**FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO**

Mestrado Integrado em Engenharia Informática e Computação – 2º ano

**Definição do Modelo Conceptual**

**Hospital HPL**

**Turma 7, Grupo 705:**

Henrique Costa Sousa - 201906681

Leandro Martins de Oliveira - 200903027

Pedro Jorge Ribeiro Botelho de Moniz Pacheco - 201806824

**Unidade Curricular:** Base de Dados

**Docentes:**

Carla Alexandra Teixeira Lopes

Gonçalo Filipe Loureiro Campos Gonçalves

**Porto**

**2020/2021**

Índice

* Descrição
* Diagrama UML
* Esquema relacional
* Análise de Dependências Funcionais e Formas Normais

**Descrição**

Pretende-se armazenar dados relativos a um hospital.

A base de dados deve ser capaz de lidar com diferentes tipos de pessoas, sendo as mesmas inseridas em duas categorias: trabalhadores e clientes.

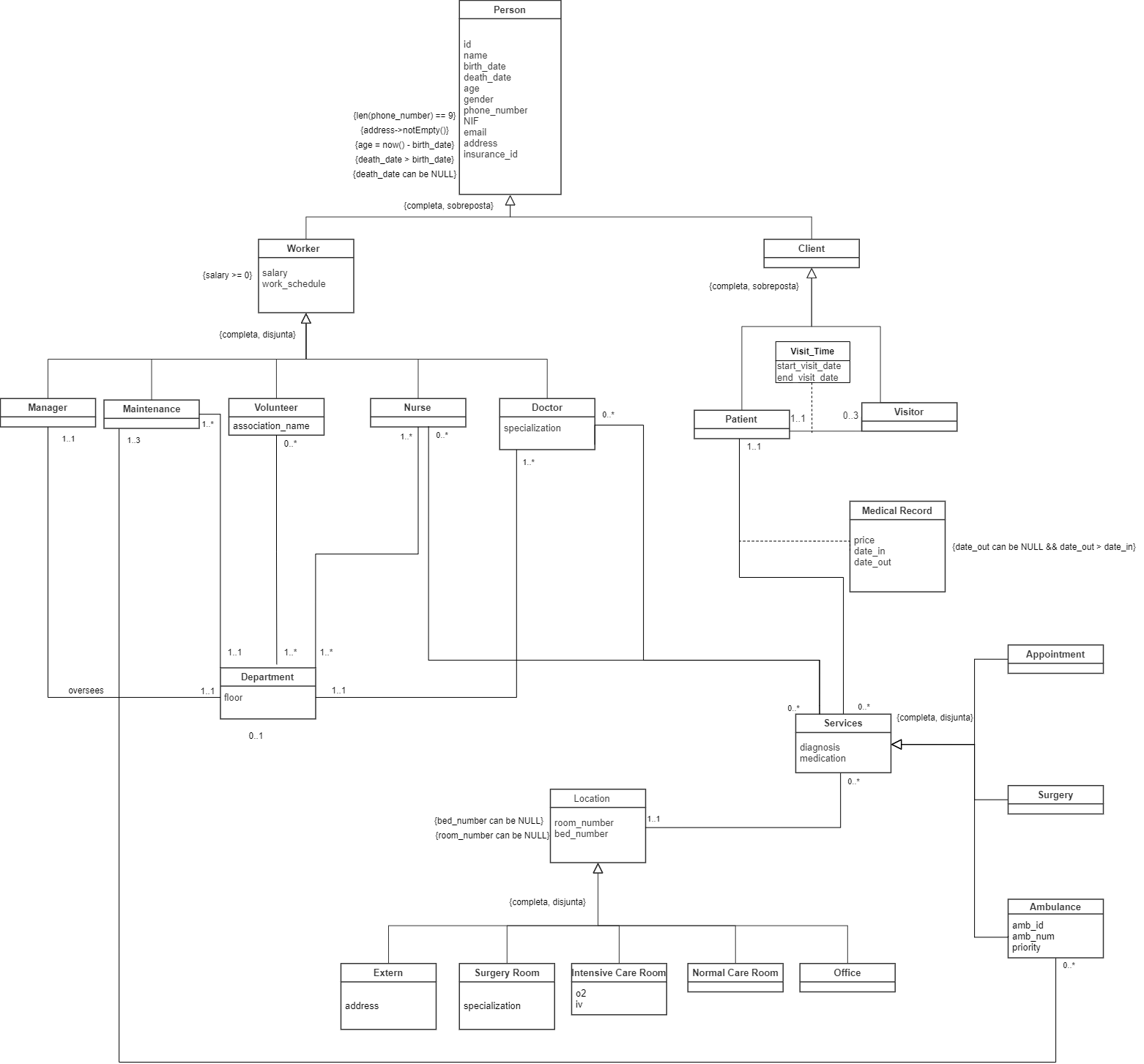
Em relação aos clientes, deverá ser feita a distinção entre visitantes e pacientes. Um visitante visita um paciente. Um paciente tem um registo médico e uma localização associada. O registo médico deve conter que trabalhadores interagiram com o paciente em questão e o que foi diagnosticado. O local pode variar entre escritório médico, sala de cuidados intensivos, sala de cuidados normais e sala de operações. Adicionalmente, um paciente pode ter um serviço designado.

Um serviço tem uma etiqueta. Esta etiqueta poderá ser algo como diagnóstico, consulta, reunião, entre outros. Tem também uma lista de pessoas associada, afetadas pelo serviço e uma lista de pessoas que providenciaram esse mesmo serviço. Por exemplo, um médico pode ter de realizar uma consulta a um paciente, ou a dois ou mais pacientes (i.e., uma mãe com dois filhos gémeos). O serviço tem uma data.

Para os trabalhadores, os mesmos podem ser de manutenção, enfermeiros, médicos (que podem ter várias especialidades), gestores e voluntários. Os trabalhadores também podem ter serviços associados. Alguns dos trabalhadores podem ter o seu próprio escritório. Enfermeiros e médicos podem pertencer a um departamento que um gestor supervisionará. Cada trabalhador deverá ter um salário.

Adicionalmente, será preciso gerir ambulâncias. As mesmas têm alguns trabalhadores associados (condutor, médico e enfermeiro) e prestarão um serviço a um paciente. Enquanto a prestação do serviço, a ambulância terá um nível de urgência associado.

**Diagrama UML**



**Esquema Relacional**

**Person** (id, name, birth\_date, death\_date, age, gender, phone\_number, NIF, email, address, insurance\_id)

**Department** (id, floor)

**Location** (id, room\_number, bed\_number)

**Shift** (id, day\_of\_the\_week\_in, time\_in, day\_of\_the\_week\_out, time\_out)

**Service** (id, diagnosis, medication, price, date\_in, date\_out, location -> Location patient -> Patient)

**Specialization** (speciality)

**Worker** (person->Person)

**WorkerShift** (worker->Worker, shift->Shift)

**Manager** (person->Worker, salary, department->Department)

**Maintenance** (person->Worker, salary, department->Department)

**Volunteer** (person->Worker, association\_name, department->Department)

**Nurse** (person->Worker, salary, department->Department)

**NurseService** (nurse->Nurse, service->Service)

**Doctor** (person->Worker, salary, department->Department)

**DoctorService** (doctor->Doctor, service->Service)

**DoctorSpecialization** (doctor->Doctor, specialization->Specialization)

**Client** (person->Person)

**Patient** (person->Client)

**Visitor** (person->Client)

**VisitTime** (patient->Patient, visitor->Visitor, start\_visit\_date, end\_visit\_date, order)

**Appointment** (service->Service)

**Surgery** (service->Service)

**Ambulance** (service->Service, amb\_id, priority)

**MaintenanceJob** (ambulance -> Ambulance, maintenance -> Maintenance, date, did\_disinfection, did\_restock)

**Extern** (location->Location, address)

**SurgeryRoom** (location->Location)

**SurgeryRoomSpecialization** (room->SurgeryRoom, specialization->Specialization)

**IntensiveCareRoom** (location->Location, o2, iv)

**NormalCareRoom** (location->Location)

**Office** (location->Location)

**Análise de Dependências Funcionais e Formas Normais**

id -> name, birth\_date, death\_date, age, gender, phone\_number, NIF, email, address, insurance\_id

id -> floor

id -> room\_number, bed\_number

id -> day\_of\_the\_week\_in, time\_in, day\_of\_the\_week\_out, time\_out

id -> diagnosis, medication, price, date\_in, date\_out, location, patient

speciality -> speciality

person -> person

worker, shift -> worker, shift

person -> department

person -> association\_name, department

person -> salary, department

nurse, service -> nurse, service

person-> salary, department

doctor, service -> doctor, service

doctor, specialization -> doctor, specialization

patient, visitor -> start\_visit\_date, end\_visit\_date, order

service -> service

service -> amb\_id, priority

ambulance, maintenance -> date, did\_disinfection, did\_restock

location -> address

location -> location

room, specializatoin -> room, specialization

location -> o2, iv

Para as relações Specialization, Worker, WorkerShift, NurseService, DoctorService, Client, Patient, Visitor, Appointment, Surgery, SurgeryRoom, SurgeryRoomSpecialization, NormalCareRoom e Office as dependências funcionais estão na BCNF e na 3ª forma normal, uma vez que se tratam de dependências funcionais triviais.

Para as restantes relações, uma vez que o lado esquerdo da relação funcional é também a sua chave, é possível concluir que se encontra na forma BCNF e na 3ª forma normal.

Deste modo, não foi preciso decompor nenhuma das relações.

Acrescentar mais cenas ao texto

O atributo age será computado numa View, e por isso, apenas implementado na próxima entrega.

Da mesma forma, existem algumas restrições que serão implementadas através de triggers, que também deixaremos para depois:

A chave para Specialization é case-insensitive

No máximo três visitantes poderão visitar um paciente

No máximo três pessoas realizarão a manutenção de uma ambulância

Uma sala de cirugia tem que ter no mínimo uma especialização

As localizações e os trabalhadores deveráo ser disjuntos, isto é, uma localização não poderá pertencer a mais do que uma subclass da mesma; e um trabalhador não poderá pertencer a mais do que uma subclasse do mesmo.

/\*Create trigger for no two visitors with the same patient that have the same order\*/