Rapport du projet de bases de données avancées

Juste Prescription des Médicaments

Description du contexte analysé

L'application développée est un gestionnaire de prescriptions. Celle-ci repose sur l'exploitation d'une base de données.

Tout d'abord, on distingue deux types d'utilisateurs. Il y a l'utilisateur médecin et l'utilisateur patient. Un patient reçoit des médicaments suite à un ou plusieurs traitements prescrits par des médecins pour des observations effectuées lors de consultations.

Un traitement concerne donc un patient et un médecin dont celui-ci inclut une ou plusieurs recommandations accompagné de médicament(s) si nécessaire.

Une consultation intervient entre un médecin et un patient.

Chaque maladie est identifiée par un ou des symptômes qui peuvent être similaires selon les maladies. Afin de soigner une maladie, il y a un médicament qui est fourni et plusieurs médicaments peuvent soigner la même maladie (concurrence sur le marché).

Le médicament quant à lui est développé par un ou plusieurs médecins qui travaillent pour un laboratoire pharmaceutique. Dans un médicament, on trouve une ou des indications, une ou des contre-indications, une ou des effets indésirables et une ou des substances actives.

Dans chaque substance active, il y a une ou plusieurs classes chimiques et pharmacologiques.

La ou les classes chimiques et pharmacologiques ainsi que la ou les substances actives peuvent générer un ou des effets indésirables.

Règle de gestions

Pour chaque table, il y a une contrainte d'entité (contrainte de relation) avec l'attribut **id**.

Pour chaque table, il y a une contrainte de domaine car chaque champ doit prendre la valeur

Les tables d'associations ou les tables référençant une autre table ont une contrainte référentielle.

Modèle en SQL2

Schéma du Modèle Entité-Association (Entity-Relationship Model)

Pour le visualiser correctement, veuillez zoomer ou bien consulter le fichier SQL2_MEA fourni en annexe.

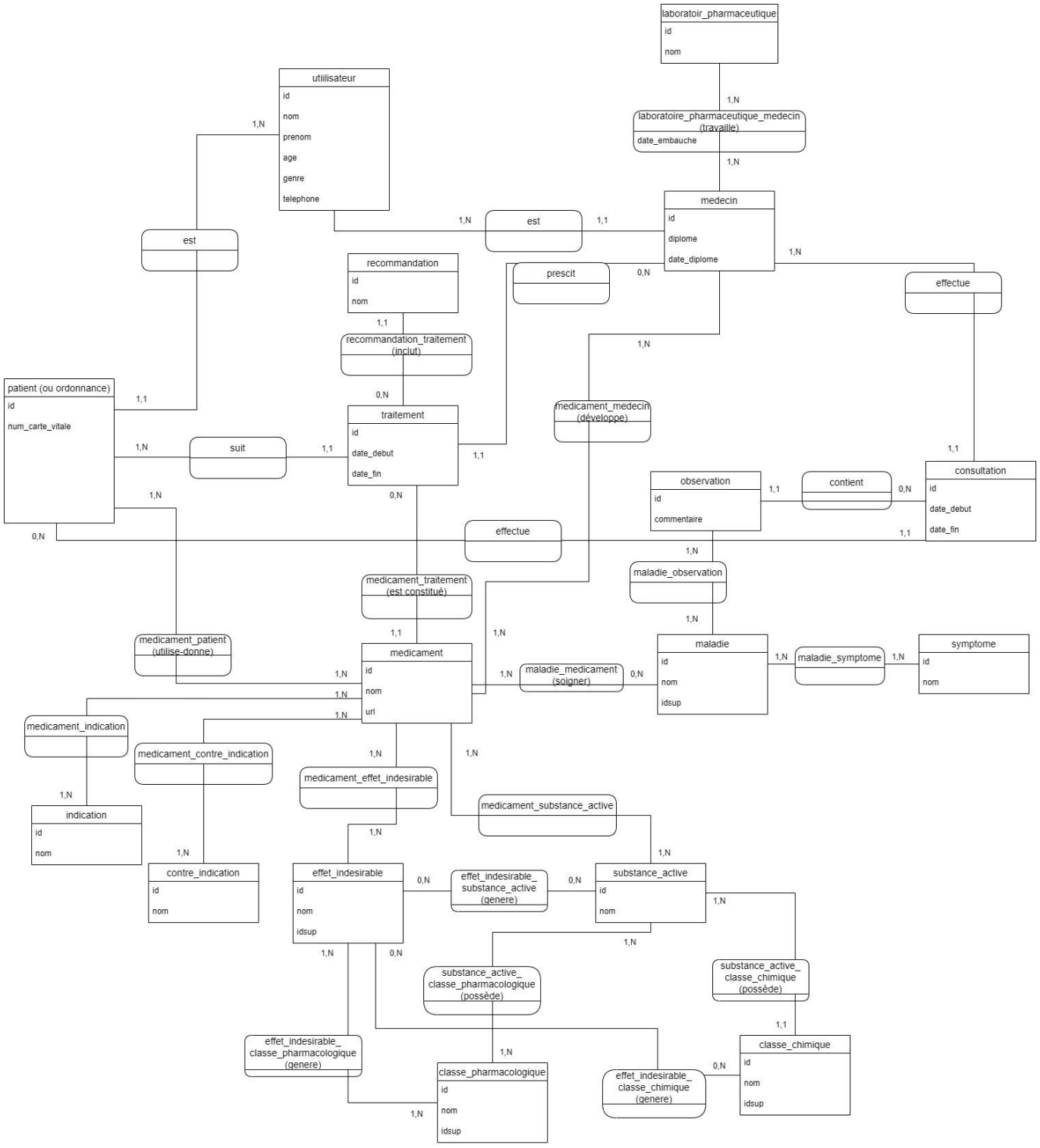
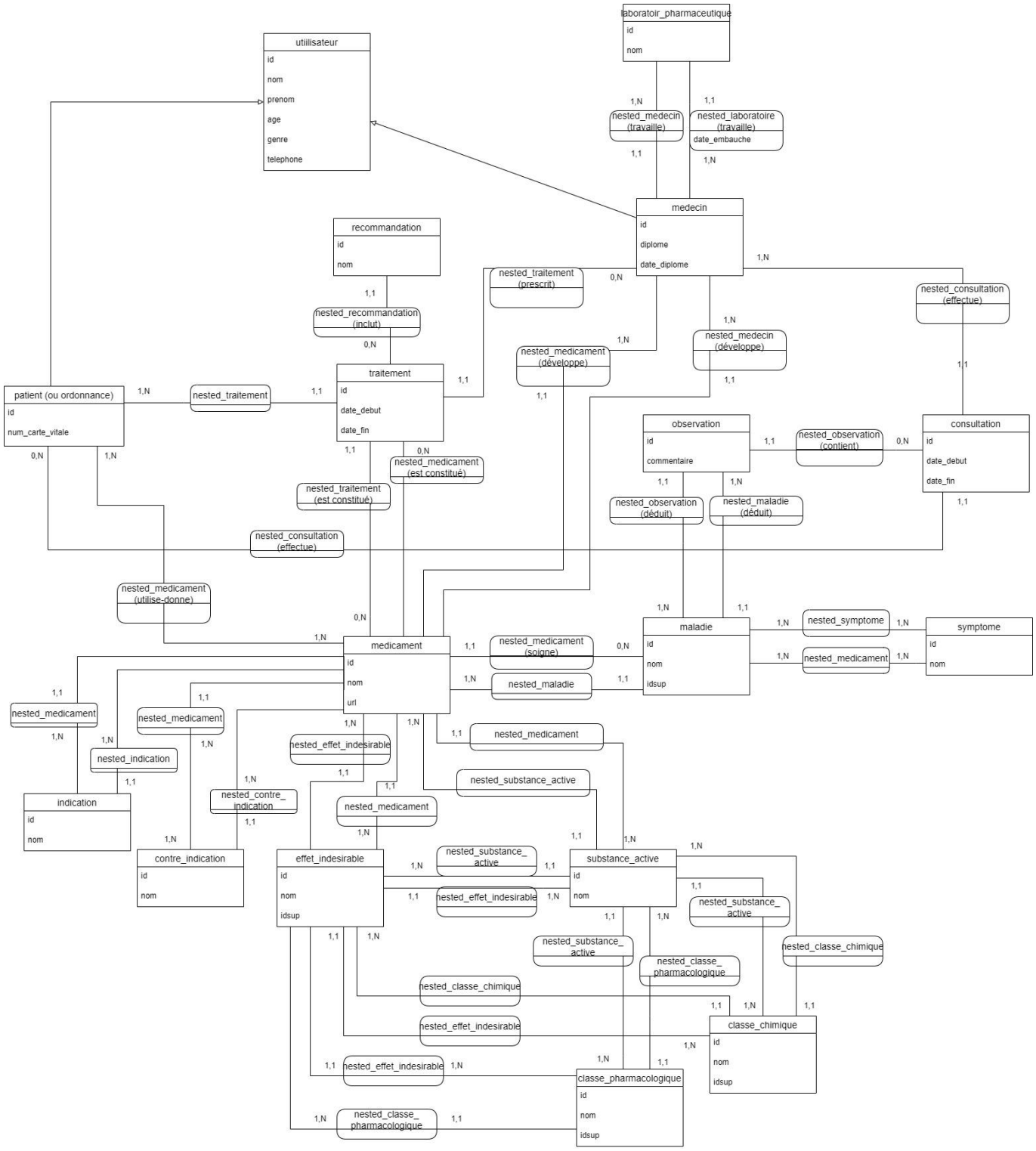


Schéma relationnel

utilisateur(id, nom, prenom, age, genre, telephone)
medecin(id, diplome, date_diplome, #utilisateur_id)
patient(id, num_carte_vitale, #utilisateur_id)
medicament(id, nom, url)
medicament_medecin(#medicament_id, #medecin_id)
medicament_traitement(#medicament_id, #traitement_id)
medicament_patient(#medicament_id, #patient_id)
traitement(id, date_debut, date_fin, #medecin_id, #patient_id)
recommandation(id, nom)
recommandation_traitement(#recommandation_id, #traitement_id)
laboratoire_pharmaceutique(id, nom)
laboratoire_pharmaceutique_medecin(#laboratoire_pharmaceutique_id, #medecin_id, date_embauche)
consultation(id, date_debut, date_fin, #medecin_id, #patient_id)
observation(id, commentaire, consultation_id)
symptome(id, nom)
maladie(id, nom, idsup)
maladie_medicament(#maladie_id, #medicament_id)
maladie_observation(#maladie_id, #observation_id)
maladie_symptome(#maladie_id, #symptome_id)
indication(id, nom)
contre_indication(id, nom)
effet_indesirable(id, nom, idsup)
substance_active(id, nom)
classe_chimique(id, nom, idsup)
classe_pharmacologique(id, nom, idsup)
medicament_indication(#medicament_id, #indication_id)
medicament_contre_indication(#medicament_id, #contre_indication_id)
medicament_effet_indesirable(#medicament_id, #effet_indesirable_id)
medicament_substance_active(#medicament_id, #substance_active_id)
effet_indesirable_classe_chimique(#effet_indesirable_id, #classe_chimique_id)
effet_indesirable_classe_pharmacologique(#effet_indesirable_id, #classe_pharmacologique_id)
substance_active_classe_chimique(#substance_active_id, #classe_chimique_id)
substance_active_classe_pharmacologique(#substance_active_id, #classe_pharmacologique_id)

Modèle en SQL3

Schéma du Modèle Entité-Association Étendu (Extended Entity-Relationship Model)



Dans le modèle SQL2, le type d'association N-N est modélisé par une table d'association.

Pour transformer ce type d'association en SQL3, vu qu' il n'y a pas de standardisation, j'ai décidé de modéliser cela par un découpage en type d'association 1-N avec une table imbriqué qui est placé dans la table ayant la cardinalité N.

Par exemple, dans le but d'indiquer qu' un médicament peut appartenir à plusieurs traitements et un traitement peut contenir plusieurs médicaments, la table médicament possède

une table imbriqué **nested_traitement** et la table traitement possède une table imbriqué **nested_medicament**.

Les types utilisées

Ci-dessous, ceux sont les types modélisant un objet.

type_utilisateur(id, nom, prenom, age, genre, telephone)
type_medecin(id, diplome, date_diplome) **hérite de type_utilisateur**
type_patient(id, num_carte_vitale) **hérite de type_utilisateur**
type_medicament(id, nom, url)
type_traitement(id, date_debut, date_fin)
type_consultation(id, date_debut, date_fin)
type_recommandation(id, nom)
type_laboratoire_pharmaceutique(id, nom)
type_observation(id, commentaire)
type_symptome(id, nom)
type_maladie(id, nom, idsup)
type_indication(id, nom)
type_contre_indication(id, nom)
type_effet_indesirable(id, nom, idsup)
type_substance_active(id, nom)
type_classe_chimique(id, nom, idsup)
type_classe_pharmacologique(id, nom, idsup)

Ci-dessous, chaque type modélise une table imbriquée.
Une table imbriquée possède des enregistrements de type de l'objet.
Par exemple, la table imbriqué medecin (type_nested_table_medecin) peut avoir des enregistrements dont chaque enregistrement est un type_medecin.

type_nested_table_medecin(type_medecin)
type_nested_table_patient(type_patient)
type_nested_table_medicament(type_medicament)
type_nested_table_traitement(type_traitement)
type_nested_table_consultation(type_consultation)
type_nested_table_recommandation(type_recommandation)
type_nested_table_laboratoire_pharmaceutique(date_embauche, type_laboratoire_pharmaceutique)
type_nested_table_observation(type_observation)
type_nested_table_symptome(type_symptome)
type_nested_table_maladie(type_maladie)
type_nested_table_indication(type_indication)
type_nested_table_contre_indication(type_contre_indication)
type_nested_table_effet_indesirable(type_effet_indesirable)
type_nested_table_substance_active(type_substance_active)
type_nested_table_classe_chimique(type_classe_chimique)
type_nested_table_classe_pharmacologique(type_classe_pharmacologique)

Ci-dessous, ceux sont les types modélisant une table.
Dans ces types, il y a le type d'entité de la table puis les types des tables imbriquées.
Par exemple, pour le type de la table medecin(type_table_medecin), il y a un type medecin qui est l'objet medecin ainsi que les tables imbriqués de consultations effectuées, de laboratoires pharmaceutiques pour lequel il travaille ou a travaillé et les traitements prescrits.

type_table_medecin(type_medecin, type_nested_table_consultation, type_nested_table_laboratoire_pharmaceutique, type_nested_table_traitement)
type_table_patient(type_patient, type_nested_table_consultation, type_nested_table_medicament, type_nested_table_traitement)
type_table_traitement(type_traitement, type_nested_table_medicament, type_nested_table_recommandation)
type_table_consultation(type_consultation, type_nested_table_observation)
type_table_laboratoire_pharmaceutique(laboratoire_pharmaceutique, type_nested_table_medecin)
type_table_observation(type_observation, type_nested_table_maladie)
type_table_maladie(type_maladie, type_nested_table_medicament, type_nested_table_observation , type_nested_table_symptome)
type_table_symptome(type_symptome, type_nested_table_maladie)
type_table_medicament(type_medicament, type_nested_table_patient, type_nested_table_traitement, type_nested_table_medecin, type_nested_table_maladie, type_nested_table_substance_active, type_nested_table_effet_indesirable, type_nested_table_contre_indication, type_nested_table_indication)
type_table_indication(type_indication, type_nested_table_medicament)
type_nested_table_contre_indication(type_contre_indication, type_nested_table_medicament)
type_table_effet_indesirable(type_effet_indesirable, type_nested_table_medicament, type_nested_table_substance_active, type_nested_table_classe_chimique, type_nested_table_classe_pharmacologique)
type_substance_active(type_substance_active, type_nested_table_medicament, type_nested_table_effet_indesirable, type_nested_table_classe_chimique, type_nested_table_classe_pharmacologique)
type_table_classe_chimique(type_classe_chimique, type_nested_table_effet_indesirable, type_nested_table_substance_active)
type_table_classe_pharmacologique(type_classe_pharmacologique, type_nested_table_effet_indesirable, type_nested_table_substance_active)

Schéma relationnel-objet

Tables imbriquées

nested_consultation(id, date_debut, date_fin)
nested_laboratoire_pharmaceutique(id, nom)
nested_traitement(id, date_debut, date_fin)

nested_medicament(id, nom, url)
nested_recommandation(id, nom)
nested_medecin(id, nom, prenom, age, genre, telephone)
nested_observation(id, commentaire)
nested_symptome(id, nom)
nested_maladie(id, nom, idsup)
nested_patient(id, nom, prenom, age, genre, telephone)
nested_substance_active(d, nom)
nested_effet_indesirable(id, nom, idsup)
nested_contre_indication(id, nom)
nested_indication(id, nom)
nested_classe_chimique(id, nom, idsup)
nested_classe_pharmacologique(id, nom, idsup)

Tables maîtres

medecin(medecin(id): type_medecin,
Ensemble(nested_consultation),
Ensemble(nested_laboratoire_pharmaceutique),
Ensemble(nested_traitement
)
patient(patient(id): type_patient,
Ensemble(nested_consultation),
Ensemble(nested_medicament),
Ensemble(nested_traitement)
)
traitement(traitement(id): type_traitement,
Ensemble(nested_medicament),
Ensemble(nested_recommandation)
)
consultation(consultation(id): type_consultation, Ensemble(nested_observation))
laboratoire_pharmaceutique(laboratoire_pharmaceutique(id): type_laboratoire_pharmaceutique,
Ensemble(nested_medecin)
)
observation(observation(id): type_observation, Ensemble(nested_medecin))
maladie(maladie(id): type_maladie,
Ensemble(nested_medicament),
Ensemble(nested_observation),
Ensemble(nested_symptome)
)
symptome(symptome(id): type_symptome, Ensemble(nested_maladie))
medicament(medicament(id): type_medicament,
Ensemble(nested_patient),
Ensemble(nested_traitement),
Ensemble(nested_medecin),
Ensemble(nested_maladie),
Ensemble(nested_substance_active),
Ensemble(nested_effet_indesirable),
Ensemble(nested_contre_indication),
Ensemble(nested_indication)
)
indication(indication(id): type_indication, Ensemble(nested_medicament))
contre_indication(contre_indication(id): type_contre_indication, Ensemble(nested_medicament),

effet_indesirable(effet_indesirable(id): type_effet_indesirable,
Ensemble(nested_medicament),
Ensemble(substance_active),
Ensemble(nested_classe_chimique),
Ensemble(nested_classe_pharmacologique)
)
substance_active(substance_active(id): type_substance_active,
Ensemble(nested_medicament),
Ensemble(nested_effet_indesirable),
Ensemble(nested_classe_chimique),
Ensemble(nested_classe_pharmacologique)
)
classe_chimique(classe_chimique(id): type_classe_chimique,
Ensemble(nested_effet_indesirable),
Ensemble(nested_substance_active)
)
classe_pharmacologique(classe_pharmacologique(id): type_classe_pharmacologique,
Ensemble(nested_effet_indesirable),
Ensemble(nested_substance_active)
)

Opérations

Opérations pour le modèle SQL2

1. une fonction/procédure qui déduit une ou plusieurs maladies à partir d'un symptôme, classées de la plus spécifique à la plus générique.

```
CREATE OR REPLACE PROCEDURE selectMaladieParSymptome(symptome_id INTEGER)
IS
      maladierow maladie%ROWTYPE;
BEGIN
      DBMS_OUTPUT.PUT_LINE('Maladie(s) pour le symptôme
'||symptome_id||':');
      FOR ml IN (
            SELECT DISTINCT m.* FROM maladie m
            INNER JOIN maladie_symptome ms ON ms.symptome_id = symptome_id
AND ms.maladie_id = m.id
            ORDER BY m.idsup DESC
      )
      LOOP
            SELECT * INTO maladierow FROM maladie m WHERE m.id = ml.id;

DBMS_OUTPUT.PUT_LINE(maladierow.id||'-'||maladierow.nom||'-'||maladierow.id
```

```
sup);
      END LOOP;
EXCEPTION
      WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Aucune maladie trouvée pour ce symptôme');
END selectMaladieParSymptome;
/
EXECUTE selectMaladieParSymptome(2);
```

2. une fonction/procédure qui permet de proposer une liste de médicaments à partir d'une maladie. Si un lien maladie-médicament n'existe pas, il faudra remonter dans la hiérarchie des maladies jusqu'à trouver un médicament à proposer. Pour chaque médicament, l'url d'accès à la notice sera également fournie en sortie.

```
CREATE OR REPLACE PROCEDURE selectProposerMedicament(maladie_id INTEGER) AS
      TYPE table_type_medicament IS TABLE OF medicament%ROWTYPE;
      table_medicament table_type_medicament;
      nom VARCHAR2(64);
      url VARCHAR2(255);
      tmp_maladie_id INTEGER;
      found BOOLEAN := false;
BEGIN
      DBMS_OUTPUT.PUT_LINE('Liste de médicament(s) pour la maladie ' ||
maladie_id || ' : ');
      WHILE NOT found
      LOOP
            BEGIN
                  SELECT mm.maladie_id INTO tmp_maladie_id FROM
maladie_medicament mm WHERE mm.maladie_id = maladie_id;
                  EXCEPTION
                        WHEN NO_DATA_FOUND THEN
                              SELECT m.id INTO tmp_maladie_id FROM maladie m
WHERE ROWNUM = 1
                                    START WITH m.id = tmp_maladie_id
                                    CONNECT BY PRIOR m.idsup = m.id;
            END;
            found := true;
      END LOOP;
      SELECT m.* BULK COLLECT INTO table_medicament FROM medicament m
            INNER JOIN maladie_medicament mm ON mm.maladie_id =
tmp_maladie_id;
      FOR i IN 1..table_medicament.COUNT
```

```
      LOOP
            nom := table_medicament(i).nom;
            url := table_medicament(i).url;
            DBMS_OUTPUT.PUT_LINE('-> ' || nom || ' : ' || url);
      END LOOP i;
END selectProposerMedicament;
/
EXECUTE selectProposerMedicament(2);
```

3. une fonction/procédure qui permet de sauvegarder le patient, son traitement (l'ensemble du ou des médicaments et/ou recommandations) et la ou les maladies diagnostiquées par un médecin. Pour contrôler les prescriptions, le système ne doit pas autoriser un médecin à prescrire un médicament pour lequel il a participé à l'élaboration. Lancez les messages d'erreurs adéquats à l'utilisateur.

```
CREATE OR REPLACE PROCEDURE sauvegarderPatientInfo(patient_id INTEGER,
traitement_id INTEGER) IS
      patient_rec patient%ROWTYPE;
      utilisateur_rec utilisateur%ROWTYPE;
      traitement_rec traitement%ROWTYPE;
      TYPE table_type_medicament IS TABLE OF medicament%ROWTYPE;
      table_medicament table_type_medicament;
      TYPE table_type_recommandation IS TABLE OF recommandation%ROWTYPE;
      table_recommandation table_type_recommandation;
      TYPE table_type_maladie IS TABLE OF maladie%ROWTYPE;
      table_maladie table_type_maladie;
BEGIN
      DBMS_OUTPUT.PUT_LINE('Sauvegarde des informations du patient ' ||
patient_id || ' avec le traitement ' || traitement_id);
      BEGIN
      -- patient
            SELECT p.* INTO patient_rec FROM patient p INNER JOIN traitement
t ON t.patient_id = patient_id WHERE p.id = patient_id AND ROWNUM = 1;
            SELECT u.* INTO utilisateur_rec FROM utilisateur u WHERE u.id =
patient_rec.utilisateur_id;
            EXCEPTION
                  WHEN NO_DATA_FOUND THEN
                        DBMS_OUTPUT.PUT_LINE('Le patient ' || patient_id || '
n''a pas été trouvé');
                        RETURN;
      END;
```

```
      DBMS_OUTPUT.PUT_LINE('Patient ' || patient_rec.id || ' - ' ||
utilisateur_rec.prenom || ' ' || utilisateur_rec.nom);
      BEGIN
            -- le traitement
            SELECT t.* INTO traitement_rec FROM traitement t WHERE t.id =
traitement_id;
            EXCEPTION
                  WHEN NO_DATA_FOUND THEN
                        DBMS_OUTPUT.PUT_LINE('Le traitement ' ||
traitement_id || ' n''a pas été trouvé');
                        RETURN;
      END;
      DBMS_OUTPUT.PUT_LINE('Traitement ' || traitement_rec.id || ' - ' ||
traitement_rec.date_debut || '-' || traitement_rec.date_fin);
      BEGIN
            -- les médicaments pour le traitement
            SELECT m.* BULK COLLECT INTO table_medicament FROM medicament m
                  INNER JOIN medicament_traitement mt ON mt.traitement_id =
traitement_id;
            FOR i IN 1..table_medicament.COUNT
            LOOP
                  DBMS_OUTPUT.PUT_LINE('Médicament : ' ||
table_medicament(i).nom || ' : ' || table_medicament(i).url);
            END LOOP i;
            EXCEPTION
                  WHEN NO_DATA_FOUND THEN
                        RETURN;
      END;
      BEGIN
            -- les recommandation pour le traitement
            SELECT r.* BULK COLLECT INTO table_recommandation FROM
recommandation r
                  INNER JOIN recommandation_traitement rt ON rt.traitement_id
= traitement_id;
            FOR i IN 1..table_medicament.COUNT
            LOOP
                  DBMS_OUTPUT.PUT_LINE('Recommandation : ' ||
table_recommandation(i).nom);
            END LOOP i;
            EXCEPTION
                  WHEN NO_DATA_FOUND THEN
                        RETURN;
```

```
      END;
      BEGIN
            -- les maladies diagnostiquées du patient
            SELECT m.* BULK COLLECT INTO table_maladie FROM maladie m INNER
JOIN maladie_observation mo ON m.id = mo.maladie_id
           WHERE mo.observation_id IN (SELECT o.id FROM observation o INNER
JOIN consultation c ON o.consultation_id = c.id AND c.patient_id =
patient_id);
            FOR i IN 1..table_maladie.COUNT
            LOOP
                  DBMS_OUTPUT.PUT_LINE('Maladie : ' || table_maladie(i).nom);
            END LOOP i;
            EXCEPTION
                  WHEN NO_DATA_FOUND THEN
                        RETURN;
      END;
END sauvegarderPatientInfo;
/
```

4. une fonction/procédure qui détermine pour un médicament la liste des effets indésirables connus et possibles qui seront déduits à partir des hiérarchies des classes chimiques et des classes pharmacologiques des substances actives.

Je n'ai pas traité cette question, je ne sais pas comment faire le lien entre le médicament et les classes chimiques et pharmacologiques parce qu' il n'y a qu'un lien médicament-substance active (peut être à cause d'une erreur de conception du mcd).

```
CREATE OR REPLACE PROCEDURE selectEffetIndesirableMedicament (medicament_id
INTEGER) AS
BEGIN
      NULL;
END selectEffetIndesirableMedicament;
/
```

5. une fonction/procédure qui permet à chaque médecin de connaître la liste de tous les médicaments qu'il a prescrits.

```
CREATE OR REPLACE PROCEDURE selectMedicamentMedecin IS
      TYPE table_type_medicament IS TABLE OF medicament%ROWTYPE;
      table_medicament table_type_medicament;
      nom VARCHAR2(64);
```

```
      url VARCHAR2(255);
BEGIN
      SELECT DISTINCT m.* BULK COLLECT INTO table_medicament FROM medicament
m WHERE m.id IN (
            SELECT mt.medicament_id FROM medicament_traitement mt WHERE
mt.traitement_id IN (
                  SELECT t.id FROM traitement t WHERE t.medecin_id IN (
                        SELECT id FROM medecin
                  )
            )
      );
      FOR i IN 1..table_medicament.COUNT
      LOOP
            nom := table_medicament(i).nom;
            url := table_medicament(i).url;
            DBMS_OUTPUT.PUT_LINE('-> ' || nom || ' : ' || url);
      END LOOP i;
EXCEPTION
      WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Aucun médicament trouvée');
END selectMedicamentMedecin;
/
EXECUTE selectMedicamentMedecin;
```

Opérations pour le modèle SQL3

1. une fonction/procédure qui déduit une ou plusieurs maladies à partir d'un symptôme, classées de la plus spécifique à la plus générique.

```
CREATE OR REPLACE PROCEDURE selectMaladieParSymptome(symptome_id INTEGER)
IS
      nm type_nested_table_maladie;
BEGIN
      DBMS_OUTPUT.PUT_LINE('Maladie(s) pour le symptôme
'||symptome_id||':');
    SELECT s.nested_maladie INTO nm FROM symptome s WHERE s.symptome.id =
symptome_id;
```

```
    FOR i IN nm.first..nm.last
    LOOP
        DBMS_OUTPUT.PUT_LINE(nm(i).nom);
    END LOOP;
EXCEPTION
     WHEN NO_DATA_FOUND THEN
          DBMS_OUTPUT.PUT_LINE('Aucune maladie trouvée pour ce symptôme');
END selectMaladieParSymptome;
/
EXECUTE selectMaladieParSymptome(1);
```

2. une fonction/procédure qui permet de proposer une liste de médicaments à partir d'une maladie. Si un lien maladie-médicament n'existe pas, il faudra remonter dans la hiérarchie des maladies jusqu'à trouver un médicament à proposer. Pour chaque médicament, l'url d'accès à la notice sera également fournie en sortie.

```
CREATE OR REPLACE PROCEDURE selectProposerMedicament(maladie_id INTEGER) AS
    TYPE table_type_nested_medicament IS TABLE OF
maladie.nested_medicament%ROWTYPE;
     table_nested_medicament table_type_nested_medicament;
BEGIN
     DBMS_OUTPUT.PUT_LINE('Liste de médicament(s) pour la maladie ' ||
maladie_id || ' : ');
    SELECT DISTINCT m.table_nested_medicament.* BULK COLLECT INTO
table_medicament FROM maladie m WHERE m.maladie.id = maladie_id;
    FOR i IN 1..table_nested_medicament.COUNT
     LOOP
          nom := table_nested_medicament(i).nom;
          url := table_nested_medicament(i).url;
          DBMS_OUTPUT.PUT_LINE('Médicament : ' || nom || ' : ' || url);
     END LOOP i;
EXCEPTION
     WHEN NO_DATA_FOUND THEN
          DBMS_OUTPUT.PUT_LINE('Aucun médicament à proposer pour la
maladie ' || maladie_id);
END selectProposerMedicament;
/
```

3. une fonction/procédure qui permet de sauvegarder le patient, son traitement (l'ensemble du ou des médicaments et/ou recommandations) et la ou les maladies diagnostiquées par un médecin. Pour contrôler les prescriptions, le système ne doit pas autoriser un médecin à prescrire un médicament pour lequel il a participé à l'élaboration. Lancez les messages d'erreurs adéquats à l'utilisateur.

Je n'ai pas implémenté, car je ne sais pas si la conception du modèle relationnel objet avec les tables imbriqués semble correct, car la table patient possède une table imbriqué pour le traitement, les médicaments mais pas pour les maladies diagnostiquées car elles sont inclusent dans la table maître médicament.
Ma solution aurait été d'utiliser des jointures sur ces tables imbriqués. De plus, en SQL3 les jointures avec les clés étrangères n'existe pas.

```
CREATE OR REPLACE PROCEDURE sauvegarderPatientInfo IS
BEGIN
    NULL;
END sauvegarderPatientInfo;
/
EXECUTE sauvegarderPatientInfo();
```

4. une fonction/procédure qui détermine pour un médicament la liste des effets indésirables connus et possibles qui seront déduits à partir des hiérarchies des classes chimiques et des classes pharmacologiques des substances actives.

Je n'ai pas traité cette question, je ne sais pas comment faire le lien entre le médicament et les classes chimiques et pharmacologiques parce qu' il n'y a qu'un lien médicament-substance active (peut être à cause d'une erreur de conception du modèle relationnel-objet).

```
CREATE OR REPLACE PROCEDURE selectEffetIndesirableMedicament IS
BEGIN
    NULL;
END selectEffetIndesirableMedicament;
/
EXECUTE selectEffetIndesirableMedicament();
```

5. une fonction/procédure qui permet à chaque médecin de connaître la liste de tous les médicaments qu'il a prescrits.

Je n’ai pas implémenté car en SQL3 les jointures avec les clés étrangères n’existe pas.
Ma solution aurait été d’utiliser une jointure avec le traitement_id de la table traitement imbriquée dans medecin avec la table traitement avec le id (traitement) qui contient une table imbriquée médicament pour savoir les médicaments prescrits, mais en SQL3 les jointures avec les clés étrangères n’existe pas.

```
CREATE OR REPLACE PROCEDURE selectMedicamentMedecin IS
BEGIN
    NULL;
END selectMedicamentMedecin;
/
EXECUTE selectMedicamentMedecin();
```