

**INF3710 –Fichiers et Bases de données**

**Hiver 2019**

**TP No. 5**

**Groupe 4**

1340155 – Alassane Maiga

1862313 – Mohamed Esseddik BENYAHIA

**Soumis à : Manel Grinchi.**

**14 Avril 2019**

I) Une brève introduction

Ceci est notre rapport du projet du cours de bases de données inf3710, session d’hiver 2019. Le projet consiste à développer un système de bases de données centralisées pour la gestion et la communication entre les cliniques d’une organisation appelée VetoSansFrontieres. De plus, il nous a été demandé de créer une application web permettant l’interrogation de la base de données.

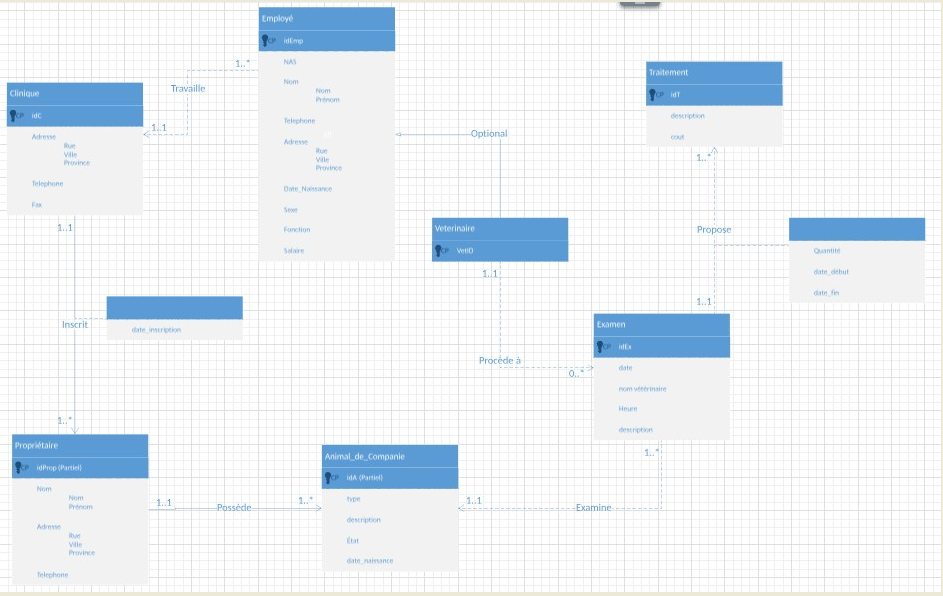
Ce projet est important par le fait qu’il nous permet de revoir les principales notions de base vues dans le cours en les mettant en pratique sur un exemple réel. À cela s’ajouter le fait que le projet nous permet de passer par toutes les étapes de création d’un système de base de données, en allant de l’étape de conception jusqu’à l’étape de création des requêtes de manipulation de la table.

Dans ce qui suit nous détaillerons les différentes tâches qui nous ont été assignées dans le projet.

II) Le modèle conceptuel UML incluant les hypothèses et commentaires

Après avoir lu et noté les besoins de l’organisme, nous avons proposé notre modèle entité association à l’aide du logiciel Visio et en utilisant la notation UML.

La figure 1 présente le modèle en question.



Nous avons rajouté l’attribut *années d’expériences* à l’entité *Gestionnaire* et l’attribut *License* à l’entité *Vétérinaire,* car selon la chargée de laboratoire une entité ne peut rester sans un attribut (autre que la clé primaire).

Nous n’avons pas rajouté le numéro d’animale ni celui du vétérinaire dans l’examen car cela est inclus dans la multiplicité One to Meany.

Voici une description des hypothèses considérées dans notre modèle:

-) Un personnel travail uniquement dans une et une seule clinique

-)

III) Le modèle relationnel en syntaxe abstraite

Clinic (icD, phone, fax, road, city, province, PC)

Employe (idE, idC, NAS, name, surname, phone, DoB, sex, function, salary, road, city, provice, PC)

Primary Key(idE)

Foreign Key(idC) REFERENCES Clinic(idC)

Owner (idO, idC, name, surname, phone, road, city, province, PC)

Primary Key(idO, idC)

Foreign Key (idC) REFERENCES Clinic(idC)

Pet (idP, idO, idC, specie, description, DoB, status)

Primary Key (idP, idO, idC)

Foreign Key (idO, idC) REFERENCES Owner(idO, idC)

Enrollment (idC, idO, idP, enrollment\_date)

Primary Key(idC, idO, idP)

Foreign Key(idC) REFERENCES Clinic(idC)

Foreign Key(idO) REFERENCES Owner(idO)

Foreign Key(idP) REFERENCES Pet(idP)

PhysicalExam (idE, idP, idExam, examDate, examTime, description)

Primary Key (idExam)

Foreign Key(idE) REFERENCES Employee(idE)

Foreign Key(idP) REFERENCES Pet(idP)

Treatment (idT, description, cost)

Primary Key(idT)

ProposedTreatment (idT, idExam, idP, quantity, startDate, endDate)

Primary Key (idT, idExam, idP)

Foreign Key (idT) REFERENCES Treatment(idT)

Foreign Key (idExam) REFERENCES PhysicalExam(idExam)

Foreign Key (idP) REFERENCES Pet(idP)

IV) Les dépendances fonctionnelles et l’explication de la forme normale de votre BD tel que requis dans la liste des tâches à effectuer

Basé sur la BD psql que l'on a faite :

numClinique à ClinNom, ClinAdresse, ClinNumTelephone, ClinNumTelecopieur, numGestionnaire

numEmploye à numClinique, NAS, EmPrenom, EmNom, EmAdresse, EmNumTelephone, EmDateNaissance, EmSexe, fonction, salaire

NAS à EmPrenom, EmNom, EmAdresse, EmNumTelephone, EmDateNaissance, EmSexe

numProprio, numClinique à ProPrenom, ProNom, ProAdresse, ProNumTelephone

numAnimal. numProprio, numClinique à AnNom, typeAnimal, AnDescription, AnDateNaissance, AnDateInscription, etatActuel

numExamen à dateExam, descResultats, numVeterinaire, numAnimal

numTraitement à TrDescription, TrCout

numTraitement, numExamen à quantite, dateDebut, dateFin

V) Création de la base de donnée PostgreSQL

VI) Insertion des données

VII) Création des requêtes

SET search\_path=tp5db;

/\*1) Lister les le numéro et nom des cliniques, leur adresse et leur gestionnaire, ordonnés par le numéro de clinique\*/

SELECT numClinique,nom,adresse,numGestionnaire FROM vetosfDB.Clinique ORDER BY numClinique;

/\*2) Lister les noms des animaux sans doublons dans toutes les cliniques\*/

SELECT DISTINCT nom FROM vetosfDB.Animal;

/\*3) Lister les numéros et noms des propriétaires d’animaux ainsi que les détails de leurs

animaux dans une clinique donnée (J'ai choisi la clinique C001)\*/

SELECT pr.numProprio, pr.prenom, pr.nom, an.numAnimal, an.numClinique, an.nom, an.typeAnimal, an.description, an.dateNaissance, an.dateInscription, an.etatActuel FROM vetosfdb.Animal AS an, vetosfdb.Proprietaire AS pr

WHERE an.numClinique='C001' AND pr.numClinique=an.numClinique AND an.numProprio=pr.numProprio;

/\*4) Lister l’ensemble des examens d’un animal donné (je choisi l'animal A001)\*/

SELECT \* FROM vetosfDB.Examen WHERE numAnimal='A001';

/\*5) Lister les détails des traitements d’un animal suite à un examen donné (je choisi l'exam EX002)\*/

SELECT \* FROM vetosfDB.Traitement NATURAL INNER JOIN vetosfDB.PropositionTraitement WHERE numExamen='EX002';

/\*6) Lister le salaire total des employés par clinique ordonné par numéro de clinique\*/

SELECT cl.numClinique, cl.nom, SUM(e.salaire) AS salaireTotal FROM vetosfdb.Employe as e, vetosfdb.Clinique as cl

WHERE e.numClinique=cl.numClinique

GROUP BY cl.numClinique

ORDER BY cl.numClinique;

/\*7) Lister le nombre total d’animaux d’un type donné (j'ai choisi chat) dans chaque

clinique\*/

SELECT cl.numClinique, cl.nom, COUNT (an.\*) as NombreDeChat FROM vetosfdb.Clinique as cl, vetosfDB.Animal as an

WHERE an.typeAnimal='chat' and an.numClinique=cl.numClinique

GROUP BY cl.numClinique;

/\*8) Lister le coût minimum, maximum et moyen des traitements\*/

SELECT MIN(cout)as coutMinimum,MAx(cout)as coutMaximum, CAST(AVG(cout) as NUMERIC(5,2)) as coutMoyen FROM vetosfdb.traitement;

/\*9) Quels sont les noms des employés de plus de 50 ans ordonnés par nom ?\*/

SELECT nom, prenom FROM vetosfdb.Employe

GROUP BY numEmploye

HAVING (select date\_part('year',now()::date)-DATE\_Part('year',dateNaissance::date))>50

ORDER BY nom;

/\*10) Quels sont les propriétaires dont le nom contient « blay » ?\*/

SELECT \* FROM vetosfdb.Proprietaire WHERE nom LIKE '%blay%';

/\*11) Supprimez le vétérinaire « Jean Tremblay »\*/

DELETE FROM vetosfdb.Employe WHERE fonction='Veterinaire' AND nom='Tremblay' AND prenom='Jean';

/\*12) Lister les détails des propriétaires qui ont un chat et un chien\*/

SELECT \* FROM vetosfdb.Proprietaire

WHERE numProprio IN (SELECT numProprio FROM vetosfdb.Animal WHERE typeAnimal='chat')

AND numProprio IN (SELECT numProprio FROM vetosfdb.Animal WHERE typeAnimal='chien');

/\*13) Lister les détails des propriétaires qui ont un chat ou un chien\*/

SELECT \* FROM vetosfdb.Proprietaire

WHERE numProprio IN (SELECT DISTINCT numProprio FROM vetosfdb.Animal WHERE typeAnimal='chat' OR typeAnimal='chien');

/\*14) Lister les détails des propriétaires qui ont un chat mais pas de chien vacciné contre la

grippe (la condition vacciné contre la grippe ne s’applique qu’aux chiens)\*/

SELECT \* FROM vetosfdb.Proprietaire as pr

WHERE pr.numProprio IN (SELECT numProprio FROM vetosfdb.Animal

WHERE typeAnimal='chat') AND

((pr.numProprio NOT IN (SELECT numProprio FROM vetosfdb.Animal WHERE typeAnimal='chien')) OR

(pr.numProprio IN (SELECT a1.numProprio FROM vetosfdb.Animal as a1 WHERE a1.typeAnimal='chien' AND a1.numAnimal NOT IN (SELECT numAnimal FROM vetosfdb.Examen WHERE numExamen IN (SELECT numExamen FROM vetosfdb.PropositionTraitement WHERE numTraitement='T112' AND dateFIN<now())))));

/\*15) Lister tous les animaux d’une clinique donnée (je choisi clinique C001) avec leurs traitements s’ils existent. Dans le

cas contraire, affichez null.\*/

SELECT an.numAnimal, an.nom, trCl.numTraitement, trCl.description, trCl.cout, trCl.numExamen, trCl.quantite, trCl.datedebut, trCl.datefin

FROM vetosfDB.Animal as an LEFT OUTER JOIN (SELECT \* FROM vetosfdb.Examen NATURAL INNER JOIN

(SELECT \* FROM vetosfDB.Traitement NATURAL INNER JOIN vetosfdb.PropositionTraitement) as tr) as trCl

ON an.numAnimal=trCl.numAnimal

WHERE an.numClinique='C001'

ORDER BY numAnimal;

VIII) Création de l’application web