Ophélie DURDON
Maxim MORARU
Dirigé par Beatrice BEAUJET

Maxlie

GESTION D'UN CABINET MEDICAL

PROJET INFO0502

Sommaire

Introduction

Dans le cadre de l'ouverture imminente d'un cabinet médical, nous avons été contactés afin de créer une base de données qui permet de stocker de gros volumes d'informations et partager celles-ci à une communauté de personnes. De nos jours beaucoup de secteurs utilisent ce système de base de données pour faciliter la gestion générale de leurs entreprises. En particulier, la complexité d'un cabinet médical et la multitude des professions liées à la médecine rendent l'utilisation des bases de données très utile pour mettre en forme ce type de réseau et comprendre son fonctionnement.

Nous avons dû observer longuement le cabinet médical avant de créer notre base de données. Nous avons dû relever tous les employés ainsi qu'examiner la structure de l'établissement.

Nous avons pu noter que le bâtiment était sur 3 étages, sur chacun d'entre eux nous avons 10 salles, où se font les consultations. Au rez-de-chaussée se trouve les secrétaires, qui sont regroupées dans une grande pièce, ainsi que quelques salles d'attentes. Les patients attends dans une certaine salle selon le médecin qu'il vienne consulter. Les secrétaires leurs indiquent bien évidement où se placer. Elles ont un rôle essentiel dans le cabinet médical, car ce sont elles qui organisent, les emplois du temps des médecins ainsi que la répartition des salles. Chaque médecin a à sa disposition un assistant pour l'aider lors de ses consultations.

- **Nom Cabinet:** Maxlie
- **Logo :**

- **Salles et personnel :**

Nombres de salles		**Nombres de médecins**		**Nombres de secrétaires**
De consultation	D'attentes	Avec 1 seule spécialités	Avec Plusieurs spécialités	
30	3	25	7	3

Analyse

Cahier des charges

Dans l'imminente ouverture du cabinet Maxlie, pour lequel nous sommes chargés de créer une base de données, nous avons dû étudier toute l'organisation de cette entreprise.

Ce cabinet médical est composé de plusieurs médecins, certains sont généralistes, d'autres sont spécialisés dans un ou plusieurs domaines (par exemple : cardiologie, angiologie, urologie, dermatologie, gastro-entérologie). Chaque médecin possède un assistant pour l'aider lors de ses consultations. Il y a environ 30 salles de consultations à la disposition des médecins, chacune numérotée en fonction de son étage. Ces médecins reçoivent des patients pour des consultations (le patient précise le motif lorsqu'il prend un rendez-vous). Chaque patient est affilié à un régime obligatoire et possède une ou plusieurs mutuelles (ou aucune).

Un médecin peut visualiser, à tout moment, les vaccins subis par le patient X et les différents antécédents de ce patient (par exemple : une allergie), ainsi que les dates correspondantes.

Lors d'un rendez-vous le médecin peut demander à son patient de faire un examen. Le patient peut choisir de faire l'examen dans un autre cabinet médical et il doit reprendre un rendez-vous pour revenir avec le résultat de l'examen. En ayant toutes les informations, le médecin peut rendre un diagnostic qui peut comporter plusieurs maladies, ainsi qu'une ordonnance contenant plusieurs médicaments ou/et traitements. On veut garder un historique des ordonnances prescrites.

Le planning des médecins et de l'occupation des salles est géré par des secrétaires.
Un accès handicapé a été aménagé.

Matrice de flux

Une matrice de flux nous permet de structurer les échanges entre nos différents acteurs.
Notre domaine d'étude étant un cabinet médical. Nous avons pu déterminer 6 acteurs : Patient, Médecin, Secrétaire, Assistant, Mutuelle et le Régime_Obligatoire.Par exemple on peut voir les échanges d'un médecin vis à vis d'un patient : Un patient consulte un médecin et reçoit un traitement de sa part.

→	Patient	Medecin	Secretaire	Assistant	Mutuelle	Regime Obligatoire
Patient	-	~Etre consulter ~Recevoir traite-ment	~Demander consultation		~Transmettre papiers	~Transmettre papiers
Medecin	~Consulter ~Prescrire traitement ~Donner ordonnance	-		~Donner planing		
Secretaire	~Donner date rendez-vous ~Enregistre ~Donner papiers	~Donner le planing	-			
Assistant		~Recevoir le planing		-		
Mutuelle	~Recevoir les papiers ~Rembourser				-	~Transmettre papiers
Regime Obligatoire	~Recevoir les papiers ~Rembourser		~Transmettre papiers			-

Dictionnaire de données

Les entités

- Patient :

Attribut	Definition attribut	Type/Taille	Exemple de valeur
IdPatient	Identifiant du patient	Int(11)	4
NomPatient	Nom du patient	Varchar(25)	MARA
PrenomPatient	Prenom du patient	Varchar(25)	Anais
DateNaissance	Date de naissance du patient	Datetime	08/06/97 11:05
Adresse	Dernier adresse connu du patient	Varchar(250)	3 rue de la vallée

- Régime_Obligatoire :

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdRegime	Identifiant du régime	Int(11)	2
PourcentageRembourser	Précise le pourcentage rembourser au patient	Int	70
NomRegime	Précise quel est le type du régime	Varchar(25)	mgel

- Mutuelle:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdMutuelle	Identifiant de la mutuelle	Int(11)	12
NomMutuelle	Nom de la mutuelle	Varchar(25)	maaf
TauxRemboursementMutuelle	Précise a quelle hauteur la mutuelle va rembourser le patient	Int	30
Forfait	Précise les « offres »	Varchar(250)	Optique: 300 euros tout les 2 ans pour une paire de lunette

- Salle:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdSalle	Identifiant de la salle	Int(11)	9
EtageSalle	Donne le numéro de l'étage ou la consultation va se dérouler	Int	2
NumeroSalle	Donne le numéro de la salle ou la consultation va se dérouler	Int	103

- Rendez_vous:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdRendezVous	Identifiant du rendez vous	Int(11)	28
Motif	Permet de savoir quel est la raison du rendez vous	Varchar(25)	Toux

- Médecin:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdMédecin	Identifiant du médecin	Int(11)	32
NomMedecin	Nom du médecin	Varchar(25)	Vasser
PrenomMedecin	Prénom du médecin	Varchar(25)	pierre
Anciennete	Nombre d'année d'ancienneté dans le métier	Int	20

- Specialité:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdSpécialité	Identifiant de la spécialité	Int(11)	2
Domaine	Dans quel domaine le médecin est spécialisé	Varchar(25)	neurologie

- Ordonnance:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdOrdonnance	Identifiant de l'ordonnance	Int(11)	25

- Examen:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdExamen	Identifiant de l'examen	Int(11)	4
NomExamen	Donne l'intitulé exact de l'examen que le patient doit subir	Varchar(25)	irm
DescriptionExamen	Décrit avec plus de précision l'examen à passer	Varchar(250)	Image par résonance magnétique

- Médicaments:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdMedicament	Identifiant du médicament	Int(11)	7
NomMedicament	Nom du médicament	Varchar(25)	Doliprane
QuantiteMedicament	Quantité à prendre par jour	Varchar(100)	3 pendant 2 jours
TypeMedicament	Quel genre de médicament c'est	Varchar(25)	antalgique
Renouvelement	Combien de fois nous pouvons aller rechercher la prescription	Int	4

- Assistant:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdAssistant	Identifiant de l'assistant	Int(11)	3
NomAssistant	Nom de l'assistant	Varchar(25)	Donova
PrenomAssistant	Prénom de l'assistant	Varchar(25)	Rosie

- Vaccin:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdVaccin	Identifiant du vaccin	Int(11)	6
NomVaccin	Nom du vaccin	Varchar(25)	gardasil
ContreIndication	Contre indication liée au vaccin	Varchar(250)	Hyper sensibilité à l'un des composant

- Antecedent:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdAntécédent	Identifiant de l'antécédent	Int(11)	5
TypeAntecedent	Type de l'antécédent	Varchar(25)	Cardio-vasculaire
DescriptionAntecedent	Description	Varchar(250)	Infarctus du myocarde

- Diagnostique:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdDiagnostic	Identifiant du diagnostique	Int(11)	30
DescriptionDiagnostique	Description du diagnostique qu'a donner le médecin	Varchar(250)	Disjonction acromio-claviculaire

- Maladie:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdMaladie	Identifiant de la maladie	Int(11)	4
GraviteMaladie	Détermine la gravité de la maladie	Varchar(25)	Maligne
NomMaladie	Nom de la maladie	Varchar(25)	Adénocarcinome bronchique

- Traitement:

Attribut	Définition attribut	Type/Taille	Exemple de valeur
IdTraitement	Identifiant du traitement	Int(11)	20
DescriptionTraitement	Description du traitement	Varchar(250)	Prescription de séance de kinésithérapie
DureeTraitement	Combien de temps le patient doit il suivre le traitement prescrit par le médecin	Varchar(250)	1 séance par semaine pendant 6 mois

- Souscrire:

Attributs	Etre la relation entre	Nom en SQL
-	Medecin et Mutuelle	souscrire

- Avoir:

Attributs	Etre la relation entre	Nom en SQL
-	Medecin et Specialite	specialite_medecin

- Prescrire:

Attributs	Etre la relation entre	Nom en SQL
-	Ordonnance et Medicament	prescrire

- Ordonner:

Attributs	Etre la relation entre	Nom en SQL
-	Ordonnance et Examen	liste_examens

- Engager:

Attributs	Etre la relation entre	Nom en SQL
-	Medecin et Assistant	engager

- Avoir_Fait:

Attributs	Etre la relation entre	Nom en SQL
DateVaccin (Type: Date) DateRapel (Type: Date)	Vaccin et Patients	vaccins_patient

- Avoir_eu:

Attributs	Etre la relation entre	Nom en SQL
DateAntecedant (Type: Date)	Antecedent et Patients	antecedent_patient

- Donner:

Attributs	Etre la relation entre	Nom en SQL
-	RendezVous et Diagnostic	donner

- Comporte:

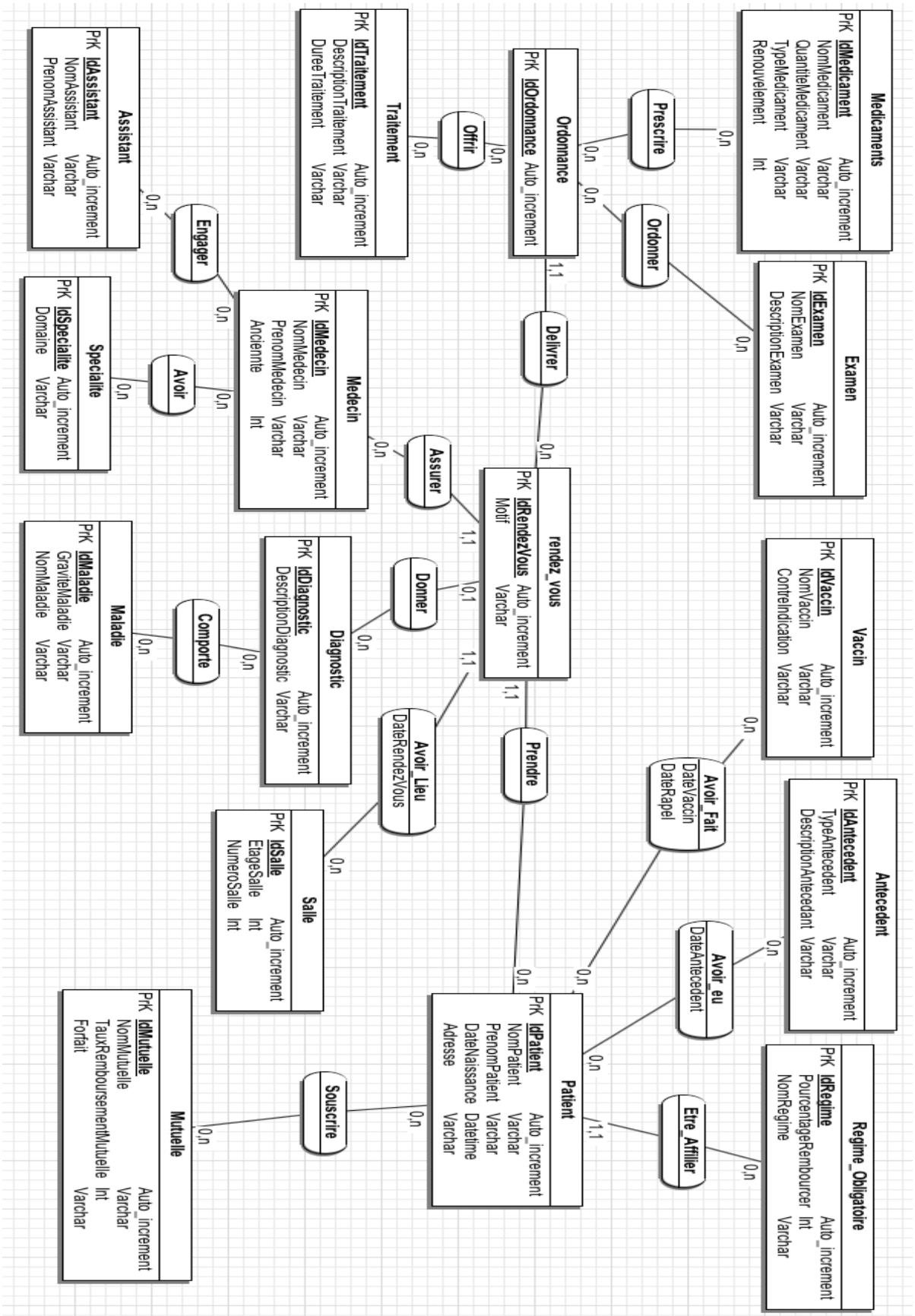
Attributs	Etre la relation entre	Nom en SQL
-	Diagnostic et Maladie	liste_maladies

- Offrir:

Attributs	Etre la relation entre	Nom en SQL
-	Traitement et Ordonnance	liste_traitements

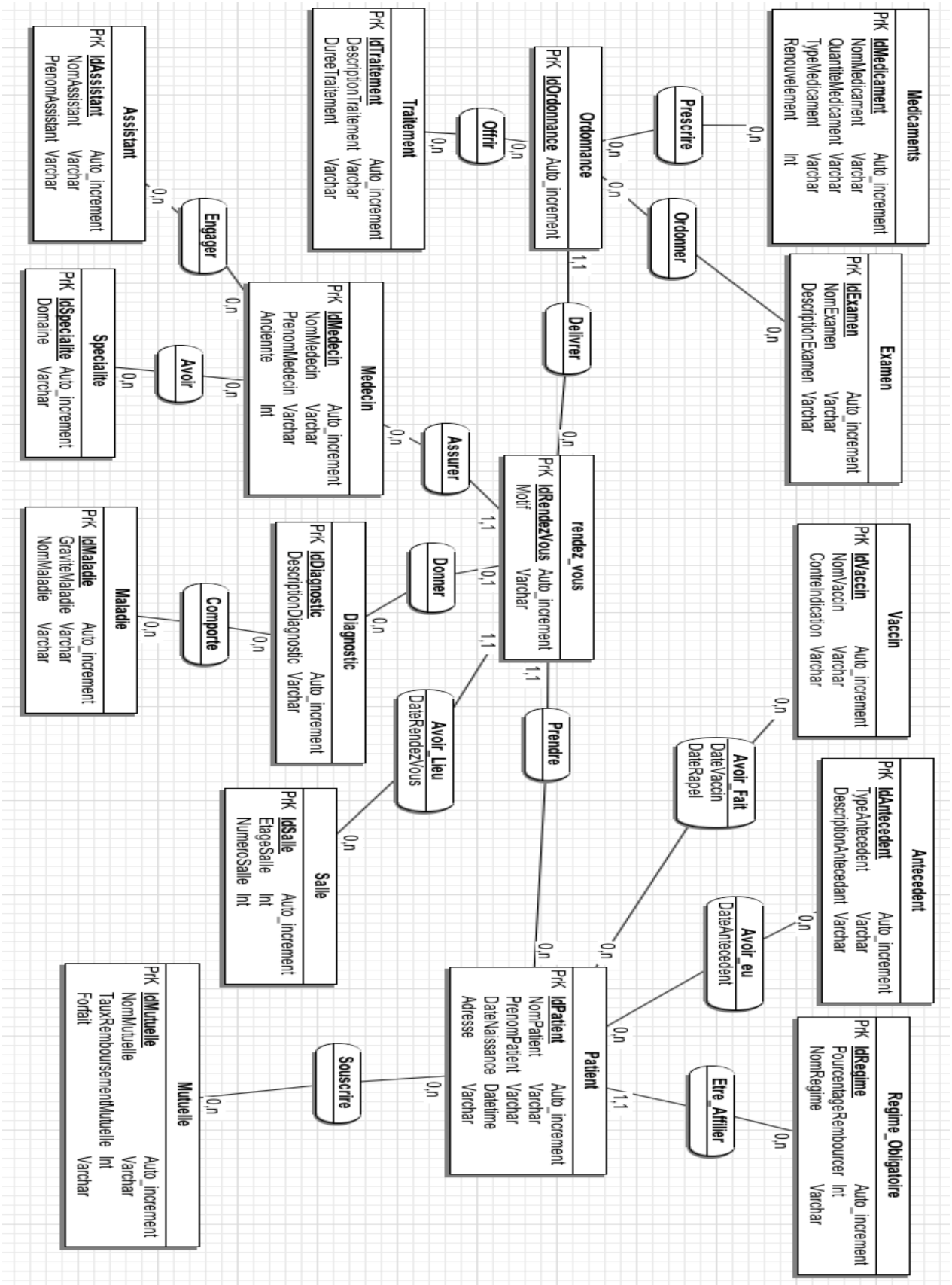
Modèle conceptuel de données

Modèle conceptuel de données ou MCD est une représentation statique, structurée du système d'information d'une entreprise.



Modèle logique de données

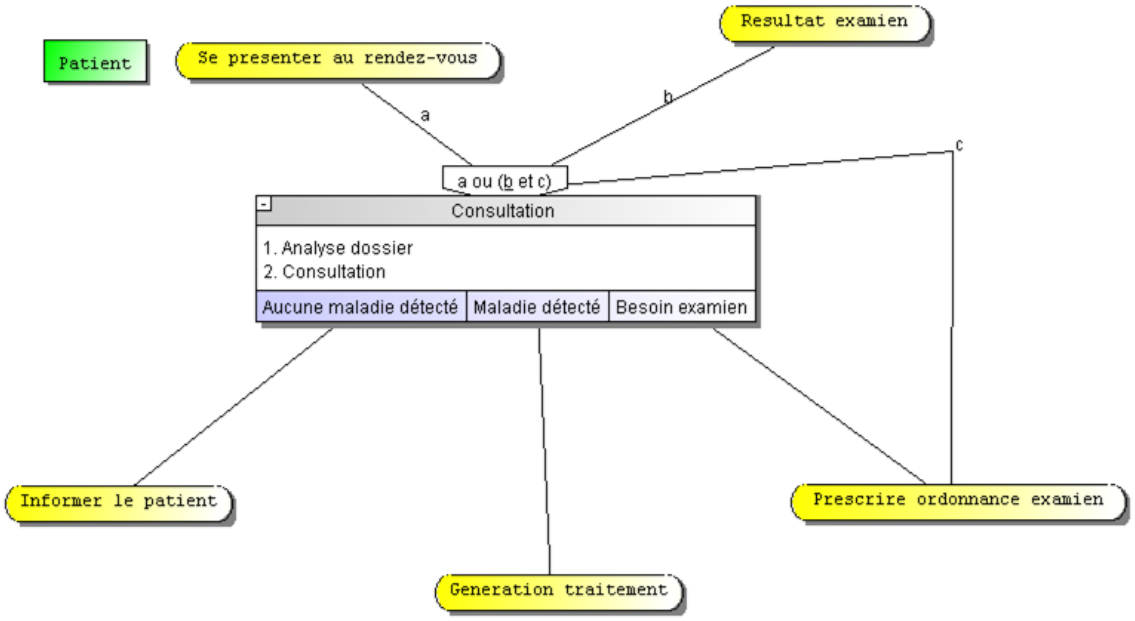
Modèle logique de données ou MLD est la modélisation logique des données qui tient compte de l'organisation des données.



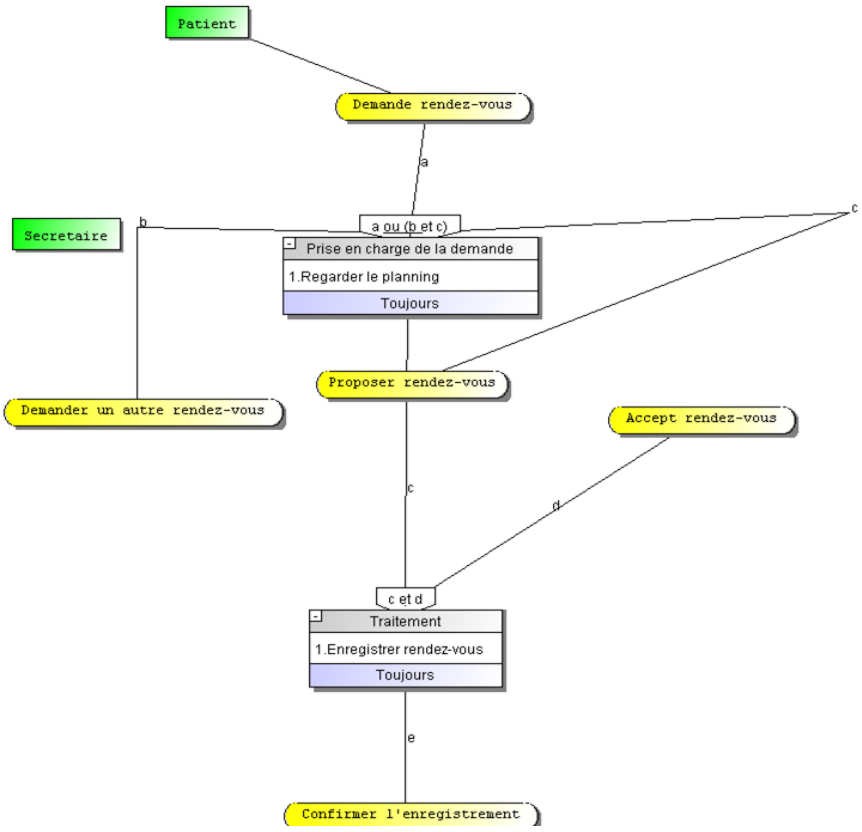
Modèle conceptuel de traitement

Modèle conceptuel de traitement ou MCT est la représentation dynamique du système d'information. Il permet de savoir quoi faire selon l'évènement en cours.

MCT Consultation



MCT Rendez-vous



Modèle relationnelle

Modèle relationnelle ou MR est une façon de modéliser les différentes informations contenues dans une base de données.

Medicaments	*(IdMedicament,NomMedicament,QuantiteMedicament,TypeMedicament,Renouvelement)*
Prescrire	*(#IdOrdonnance,#IdMedicament)*
Ordonnance	*(IdOrdonnance,DateOrdonnance,#IdRendezVous)*
Traitement	*(IdTraitement, DescriptionTraitement, DureeTraitement)*
Offrir	*(#IdTraitement,#IdOrdonnance)*
Engager	*(# IdMedecin,#IdAssistant)*
Assistant	*(IdAssistant, NomAssistant,PrenomAssistant)*
Examen	*(IdExamen, NomExamen ,DescriptionExamen)*
Ordonner	*(#IdOrdonnance,#IdExamen)*
rendez_vous	*(IdRendezVous, Motif, DateRendezVous,#IdSalle,#IdMedecin,#IdPatient)*
Medecin	*(IdMedecin, NomMedecin, PrenomMedecin, Anciennete)*
Avoir	*(#IdMedecin, #IdSpecialite)*
Specialite	*(IdSpecialite, Domaine)*
Vaccin	*(IdVaccin,NomVaccin, ContreIndication , Rappel)*
Avoir_Fait	*(DateVaccin,#IdVaccin ,#IdPatient)*
Avoir_eu	*(#IdAntécédent, #IdPatient)*
Donner	*(#IdRendezVous, #IdDiagnostic)*
Diagnostic	*(IdDiagnostic, DescriptionDiagnostic)*
Comporte	*(#IdDiagnostic,#IdMaladie)*
Maladie	*(IdMaladie, GraviteMaladie, NomMaladie)*
Antecedent	*(IdAntecedent, TypeAntecedent, DescriptionAntecedent)*
Patient	*(IdPatient, NomPatient, PrenomPatient,DateNaissance, Adresse, #IdRegime)*
Souscrire	*(#IdPatient, #IdMutuelle)*
Mutuelle	*(IdMutuelle, NomMutuelle, TauxRemboursementMutuelle,Forfait)*
Salle	*(IdSalle,EtageSalle, NumeroSalle)*
Regime_Obligatoire	*(IdRegime,PourcentageRembourse,TypeRegime)*

Normalisation

« Le but essentiel de la normalisation est d'éviter les anomalies transactionnelles pouvant découler d'une mauvaise modélisation des données et ainsi éviter un certain nombre de problèmes potentiels tels que les anomalies de lecture, les anomalies d'écriture, la redondance des données et la contre-performance. »

Wikipedia

- Vérifions que la base de données est en 3ème forme normale:

1 NF :

Tous les attributs de toutes les relations sont des valeurs atomiques donc nous sommes en 1 NF.

2 NF :

Dans chacune de nos relations, les attributs non clés ne dépendent pas d'une partie de la clé donc nous sommes bien en 2 NF.

3 NF :

Dans toutes nos relations, aucun attribut non clé ne dépend pas d'autres attributs n'appartenant pas à la clé, donc nous sommes bien en 3 NF.

De plus nous avons utilisé la **méthode Merise** pour développer notre base de données, nous sommes donc **forcément en 3 NF**.

En effet, pour vérifier que la base de données est en 3ème forme normale il faut que chaque relation, de la MLD, soit en 3ème forme normale.

Exemple (pour les entités : **rendez_vous** et **Examen**):

- **rendez_vous***(IdRendezVous, Motif, DateRendezVous,IdSalle,IdMedecin,IdPatient)*

Les dépendances fonctionnelles :

F : { IdRendezVous → Motif
IdRendezVous →DateRendezVous
IdRendezVous → IdSalle
IdRendezVous → IdMedecin
IdRendezVous → IdPatient
DateRendezVous,IdSalle,IdMedecin,IdPatient → IdRendezVous
}

On calcule la fermeture transitive de IdRendezVous (pour trouver la clé) :

$(IdRendezVous)^+$: IdRendezVous,Motif, DateRendezVous,IdSalle,IdMedecin,IdPatient

On retrouve bien tous les attributs de la relation donc **IdRendezVous est clé.**

Tous les dépendances fonctionnelles sont des valeurs atomiques donc nous sommes en 1 NF. Aucun attribut non clés ne dépendent pas d'une partie de la clé, donc nous sommes bien en 2 NF. Aucun attribut non clé ne dépend pas d'autres attributs non clé, donc nous sommes bien en **3 NF**.

- **Examen**(IdExamen, NomExamen ,DescriptionExamen)

Dépendances fonctionnelles

F :{ IdExamen → NomExamen
IdExamen →DescriptionExamen
}

On calcule la fermeture transitive de IdExamen (pour trouver la clé):

$(IdExamen)^+$: IdExamen, NomExamen, DescriptionExamen

On retrouve bien tous les attributs de la relation donc **IdExamen est clé.**

Tous les dépendances fonctionnelles sont des valeurs atomiques donc nous sommes en 1 NF. Aucun attribut non clés ne dépendent pas d'une partie de la clé, donc nous sommes bien en 2 NF. Aucun attribut non clé ne dépend pas d'autres attributs non clé, donc nous sommes bien en **3 NF**.

Requetés en SQL et Algèbre Relationnelle

Domaine de la spécialité d'un médecin

- **R1 :** Domaine de la spécialité du médecin qui a consulté monsieur X a une date donnée.

SQL

```
SELECT Domaine
FROM `patient` AS P,`rendez_vous` AS R,`specialite` AS S,`medecin` AS M,`specialite_medecin` AS SM
WHERE P.nomPatient='X'
AND   R.DateRendezVous>='2016-11-07 00:00:00'
AND   R.DateRendezVous<='2016-11-08 00:00:00'
AND   R.IdPatient=P.IdPatient
AND   M.IdMedecin=R.IdMedecin
AND   M.IdMedecin=SM.IdMedecin
AND   SM.IdSpecialite=S.IdSpecialite
```

Algèbre relationnelle

T1 : L'id ou les id de/des médecin(s) qui ont consulter le patient X à une date donnée.

$$T1: \pi_{IdMedecin} (\sigma_{NomPatient='X'}(Patient) \underset{IdPatient}{\bowtie} \sigma_{DateRendezVous='date'}(rendez\_vous))$$

$$R1: \pi_{Domaine}(Specialite \underset{IdSpecialite}{\bowtie} (Avoir \underset{IdMedecin}{\bowtie} T1)))$$

Planning des médecins

- **R2 :** Planning des médecins **(ce qui sera affiché sera le planning des rendez-vous qui ne sont pas encore passé c'est-à-dire les rendez-vous à venir)**

SQL

```
SELECT NomMedecin,PrenomMedecin,NomPatient,PrenomPatient,NumeroSalle,DateRendezVous
FROM `medecin` AS M, `rendez_vous` AS R, `patient` AS P, `salle` AS S
WHERE R.DateRendezVous >= CURRENT_TIMESTAMP()
AND   M.IdMedecin=R.IdMedecin
AND   R.IdPatient=P.IdPatient
AND   R.IdSalle=S.IdSalle
```

Algèbre relationnelle

$$\pi_{\substack{\text{DateRendezVous}\\ \text{NomMedecin}\\ \text{PrenomMedecin}\\ \text{NomPatient}\\ \text{PrenomParient}\\ \text{NumeroSalle}}}(((\text{Medecin} \bowtie_{\text{IdMedecin}} \sigma_{\text{DateRendezVous}>=\text{'date\_courrante'}}(\text{rendez\_vous})) \bowtie_{\text{IdPatient}} \text{Patient}) \bowtie_{\text{IdSalle}} \text{Salle})$$

Dernier Vaccin

- **R3 : Dernier Vaccin qu'a subi monsieur X**

SQL

```
SELECT NomVaccin,MAX(DateVaccin)
FROM `vaccins_patient` AS VP, `patient` AS P ,`vaccin` AS V
WHERE VP.IdPatient=P.IdPatient
AND   P.NomPatient='Bred'
```

Algèbre relationnelle

$$\pi_{\substack{\text{DateVaccin}\\ \text{NomVaccin}}}(\sigma_{\text{Max(DateVaccin)}}((\text{Avoir\_fait} \bowtie_{\text{IdPatient}} \text{Patient}) \bowtie_{\text{IdVaccin}} \text{Vaccin}))$$

Comptage

- **R4 : Nombre des maladies qu'ont eu les patients**

SQL

```
SELECT NomPatient,PrenomPatient,COUNT(IdMaladie)
FROM `patient` AS P, `rendez_vous` AS R, `diagnostic` AS D, `liste_maladies` AS LM
WHERE P.IdPatient=R.IdPatient
AND R.IdDiagnostic=D.IdDiagnostic
AND LM.IdDiagnostic=D.IdDiagnostic
GROUP BY P.IdPatient
```

Algèbre relationnelle

Impossible en algèbre relationnelle

Division

- **R5 : Donner les Patients qui ont déjà vu tous les médecins**

SQL

```
SELECT NomPatient,PrenomPatient
From `patient` AS P
WHERE NOT EXISTS (
        SELECT IdMedecin
        FROM `medecin` AS M
        WHERE NOT EXISTS (
            SELECT *
            FROM `rendez_vous` AS R
              WHERE R.IdMedecin=M.IdMedecin
              AND   R.IdPatient=P.IdPatient
              AND   R.DateRendezVous < CURRENT_TIMESTAMP()
            )
    )
```

Algèbre relationnelle

T1 : Les **IdPatient** des patients qui ont déjà vu tous les médecins.

T1 : $(\pi_{\substack{\text{IdMedecin} \\ \text{IdPatient}}}(\text{rendez\_vous})) / ((\pi_{\text{IdMedecin}}(\text{Medecin}))$

R5 : $\pi_{\substack{\text{NomPatient} \\ \text{PrenomPatient}}} (\text{T1} \bowtie_{\text{IdPatient}} \text{Patient})$

***La division a bien un sens car le schéma de diviseur *(IdMedecin)* est inclus dans le schéma de dividende *(IdMedecin,IdPatient)*

Dossier patient

➢ R6 : Donner le dossier d'un patient X

***Note :** Cette requête retournera le résultat voulu que si toutes les tables contiennent au moins une information concernant le patient X, dans le cas contraire on aura un résultat nul. En effet, pour obtenir le dossier d'un patient on va faire des jointures entre toutes les tables et donc, on risque d'obtenir un résultat vide. Par exemple, en faisant une **restriction** sur le patient X et une **jointure** entre **Patient**(IdPatient,NomPatient…) et **rendez_vous**(IdRendezVous,IdPatient…) sur **IdPatient** , on peut avoir un résultat nul si le patient X n'a jamais eu aucun rendez-vous, et donc les jointures qui vont suivre vont donner aussi un résultat nul.

On peut régler ce problème par programmation, en faisant **des requetés à part** pour chaque information (les vaccins, les maladies, etc.). En faisant cela on pourra savoir quelle partie de la requête donne un résultat nul et donc on pourra en déduire des informations sur le dossier du patient (exemple : si la requête qui permet de récupérer tous les vaccins du patient retourne nul, ça veut dire que le patient n'a pas fait aucun vaccin). On aura besoin de découper notre requête en 15 sous requêtes (voir page 21).

SQL

```
SELECT NomPatient,PrenomPatient,DateNaissance,Adresse,PourcentageRembourcer,NomRegime,
    NomMutuelle,TauxRemboursementMutuelle,Forfait,TypeAntecedent, DescriptionAntecedant,
    DateVaccin,NomVaccin,ContreIndication,Rappel,Motif,DateRendezVous,
    EtageSalle,NumeroSalle,DescriptionDiagnostic,GraviteMaladie,NomMaladie,NomMedecin,
    PrenomMedecin,Anciennte,Domaine,NomAssistant,PrenomAssistant, DescriptionTraitement,
    DureeTraitement,NomMedicament,QuantiteMedicament,TypeMedicament,Renouvelement,
    NomExamen,DescriptionExamen
FROM `patient` AS P, `vaccins_patient` AS VP,`vaccin` AS V, `antecedent_patient` AS AP,
        `antecedent` AS A,`regime_obligatoire` AS R, `souscrire` AS S, `mutuelle` AS M,
    `rendez_vous` AS R_V,`salle` AS SALLE,`diagnostic` AS DIAG,`liste_maladies` AS L_MALAD,
    `maladie` AS MALAD, `medecin` AS MED, `specialite_medecin`AS L_SPEC, `specialite` AS SPEC,
    `engager`AS ENGAGE, `assistant`AS ASSIS, `ordonnance` AS ORD, `liste_examens` AS L_EXAM,
    `examen` AS EXAM, `prescrire` AS PRESCRIRE, `medicaments` AS MEDICAMENTS,
    `liste_traitements` AS L_TRAIT,`traitement` AS TRAIT
WHERE P.NomPatient='X'
AND   P.IdPatient=VP.IdPatient
AND   VP.IdVaccin=V.IdVaccin
AND   P.IdPatient=AP.IdPatient
AND   AP.IdAntecedent=A.IdAntecedent
AND   P.IdRegime=R.IdRegime
AND   P.IdPatient=S.IdPatient
AND   S.IdMutuelle=M.IdMutuelle
```

AND P.IdPatient=R_V.IdPatient
AND R_V.IdSalle=SALLE.IdSalle
AND R_V.IdDiagnostic=DIAG.IdDiagnostic
AND DIAG.IdDiagnostic=L_MALAD.IdDiagnostic
AND L_MALAD.IdMaladie=MALAD.IdMaladie
AND R_V.IdMedecin=MED.IdMedecin
AND MED.IdMedecin=L_SPEC.IdMedecin
AND L_SPEC.IdSpecialite=SPEC.IdSpecialite
AND MED.IdMedecin=ENGAGE.IdMedecin
AND ENGAGE.IdAssistant=ASSIS.IdAssistant
AND R_V.IdRendezVous=ORD.IdRendezVous
AND ORD.IdOrdonnance=L_EXAM.IdOrdonnance
AND L_EXAM.IdExamen=EXAM.IdExamen
AND ORD.IdOrdonnance=PRESCRIRE.IdOrdonnance
AND PRESCRIRE.IdMedicament=MEDICAMENTS.IdMedicament
AND ORD.IdOrdonnance=L_TRAIT.IdOrdonnance
AND L_TRAIT.IdTraitement=TRAIT.IdTraitement

Algèbre relationnelle

***On va récupérer au fur et à mesure les différentes données pour constituer le dossier du patient (en faisant plusieurs sous-requetés)

T1 : $(\text{Patient} \underset{\text{IdPatient}}{\bowtie} \text{Avoir\_Fait}) \underset{\text{IdVaccin}}{\bowtie} \text{Vaccin})$ → Vaccins

T2 : $(\text{T1} \underset{\text{IdPatient}}{\bowtie} \text{Avoir\_eu}) \underset{\text{IdAntecedent}}{\bowtie} \text{Antecedent})$ → + Antecedents

T3 : $(\text{T2} \underset{\text{IdRegime}}{\bowtie} \text{Regime\_Obligatoire})$ → + Regime obligatoire

T4 : $(\text{T3} \underset{\text{IdPatient}}{\bowtie} \text{Souscrire}) \underset{\text{IdMutuelle}}{\bowtie} \text{Mutuelle})$ → + Mutuelle

T5 : (T4 $\bowtie_{\text{IdPatient}}$ Rendez_Vous) + Rendez_Vous

T6 : (T5 $\bowtie_{\text{IdSalle}}$ Salle) + Salles

T7 : (T6 $\bowtie_{\text{IdDiagnostic}}$ Diagnostic) + Diagnostics

T8 : (T7 $\bowtie_{\text{IdDiagnostic}}$ Comporte) $\bowtie_{\text{IdMaladie}}$ Maladie) + Maladies

T9 : (T8 $\bowtie_{\text{IdMedecin}}$ Medecin) + Medecins

T10 : (T9 $\bowtie_{\text{IdMedecin}}$ Avoir) $\bowtie_{\text{IdSpecialite}}$ Specialite) + Specialités

T11 : (T10 $\bowtie_{\text{IdMedecin}}$ Engager) $\bowtie_{\text{IdAssistant}}$ Assistant) + Assistants

T12 : (T11 $\bowtie_{\text{IdOrdonnance}}$ Ordonnance) + Ordonnances

T13 : (T12 $\bowtie_{\text{IdOrdonnance}}$ Ordonnance) $\bowtie_{\text{IdExamen}}$ Examen) + Examens

T14 : (T13 $\bowtie_{\text{IdOrdonnance}}$ Prescrire) $\bowtie_{\text{IdMedicament}}$ Medicament) + Medicaments

T15 : (T14 $\bowtie_{\text{IdOrdonnance}}$ Offrir) $\bowtie_{\text{IdTraitement}}$ Traitement) + Traitements

Conclusion

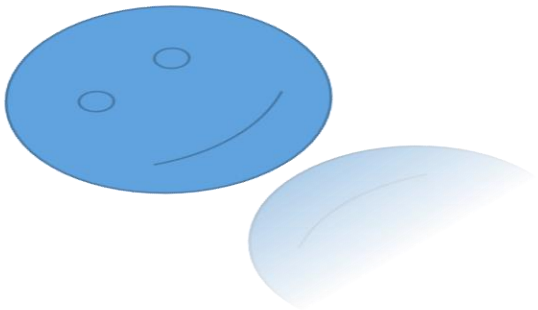
Nous avons donc notre base de données permettant la gestion d'un cabinet médical, que nous avons développé grâce à la méthode merise.
Nous n'avons cependant pas traité la partie sécurité.

Il aurait fallu donner, par exemple, la possibilité à la secrétaire de voir et de modifier tout ce qui concerne le planning mais lui bloquer la possibilité de modifier le dossier patient et même lui bloquer la vue sur certaines informations de celui-ci (par exemple la secrétaire peut voir et modifier l'adresse du patient mais ne peut pas avoir accès, du tout, aux informations concernant les antécédent familiaux).

Cependant il y a un point litigieux en ce qui concerne la sécurité. Le dossier patient contient des informations confidentielles, et l'administrateur a la vision sur tout, or il ne devrait pas. Certaines informations ne devraient être connues que par le médecin et par le patient. Mais à ce jour il est impossible de bloquer la vue ou l'accès à l'administrateur car c'est lui qui régit toute l'organisation de la base de données, donc on supposera que l'administrateur sait garder un secret médical.

Donc on aurait eu les droits suivants :

	Droit lecture	**Droit écriture**
Administrateur	Toutes les tables	Toutes les tables
Médecin	Toutes les tables	Toutes les tables sauf celles concernant les informations non-médicales
Secrétaire	Toutes les tables concernant les informations de base (adresse, mutuelle, etc.)	Toutes les tables concernant les informations de base (adresse, mutuelle, etc.)



Webographie

- http://stephanie.laporte.pagesperso-orange.fr/Pdf/introMCD.pdf
- http://www.developpez.net/forums/d793529/general-developpement/alm/modelisation/schema/definition-modele-logique-donnees/
- http://stephanie.laporte.pagesperso-orange.fr/Pdf/MCT_MCTA.pdf
- http://www.hopital.fr/Le-dico-medical/Les-specialites-medicales
- https://www.mesvaccins.net/web/vaccines
- https://fr.wikipedia.org/wiki/Normalisation
- http://www.commentcamarche.net/contents/661-merise-modele-logique-des-donnees
- http://canadiensensante.gc.ca/publications/healthy-living-vie-saine/2-canadian-immunization-guide-canadien-immunisation/index-fra.php?page=3
- http://www.passeportsante.net/problemes-et-maladies-p69/maladies-infectieuses-92
- http://www.commentcamarche.net/contents/659-merise-modele-conceptuel-des-donnees

Annexe

Script de la création de la base de données

```
#------------------------------------------------------------
#        Script MySQL.
#------------------------------------------------------------


#------------------------------------------------------------
# Table: Patient
#------------------------------------------------------------

CREATE TABLE Patient(
        IdPatient     int (11) Auto_increment  NOT NULL ,
        NomPatient    Varchar (25) NOT NULL ,
        PrenomPatient Varchar (25) NOT NULL ,
        DateNaissance Datetime NOT NULL ,
        Adresse       Varchar (250) NOT NULL ,
        IdRegime      Int NOT NULL ,
        PRIMARY KEY (IdPatient )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Regime_Obligatoire
#------------------------------------------------------------

CREATE TABLE Regime_Obligatoire(
        IdRegime             int (11) Auto_increment  NOT NULL ,
        PourcentageRembourcer Int ,
        NomRegime            Varchar (25) NOT NULL ,
        PRIMARY KEY (IdRegime )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Mutuelle
#------------------------------------------------------------

CREATE TABLE Mutuelle(
        IdMutuelle               int (11) Auto_increment  NOT NULL ,
        NomMutuelle              Varchar (25) NOT NULL ,
        TauxRemboursementMutuelle Int NOT NULL ,
        Forfait                  Varchar (250) NOT NULL ,
        PRIMARY KEY (IdMutuelle )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: Salle
#------------------------------------------------------------

CREATE TABLE Salle(
        IdSalle     int (11) Auto_increment  NOT NULL ,
        EtageSalle  Int NOT NULL ,
        NumeroSalle Int NOT NULL ,
        PRIMARY KEY (IdSalle )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: rendez_vous
#------------------------------------------------------------

CREATE TABLE rendez_vous(
        IdRendezVous   int (11) Auto_increment  NOT NULL ,
        Motif          Varchar (25) NOT NULL ,
        DateRendezVous Datetime NOT NULL ,
        IdSalle        Int NOT NULL ,
        IdMedecin      Int NOT NULL ,
        IdPatient      Int NOT NULL ,
        IdDiagnostic   Int NOT NULL ,
        PRIMARY KEY (IdRendezVous )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Medecin
#------------------------------------------------------------

CREATE TABLE Medecin(
        IdMedecin     int (11) Auto_increment  NOT NULL ,
        NomMedecin    Varchar (25) NOT NULL ,
        PrenomMedecin Varchar (25) NOT NULL ,
        Anciennte     Int NOT NULL ,
        PRIMARY KEY (IdMedecin )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Specialite
#------------------------------------------------------------

CREATE TABLE Specialite(
        IdSpecialite int (11) Auto_increment  NOT NULL ,
        Domaine      Varchar (25) NOT NULL ,
        PRIMARY KEY (IdSpecialite )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: Ordonnance
#------------------------------------------------------------

CREATE TABLE Ordonnance(
        IdOrdonnance int (11) Auto_increment  NOT NULL ,
        IdRendezVous Int NOT NULL ,
        PRIMARY KEY (IdOrdonnance )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Examen
#------------------------------------------------------------

CREATE TABLE Examen(
        IdExamen          int (11) Auto_increment  NOT NULL ,
        NomExamen         Varchar (25) NOT NULL ,
        DescriptionExamen Varchar (250) NOT NULL ,
        PRIMARY KEY (IdExamen )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Medicaments
#------------------------------------------------------------

CREATE TABLE Medicaments(
        IdMedicament       int (11) Auto_increment  NOT NULL ,
        NomMedicament      Varchar (25) NOT NULL ,
        QuantiteMedicament Varchar (100) NOT NULL ,
        TypeMedicament     Varchar (25) NOT NULL ,
        Renouvelement      Int ,
        PRIMARY KEY (IdMedicament )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Assistant
#------------------------------------------------------------

CREATE TABLE Assistant(
        IdAssistant     int (11) Auto_increment  NOT NULL ,
        NomAssistant    Varchar (25) NOT NULL ,
        PrenomAssistant Varchar (25) ,
        PRIMARY KEY (IdAssistant )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: Vaccin
#------------------------------------------------------------

CREATE TABLE Vaccin(
        IdVaccin         int (11) Auto_increment  NOT NULL ,
        NomVaccin        Varchar (25) NOT NULL ,
        ContreIndication Varchar (250) NOT NULL ,
        PRIMARY KEY (IdVaccin )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Antecedent
#------------------------------------------------------------

CREATE TABLE Antecedent(
        IdAntecedent          int (11) Auto_increment  NOT NULL ,
        TypeAntecedent        Varchar (25) NOT NULL ,
        DescriptionAntecedant Varchar (250) NOT NULL ,
        PRIMARY KEY (IdAntecedent )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Diagnostic
#------------------------------------------------------------

CREATE TABLE Diagnostic(
        IdDiagnostic          int (11) Auto_increment  NOT NULL ,
        DescriptionDiagnostic Varchar (250) NOT NULL ,
        PRIMARY KEY (IdDiagnostic )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Maladie
#------------------------------------------------------------

CREATE TABLE Maladie(
        IdMaladie      int (11) Auto_increment  NOT NULL ,
        GraviteMaladie Varchar (25) NOT NULL ,
        NomMaladie     Varchar (25) NOT NULL ,
        PRIMARY KEY (IdMaladie )
)ENGINE=InnoDB;
```

```
#------------------------------------------------------------
# Table: Traitement
#------------------------------------------------------------

CREATE TABLE Traitement(
        IdTraitement          int (11) Auto_increment  NOT NULL ,
        DescriptionTraitement Varchar (250) NOT NULL ,
        DureeTraitement       Varchar (250) NOT NULL ,
        PRIMARY KEY (IdTraitement )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Souscrire
#------------------------------------------------------------

CREATE TABLE Souscrire(
        IdPatient  Int NOT NULL ,
        IdMutuelle Int NOT NULL ,
        PRIMARY KEY (IdPatient ,IdMutuelle )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Avoir
#------------------------------------------------------------

CREATE TABLE Avoir(
        IdMedecin    Int NOT NULL ,
        IdSpecialite Int NOT NULL ,
        PRIMARY KEY (IdMedecin ,IdSpecialite )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Prescrire
#------------------------------------------------------------

CREATE TABLE Prescrire(
        IdOrdonnance Int NOT NULL ,
        IdMedicament Int NOT NULL ,
        PRIMARY KEY (IdOrdonnance ,IdMedicament )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Ordonner
#------------------------------------------------------------

CREATE TABLE Ordonner(
        IdOrdonnance Int NOT NULL ,
        IdExamen     Int NOT NULL ,
```

```
    PRIMARY KEY (IdOrdonnance ,IdExamen )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Engager
#------------------------------------------------------------

CREATE TABLE Engager(
    IdMedecin    Int NOT NULL ,
    IdAssistant Int NOT NULL ,
    PRIMARY KEY (IdMedecin ,IdAssistant )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Avoir_Fait
#------------------------------------------------------------

CREATE TABLE Avoir_Fait(
    DateVaccin Date NOT NULL ,
    DateRapel  Date ,
    IdVaccin    Int NOT NULL ,
    IdPatient  Int NOT NULL ,
    PRIMARY KEY (IdVaccin ,IdPatient )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Avoir_eu
#------------------------------------------------------------

CREATE TABLE Avoir_eu(
    DateAntecedent Date ,
    IdAntecedent   Int NOT NULL ,
    IdPatient      Int NOT NULL ,
    PRIMARY KEY (IdAntecedent ,IdPatient )
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: Comporte
#------------------------------------------------------------

CREATE TABLE Comporte(
    IdDiagnostic Int NOT NULL ,
    IdMaladie    Int NOT NULL ,
    PRIMARY KEY (IdDiagnostic ,IdMaladie )
)ENGINE=InnoDB;


#------------------------------------------------------------
```

```
# Table: Offrir
#------------------------------------------------------------

CREATE TABLE Offrir(
        IdTraitement Int NOT NULL ,
        IdOrdonnance Int NOT NULL ,
        PRIMARY KEY (IdTraitement ,IdOrdonnance )
)ENGINE=InnoDB;

ALTER TABLE Patient ADD CONSTRAINT FK_Patient_IdRegime FOREIGN KEY (IdRegime)
REFERENCES Regime_Obligatoire(IdRegime);
ALTER TABLE rendez_vous ADD CONSTRAINT FK_rendez_vous_IdSalle FOREIGN KEY (IdSalle)
REFERENCES Salle(IdSalle);
ALTER TABLE rendez_vous ADD CONSTRAINT FK_rendez_vous_IdMedecin FOREIGN KEY
(IdMedecin) REFERENCES Medecin(IdMedecin);
ALTER TABLE rendez_vous ADD CONSTRAINT FK_rendez_vous_IdPatient FOREIGN KEY (IdPatient)
REFERENCES Patient(IdPatient);
ALTER TABLE rendez_vous ADD CONSTRAINT FK_rendez_vous_IdDiagnostic FOREIGN KEY
(IdDiagnostic) REFERENCES Diagnostic(IdDiagnostic);
ALTER TABLE Ordonnance ADD CONSTRAINT FK_Ordonnance_IdRendezVous FOREIGN KEY
(IdRendezVous) REFERENCES rendez_vous(IdRendezVous);
ALTER TABLE Souscrire ADD CONSTRAINT FK_Souscrire_IdPatient FOREIGN KEY (IdPatient)
REFERENCES Patient(IdPatient);
ALTER TABLE Souscrire ADD CONSTRAINT FK_Souscrire_IdMutuelle FOREIGN KEY (IdMutuelle)
REFERENCES Mutuelle(IdMutuelle);
ALTER TABLE Avoir ADD CONSTRAINT FK_Avoir_IdMedecin FOREIGN KEY (IdMedecin)
REFERENCES Medecin(IdMedecin);
ALTER TABLE Avoir ADD CONSTRAINT FK_Avoir_IdSpecialite FOREIGN KEY (IdSpecialite)
REFERENCES Specialite(IdSpecialite);
ALTER TABLE Prescrire ADD CONSTRAINT FK_Prescrire_IdOrdonnance FOREIGN KEY
(IdOrdonnance) REFERENCES Ordonnance(IdOrdonnance);
ALTER TABLE Prescrire ADD CONSTRAINT FK_Prescrire_IdMedicament FOREIGN KEY
(IdMedicament) REFERENCES Medicaments(IdMedicament);
ALTER TABLE Ordonner ADD CONSTRAINT FK_Ordonner_IdOrdonnance FOREIGN KEY
(IdOrdonnance) REFERENCES Ordonnance(IdOrdonnance);
ALTER TABLE Ordonner ADD CONSTRAINT FK_Ordonner_IdExamen FOREIGN KEY (IdExamen)
REFERENCES Examen(IdExamen);
ALTER TABLE Engager ADD CONSTRAINT FK_Engager_IdMedecin FOREIGN KEY (IdMedecin)
REFERENCES Medecin(IdMedecin);
ALTER TABLE Engager ADD CONSTRAINT FK_Engager_IdAssistant FOREIGN KEY (IdAssistant)
REFERENCES Assistant(IdAssistant);
ALTER TABLE Avoir_Fait ADD CONSTRAINT FK_Avoir_Fait_IdVaccin FOREIGN KEY (IdVaccin)
REFERENCES Vaccin(IdVaccin);
ALTER TABLE Avoir_Fait ADD CONSTRAINT FK_Avoir_Fait_IdPatient FOREIGN KEY (IdPatient)
REFERENCES Patient(IdPatient);
ALTER TABLE Avoir_eu ADD CONSTRAINT FK_Avoir_eu_IdAntecedent FOREIGN KEY
(IdAntecedent) REFERENCES Antecedent(IdAntecedent);
ALTER TABLE Avoir_eu ADD CONSTRAINT FK_Avoir_eu_IdPatient FOREIGN KEY (IdPatient)
REFERENCES Patient(IdPatient);
ALTER TABLE Comporte ADD CONSTRAINT FK_Comporte_IdDiagnostic FOREIGN KEY
(IdDiagnostic) REFERENCES Diagnostic(IdDiagnostic);
ALTER TABLE Comporte ADD CONSTRAINT FK_Comporte_IdMaladie FOREIGN KEY (IdMaladie)
```

```
REFERENCES Maladie(IdMaladie);
ALTER TABLE Offrir ADD CONSTRAINT FK_Offrir_IdTraitement FOREIGN KEY (IdTraitement)
REFERENCES Traitement(IdTraitement);
ALTER TABLE Offrir ADD CONSTRAINT FK_Offrir_IdOrdonnance FOREIGN KEY (IdOrdonnance)
REFERENCES Ordonnance(IdOrdonnance);
```