrist X-Ray');
insert into Series values(8419, 'B', 'results.com/b8419', 8457, 'Left Leg X-Ray');

--This populates Element table

insert into Element values(6201, 1);
insert into Element values(1402, 1);
insert into Element values(1402, 2);
insert into Element values(5203, 1);
insert into Element values(3404, 1);
insert into Element values(3404, 2);
insert into Element values(2505, 1);
insert into Element values(2505, 2);
insert into Element values(2506, 1);
insert into Element values(2506, 2);
insert into Element values(3207, 1);
insert into Element values(3708, 1);
insert into Element values(3509, 1);
insert into Element values(3509, 2);
insert into Element values(3509, 3);
insert into Element values(7210, 1);
insert into Element values(7210, 2);
insert into Element values(7210, 3);
insert into Element values(1211, 1);
insert into Element values(5212, 1);
insert into Element values(7213, 1);
insert into Element values(7213, 2);
insert into Element values(3214, 1);
insert into Element values(3214, 2);
insert into Element values(8215, 1);
insert into Element values(5216, 1);
insert into Element values(5216, 2);
insert into Element values(5216, 3);
insert into Element values(8217, 1);
insert into Element values(5618, 1);
insert into Element values(8419, 1);
insert into Element values(8419, 2);

--This populates Region

insert into Region values(6201, 1, 0.5, 0.5, 0.7, 0.9);
insert into Region values(1402, 1, 0.3, 0.6, 0.7, 0.8);
insert into Region values(1402, 2, 0.1, 0.4, 0.2, 0.6);
insert into Region values(2505, 1, 0.1, 0.2, 0.5, 0.6);
insert into Region values(2505, 2, 0.2, 0.3, 0.6, 0.7);
insert into Region values(2506, 1, 0.05, 0.3, 0.55, 0.9);
insert into Region values(2506, 2, 0.05, 0.3, 0.55, 0.9);
insert into Region values(3207, 1, 0.2, 0.3, 0.3, 0.5);
insert into Region values(3708, 1, 0.1, 0.5, 0.3, 0.8);
insert into Region values(1211, 1, 0.45, 0.5, 0.7, 0.9);
insert into Region values(7213, 1, 0.15, 0.5, 0.3, 0.75);
insert into Region values(7213, 2, 0.25, 0.65, 0.45, 0.85);
insert into Region values(5216, 1, 0.05, 0.35, 0.25, 0.7);
insert into Region values(5216, 2, 0.1, 0.25, 0.35, 0.65);
insert into Region values(5216, 3, 0.2, 0.5, 0.5, 0.9);
insert into Region values(8217, 1, 0.4, 0.7, 0.6, 0.9);
```

3.2 Comments

The table population can be found in the *population.sql*.
We've populated the table with the values we saw fit to test the different queries and triggers. Note that the *population.sql* should be run after the *triggers.sql* since it contains some lines that activate the triggers.

4 Queries

4.1 Code

Query 1:

```
SELECT
  Patient.name
FROM
  Patient,
  Wears,
  Sensor,
  Reading
WHERE
  Patient.number=Wears.patient
AND
  Wears.snum=Sensor.snum
AND
  Wears.manuf=Sensor.manuf
AND
  Sensor.snum=Reading.snum
AND
  Sensor.manuf=Reading.manuf
AND
  Sensor.units='LDL cholesterol in mg/dL'
AND
  Wears.start<=Reading.datetime
AND
  Wears.end>=Reading.datetime
AND
  Reading.value>200
AND
  DateDiff(current_date,cast(Reading.datetime AS date)) <= 90
GROUP BY
  Patient.name
HAVING
  COUNT(*) >= all(SELECT COUNT(*) FROM Patient, Reading, Wears, Sensor
WHERE
  Patient.number = Wears.patient
AND
  Wears.snum = Sensor.snum
AND
  Wears.manuf=Sensor.manuf
AND
  Sensor.snum=Reading.snum
AND
  Sensor.manuf=Reading.manuf
AND
  Sensor.units = 'LDL cholesterol in mg/dL'
AND
  Wears.start<Reading.datetime
AND
  Wears.end>Reading.datetime
AND
  Reading.value>200
AND
  DateDiff(current_date(), cast(Reading.datetime AS date)) <= 90
GROUP BY
  Patient.name
  );
```

Query 2:

```
SELECT
    Patient.name
FROM
    Patient,
    Study,
    Request
WHERE
    Patient.number = Request.patient_id
    AND
      Request.number = Study.request_number
    AND
      Study.manufacturer = 'Medtronic'
    AND
      YEAR(Study.date) = YEAR(current_date - INTERVAL 1 YEAR)
    GROUP BY
      name
    HAVING
    COUNT(DISTINCT Study.serial_number)=(SELECT
                        COUNT(DISTINCT Device.serialnum)
                        FROM
                                Device
                        WHERE
                                Device.manufacturer = 'Medtronic');
```

4.2 Comments

The requested queries are presented in *query*1.*sql* and *query*2.*sql*.
We were to use the $max()$ function in the first query but opted to use $>= all()$ since this can return multiple results, which can happen to be true.
The queries return what was expected according with our database population.

5 Triggers

5.1 Code:

```
--Triggers

--Triggers that blocks invalid Period entries (not requested)
--Insert


DROP TRIGGER IF EXISTS insertPeriod;
DELIMITER $$
CREATE TRIGGER insertPeriod
BEFORE INSERT ON Period
FOR EACH ROW
BEGIN
  IF (new.start>new.end)
   THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = "INSERT Period start can't be after Period end";
   END IF;
END$$
DELIMITER ;


--Update (not requested)

DROP TRIGGER IF EXISTS updatePeriod;
DELIMITER $$
```

```
26 CREATE TRIGGER updatePeriod
27 BEFORE UPDATE ON Period
28 FOR EACH ROW
29 BEGIN
30   IF (new.start>new.end)
31    THEN
32       SIGNAL SQLSTATE '45000'
33       SET MESSAGE_TEXT = "UPDATE Period start can't be after Period end";
34    END IF;
35 END$$
36 DELIMITER ;
37
38
39 --Trigger if the doctor that prescribed the exam is set to conduct the exam thorugh an
       Insert in the Study table (requested).
40
41 --The trigger doesn't allow this to happen and block the insert.
42
43 DROP TRIGGER IF EXISTS insertDiffDoctor;
44 DELIMITER $$
45 CREATE TRIGGER insertDiffDoctor
46 BEFORE INSERT ON Study
47 FOR EACH ROW
48 BEGIN
49   IF (
50     new.doctor_id=(SELECT Request.doctor_id FROM Request WHERE Request.number = new.
           request_number))
51   THEN
52     SIGNAL SQLSTATE '45000'
53     SET MESSAGE_TEXT = "Sorry! Can't insert. The doctor that requests an exam cannot conduct
           that exam.";
54   END IF;
55 END$$
56 DELIMITER ;
57
58
59 --Trigger if the doctor that prescribed the exam is set to conduct the exam thorugh an
       Update in the Study table (requested).
60
61 --The trigger doesn't allow this to happen and blocks the update.
62
63 DROP TRIGGER IF EXISTS updateDiffDoctor;
64 DELIMITER $$
65 CREATE TRIGGER updateDiffDoctor
66 BEFORE UPDATE ON Study
67 FOR EACH ROW
68 BEGIN
69   IF (
70     new.doctor_id=(SELECT Request.doctor_id FROM Request WHERE Request.number = new.
           request_number))
71   THEN
72     SIGNAL SQLSTATE '45000'
73     SET MESSAGE_TEXT = "Sorry! Can't update. The doctor that requests an exam cannot conduct
           that exam.";
74   END IF;
75 END$$
76 DELIMITER ;
77
78
79 --Triggers if a device is to different patients during overlapping periods of time thorugh
       an Insert in the Wears table (requested).
80
81 --This trigger doesn't allow this to happen and blocks the insert.
82
```

```
83  DROP TRIGGER IF EXISTS insertDevicePeriod;
84
85  DELIMITER $$
86  CREATE TRIGGER insertDevicePeriod
87  BEFORE INSERT ON Wears
88  FOR EACH ROW
89  BEGIN
90     IF EXISTS(
91        SELECT * FROM Wears WHERE (((new.manuf=Wears.manuf)
92        AND (new.snum=Wears.snum)
93        AND ((new.end<Wears.end AND new.end>Wears.start) OR (new.start>Wears.start AND new.
              start<Wears.end)))))
94     THEN
95        SIGNAL SQLSTATE '45000'
96        SET MESSAGE_TEXT = "Overlapping Periods";
97     END IF;
98  END$$
99  DELIMITER ;
100
101
102 --This trigger doesn't allow this to happen and blocks the update (requested).
103 DROP TRIGGER IF EXISTS updateDevicePeriod;
104 DELIMITER $$
105 CREATE TRIGGER updateDevicePeriod
106 BEFORE UPDATE ON Wears
107 FOR EACH ROW
108 BEGIN
109    IF EXISTS(
110       SELECT * FROM Wears WHERE (((new.manuf=Wears.manuf)
111       AND (new.snum=Wears.snum)
112       AND ((new.end<Wears.end AND new.end>Wears.start) OR (new.start>Wears.start AND new.
              start<Wears.end)))))
113    THEN
114       SIGNAL SQLSTATE '45000'
115       SET MESSAGE_TEXT = "Overlapping Periods";
116    END IF;
117 END$$
118 DELIMITER ;
```

5.2 Comments:

The triggers requested are presented below as well as some we deemed relevant to the project and can be also found in *triggers.sql*. We've added two triggers that we deemed relevant for the project that ensure that the start of time period must be prior to its end (*insertPeriod* and *updatePeriod*). After testing we concluded that the triggers work as expected. Note that *triggers.sql* should be run before *population.sql* to see the triggers activate while *population.sql* is running

6 Function region_overlaps_element

6.1 Code:

```
1  --Function that returns TRUE if any Region B intersects with any Region A
2
3  DROP FUNCTION IF EXISTS region_overlaps_element;
4  DELIMITER $$
5  CREATE FUNCTION region_overlaps_element
6  (series_id int, e_index int, x1b int, y1b int, x2b int, y2b int)
7  RETURNS BOOLEAN
8  BEGIN
9     DECLARE x1a integer;
10    DECLARE y1a integer;
11    DECLARE x2a integer;
12    DECLARE y2a integer;
13    IF NOT EXISTS(
```

```
14      SELECT Region.x1, Region.y1, Region.x2, Region.y2 as x1a, y1a, x2a, y2a
15      FROM Region
16      WHERE (Region.series_id=series_id AND Region.elem_index=e_index AND
17            ((x1a<x1b AND x1a<x2b AND x2a<x1b AND x2a<x2b)
18          OR (x1a>x1b AND x1a>x2b AND x2a>x1b AND x2a>x2b)
19          OR (y1a<y1b AND y1a<y2b AND y2a<y1b AND y2a<y2b)
20          OR (y1a>y1b AND y1a>y2b AND y2a>y1b AND y2a>y2b))))
21      THEN
22         RETURN TRUE;
23      ELSE
24         RETURN FALSE;
25      END IF;
26  END $$
27  DELIMITER ;
```

6.2 Comments

The requested function is given in *region_overlaps_element.sql*.
The function was tested and the results were as expected. We tried the to use other approaches but this one was what we came up with. Basically the function returns true if *regionA* overlaps with *regionB*, but for this we used the conditions where they do not intersect and we negate the IF statement.