UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Database Management Systems, A.Y. 2017/2018
Master Degree in Computer Engineering
Master Degree in Telecommunication Engineering

Homework 4 – Physical Design

Deadline: May 25, 2017

LeoForFriendsDB	TreatIT	
Da Lio	Edoardo	1174591
Giovannini	Stefano	1178044
Kalakonda	Srikanth Reddy	1178232
Rossi	Gianmaria	1178526
Zhang	Guangzheng	1178043

Variations to the Relational Schema

No variations on the Relational Schema are needed.
Since the Relational Schema does not fit in a single page, it is reported in two parts in Figure 1 and 2.

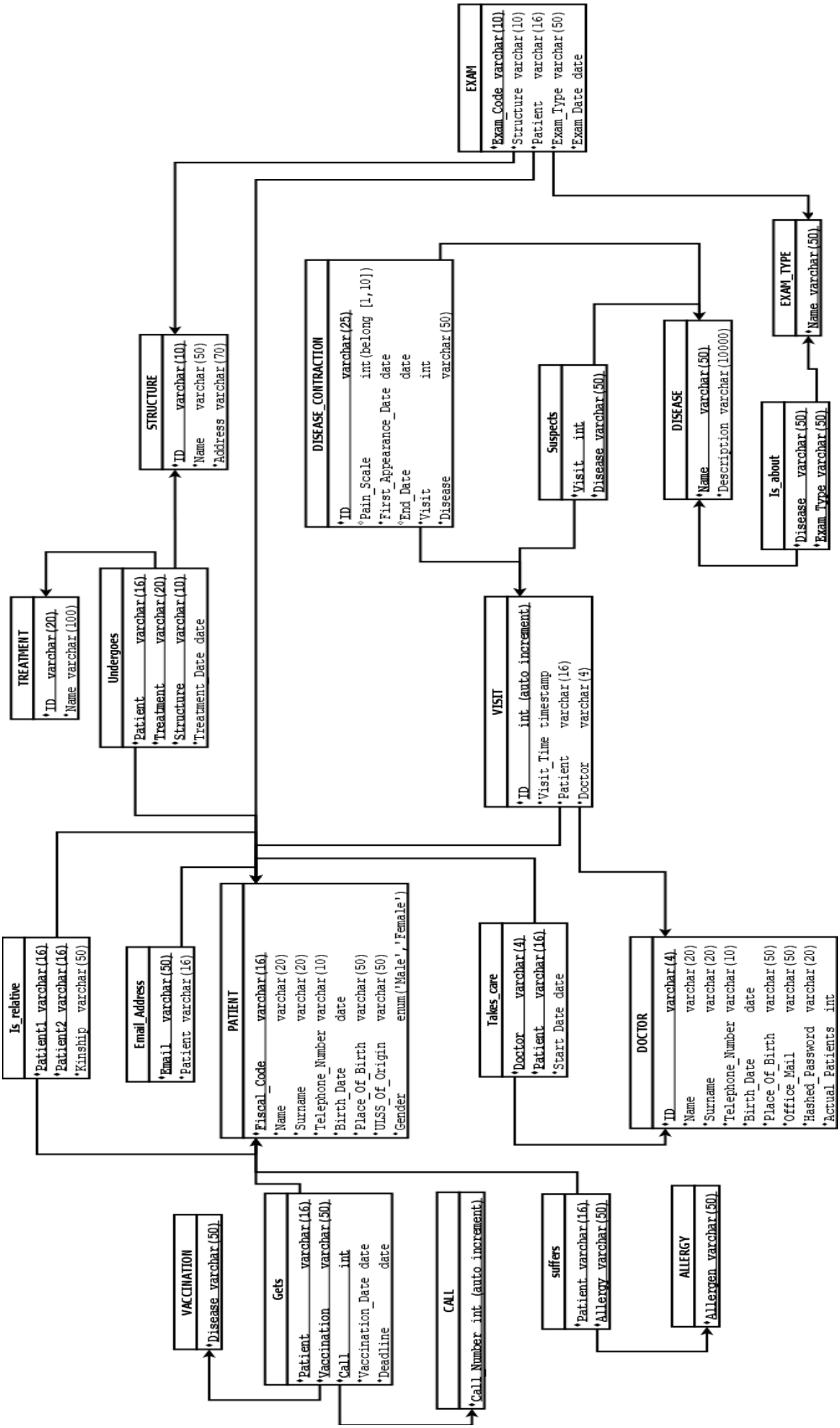

Figure 1: Relational Schema, Part 1

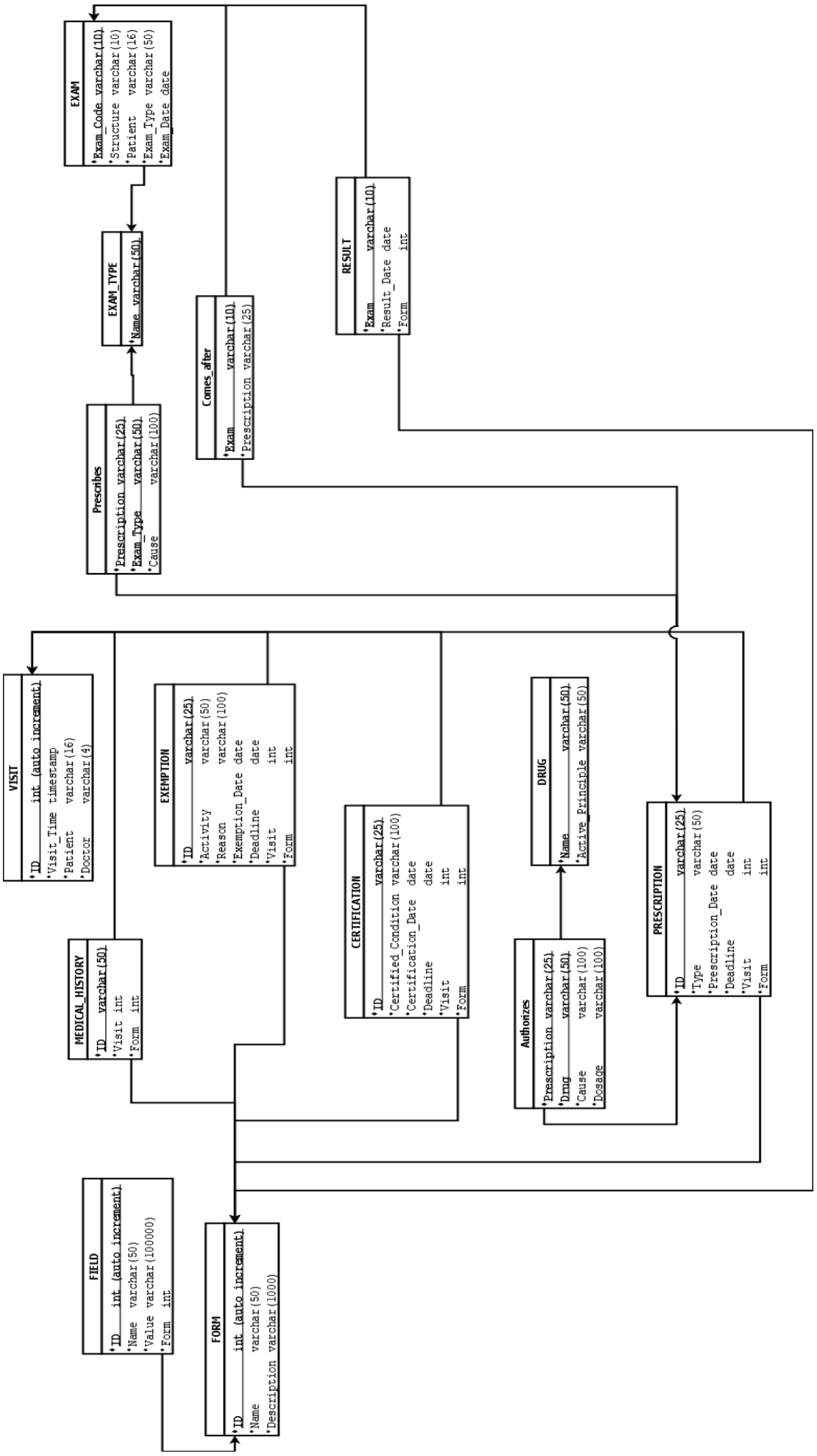

Figure 1: Relational Schema, Part 2

Physical Schema

Here follow the SQL instructions needed for the complete creation of the Database reported by the Relational Schema provided in the previous section.

```
-- Droppping the schema if already exist
drop schema if exists treatit cascade;

-- Creation of the schema
create schema treatit;
comment on schema treatit is 'Schema for containing the objects of the DBMS project
of treatit group';

-- Patient table
create type gender1 as enum ('Male','Female');
create table treatit.Patient
(
    fiscal_code varchar(16) primary key,
    name varchar(20) not null,
    surname varchar(20) not null,
    telephone_number varchar(10) not null,
    birth_date date not null,
    place_of_birth varchar(50) not null,
    ULSS_of_origin varchar(50) not null,
    gender gender1 not null
);

comment on table treatit.Patient is 'Represents a patient';
comment on column treatit.Patient.fiscal_code is 'Unique alphanumerical string
provided by the Country Administration';
comment on column treatit.Patient.name is 'The name of the patient';
comment on column treatit.Patient.surname is 'The surname of the patient';
comment on column treatit.Patient.telephone_number is 'The telephone number of the
patient';
comment on column treatit.Patient.birth_date is 'The date of birth of the patient';
comment on column treatit.Patient.place_of_birth is 'The place where the patient is
born';
comment on column treatit.Patient.ULSS_of_origin is 'The ULSS where the patient
comes from';
comment on column treatit.Patient.gender is 'The gender of the patient';

-- Doctor table
create table treatit.Doctor
(
    id varchar(4) primary key,
    name varchar(20) not null,
    surname varchar(20) not null,
    telephone_number varchar(10) not null,
    birth_date date not null,
```

```
    place_of_birth varchar(50) not null,
    office_mail varchar(50) not null,
    hashed_password varchar(20) not null,
    actual_patients int not null
);

comment on table treatit.Doctor is 'Represents a doctor';
comment on column treatit.Doctor.id is 'Unique identifier for the doctor';
comment on column treatit.Doctor.name is 'The name of the doctor';
comment on column treatit.Doctor.surname is 'The surname of the doctor';
comment on column treatit.Doctor.telephone_number is 'The telephone number of the
doctor';
comment on column treatit.Doctor.birth_date is 'The date of birth of the doctor';
comment on column treatit.Doctor.place_of_birth is 'The place of birth of the
doctor';
comment on column treatit.Doctor.office_mail is 'The professional email address of
the doctor';
comment on column treatit.Doctor.hashed_password is 'Doctor's personal password
saved in hashed form for security reasons';
comment on column treatit.Doctor.actual_patients is 'The number of patients
assigned to the doctor';

-- Patient Email table
create table treatit.Email_Address
(
    email varchar(50) primary key,
    patient varchar(16) not null,
    foreign key (patient) references treatit.Patient(fiscal_code) on delete cascade
on update cascade
);

comment on table treatit.Email_Address is 'Represents an email address of a
patient';
comment on column treatit.Email_Address.email is 'The email of the patient';
comment on column treatit.Email_Address.patient is 'Unique alphanumerical string
provided by the Country Administration';

-- Is relative table
create table treatit.Is_relative
(
    patient1 varchar(16),
    patient2 varchar(16),
    kinship varchar(50) not null,
    primary key (patient1,patient2),
    foreign key (patient1) references treatit.Patient(fiscal_code),
    foreign key (patient1) references treatit.Patient(fiscal_code)
);
```

```
comment on table treatit.Is_relative is 'Represents the kinship between two
patients';
comment on column treatit.Is_relative.patient1 is 'Unique alphanumerical string
provided by the Country Administration';
comment on column treatit.Is_relative.patient2 is 'Unique alphanumerical string
provided by the Country Administration';
comment on column treatit.Is_relative.kinship is 'The degree of relationship
connecting the two patients';

-- Takes care table
create table treatit.Takes_care
(
    doctor varchar(4),
    patient varchar(16),
    start_date date not null,
    primary key (doctor,patient),
    foreign key (doctor) references treatit.Doctor(id) on delete cascade on update
cascade,
    foreign key (patient) references treatit.Patient(fiscal_code) on delete cascade
on update cascade
);

comment on table treatit.Takes_care is 'Represents the patients who are taken care
by a doctor';
comment on column treatit.Takes_care.doctor is 'Unique identifier for the doctor';
comment on column treatit.Takes_care.patient is 'Unique alphanumerical string
provided by the Country Administration';
comment on column treatit.Takes_care.start_date is 'The day from which the doctor
has started taking care of the patient';

-- Treatment table
create table treatit.Treatment
(
    id varchar(20) primary key,
    name text not null
);

comment on table treatit.Treatment is 'Represents a treatment';
comment on column treatit.Treatment.id is 'Combination of the unique identifier
provided by the structure and the structure name';
comment on column treatit.Treatment.name is 'The name of the treatment';

-- Structure table
create table treatit.Structure
(
    id varchar(10) primary key,
    name varchar(50) not null,
    address varchar(70) not null
);
```

```
comment on table treatit.Structure is 'Represents a medical structure';
comment on column treatit.Structure.id is 'Unique identifier of the structure
provided by the National System';
comment on column treatit.Structure.name is 'The name of the structure';
comment on column treatit.Structure.address is 'The address of the structure';

-- Undergoes table
create table treatit.Undergoes
(
    patient varchar(16),
    treatment varchar(20),
    structure varchar(10),
    treatment_date date not null,
    primary key (patient,treatment,structure),
    foreign key (patient) references treatit.Patient(fiscal_code) on delete cascade
on update cascade,
    foreign key (treatment) references treatit.Treatment(id) on delete cascade on
update cascade,
    foreign key (structure) references treatit.Structure(id) on delete cascade on
update cascade
);

comment on table treatit.Undergoes is 'Represents a patient who undergoes a
treatment in a certain structure';
comment on column treatit.Undergoes.patient is 'Unique alphanumerical string
provided by the Country Administration';
comment on column treatit.Undergoes.treatment is 'Combination of the unique
identifier provided by the structure and the structure name';
comment on column treatit.Undergoes.structure is 'Unique identifier of the
structure provided by the National System';
comment on column treatit.Undergoes.treatment_date is 'The date on which the
patient undergoes the treatment';

-- Allergy table
create table treatit.Allergy
(
    allergen varchar(50) primary key
);

comment on table treatit.Allergy is 'Represents an allergy';
comment on column treatit.Allergy.allergen is 'The name of the substance the
patient is allergic to';

-- Suffers table
create table treatit.Suffers
(
    patient varchar(16),
    allergy varchar(50),
```

```
    primary key (patient,allergy),
    foreign key (patient) references treatit.Patient(fiscal_code) on update
cascade,
    foreign key (allergy) references treatit.Allergy(allergen) on update cascade
);

comment on table treatit.Suffers is 'Represents which allergies a patient suffers';
comment on column treatit.Suffers.patient is 'Unique alphanumerical string provided
by the Country Administration';
comment on column treatit.Suffers.allergy is 'The name of the substance the patient
is allergic to';

-- Call table
create table treatit.Call
(
    call_number serial primary key
);

comment on table treatit.Call is 'Represents the number of times a vaccination has
been repeated';
comment on column treatit.Call.call_number is 'The number of the call';

-- Vaccination table
create table treatit.Vaccination
(
    disease varchar(50) primary key
);

comment on table treatit.Vaccination is 'Represents a vaccine';
comment on column treatit.Vaccination.disease is 'The name of the disease the
vaccination is about';

-- Gets table
create table treatit.Gets
(
    patient varchar(16),
    vaccination varchar(50),
    call int,
    vaccination_date date not null,
    deadline date not null,
    primary key (patient,vaccination,call),
    foreign key (patient) references treatit.Patient(fiscal_code) on update
cascade,
    foreign key (vaccination) references treatit.Vaccination(disease) on update
cascade,
    foreign key (call) references treatit.Call(call_number)
);
```

```
comment on table treatit.Gets is 'Represents a patient who gets a vaccination in a
specific call';
comment on column treatit.Gets.patient is 'Unique alphanumerical string provided by
the Country Administration';
comment on column treatit.Gets.vaccination is 'The name of the disease the
vaccination is about';
comment on column treatit.Gets.call is 'The number of the call';
comment on column treatit.Gets.vaccination_date is 'The date on which the
vaccination was performed';
comment on column treatit.Gets.deadline is 'The date when the vaccination expires';

-- Visit table
create table treatit.Visit
(
    id serial primary key,
    visit_time timestamp not null,
    patient varchar(16) not null,
    doctor varchar(4) not null,
    foreign key (patient) references treatit.Patient(fiscal_code) on update
cascade,
    foreign key (doctor) references treatit.Doctor(id) on update cascade
);

comment on table treatit.Visit is 'Represents a visit that a doctor gives to a
patient';
comment on column treatit.Visit.id is 'An increasing number';
comment on column treatit.Visit.visit_time is 'The timestamp of the visit';
comment on column treatit.Visit.patient is 'Unique alphanumerical string provided
by the Country Administration';
comment on column treatit.Visit.doctor is 'Unique identifier for the doctor';

-- Disease table
create table treatit.Disease
(
    name varchar(50) primary key,
    description text not null
);

comment on table treatit.Disease is 'Represents a disease';
comment on column treatit.Disease.name is 'The name of the disease';
comment on column treatit.Disease.description is 'The description of the disease';

-- Disease contraction table
create table treatit.Disease_contraction
(
    id varchar(25) primary key,
    pain_scale int check(pain_scale > 0 and pain_scale < 11),
    first_appearance_date date not null,
    end_date date,
```

```
    visit int not null,
    disease varchar(50) not null,
    foreign key (visit) references treatit.Visit(id),
    foreign key (disease) references treatit.Disease(name) on update cascade
);

comment on table treatit.Disease_contraction is 'Represents the act of getting a
disease';
comment on column treatit.Disease_contraction.id is 'Combination of patient ID,
disease name and a counter';
comment on column treatit.Disease_contraction.pain_scale is 'Subjective number from
1 to 10 with which the patient describes how much he/she is suffering';
comment on column treatit.Disease_contraction.first_appearance_date is 'Date of the
day in which the first symptom of the disease occurred';
comment on column treatit.Disease_contraction.end_date is 'The date when the
disease ended';
comment on column treatit.Disease_contraction.visit is 'The identifier of the
visit';
comment on column treatit.Disease_contraction.disease is 'The name of the disease';

-- Suspects table
create table treatit.Suspects
(
    disease varchar(50),
    visit int,
    primary key (disease,visit),
    foreign key (disease) references treatit.Disease(name) on update cascade,
    foreign key (visit) references treatit.Visit(id)
);

comment on table treatit.Suspects is 'Represents which disease a doctor suspects
that a patient has';
comment on column treatit.Suspects.disease is 'The name of the disease';
comment on column treatit.Suspects.visit is 'The identifier of the visit';

-- Exam type table
create table treatit.Exam_type
(
    name varchar(50) primary key
);

comment on table treatit.Exam_type is 'Represents a type of exam';
comment on column treatit.Exam_type.name is 'The specific name of the exam
typology';

-- Is about table
create table treatit.Is_about
(
    disease varchar(50),
```

```
    exam_type varchar(50),
    primary key (disease,exam_type),
    foreign key (disease) references treatit.Disease(name) on update cascade,
    foreign key (exam_type) references treatit.Exam_type(name) on update cascade
);

comment on table treatit.Is_about is 'Represents for which disease is about the
type of the exam';
comment on column treatit.Is_about.disease is 'The name of the disease';
comment on column treatit.Is_about.exam_type is 'The specific name of the exam
typology';

-- Exam table
create table treatit.Exam
(
    exam_code varchar(10) primary key,
    structure varchar(10),
    patient varchar(16),
    exam_type varchar(50),
    exam_date date,
    foreign key (structure) references treatit.Structure(id) on update cascade,
    foreign key (patient) references treatit.Patient(fiscal_code) on update
cascade,
    foreign key (exam_type) references treatit.Exam_type(name) on update cascade
);

comment on table treatit.Exam is 'Represents the act of getting an exam';
comment on column treatit.Exam.exam_code is 'Unique identifier provided by the
National System';
comment on column treatit.Exam.structure is 'Unique identifier pf the structure
provided by the National System';
comment on column treatit.Exam.patient is 'Unique alphanumerical string provided by
the Country Administration';
comment on column treatit.Exam.exam_type is 'The specific name of the exam
typology';
comment on column treatit.Exam.exam_date is 'The date in which the exam was
performed';

-- Form table
create table treatit.Form
(
    id serial primary key,
    name varchar(50) not null,
    description text not null
);

comment on table treatit.Form is 'Represents a form used by the doctors';
comment on column treatit.Form.id is 'Progressive number identifying the form for a
specific exam result, medical history or document';
```

```
comment on column treatit.Form.name is 'The name of the form';
comment on column treatit.Form.description is 'A brief description of what the form
describes';

-- Field table
create table treatit.Field
(
    id serial primary key,
    name varchar(50) not null,
    field_value text not null,
    form int not null,
    foreign key (form) references treatit.Form(id) on update cascade
);

comment on table treatit.Field is 'Represents a field of a specific form';
comment on column treatit.Field.id is 'Progressive number identifying the field for
a specific form';
comment on column treatit.Field.name is 'The name of the field';
comment on column treatit.Field.field_value is 'The value of the field';
comment on column treatit.Field.form is 'The form identifier';

-- Medical History table
create table treatit.Medical_history
(
    id varchar(50) primary key,
    visit int not null,
    form int not null,
    foreign key (form) references treatit.Form(id) on update cascade,
    foreign key (visit) references treatit.Visit(id) on update cascade
);

comment on table treatit.Medical_history is 'Represents the medical history of a
patient';
comment on column treatit.Medical_history.id is 'Unique identifier computed by
combining the Fiscal Code of the patient and an incremental number, to take into
account of previous versions';
comment on column treatit.Medical_history.visit is 'The identifier of the visit';
comment on column treatit.Medical_history.form is 'The medical history form
identifier';

-- Exemption table
create table treatit.Exemption
(
    id varchar(25) primary key,
    activity varchar(50) not null,
    reason text not null,
    exemption_date date not null,
    deadline date not null,
    visit int not null,
```

```
    form int not null,
    foreign key (form) references treatit.Form(id) on update cascade,
    foreign key (visit) references treatit.Visit(id) on update cascade
);

comment on table treatit.Exemption is 'Represents an exemption gived by a doctor in
a visit';
comment on column treatit.Exemption.id is 'Unique identifier for the exemption,
derived from the visit one';
comment on column treatit.Exemption.activity is 'The activity targeted by the
exemption';
comment on column treatit.Exemption.reason is 'The motivation for which the
exemption is given';
comment on column treatit.Exemption.exemption_date is 'The date when the exemption
is released';
comment on column treatit.Exemption.deadline is 'The date when the exemption
expires';
comment on column treatit.Exemption.visit is 'The identifier of the visit';
comment on column treatit.Exemption.form is 'The exemption form identifier';

-- Certification table
create table treatit.Certification
(
    id varchar(25) primary key,
    certified_condition varchar(100) not null,
    certification_date date not null,
    deadline date not null,
    visit int not null,
    form int not null,
    foreign key (form) references treatit.Form(id) on update cascade,
    foreign key (visit) references treatit.Visit(id) on update cascade
);

comment on table treatit.Certification is 'Represents a certification gived by a
doctor in a visit';
comment on column treatit.Certification.id is 'Unique identifier for the
certificate, derived from the visit one';
comment on column treatit.Certification.certified_condition is 'The condition for
which the certificate is released';
comment on column treatit.Certification.certification_date is 'The date when the
certificate is released';
comment on column treatit.Certification.deadline is 'The date when the certificate
expires';
comment on column treatit.Certification.visit is 'The identifier of the visit';
comment on column treatit.Certification.form is 'The certification form
identifier';

-- Prescription table
create table treatit.Prescription
```

```
(
    id varchar(25) primary key,
    type varchar(50) not null,
    prescription_date date not null,
    deadline date not null,
    visit int not null,
    form int not null,
    foreign key (form) references treatit.Form(id) on update cascade,
    foreign key (visit) references treatit.Visit(id) on update cascade
);

comment on table treatit.Prescription is 'Represents a prescription gived by a
doctor in a visit';
comment on column treatit.Prescription.id is 'Unique identifier for the
prescription, derived from the visit one';
comment on column treatit.Prescription.type is 'The type of the medical
prescription';
comment on column treatit.Prescription.prescription_date is 'The date when the
prescription is released';
comment on column treatit.Prescription.deadline is 'The date when the prescription
expires';
comment on column treatit.Prescription.visit is 'The identifier of the visit';
comment on column treatit.Prescription.form is 'The prescription form identifier';

-- Drug table
create table treatit.Drug
(
    name varchar(50) primary key,
    active_principle varchar(50) not null
);

comment on table treatit.Drug is 'Represents a drug';
comment on column treatit.Drug.name is 'The commercial name of the drug';
comment on column treatit.Drug.active_principle is 'The active principle of the
drug';

-- Authorizes table
create table treatit.Authorizes
(
    prescription varchar(25),
    drug varchar(50),
    cause text not null,
    dosage text not null,
    primary key (prescription,drug),
    foreign key (prescription) references treatit.Prescription(id) on update
cascade,
    foreign key (drug) references treatit.Drug(name) on update cascade
);
```

```
comment on table treatit.Authorizes is 'Represents the authorization gived by a
doctor to a patient to assume a drug';
comment on column treatit.Authorizes.prescription is 'Identifier of the
prescription';
comment on column treatit.Authorizes.drug is 'The commercial name of the drug';
comment on column treatit.Authorizes.cause is 'The reason why the drug was
prescribed';
comment on column treatit.Authorizes.dosage is 'The dosage of the drug to be taken
by the patient';

-- Prescribes table
create table treatit.Prescribes
(
    prescription varchar(25),
    exam_type varchar(50),
    cause text not null,
    primary key (prescription,exam_type),
    foreign key (prescription) references treatit.Prescription(id) on update
cascade,
    foreign key (exam_type) references treatit.Exam_type(name) on update cascade
);

comment on table treatit.Prescribes is 'Represents the type of exam that a doctor
prescribes to a patient to do';
comment on column treatit.Prescribes.prescription is 'Identifier of the
prescription';
comment on column treatit.Prescribes.exam_type is 'The specific name of the exam
typology';
comment on column treatit.Prescribes.cause is 'The reason why the exam object of
the prescription was prescribed';

-- Comes after table
create table treatit.Comes_after
(
    exam varchar(10) primary key,
    prescription varchar(25) not null,
    foreign key (exam) references treatit.Exam(exam_code) on update cascade,
    foreign key (prescription) references treatit.Prescription(id) on update
cascade
);

comment on table treatit.Comes_after is 'Represents an exam that comes after a
doctor prescription';
comment on column treatit.Comes_after.exam is 'Unique identifier for the exam
provided by the National System';
comment on column treatit.Comes_after.prescription is 'The identifier of the
prescription';
```

```
-- Result table
create table treatit.Result
(
    exam varchar(10) primary key,
    result_date date not null,
    form int not null,
    foreign key (exam) references treatit.Exam(exam_code) on update cascade,
    foreign key (form) references treatit.Form(id) on update cascade
);

comment on table treatit.Result is 'Represents the result of an exam';
comment on column treatit.Result.exam is 'Unique identifier for the exam provided
by the National System';
comment on column treatit.Result.result_date is 'The date in which the result for
the exam is delivered to the doctor';
comment on column treatit.Result.form is 'The identifier of the form';
```

Triggers

Two triggers are needed for the proper functioning of the Database as a consequence of the modifications induced by the Load Analysis performed in the Logical Design.
The two triggers are activated in approximately the same context, that is when a new patient is inserted or deleted from the “takes _care” table in the Database.

The first one is fired on insertion of a new entry in the table and calls the stored procedure “AddPatient()”, which increments the “actual_patients” attribute of the Doctor table.
The code for the trigger and the function is the subsequent:

```
-- Create function for update the actual patients of a doctor
create function AddPatient() returns trigger as $$
declare
    begin
    -- If I have inserted a new row in Takes_care (i.e. the doctor of the
corresponding row
    -- has a new patient, so I need to increment his actual patients by one unit
    -- I called an insert, so the number of patients of the corresponding doctor
increased by one unit
        update treatit.Doctor
        set actual_patients=actual_patients+1
        where id=new.doctor;
        return new;
    end
$$ language plpgsql;
```

```
-- Adding a trigger to automatically increment the number of patients of a single
doctor

create trigger AddPatientTrigger
    after insert on treatit.Takes_care
    for each row
    execute procedure AddPatient();
```

The second one is fired on deletion of a new entry in the table and calls the stored procedure "DelPatient()", which decrements the "actual_patients" attribute of the Doctor table.
The code for the trigger and the function is the subsequent:

```
-- Create function for update the actual patients of a doctor
create function DelPatient() returns trigger as $$
declare
    begin
    -- If I have deleted a row in Takes_care (i.e. the doctor of the corresponding
row
    -- looses a patient, so I need to decrement his actual patients by one unit
    -- I called a deletion, so the number of patients of the corresponding doctor
decreased by one unit
        update treatit.Doctor
        set actual_patients=actual_patients-1
        where id=old.doctor;
        return old;
    end
$$ language plpgsql;


-- Adding a trigger to automatically decrement the number of patients of a single
doctor
create trigger DelPatientTrigger
    after delete on treatit.Takes_care
    for each row
    execute procedure DelPatient();
```

Populate the Database: Example

The final users of the systems are the Doctors which need to access and modify the data via an interface running on their personal computer. Doctors are interested in inserting new patients and all the information regarding them (diseases, vaccinations, medical history, exams, exam results), registering new visits and completing forms for prescriptions, exemptions and certificates.
Some examples of main instructions for inserting data in these tables are listed hereafter.

```
-- Populating Patient table
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('ASSMRA85T10A569S','AARIO','ROSSI','394839488','1985-11-10','SAN
GIULIANO TERME','ULSS 1','Male');
```

```
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('BSSMRA85T10A568S','BARIO','BRUNI','384839488','1985-12-
10','CAORLE','ULSS 2','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('CSSMRA85T10A597S','CARIO','VERDI','374839488','1985-10-
10','SOTTOMARINA','ULSS 3','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('DSSMRA85T10A566S','DARIO','ROSSI','364839488','1985-12-
10','BRINDISI','ULSS 3','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('ESSMRA85T10A565S','EARIO','BIANCHI','354839498','1985-12-10','SAN
GIULIANO TERME','ULSS 5','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('FSSMRA85T10A564S','FARIA','BIANCHI','344839488','1985-12-16','SAN
GIULIANO TERME','ULSS 3','Female');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('DSSMRA85T10A553S','DARIO','BRUNI','334839488','1985-08-10','SAN
GIULIANO TERME','ULSS 3','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('HSSMRA85T10A562S','HARIO','NERI','324839488','1985-12-01','SAN
GIULIANO TERME','ULSS 6','Male');
insert into treatit.Patient (fiscal_code, name, surname, telephone_number,
birth_date, place_of_birth, ULSS_of_origin, gender)
    values ('ISSMRA85T10A561S','ILARIA','ROSSI','314839488','1985-02-10','SAN
GIULIANO TERME','ULSS 3','Female');

-- Populating Email_Address table
insert into treatit.Email_Address (email, patient)
    values ('aariorossi@gmail.com','ASSMRA85T10A569S');
insert into treatit.Email_Address (email, patient)
    values ('aariorossi2@gmail.com','ASSMRA85T10A569S');
insert into treatit.Email_Address (email, patient)
    values ('aariorossi3@gmail.com','ASSMRA85T10A569S');
insert into treatit.Email_Address (email, patient)
    values ('fariabianchi@gmail.com','FSSMRA85T10A564S');
insert into treatit.Email_Address (email, patient)
    values ('fariabianchi2@gmail.com','FSSMRA85T10A564S');

-- Populating kinship Is_Relative table
insert into treatit.Is_Relative (patient1, patient2, kinship)
    values ('ASSMRA85T10A569S','HSSMRA85T10A562S','PARENT');
insert into treatit.Is_Relative (patient1, patient2, kinship)
```

```
    values ('ASSMRA85T10A569S','ISSMRA85T10A561S','SIBILING');

-- Populating Takes_care table
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0002','ASSMRA85T10A569S','2017-12-20');
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0001','BSSMRA85T10A568S','2016-12-20');
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0002','CSSMRA85T10A597S','2015-12-20');
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0003','DSSMRA85T10A566S','2014-12-20');
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0002','ESSMRA85T10A565S','2013-12-20');
insert into treatit.Takes_care (doctor, patient, start_date)
    values ('0003','FSSMRA85T10A564S','2017-12-20');

-- Populating Treatment table
insert into Treatit.Treatment (id, name)
    values ('STRUCT039ABLARR','ABLATION OF ARRHYTHMIA');
insert into Treatit.Treatment (id, name)
    values ('STRUCT039ACN','ACNE');
insert into Treatit.Treatment (id, name)
    values ('STRUCT045ARMACDEG','AGE-RELATED MACULAR DEGENERATION TREATMENT');
insert into Treatit.Treatment (id, name)
    values ('STRUCT039ALTTREAD','ALTERG TREADMILL');
insert into Treatit.Treatment (id, name)
    values ('STRUCT031ANKARTH','ANKLE ARTHROSCOPY');

-- Populating Undergoes table
insert into treatit.Undergoes (patient, treatment, structure, treatment_date)
    values ('ASSMRA85T10A569S','STRUCT039ABLARR','STRUCT039','2017-10-10');
insert into treatit.Undergoes (patient, treatment, structure, treatment_date)
    values ('ASSMRA85T10A569S','STRUCT039ALTTREAD','STRUCT039','2017-11-10');
insert into treatit.Undergoes (patient, treatment, structure, treatment_date)
    values ('ASSMRA85T10A569S','STRUCT031ANKARTH','STRUCT031','2017-10-09');

-- Populating Suffers table
insert into treatit.Suffers (patient, allergy) values ('ASSMRA85T10A569S','AA');
insert into treatit.Suffers (patient, allergy) values ('ASSMRA85T10A569S','ABE');
insert into treatit.Suffers (patient, allergy) values ('ASSMRA85T10A569S','ABG');
insert into treatit.Suffers (patient, allergy) values ('CSSMRA85T10A597S','AA');

-- Populating Call table
insert into treatit.Call (call_number) values (default);
insert into treatit.Call (call_number) values (default);
insert into treatit.Call (call_number) values (default);
insert into treatit.Call (call_number) values (default);
insert into treatit.Call (call_number) values (default);
insert into treatit.Call (call_number) values (default);
```

```
insert into treatit.Call (call_number) values (default);

-- Populating Gets table
insert into treatit.Gets (patient, vaccination, call, vaccination_date, deadline)
    values ('ASSMRA85T10A569S','ADENOVIRUS',2,'2017-11-10','2020-11-10');
insert into treatit.Gets (patient, vaccination, call, vaccination_date, deadline)
    values ('ASSMRA85T10A569S','ANTHRAX',2,'2013-11-10','2016-11-10');
insert into treatit.Gets (patient, vaccination, call, vaccination_date, deadline)
    values ('ASSMRA85T10A569S','CHOLERA',2,'2013-11-10','2016-11-10');
insert into treatit.Gets (patient, vaccination, call, vaccination_date, deadline)
    values ('ASSMRA85T10A569S','DIPHTHERIA',2,'2013-11-10','2017-11-10');
insert into treatit.Gets (patient, vaccination, call, vaccination_date, deadline)
    values ('ASSMRA85T10A569S','ADENOVIRUS',3,'2020-11-10','2022-11-10');

-- Populating Visit table
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0001');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'BSSMRA85T10A568S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0003');
insert into treatit.Visit (visit_time, patient, doctor)
    values (current_timestamp,'ASSMRA85T10A569S','0002');

--Populating Disease_contraction table
insert into treatit.Disease_contraction (id, pain_scale, first_appearance_date,
end_date, visit, disease)

    values ('ASSMRA85T10A569SARTHR001',6,'2016-05-20','2017-03-11',2,'ARTHRITIS');
insert into treatit.Disease_contraction (id, pain_scale, first_appearance_date,
end_date, visit, disease)

    values ('ASSMRA85T10A569SCANCE001',10,'2018-04-20',NULL,5,'CANCER');
insert into treatit.Disease_contraction (id, pain_scale, first_appearance_date,
end_date, visit, disease)

    values ('ASSMRA85T10A569SARTHR002',7,'2018-01-20',NULL,8,'ARTHRITIS');
```

```
insert into treatit.Disease_contraction (id, pain_scale, first_appearance_date,
end_date, visit, disease)

    values ('ASSMRA85T10A569SLUNDI001',4,'2018-05-20',NULL,8,'LUNG DISEASE');

-- Populating Suspects table
insert into treatit.Suspects (disease, visit) values ('CHRONIC PAIN',7);
insert into treatit.Suspects (disease, visit) values ('LUNG DISEASE',6);
insert into treatit.Suspects (disease, visit) values ('HIV',3);
insert into treatit.Suspects (disease, visit) values ('CANCER',2);

-- Populating Exam table
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('TAC201','STRUCT039','ASSMRA85T10A569S','TAC','2018-01-20');
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('BIOPSY601','STRUCT031','ASSMRA85T10A569S','BIOPSY','2018-01-20');
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('TAC202','STRUCT039','ASSMRA85T10A569S','TAC','2018-01-10');
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('HIVTEST222','STRUCT039','ASSMRA85T10A569S','HIV TEST','2017-01-20');
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('TAC111','STRUCT039','ASSMRA85T10A569S','TAC','2016-01-20');
insert into treatit.Exam (exam_code, structure, patient, exam_type, exam_date)
    values ('TAC101','STRUCT045','ASSMRA85T10A569S','TAC','2015-01-20');

-- Populating Medical_history table
insert into treatit.Medical_history (id, visit, form) values
('ASSMRA85T10A569S01',2,3);
insert into treatit.Medical_history (id, visit, form) values
('ASSMRA85T10A569S02',2,5);

-- Populating Exemption table
insert into treatit.Exemption (id, activity, reason, exemption_date, deadline,
visit, form)
    values ('26','TICKET PAYEMENT','THE PATIENT IS FULL OF ILLNESSES','2015-10-
10','2019-10-10',2,6);

-- Populating Certification table
insert into treatit.Certification (id, certified_condition, certification_date,
deadline, visit, form)
    values ('24','BLINDNESS','2016-10-09','2017-10-10',2,4);
insert into treatit.Certification (id, certified_condition, certification_date,
deadline, visit, form)
    values ('37','CRONIC BRONCOPNEUNOPHATY','2018-10-09','2039-10-10',3,7);

-- Populating Prescription table
insert into  treatit.Prescription (id, type, prescription_date, deadline, visit,
form)
```

```
    values ('28','PRESCRIPTION FOR A DRUG ABOUT STOMACHACHE','2018-05-23','2018-06-
01',2,8);
insert into  treatit.Prescription (id, type, prescription_date, deadline, visit,
form)
    values ('39','PRESCRIPTION FOR DOING A BIOPSY','2018-01-01','2018-07-01',3,9);

-- Populating Form table
insert into treatit.Form (name, description)
    values ('RESULTS FROM TAC','THE RESULTS FROM THE TAC EXAM PERFORMED BY DR. UGO
BASSI');
insert into treatit.Form (name, description)
    values ('RESULTS FROM HIV TEST','THE RESULTS FROM THE HIV TEST OF AARIO
ROSSI');
insert into treatit.Form (name, description)
    values ('LIST OF EREDITATED DISEASES','THE LIST OF THE DISEASES EREDITATED BY
AARIO ROSSI');
insert into treatit.Form (name, description)
    values ('BLINDNESS CERTIFICATE','THE CERTIFICATION OF BLINDNESS OF AARIO
ROSSI');
insert into treatit.Form (name, description)
    values ('LIST OF TRAUMA','THE LIST OF THE TRAUMAS TAKEN BY AARIO ROSSI');
insert into treatit.Form (name, description)
    values ('ADDITIONAL INFORMATION','ADDITIONAL INFORMATIONS REGARDING THE LIST OF
THE EXEMPTIONS OF BY AARIO ROSSI');
insert into treatit.Form (name, description)
    values ('CERTIFICATE OF CRONIC BRONCOPNEUNOPHATY','THE CERTIFICATION OF A
CRONIC LUNG DISEASE OF AARIO ROSSI');
insert into treatit.Form (name, description)
    values ('PRESCRIPTION OF STOMACH DRUG','A PRESCRIPTION FOR A SPECIFIC DRUG THAT
DEALS WITH REALLY PAINFUL STOMACHACHE');
insert into treatit.Form (name, description)
    values ('PRESCRIPTION OF A BIOPSY','A PRESCRIPTION FOR A SPECIFIC EXAM ABOUT
CANCER');

-- Populating Field table
insert into treatit.Field (name, field_value, form) values ('responseA','true',1);
insert into treatit.Field (name, field_value, form) values ('responseB','false',1);
insert into treatit.Field (name, field_value, form) values ('result:
','negative',2);
insert into treatit.Field (name, field_value, form) values ('first disease','skin
disease',3);
insert into treatit.Field (name, field_value, form) values ('second
disease','HIV',3);
insert into treatit.Field (name, field_value, form) values ('from mother
branch','chronic disease, eyes disease',3);
insert into treatit.Field (name, field_value, form) values ('percentage of
blindness','89%',4);
insert into treatit.Field (name, field_value, form) values ('traumas in
childhood','domestic violence, arm broken',5);
```

```
insert into treatit.Field (name, field_value, form) values ('traumas in youth','leg
broken, arm broken',5);
insert into treatit.Field (name, field_value, form) values ('recent traumas','heart
attack, nose broken',5);
insert into treatit.Field (name, field_value, form) values ('law that is exploited
in the exemption: ','ART495 COMMA B',6);
insert into treatit.Field (name, field_value, form) values ('affected
mobility','30%',7);
insert into treatit.Field (name, field_value, form) values ('certification tests
was performed by: ','ULSS14',7);


-- Populating Drug table
insert into treatit.Drug (name, active_principle)
    values ('AZTREONAM','Bezellanius Acheolaptus, Lombagenanthes');

-- Populating Authorizes table
insert into treatit.Authorizes (prescription, drug, cause, dosage)
    values ('28','AZTREONAM','PRESCRIBED BECAUSE THE PATIENT HAS A REALLY DANGEROUS
STOMACHACHE','100mL/DAY UNTIL HE FEELS BETTER');

-- Populating Prescribes table
insert into treatit.Prescribes (prescription, exam_type, cause)
    values ('39','BIOPSY','THE PATIENT STARTED TO FEEL DIZZY AND STUFF, SO HE BADLY
NEEDS TO DO THIS EXAM.');

-- Populating Comes_after table
insert into treatit.Comes_after (exam, prescription) values ('BIOPSY601','39');

-- Populating Result table
insert into treatit.Result (exam, result_date, form) values ('TAC111','2018-02-
20',1);
insert into treatit.Result (exam, result_date, form) values ('HIVTEST222','2017-02-
20',2);
```

All the other information regarding the Relational Schema reported in the first section shall be already present in the Database before making it available to the final users.
The SQL code for the population of these tables is reported below:

```
-- Populating Doctor table
insert into treatit.Doctor (id, name, surname, telephone_number, birth_date,
place_of_birth, office_mail, hashed_password, actual_patients)
    values ('0001','Giuseppe','Casari','394859424','1982-11-
28','NOALE','g.casari@gmail.com','-946852072',0);
insert into treatit.Doctor (id, name, surname, telephone_number, birth_date,
place_of_birth, office_mail, hashed_password, actual_patients)
    values ('0002','Marco','Zambon','374859424','1972-10-
28','TREVISO','m.zambon@gmail.com','-946852072',0);
```

```
insert into treatit.Doctor (id, name, surname, telephone_number, birth_date,
place_of_birth, office_mail, hashed_password, actual_patients)
    values ('0003','Alessandro','Gottardo','384859424','1962-10-
8','NOALE','a.gattaro@gmail.com','-946852072',0);

-- Populating Structure table
insert into treatit.Structure (id, name, address)
    values ('STRUCT039','Centro Molliego','Via Aldo Moro 17 56122 PISA PI');
insert into treatit.Structure (id, name, address)
    values ('STRUCT045','Centro Colle','Via Roma 17 56122 ROVIGO RO');
insert into treatit.Structure (id, name, address)
    values ('STRUCT031','Centro Urso','Via Genova 13 56522 TREVISO TV');
insert into treatit.Structure (id, name, address)
    values ('STRUCT030','Centro Teresin','Via Trieste 17 56622 VENEZIA VE');
insert into treatit.Structure (id, name, address)
    values ('STRUCT088','Centro Molesan','Via Pescara 17 56172 NAPOLI NA');
insert into treatit.Structure (id, name, address)
    values ('STRUCT078','Centro Meggin','Via Provenza 14 56182 TORINO TO');

-- Populating Allergy table
insert into treatit.Allergy (allergen) values ('AA');
insert into treatit.Allergy (allergen) values ('ABD');
insert into treatit.Allergy (allergen) values ('ABE');
insert into treatit.Allergy (allergen) values ('ABF');
insert into treatit.Allergy (allergen) values ('ABG');

-- Populating Vaccination table
insert into treatit.Vaccination (disease) values ('ADENOVIRUS');
insert into treatit.Vaccination (disease) values ('ANTHRAX');
insert into treatit.Vaccination (disease) values ('CHOLERA');
insert into treatit.Vaccination (disease) values ('DIPHTHERIA');

-- Populating Disease table
insert into treatit.Disease (name, description)
    values ('ARTHRITIS','DISEASE INVOLVING BONES');
insert into treatit.Disease (name, description)
    values ('CANCER','DISEASE INVOLVING ROGUE CELLULAR MULTIPLICATION');
insert into treatit.Disease (name, description)
    values ('CHOLESTEROL','DISEASE INVOLVING SUGAR IN BLOOD');
insert into treatit.Disease (name, description)
    values ('CHRONIC PAIN','DISEASE INVOLVING STRESS, PREVIOUS TRAUMAS AND OTHER
FACTORS');
insert into treatit.Disease (name, description)
    values ('LUNG DISEASE','DISEASE INVOLVING PAIN IN THE LUNG');
insert into treatit.Disease (name, description)
    values ('HIV','DISEASE INVOLVING THE HIV VIRUS');

--Populating Exam_type table
insert into treatit.Exam_type (name) values ('BIOPSY');
```

```
insert into treatit.Exam_type (name) values ('TAC');
insert into treatit.Exam_type (name) values ('HIV TEST');
insert into treatit.Exam_type (name) values ('BLOOD EXAM');
insert into treatit.Exam_type (name) values ('CARDIOVASCULAR EXAMINATION');

-- Populating Is_about table
insert into treatit.Is_about (disease, exam_type) values ('HIV','HIV TEST');
insert into treatit.Is_about (disease, exam_type) values ('CHRONIC PAIN','TAC');
insert into treatit.Is_about (disease, exam_type) values ('ARTHRITIS','TAC');
insert into treatit.Is_about (disease, exam_type) values ('CANCER','BIOPSY');
```

Principal Queries

The principal queries for which the system has to be exploited are reported in this paragraph, alongside with a brief comment on what they perform and an image showing the result.

Retrieving data about diseases affecting a certain patient:

```
-- Extract disease name, first appearance date, (eventually) end date and
description of the desease for a certain patient (the research is done by fiscal
code)
-- the disease manifestations are sorted in descending order (the more recent ones
first)
select disease, first_appearance_date, end_date, description
from treatit.Disease as D inner join treatit.Disease_contraction as DC on
D.name=DC.disease
        inner join treatit.Visit as V on DC.visit=V.id
where V.patient='ASSMRA85T10A569S'
order by first_appearance_date desc;
```

	disease character varying (50)	first_appearance_date date	end_date date	description text
1	LUNG DISEASE	2018-05-20	[null]	DISEASE INVOLVING PAIN IN THE LUNG
2	CANCER	2018-04-20	[null]	DISEASE INVOLVING ROGUE CELLULAR MULTIPLICATION
3	ARTHRITIS	2018-01-20	[null]	DISEASE INVOLVING BONES
4	ARTHRITIS	2016-05-20	2017-03-11	DISEASE INVOLVING BONES

Retrieving data about exams of a certain patient:

```
-- Extract all the exams of a certain patient (i.e. code, type, date, and
eventually the result date) starting from the most recent ones
select exam_code, exam_type, exam_date, result_date
from treatit.Exam as E inner join treatit.Result as R on E.exam_code=R.exam
where E.patient='ASSMRA85T10A569S'
order by exam_date desc;
```

	exam_code character varying (10)	exam_type character varying (50)	exam_date date	result_date date
1	HIVTEST222	HIV TEST	2017-01-20	2017-02-20
2	TAC111	TAC	2016-01-20	2018-02-20

Retrieving data about fields of a form:

```
-- Extract all the fields (each of them composed by name and value) associated to a
form which corresponds to the result of a desired exam (search by exam code)
select FI.name, FI.field_value
from treatit.Result as R inner join treatit.Form as FO on R.form=FO.id
    inner join treatit.Field as FI on FO.id=FI.form
where R.exam='TAC111';
```

	name character varying (50)	field_value text
1	responseA	true
2	responseB	false

Retrieving data about medical histories:

```
--extract all the fields of medical history of a certain patient
--in each visit, the doctor can submit multiple form of medical history, each of
them are composed by one or more fields
--with this query the doctor retrieves all the forms about medical history which
concern a specific patient
select V.visit_time, FO.name as form_name, FI.name as field_name, FI.field_value
from treatit.Visit as V inner join treatit.Medical_History as M on M.visit=V.id
    inner join treatit.Form as FO on FO.id=M.form
        inner join treatit.Field as FI on FO.id=FI.form
where V.patient='ASSMRA85T10A569S'
order by V.visit_time desc;
```

	visit_time timestamp without time zone	form_name character varying (50)	field_name character varying (50)	field_value text
1	2018-05-24 20:45:02.975295	LIST OF EREDITATED DISEASES	first disease	skin disease
2	2018-05-24 20:45:02.975295	LIST OF EREDITATED DISEASES	second disease	HIV
3	2018-05-24 20:45:02.975295	LIST OF EREDITATED DISEASES	from mother branch	chronic disease, eyes disease
4	2018-05-24 20:45:02.975295	LIST OF TRAUMA	traumas in childhood	domestic violence, arm broken
5	2018-05-24 20:45:02.975295	LIST OF TRAUMA	traumas in youth	leg broken, arm broken
6	2018-05-24 20:45:02.975295	LIST OF TRAUMA	recent traumas	heart attack, nose broken

Retrieving data about certifications of a certain patient:

```
--extract all the current certifications of a certain patient
--(i.e. the id, the certified condition, the doctor who certified them, the
certification date and deadline)
select C.id, C.certified_condition, D.name ||' '|| D.surname as doctor,
C.certification_date, C.deadline
```

```
from treatit.Certification as C inner join treatit.Visit as V on C.Visit=V.id
    inner join treatit.doctor as D on V.doctor=D.id
where V.patient='ASSMRA85T10A569S' and C.deadline < current_date
order by C.certification_date;
```

	id character varying (25)	certified_condition character varying (100)	doctor text	certification_date date	deadline date
1	24	BLINDNESS	Marco Zambon	2016-10-09	2017-10-10

Retrieving data about vaccinations of certain patient:

```
--extract all the vaccinations (i.e. name of the corresponding disease, number of
call, date in which it was/will be performed, deadline
--it is adopted the descending order in the deadline date
select vaccination, call, vaccination_date, deadline
from treatit.Gets
where patient='ASSMRA85T10A569S'
order by deadline desc;
```

	vaccination character varying (50)	**call** integer	**vaccination_date** date	**deadline** date
1	ADENOVIRUS	3	2020-11-10	2022-11-10
2	ADENOVIRUS	2	2017-11-10	2020-11-10
3	DIPHTHERIA	2	2013-11-10	2017-11-10
4	ANTHRAX	2	2013-11-10	2016-11-10
5	CHOLERA	2	2013-11-10	2016-11-10

Retrieving data about emails of a certain patient:

```
--extract all the emails of a certain patient (research is done by using patient's
fiscal code)
select email
from treatit.Email_Address
where patient='ASSMRA85T10A569S';
```

	email character varying (50)
1	aariorossi@gmail.com
2	aariorossi2@gmail.com
3	aariorossi3@gmail.com

Retrieving data about visits involving a certain patient:

```
--extract all the visit id and for a certain patient
--decreasing order following visit id put in evidence the recent visits
--this is a useful query that allows the insertion of events performed during a
certain visit
select id, visit_time
from treatit.Visit
where patient='ASSMRA85T10A569S'
order by visit_time desc;
```

	id integer	visit_time timestamp without time zone
1	11	2018-05-24 20:45:56.392788
2	10	2018-05-24 20:45:49.976605
3	9	2018-05-24 20:45:43.692316
4	8	2018-05-24 20:45:38.576594
5	7	2018-05-24 20:45:33.025389
6	5	2018-05-24 20:45:22.742144
7	4	2018-05-24 20:45:16.593731
8	3	2018-05-24 20:45:10.242203
9	2	2018-05-24 20:45:02.975295
10	1	2018-05-24 20:44:52.004282

JDBC Implementations of the Principal Queries and Visualization

A Java class executing all the queries listed in the previous section performing some queries and insertions by using some auxiliary classes to wrap the contents extracted from the DB. Such auxiliary classes, provided together with this document, are the ones effectively managing the connections to the DB.

```
package it.unipd.dei.bding.example;

import it.unipd.dei.bding.database.*;
import it.unipd.dei.bding.resource.*;
import org.apache.tomcat.jdbc.pool.DataSource;
import org.apache.tomcat.jdbc.pool.PoolProperties;

import java.sql.Date;
import java.util.List;

/**
 * Testing the treatit database.
 *
 * @author leoforfriendsDB
 * @version 1.00
```

```
 */
public class UseTreatit
{

    public static void main(String[] args)
    {
        // create a pool of connections
        PoolProperties p = new PoolProperties();

        // setup connection properties
        p.setUrl("jdbc:postgresql://localhost/dbms1718");
        p.setDriverClassName("org.postgresql.Driver");
        p.setUsername("leoforfriendsdb");
        p.setPassword("dbmshome");
        /*p.setUrl("jdbc:postgresql://dbstud.dei.unipd.it:5434/dbms1718");
        p.setDriverClassName("org.postgresql.Driver");
        p.setUsername("webdb");
        p.setPassword("webdb");*/

        // setup the size of the pool
        p.setInitialSize(1);
        p.setMaxActive(2);
        p.setMaxIdle(2);
        p.setMinIdle(1);

        // setup how and when to test that a connection is still working
        p.setValidationQuery("SELECT 1");
        p.setTestOnBorrow(true);
        p.setTestOnReturn(false);
        p.setTestWhileIdle(false);

        // create a datasource based on that pool
        DataSource datasource = new DataSource();
        datasource.setPoolProperties(p);

        // using classes for querying the database
        try
        {

            // Search all the diseases that a patient contracted by his/her fiscal
            code
            try
            {
                // searches all the diseases that a patient contracted by his/her
                  fiscal code
                SearchDiseaseByFiscalCodeDatabase di = new
                     SearchDiseaseByFiscalCodeDatabase(datasource.getConnection(),
                  "ASSMRA85T10A569S");
                di.searchExamByFiscalCode();
```

```
}
catch (Exception e)
{
    System.out.printf("Unable to search the diseases of this patient:
      %s%n", e.getMessage());
}


// Search all the exams that a patient got by his/her fiscal code
try
{
    // searches all the exams that a patient got by his/her fiscal code
    SearchExamByFiscalCodeDatabase ex = new
             SearchExamByFiscalCodeDatabase(datasource.getConnection(),
      "ASSMRA85T10A569S");
    ex.searchExamByFiscalCode();
}
catch (Exception e)
{
    System.out.printf("Unable to search the exams of this patient:
      %s%n", e.getMessage());
}


// Search all the fields associated to a form which corresponds to the
      result of a desired exam by its exam code
try
{
    // searches all the fields associated to a form which corresponds
      to the result of a desired exam by its exam code
    SearchFieldByExamTypeDatabase f = new
             SearchFieldByExamTypeDatabase(datasource.getConnection(),
      "TAC111");
    f.searchFieldByExamType();
}
catch (Exception e)
{
    System.out.printf("Unable to search the fields of this Exam: %s%n",
      e.getMessage());
}


// Search all the medical histories that a patient got by his/her
      fiscal code
try
{
    // searches all the medical histories that a patient got by his/her
      fiscal code
    SearchMedicalHistoryByFiscalCodeDatabase m = new
```

```
      SearchMedicalHistoryByFiscalCodeDatabase(datasource.getConnection
(), "ASSMRA85T10A569S");
    m.searchMedicalHistoryByFiscalCode();
}
catch (Exception e)
{
    System.out.printf("Unable to search the medical histories of this
      patient: %s%n", e.getMessage());
}


// Search all the vaccinations that a patient got by his/her fiscal
      code
try
{
    // searches all the vaccinations that a patient got by his/her
      fiscal code
    SearchCertificationByFiscalCodeDatabase c = new
      SearchCertificationByFiscalCodeDatabase(datasource.getConnection(
), "ASSMRA85T10A569S");
    c.searchCertificationByFiscalCode();
}
catch (Exception e)
{
    System.out.printf("Unable to search the certifications of this
      patient: %s%n", e.getMessage());
}


// Search all the vaccinations that a patient got by his/her fiscal
      code
try
{
    // searches all the vaccinations that a patient got by his/her
      fiscal code
    List<Gets> vaccinations = new
      SearchVaccinationByFiscalCodeDatabase(datasource.getConnection(),
"ASSMRA85T10A569S").searchVaccinationByFiscalCode();

    // printing the results
    System.out.printf("%nVaccinations successfully searched. Found
      Vaccinations:%n");
    System.out.printf("- %-15s %-5s %-20s %s%n", "Vaccination", "Call",
      "Vaccination date", "Deadline");
    for (Gets e : vaccinations)
    {
        System.out.printf("- %-15s %-5s %-20s %s%n",
      e.getVaccination(), e.getCall(), e.getVaccination_date(),
e.getDeadline());
```

```
    }
}
catch (Exception e)
{
    System.out.printf("Unable to search the vaccinations of this
      patient: %s%n", e.getMessage());
}


// Search email addresses of a patient by his/her fiscal code
try
{
    // searches email addresses of a patient by his/her fiscal code
    List<EmailAddress> emailAddresses = new
      SearchEmailAddressByFiscalCodeDatabase(datasource.getConnection()
, "ASSMRA85T10A569S").searchEmailAddressByFiscalCode();

    // printing the results
    System.out.printf("%nEmail addresses successfully searched. Found
      email addresses:%n");
    System.out.printf("- %s%n", "Email Address");
    for (EmailAddress e : emailAddresses)
    {
        System.out.printf("- %s%n", e.getEmail());
    }
}
catch (Exception e)
{
    System.out.printf("Unable to search the email addresses of this
      patient: %s%n", e.getMessage());
}


// Search visit ids of a patient by his/her fiscal code
try
{
    // searches visit ids of a patient by his/her fiscal code
    List<Visit> visits = new
      SearchVisitByFiscalCodeDatabase(datasource.getConnection(),
"ASSMRA85T10A569S").searchVisitByFiscalCode();

    // printing the results
    System.out.printf("%nVisit ids successfully searched. Found visit
      ids:%n");
    System.out.printf("- %-10s %s%n", "Visit id", "Visit time");
    for (Visit v : visits)
    {
        System.out.printf("- %-10s %s%n", v.getId(),
      v.getVisit_time());
```

```
    }
}
catch (Exception e)
{
    System.out.printf("Unable to search the visit ids of this patient:
      %s%n", e.getMessage());
}


// insertion of data in the database

// Adds a new doctor
try
{
    Date birthDate = new Date(70, 3, 16);
    Doctor doctor = new Doctor("1024", "Augusto",
        "Rossi", "0445368596",
        birthDate, "Bassano del Grappa",
        "augusto.rossi@gmail.com", "qwerty",
        0);

    // creates a new object for accessing the database and stores the
      doctor
    new CreateDoctorDatabase(datasource.getConnection(),
      doctor).createDoctor();

    // if the insert is successfully done then prints the new added row
      values
    System.out.printf("%nNew doctor successfully created.%n%s%n",
      doctor.toString());
}
catch (Exception e)
{
    System.out.printf("Unable to add the doctor: %s%n",
      e.getMessage());
}


// Adds a new patient
try
{
    Date birthDate = new Date(93, 5, 25);
    Patient patient = new Patient("ANHMR93T10B5679I", "Anita",
        "Moratti", "0444368336", birthDate,
        "Vicenza", "ULSS 4",
        "Female");

    // creates a new object for accessing the database and stores the
      patient
```

```
                new CreatePatientDatabase(datasource.getConnection(),
                  patient).createPatient();

                // if the insert is successfully done then prints the new added row
                  values
                System.out.printf("%nNew patient successfully created.%n%s%n",
                  patient.toString());
            }
            catch (Exception e)
            {
                System.out.printf("Unable to add the patient: %s%n",
                  e.getMessage());
            }


            // A doctor takes care a patient
            try
            {

                Date startDate = new Date(117, 5, 25);
                TakesCare takesCare = new TakesCare("1024", "ANHMR93T10B5679I",

                  startDate);

                // A doctor takes care the patient
                                    new
             CreateTakesCareDatabase(datasource.getConnection(),
             takesCare).createTakesCare();

                // if the insert is successfully done then prints the new added row
                  values
                System.out.printf("%nA doctor successfully takes care the
                  patient.%n%s%n", takesCare.toString());
            }
            catch (Exception e)
            {
                System.out.printf("A doctor is unable to take care the patient:
                  %s%n", e.getMessage());
            }
        }
        finally
        {
            //close the datasource
            datasource.close();
        }

    }
}
```