U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

POSTO VETERINÁRIO

Bases de Dados

Turma 1 Grupo 05

João Leite / up201705312@fe.up.pt

João Campos / up201704982@fe.up.pt

Márcia Teixeira / up201706065@fe.up.pt

17 de março de 2019

Índice

Introdução

Este trabalho, realizado para a unidade curricular de Bases de Dados, tem como objetivo avaliar a capacidade dos estudantes criarem uma base de dados, tendo esta segunda entrega como objetivo o melhoramento do diagrama UML, o mapeamento do modelo anteriormente criado para um esquema relacional e implementação deste esquema numa base de dados SQLite. Este relatório tem como objetivo descrever o modo como a base de dados pretendida estará organizada.

1. Contexto

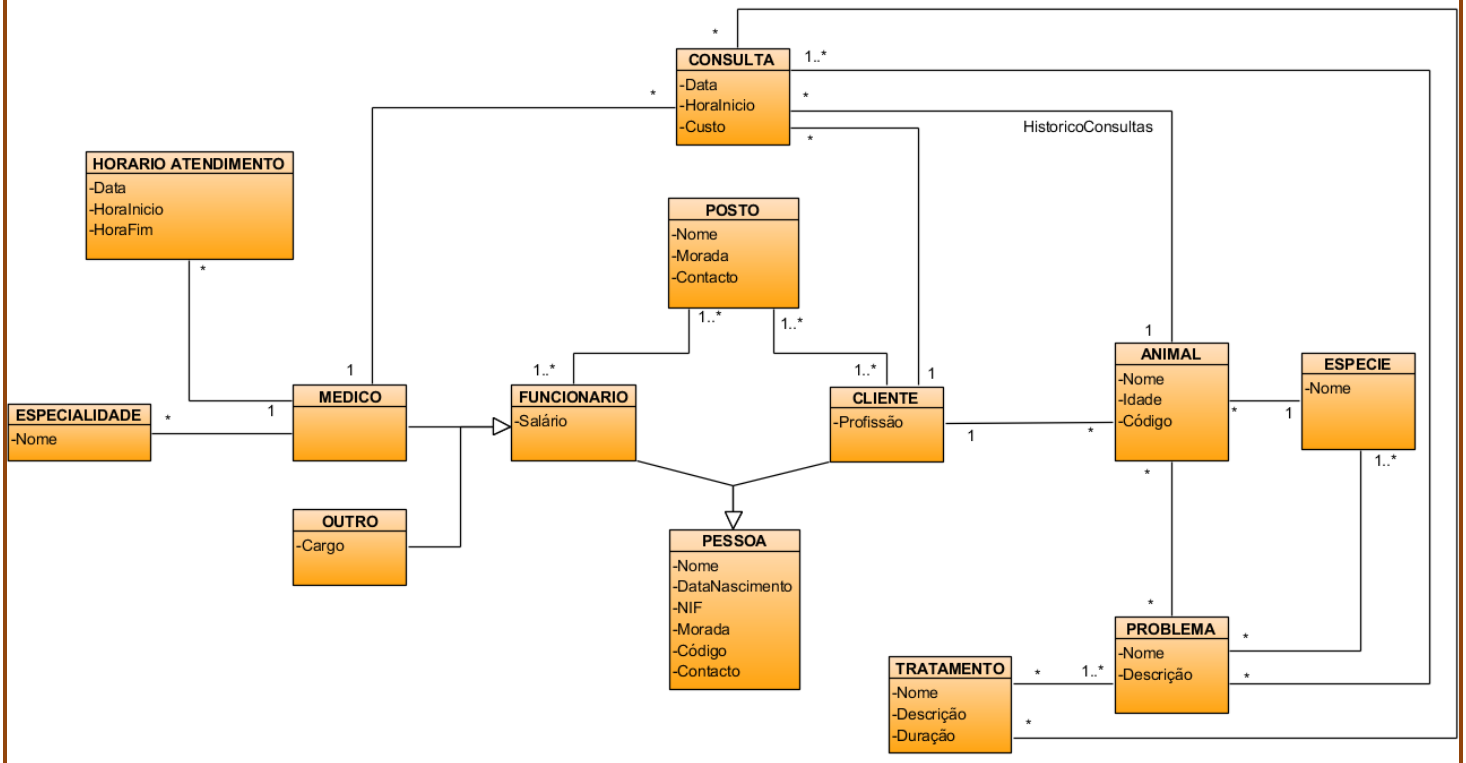
Este trabalho consistirá em fazer uma base de dados para um posto veterinário. Cada posto terá um nome, uma morada e um contacto.

Ao posto estarão associadas pessoas, caracterizadas por um nome, um NIF, um código identificador, uma morada, a sua data de nascimento e um contacto. As pessoas podem ser divididas em dois tipos, funcionários ou clientes, sendo que os clientes terão também a sua profissão e os funcionários terão o seu salário, sendo também divididos em dois tipos, médicos ou “outro”. O “outro” será identificado pelo seu cargo, já o médico terá associadas uma especialidade e um horário de atendimento - que guarda a data e horas de início e fim de atendimento para essa data.

Os clientes terão um conjunto de animais, caracterizados por um nome, a sua idade e um código identificativo, assim como uma espécie associada. Aos animais estarão associados diversos problemas, com um determinado nome e descrição. Estes problemas terão diferentes tratamentos possíveis.

O posto terá ainda consultas, com uma certa data, hora e custo, associadas a um determinado médico, animal e cliente (que pode ou não ser o dono do animal) e pode estar associada a um conjunto de problemas abordados e tratamentos recomendados.

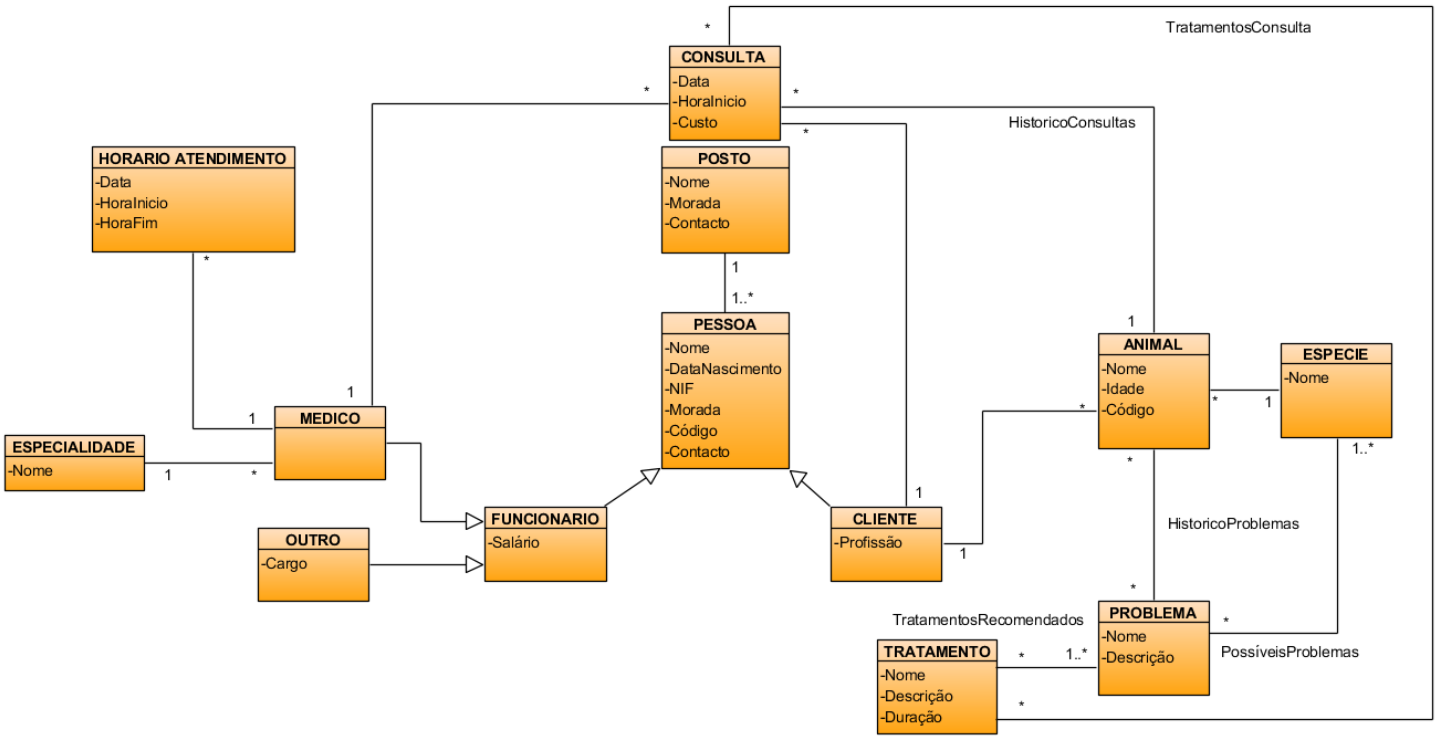
2. Diagrama UML Inicial



Acima encontra-se o diagrama UML criado pelo nosso grupo na primeira fase do trabalho, tendo em conta uma ideia geral de um posto veterinário.

Este UML foi depois revisto, tendo sido resolvidos alguns problemas do mesmo.

3. Diagrama UML Revisto



4. Classes

Todas as classes têm um atributo ID, único e usado para identificar os objetos da classe. No entanto, embora os IDs sejam usados como primary key das classes, estas podem ter mais chaves candidatas.

4.1. Posto

Um objeto desta classe representa um posto veterinário, identificado pelo seu nome e morada, tendo também um contacto.

Restrições: O ID deve ser único (PRIMARY KEY), um posto tem que ter um nome e uma morada (NOT NULL) e o contacto deve ser também único (UNIQUE) e um número com 9 algarismos (CHECK).

4.2. Pessoa

Uma pessoa é, no contexto do posto veterinário, identificada por um código. Uma pessoa é também caracterizada pelo seu nome, data de nascimento, morada, NIF e contacto.

4.2.1. Funcionário

Um funcionário é um tipo de pessoa que trabalha no posto veterinário, tendo, para além dos atributos de pessoa, um salário.

4.2.1.1. Médico

Um objeto desta classe representa um médico veterinário, que é um tipo de funcionário, tendo os mesmos atributos. Um médico tem também uma especialidade associada.

Restrições: O ID deve ser único (PRIMARY KEY), um médico tem que ter um nome, uma morada, um contacto, uma data de nascimento, um salário (valor maior que 0 – CHECK), um posto e uma especialidade associados (NOT NULL) e tanto o contacto como o NIF devem ser também únicos (UNIQUE) e um número com 9 algarismos (CHECK).

4.2.1.2. Outro

Esta classe representa “outros” funcionários, isto é, funcionários que não são médicos. Este tipo de funcionários tem como atributo o cargo que exerce (para além dos atributos de funcionário). Este cargo não pode ser “médico” ou “médico veterinário”, já que estes fazem parte da classe referida acima.

Restrições: O ID deve ser único (PRIMARY KEY), um “outro” tem que ter um nome, um contacto, uma data de nascimento, uma morada, um salário (valor maior que 0 – CHECK), um cargo e um posto associado (NOT NULL) e tanto o contacto como o NIF devem ser também únicos (UNIQUE) e um número com 9 algarismos (CHECK). O cargo não pode ser “médico (veterinário)”, já que estes funcionários fazem parte da classe Médico (CHECK).

4.2.2. Cliente

Um cliente é um tipo de pessoa que é cliente do posto veterinário. Para além dos atributos de pessoa, tem também como atributo a sua profissão.

Restrições: O ID deve ser único (PRIMARY KEY), uma pessoa tem que ter um nome, um contacto, uma morada, uma data de nascimento, uma profissão e um posto associado (NOT NULL) e tanto o contacto como o NIF devem ser também únicos (UNIQUE) e um número com 9 algarismos (CHECK).

4.3. Especialidade

Esta classe representa a especialidade de um médico do posto, tendo como atributo o nome da especialidade.

Restrições: O ID deve ser único (PRIMARY KEY), e a especialidade tem que ter um nome (NOT NULL) único (UNIQUE) .

4.4. Animal

Um animal é identificado, no contexto da clínica, por um código. Tem também como atributos o seu nome e idade, e está associado à sua espécie e ao seu dono (cliente do posto). Pode também ter associados vários problemas, que representam o histórico de problemas (problemas passados e presentes) do animal.

Restrições: O ID deve ser único (PRIMARY KEY), um animal tem que ter um nome, uma idade e um cliente e uma espécie associados (NOT NULL).

4.5. Problema

Um objeto desta classe representa um problema de saúde que pode afetar os animais, sendo caracterizado pelo seu nome e descrição. Um problema pode ter vários tratamentos associados, que representam os tratamentos que podem ser usados para tratar o problema.

Restrições: O ID deve ser único (PRIMARY KEY), um problema tem que ter um nome e uma descrição (NOT NULL) , sendo que o nome terá de ser único (UNIQUE). Sempre que um animal apresenta um problema numa consulta, este problema deve poder afetar a sua espécie (estar associado à mesma) (restrição ainda não implementada).

4.6. Espécie

Esta classe representa a espécie de um animal, sendo identificada pelo seu nome. Uma espécie pode estar associada a vários problemas, que representam os problemas que podem afetar animais daquela espécie.

Restrições: O ID deve ser único (PRIMARY KEY), e a espécie tem que ter um nome (NOT NULL) único (UNIQUE).

4.7. Tratamento

Esta classe representa um tratamento que pode ser recomendado para tratar certos problemas, tendo como atributos o seu nome, uma descrição, e a sua duração (em dias), sendo que uma duração de 0 corresponde a um tratamento instantâneo (por exemplo uma vacina).

Restrições: O ID deve ser único (PRIMARY KEY), um tratamento tem que ter um nome único (UNIQUE) e uma descrição (NOT NULL), e a duração tem que ser maior ou igual que 0 dias (CHECK). Sempre que, numa consulta, é recomendado um tratamento, este deve ser adequado a um dos problemas que o animal apresentou nesta consulta (restrição ainda não implementada).

4.8. Consulta

Esta classe representa uma consulta dada por um médico veterinário a um animal, tendo como atributos a data da consulta, a sua hora de início e o seu custo. A uma consulta estão associados um médico (que dá a consulta), um animal (paciente) e um cliente (que paga a consulta). Podem também estar associados vários tratamentos que tenham sido feitos ou prescritos pelo médico na consulta.

Restrições: O ID deve ser único (PRIMARY KEY), uma consulta tem que ter uma data, hora de início, um animal, médico e cliente associados (NOT NULL). A hora de início da consulta não pode ser anterior à hora de início de atendimento do médico que dá a consulta para aquela data (restrição ainda não implementada). Também não deve ser nem igual ou posterior à hora de fim (restrição ainda não implementada). A hora de início não pode também coincidir com a hora de início de outra consulta do mesmo médico naquela data (restrição ainda não implementada).

4.9. Horário Atendimento

Esta classe representa o horário de atendimento de um médico para um determinado dia, tendo como atributos a data a que se refere, a hora em que se inicia o atendimento e a hora em que este termina.

Restrições: O ID deve ser único (PRIMARY KEY), um horário de atendimento tem que ter uma data, hora de início, hora de fim e um médico associado (NOT NULL). A hora de fim de atendimento deve ser posterior à hora de início de atendimento (CHECK).

5. Esquema relacional

Posto (idPosto, nome, morada, contacto)

Pessoa (idPessoa, codigo, nome, dataNasc, NIF, morada, contacto, idPosto->Posto)

Cliente (idCliente->Pessoa, nome, dataNasc, NIF, morada, contacto, profissão, idPosto->Posto)

Funcionario (idFuncionario->Pessoa, nome, dataNasc, NIF, morada, contacto, salario, idPosto->Posto)

Medico (idMedico->Funcionario, nome, dataNasc, NIF, morada, contacto, salario, nomeEspecialidade->Especialidade, idPosto->Posto)

Outro (idFuncionario-> Funcionario, nome, dataNasc, NIF, morada, contacto, salario, cargo, idPosto->Posto)

Especialidade (idEspecialidade, nomeEsp)

HorarioAtendimento (idHorario, idMedico-> Medico, data, horaInicio, horaFim)

Animal (idAnimal, codigoCliente->Cliente, idade, nome, nomeEspecie->Especie)

Especie (idEspecie, nomeEspecie)

Problema (idProblema, nomeProblema, descricao)

HistoricoProblemas (codigoAnimal-> Animal, nomeProblema-> Problema)

PossiveisProblemas (nomeEspecie-> Especie, nomeProblema-> Problema)

Tratamento (idTratamento, nomeTratamento, descricao, duracao)

TratamentosRecomendados (nomeProblema-> Problema, nomeTratamento-> Tratamento)

Consulta (idConsulta, data, codigoAnimal-> Animal, codigoMedico-> Medico, codigo-> Cliente, custo)

TratamentosConsulta (data-> Consulta, codigoAnimal-> Animal-> Consulta, nomeTratamento-> Tratamento)

6. Análise de Dependências Funcionais e Formas Normais

6.1. Relação Posto

idPosto->nome, morada, contacto
nome, morada-> contacto, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.2. Relação Pessoa

idPessoa-> nome, dataNasc, NIF, morada, contacto, idPosto
NIF-> idPessoa, nome, dataNasc, morada, contacto, idPosto
nome, dataNasc, morada, contacto -> idPessoa, NIF, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.3. Relação Cliente

idCliente-> nome, dataNasc, NIF, morada, contacto, profissão, idPosto
NIF-> idCliente, nome, dataNasc, morada, contacto, profissão, idPosto
nome, dataNasc, morada, contacto -> idCliente, NIF, profissão, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.4. Relação Funcionario

idFuncionario-> nome, dataNasc, NIF, morada, contacto, salario, idPosto
NIF-> idFuncionario, nome, dataNasc, morada, contacto, salario, idPosto
nome, dataNasc, morada, contacto -> idFuncionario, NIF, salario, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.5. Relação Medico

idMedico-> nome, dataNasc, NIF, morada, contacto, salario, nomeEspecialidade, idPosto
NIF-> idMedico, nome, dataNasc, morada, contacto, salario, nomeEspecialidade, idPosto
nome, dataNasc, morada, contacto -> idMedico, NIF, salario, nomeEspecialidade, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.6. Relação Outro

idFuncionario-> nome, dataNasc, NIF, morada, contacto, salario, cargo, idPosto
NIF-> idFuncionario, nome, dataNasc, morada, contacto, salario, cargo, idPosto
nome, dataNasc, morada, contacto -> idFuncionario, NIF, salario, cargo, idPosto

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.7. Relação Especialidade

idEspecialidade -> nomeEsp
nomeEsp -> idEspecialidade

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.8. Relação HorarioAtendimento

idHorario-> idMedico, data, horaInicio, horaFim
idMedico, data-> horaInicio, horaFim, idHorario

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.9. Relação Animal

idAnimal-> idCliente, idade, nome, nomeEspecie

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.10. Relação Especie

idEspecie-> nomeEspecie;
nomeEspecie-> idEspecie;

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.11. Relação Problema

idProblema-> nomeProblema, descricao
nomeProblema-> idProblema, descricao

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.12. Relação HistoricoProblemas

Não existem Dependências Funcionais não-triviais.

6.13. Relação PossiveisProblemas

Não existem Dependências Funcionais não-triviais.

6.14. Relação Tratamento

idTratamento-> nomeTratamento, descricao, duracao
nomeTratamento-> descricao, duração, idTratamento

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.15. Relação TratamentosRecomendados

Não existem Dependências Funcionais não-triviais.

6.16. Relação Consulta

idConsulta-> data, idAnimal, idMedico, idCliente, horaInicio, custo
data, codigoAnimal-> codigoMedico, codigoCliente, horaInicio, custo
data, codigoMedico, horaInicio-> codigoAnimal, codigoCliente, custo

Esta relação está na BCNF, pois para cada dependência funcional não-trivial A->B, A é uma (super)key. Estando provado que está na BCNF, pode-se também concluir que está na 3NF.

6.17. Relação TratamentosConsulta

Não existem Dependências Funcionais não-triviais.