BASE DE DONNEE 2 - PROJET

Réalisé par KARTHIGESU Prousoth et Maréchal Théo

Table des matières

Introduction

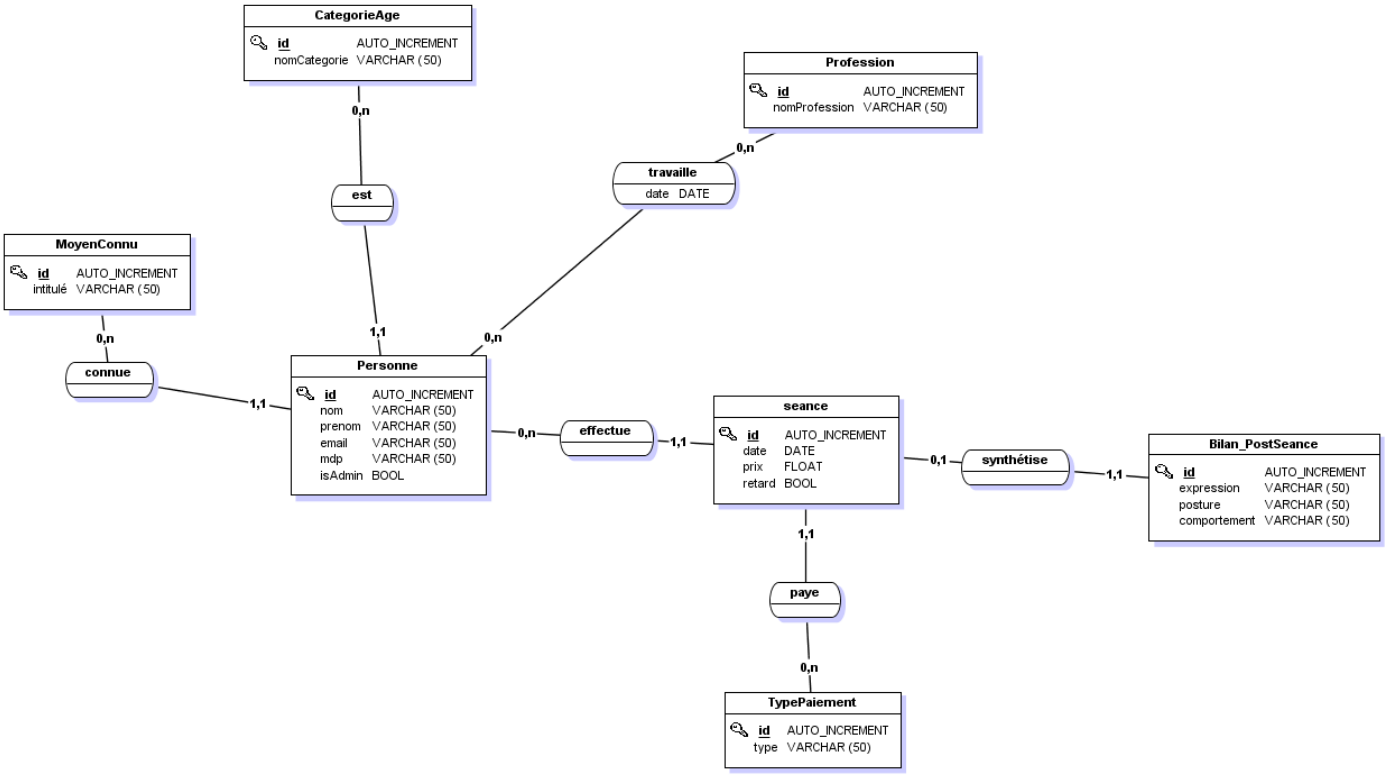
Dans le cadre de notre formation en base de données, nous devions réaliser un projet informatique. Nous devions concevoir et développé une application avec sa base de données.

Objectif du projet

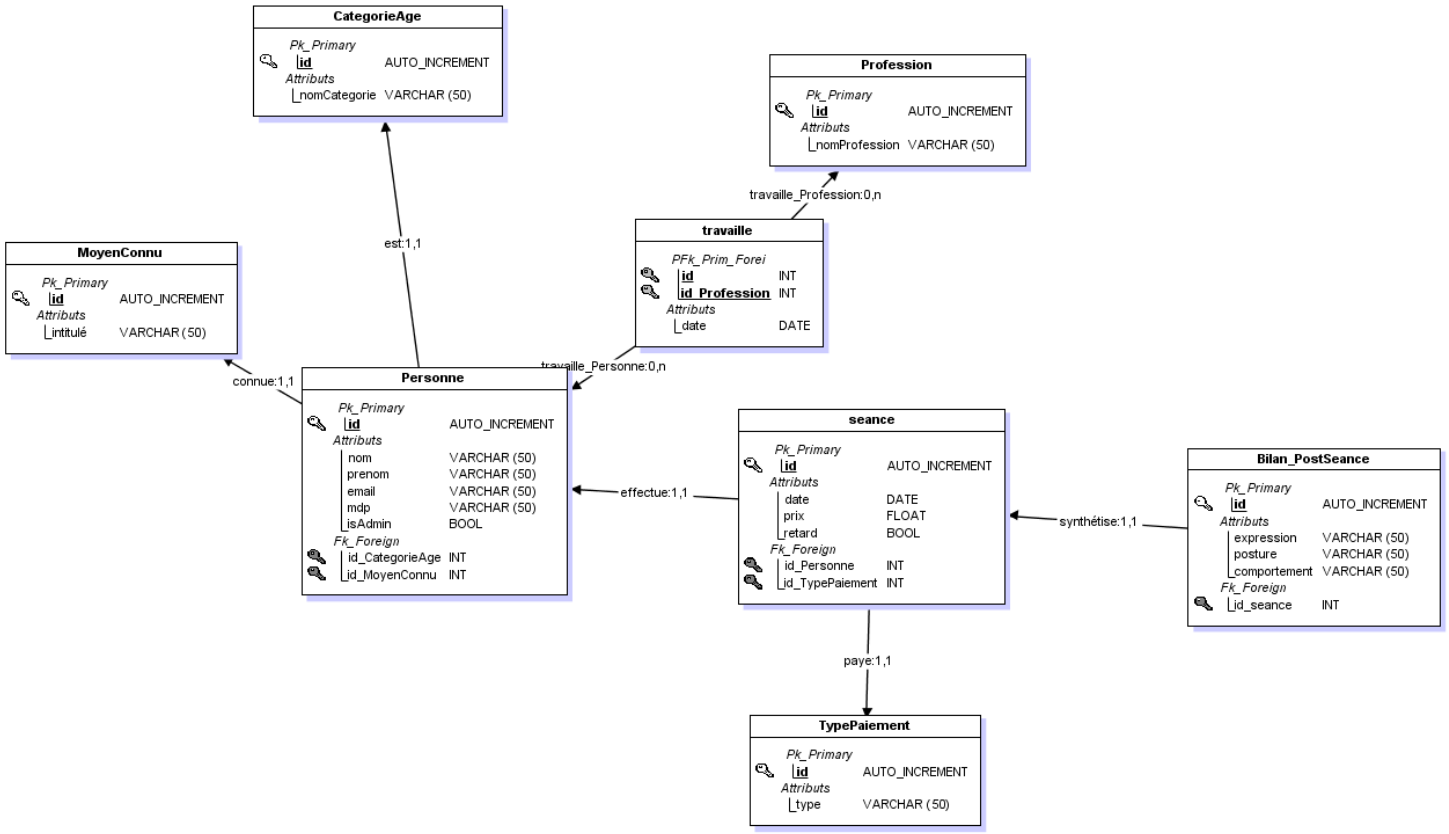
L'objectif est de réalisé une application pour une psychologue. L'ensemble des fonctionnalités sont détaillé dans le sujet fournis. Cette application devra permettre la création de rendez-vous et leur consultation par les patients et la psy.

Conception de la base de données

Nous devions tout d'abord réaliser le modèle MCD de la base de données. Après avoir relue à multiple reprise nous avons adopté le modèle suivant :



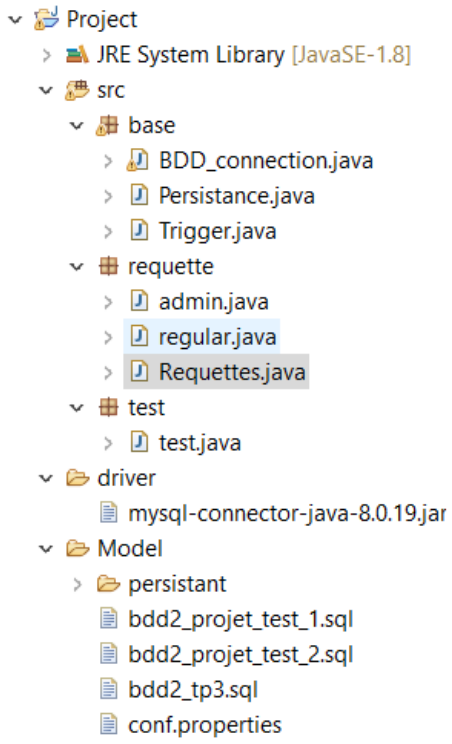
Ce modèle a été valider après quelque correctif par notre enseignant. Voici le modèle MLD associé :



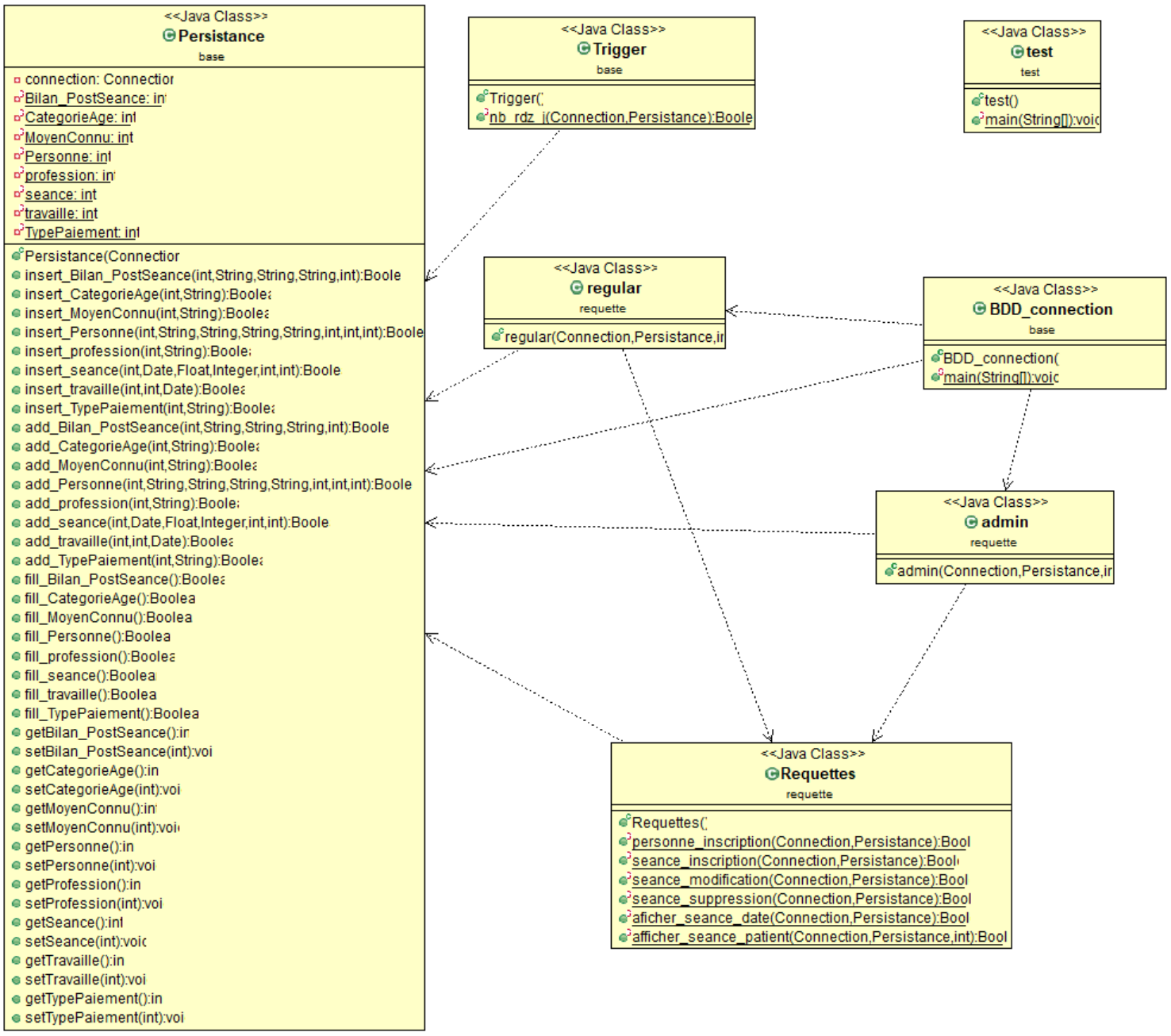
Structure du code JAVA

Le code l'application qui se situe dans le dossier « src » a été segmenter dans 3 package. Le package « base » contient les fichiers liés à la base de données. Le package « requettes » contient les fichiers liés à l'interface de l'application. Et enfin le package « test » contient le main.

Le dossier « Model » contient quand a lui l'ensemble des scripts SQL.



Voici le diagramme de classe de l'application :



Les fonctionnalités implémentées

Le programme permet de charger une BDD à partir d'un fichier SQL décrivant sa structure et de fichier csv décrivant le contenue de la base, c'est fichier sont

constamment mis à jour de manière à ce que quel que soit le moment où la base de données est arrêtée elle puisse être rechargé simplement en la relançant

La BDD possède 2 niveaux d'utilisateur : « regular », destiné au patient qui leur permet seulement de voir leur rendez-vous passé et futur. ET admin destiné à la psy et c'est éventuel associé qui offre toutes les fonctions du niveau « regular », mais permet de voir le planning de n'importe qui et de le voir en fonction d'une date, de créer des « compte », de mettre en place un rendez-vous, le supprimer et le modifier.

Il est possible de facilement changer le système de base de donné en ne modifiant que le fichier « comf.properties » et la logiciel fait attention à demander plusieurs fois à l'utilisateur si il souhaite vraiment quitter.

Les difficultés rencontrées

Nous avons essentiellement rencontré des difficultés sur la mise en place de la base de données et la liaison de celle-ci avec le code JAVA.

Ayant des bases avancées en informatique, le reste du projet a pu être développé assez aisément.

Scenario d'exécution

- Lancement de l'application et demande de connexion

```
Connection avec la base de données établie.
Table remplis.
Que voulez vous faire :
- 1) Se connecter
- 2) Quitter
```

- Quitter l'application après son lancement

```
Que voulez vous faire :
- 1) Se connecter
- 2) Quitter
2
Etes-vous sur de vouloir quiter (oui/non):
oui
Aurevoir
```

- Connexion en tant qu'administrateur sur l'application

```
Table remplis.
Que voulez vous faire :
- 1) Se connecter
- 2) Quitter
1
E-mail de l'utilisateur :
reneclaude
Mot de passe de l'utilisateur :
p
Que voulez vous faire :
- 1) Consulter planing passé
- 2) Consulter planing futur
- 3) Créer patient
- 4) Créer séance
- 5) Modifier séance
- 6) Supprimer séance
- 7) Se deconnecter
```

- Création d'un patient depuis l'espace administrateur

```
- 1) Consulter planing passé
- 2) Consulter planing futur
- 3) Créer patient
- 4) Créer séance
- 5) Modifier séance
- 6) Supprimer séance
- 7) Se deconnecter
3
Nom de la personne:
tru
Prenom de la personne:
zer
E-mail de la personne:
truzer
Mode de passe de la personne:
pp
Est ce que le patientest un admin (Vrai/Faux):
Faux
Categorie d'age de la personne:
1
Moyen connue de la personne:
1
```

- Afficher le planning d'un patient depuis l'espace administrateur

```
- 1) Consulter planing
- 2) Créer patient
- 3) Créer séance
- 4) Modifier séance
- 5) Supprimer séance
- 6) Se deconnecter
1
Donné l'id du patient dont vous souhaitez voir les séances:
3
id: 1, nom: 1111-01-01, prix: 1.0, retard: false, id patient: 3, id type de payment: 1
id: 2, nom: 1111-01-01, prix: 1.0, retard: true, id patient: 3, id type de payment: 1
```

- Connexion en tant que patient sur l'application

```
Que voulez vous faire :
- 1) Se connecter
- 2) Quitter
1
E-mail de l'utilisateur :
sdd
Mot de passe de l'utilisateur :
w
Que voulez vous faire :
- 1) Consulter planing
- 2) Se déconnecter
```

- Afficher le planning depuis l'espace patient

```
Que voulez vous faire :
- 1) Consulter planing
- 2) Se déconnecter
1
id: 3, nom: 2222-02-02, prix: 2.0, retard: true, id patient: 5, id type de payment: 1
```

Annexe

Script de la base de données

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

# Table: Profession

#------------------------------------------------------------


CREATE TABLE Profession(

      id          Int  Auto_increment  NOT NULL ,

      nomProfession Varchar (50) NOT NULL

          ,CONSTRAINT Profession_PK PRIMARY KEY (id)

)ENGINE=InnoDB;



#------------------------------------------------------------

# Table: CategorieAge

#------------------------------------------------------------


CREATE TABLE CategorieAge(

      id          Int  Auto_increment  NOT NULL ,

      nomCategorie Varchar (50) NOT NULL

          ,CONSTRAINT CategorieAge_PK PRIMARY KEY (id)

)ENGINE=InnoDB;



#------------------------------------------------------------

# Table: MoyenConnu
```

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

CREATE TABLE MoyenConnu(
    id      Int  Auto_increment  NOT NULL ,
    intitule Varchar (50) NOT NULL
        ,CONSTRAINT MoyenConnu_PK PRIMARY KEY (id)
)ENGINE=InnoDB;



#------------------------------------------------------------
# Table: Personne
#------------------------------------------------------------

CREATE TABLE Personne(
    id            Int  Auto_increment  NOT NULL ,
    nom           Varchar (50) NOT NULL ,
    prenom        Varchar (50) NOT NULL ,
    email         Varchar (50) NOT NULL ,
    mdp           Varchar (50) NOT NULL ,
    isAdmin       Bool NOT NULL ,
    id_CategorieAge Int NOT NULL ,
    id_MoyenConnu   Int NOT NULL
        ,CONSTRAINT Personne_PK PRIMARY KEY (id)


        ,CONSTRAINT Personne_CategorieAge_FK FOREIGN KEY (id_CategorieAge) REFERENCES
CategorieAge(id)
        ,CONSTRAINT Personne_MoyenConnu0_FK FOREIGN KEY (id_MoyenConnu) REFERENCES
MoyenConnu(id)
)ENGINE=InnoDB;



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

```
# Table: TypePaiement
#------------------------------------------------------------

CREATE TABLE TypePaiement(
        id   Int  Auto_increment  NOT NULL ,
        type Varchar (50) NOT NULL
            ,CONSTRAINT TypePaiement_PK PRIMARY KEY (id)
)ENGINE=InnoDB;



#------------------------------------------------------------
# Table: seance
#------------------------------------------------------------

CREATE TABLE seance(
        id            Int  Auto_increment  NOT NULL ,
        date          Date NOT NULL ,
        prix          Float NOT NULL ,
        retard        Bool NOT NULL ,
        id_Personne     Int NOT NULL ,
        id_TypePaiement Int NOT NULL
            ,CONSTRAINT seance_PK PRIMARY KEY (id)


            ,CONSTRAINT seance_Personne_FK FOREIGN KEY (id_Personne) REFERENCES Personne(id)
            ,CONSTRAINT seance_TypePaiement0_FK FOREIGN KEY (id_TypePaiement) REFERENCES
TypePaiement(id)
)ENGINE=InnoDB;



#------------------------------------------------------------
# Table: Bilan_PostSeance
```

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

CREATE TABLE Bilan_PostSeance(
    id          Int  Auto_increment  NOT NULL ,
    expression   Varchar (50) NOT NULL ,
    posture      Varchar (50) NOT NULL ,
    comportement Varchar (50) NOT NULL ,
    id_seance    Int NOT NULL
        ,CONSTRAINT Bilan_PostSeance_PK PRIMARY KEY (id)

        ,CONSTRAINT Bilan_PostSeance_seance_FK FOREIGN KEY (id_seance) REFERENCES seance(id)
        ,CONSTRAINT Bilan_PostSeance_seance_AK UNIQUE (id_seance)
)ENGINE=InnoDB;


#------------------------------------------------------------
# Table: travaille
#------------------------------------------------------------

CREATE TABLE travaille(
    id          Int NOT NULL ,
    id_Profession Int NOT NULL ,
    date        Date NOT NULL
        ,CONSTRAINT travaille_PK PRIMARY KEY (id,id_Profession)

        ,CONSTRAINT travaille_Personne_FK FOREIGN KEY (id) REFERENCES Personne(id)
        ,CONSTRAINT travaille_Profession0_FK FOREIGN KEY (id_Profession) REFERENCES
Profession(id)
)ENGINE=InnoDB;
```

Les Requêtes

- Créer un patient :

  INSERT INTO Personne (nom, prenom, email, mdp, isAdmin, id_CategorieAge, id_MoyenConnu)
  VALUES ('valeur 1', 'valeur 2', 'valeur 3', 'valeur 4', false,'valeur 6','valeur 7')

- Créer un rdv :

  INSERT INTO seance (date, prix, retard, id_Personne, id_TypePaiement) VALUES ('valeur 1', 'valeur 2', false, 'valeur 4', 'valeur 5')

- Modifier un rdv :

  UPDATE seance SET date= 'valeur 1', prix= 'valeur 2', retard= 'valeur 3', id_Personne= 'valeur 4',id_TypePaiement='valeur 5')
  WHERE id="valeur_id"

- Supprimer un rdv :

  CREATE OR REPLACE VIEW delete_patient (IdPatient) AS
  DELETE FROM seance
  WHERE id=IdPatient

- Toutes les séances du jour :

  CREATE OR REPLACE VIEW all_future_seance_patient(IdPatient) AS
  SELECT * FROM seance WHERE date= 'valeur 1'

- Toutes les séances futures d'un patient :

  CREATE OR REPLACE VIEW all_future_seance_patient(IdPatient) AS
  SELECT * FROM seance WHERE date=> CAST(NOW() AS DATE) and id=IdPatient

- Toutes les séances passé d'un patient :

  CREATE OR REPLACE VIEW all_seance_patient(IdPatient) AS
  SELECT * FROM seance WHERE date < CAST(NOW() AS DATE) and id=IdPatient

- Un patient ne peux avoir plus de 3 rendez-vous par jour / La psy ne peux pas travailler plus de 10h par jour :

  CREATE TRIGGER before_insert_seance BEFORE INSERT ON seance FOR EACH ROW

  DECLARE
  nb_seance_psy INT DEFAULT
  nb_seance_patient INT DEFAULT

```
BEGIN

SELECT SUM(*) INTO nb_seance_psy  FROM seance WHERE ( ( date BETWEEN TRUNC(NEW.date,
DATE) AND TRUNC(NEW.date+1, DATE)

SELECT SUM(*) INTO nb_seance_patient  FROM seance WHERE ( ( date BETWEEN
TRUNC(NEW.date, DATE) AND TRUNC(NEW.date+1, DATE) ) AND id_Personne =
NEW.id_Personne

IF (nb_seance_psy >= 20) THEN
   RAISE_APPLICATION_ERROR(-20501, 'Insertion impossible : la psy travaille deja 10h ce jour ')
ELSEIF (nb_seance_patient >= 3) THEN
   RAISE_APPLICATION_ERROR(-20501, 'Insertion impossible : le patient a plus de 3 seance le
même jour ')

END IF
END
```

- La psy ne peut pas travailler plus de 10h par jour :

```
CREATE TRIGGER max_10h_psy BEFORE INSERT ON seance FOR EACH ROW
DECLARE

nb_seance INT DEFAULT

BEGIN

SELECT SUM(*) INTO nb_seance   FROM seance WHERE ( ( date BETWEEN TRUNC(NEW.date,
DATE)
AND TRUNC(NEW.date+1, DATE) ) AND id_Personne = NEW.id_Personne

IF (nb_seance >2) THEN
   RAISE_APPLICATION_ERROR(-20501, 'Insertion impossible')
END IF
END
```