Dossier du projet de janvier 2018

Applications informatiques

Par Bastien Dewez et Wilson Weets

Table des matières

[I. Enoncé 4](#_Toc503286170)

[Gestion des différents blocs opératoires d’un hôpital 4](#_Toc503286171)

[1) Utilisateurs 4](#_Toc503286172)

[2) Tâches effectuées par le programme 4](#_Toc503286173)

[3) Cas spéciaux 4](#_Toc503286174)

[II. Analyse 5](#_Toc503286175)

[1. Diagramme des Use Cases 5](#_Toc503286176)

[2. Diagramme de classe 6](#_Toc503286177)

[3. Diagrammes de séquence 6](#_Toc503286178)

[1) Lorsque le chirurgien ajoute, modifie ou supprime une réservation 6](#_Toc503286179)

[2) Lorsque le chirurgien supprime une notification 7](#_Toc503286180)

[3) Lorsque le directeur ajoute ou supprime du personnel 8](#_Toc503286181)

[4) Lorsque le secrétaire ajoute, modifie ou supprime une réservation 9](#_Toc503286182)

[5) Lorsque le secrétaire modifie un patient 9](#_Toc503286183)

[6) Lorsque le secrétaire modifie un patient 10](#_Toc503286184)

[III. Ecrans 11](#_Toc503286185)

[1) Ecran Connexion 11](#_Toc503286186)

[2) Ecran Accueil 11](#_Toc503286187)

[3) Ecran Réservation 12](#_Toc503286188)

[4) Ecran Notification 12](#_Toc503286189)

[5) Ecran Information 13](#_Toc503286190)

[6) Ecran Patient 13](#_Toc503286191)

[7) Ecran Personnel 14](#_Toc503286192)

[IV. Base de données 15](#_Toc503286193)

[1. Schéma conceptuel de la base de données 15](#_Toc503286194)

[2. Compte Oracle 15](#_Toc503286195)

[3. Scripts 15](#_Toc503286196)

[1) Création de la base de données et des tables (CreationTable.sql) 15](#_Toc503286197)

[2) Insertion des premières données (InsertionDonnees.sql) 17](#_Toc503286198)

[3) Ajout du delete on cascade (AjoutCascades.sql) 17](#_Toc503286199)

[4) Package SelectAll (SelectAll.sql) 18](#_Toc503286200)

[5) Package SelectOne (SelectOne.sql) 19](#_Toc503286201)

[6) Package Inserts (Inserts.sql) 20](#_Toc503286202)

[7) Package Updates (Updates.sql) 22](#_Toc503286203)

[8) Package Deletes (Deletes.sql) 23](#_Toc503286204)

[V. Problèmes rencontrés 25](#_Toc503286205)

[1) Crash du serveur GlassFish 25](#_Toc503286206)

[VI. Accès au programme 25](#_Toc503286207)

[Comptes 25](#_Toc503286208)

[Directeur 25](#_Toc503286209)

[Secrétaires 25](#_Toc503286210)

[Chirurgiens 25](#_Toc503286211)

# Enoncé

## Gestion des différents blocs opératoires d’un hôpital

### Utilisateurs

* Chirurgien
* Secrétaire
* Directeur

### Tâches effectuées par le programme

#### Chirurgien

* Réserver une salle d’opération (en y ajoutant les informations du patient)
* Annuler une réservation
* Modifier une réservation
* Consulter ses réservations
* Gérer ses informations
* Consulter ses notifications

#### Secrétaire

* Réserver une salle d’opération (soit à la place d’un chirurgien, soit lors d’une urgence, …)
* Prévenir un chirurgien d’une modification des réservations
* Modifier une réservation
* Supprimer une réservation
* Gérer ses informations
* Consulter les réservations en fonction des salles d’opération
* Consulter la liste des patients inscrit dans des réservations

#### Directeur

* Consulter les réservations en fonction des salles d’opération
* Consulter la liste des patients inscrit dans des réservations
* Ajouter du personnel (chirurgiens, secrétaires)
* Supprimer du personnel
* Consulter la liste de son personnel

### Cas spéciaux

#### Urgences

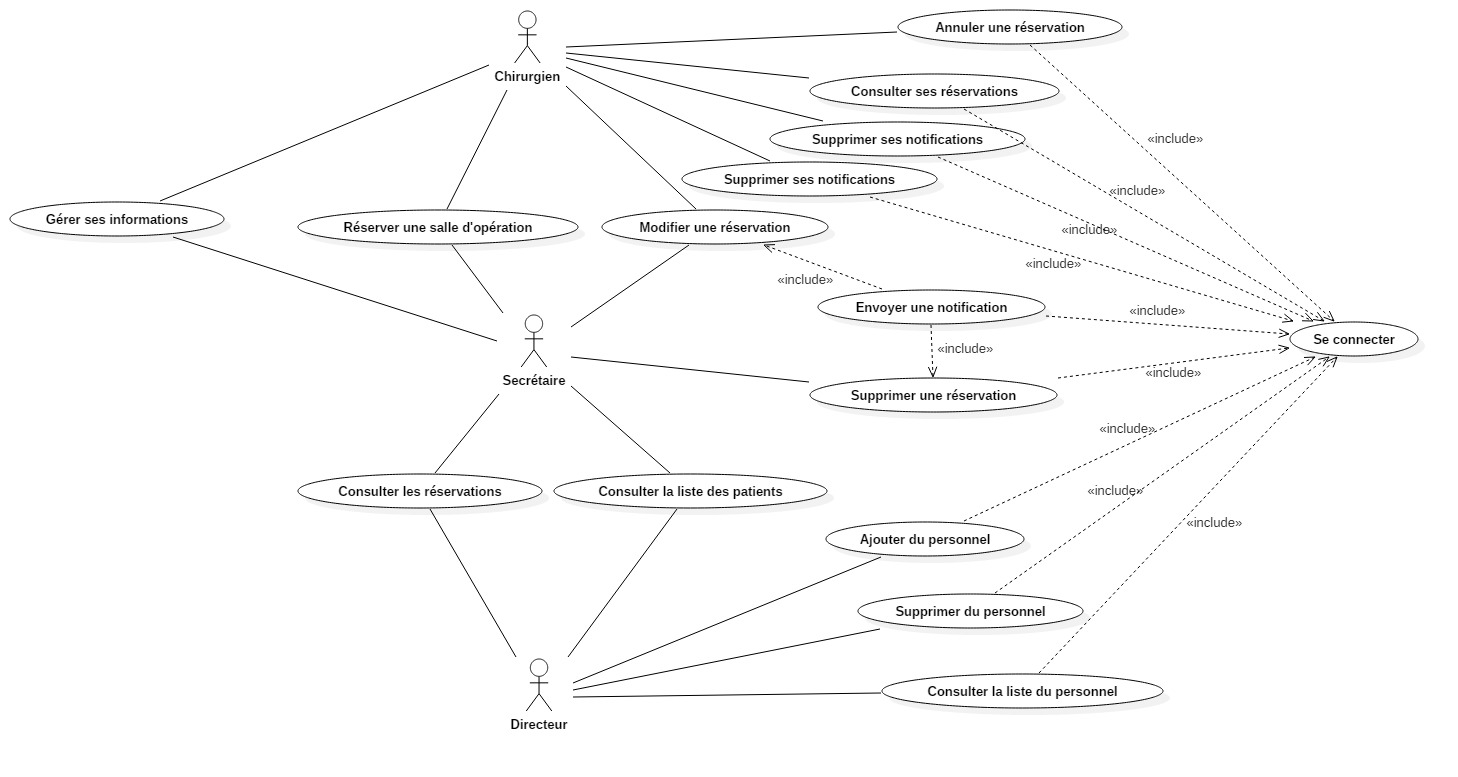
Lorsqu’un patient arriver aux urgences et doit se faire opérer un/une secrétaire doit réserver une salle d’opération en urgence. Si une réservation a dû être enlevée, le chirurgien concerné sera automatiquement prévenu grâce au système de notification.

#### Priorités

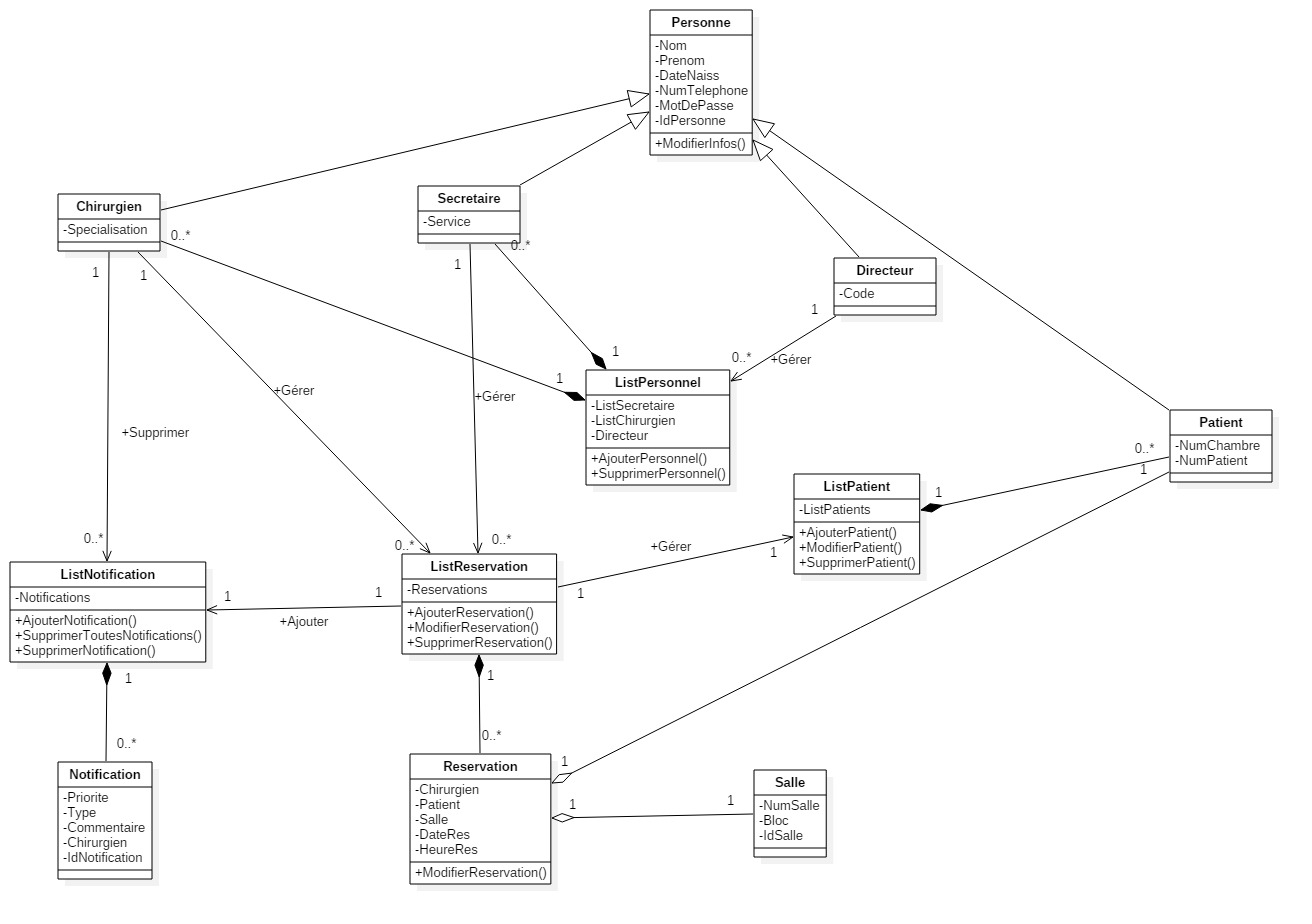
Chaque réservation possèdera une priorité en rapport avec la gravité de la pathologie du patient. Si une réservation a une priorité plus haute qu’une autre, elle sera considérée comme plus urgente et pourra dès lors la remplacer dans l’agenda. Seul les secrétaires peuvent effectuer cette opération. Si un chirurgien veut remplacer une réservation par la sienne, il devra s’adresser au secrétariat.

# Analyse

## Diagramme des Use Cases

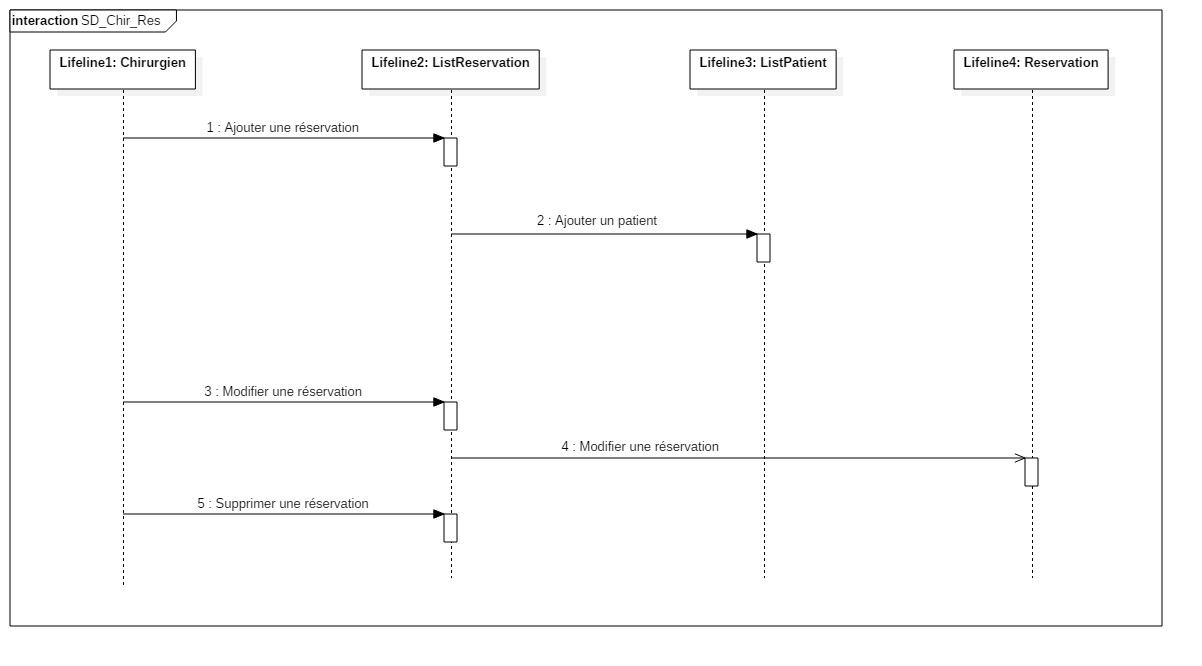


## Diagramme de classe

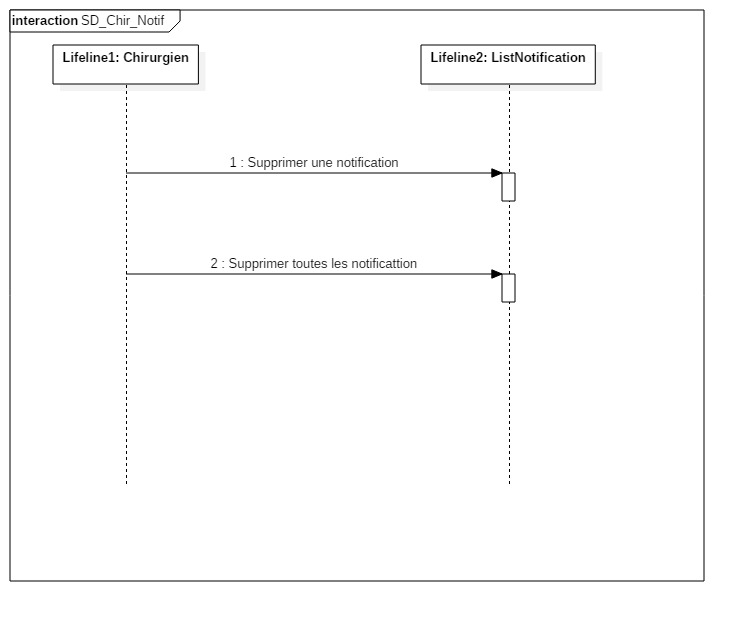


## Diagrammes de séquence

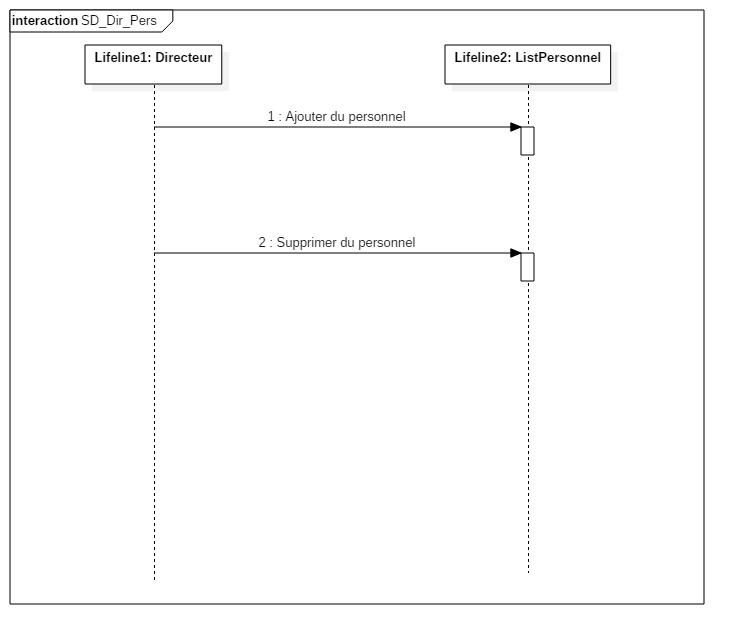
### Lorsque le chirurgien ajoute, modifie ou supprime une réservation



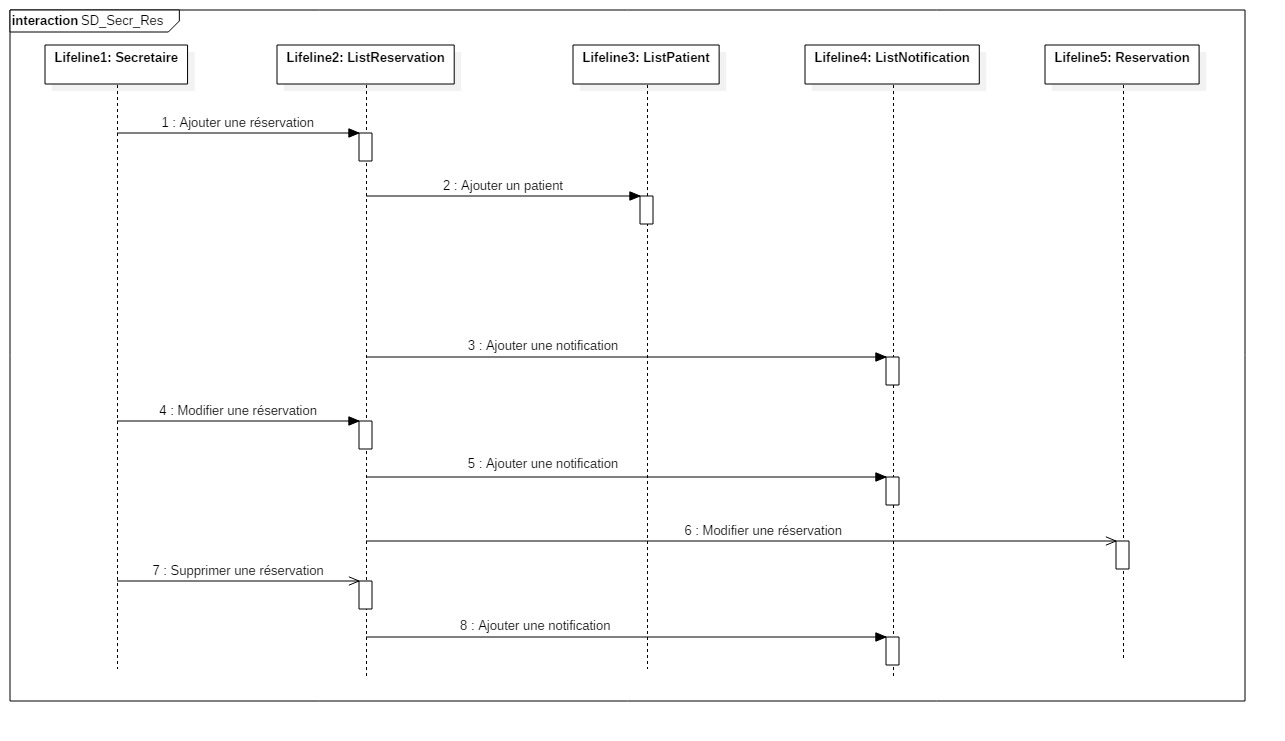
### Lorsque le chirurgien supprime une notification



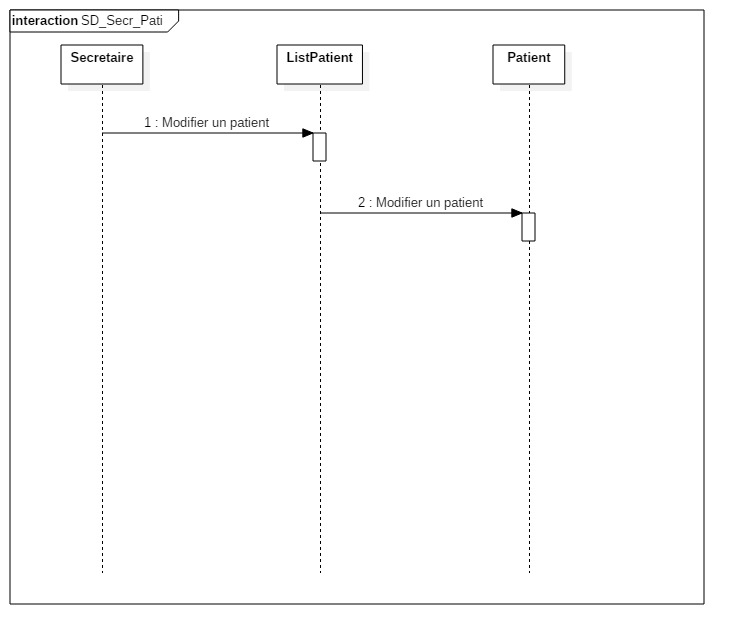
### Lorsque le directeur ajoute ou supprime du personnel



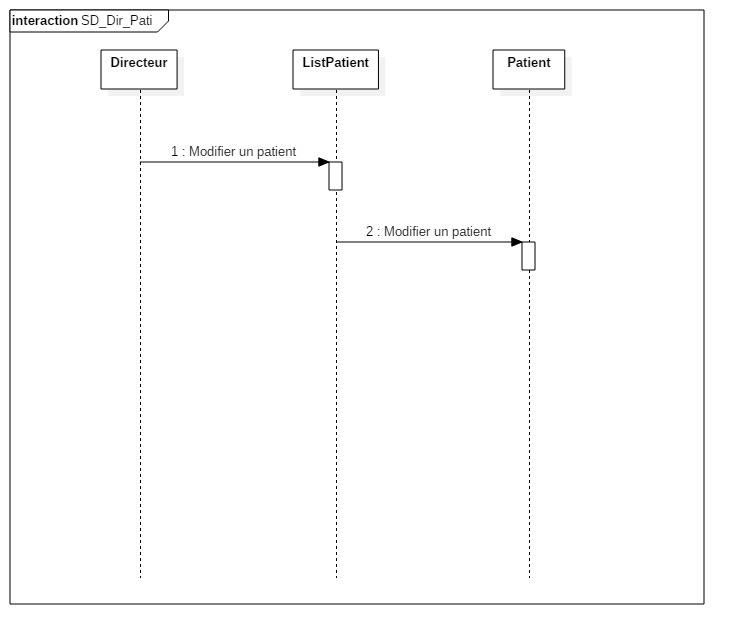
### Lorsque le secrétaire ajoute, modifie ou supprime une réservation



### Lorsque le secrétaire modifie un patient



### Lorsque le secrétaire modifie un patient



# Ecrans

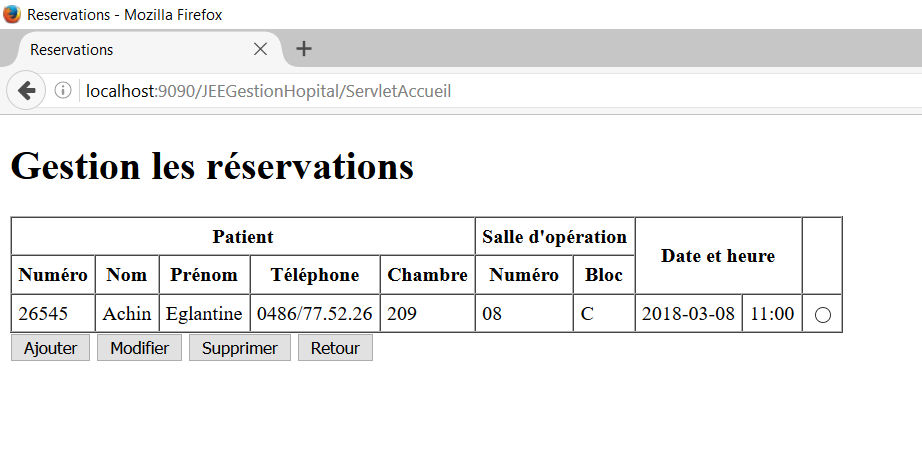
## Ecran Connexion



## Ecran Accueil



## Ecran Réservation



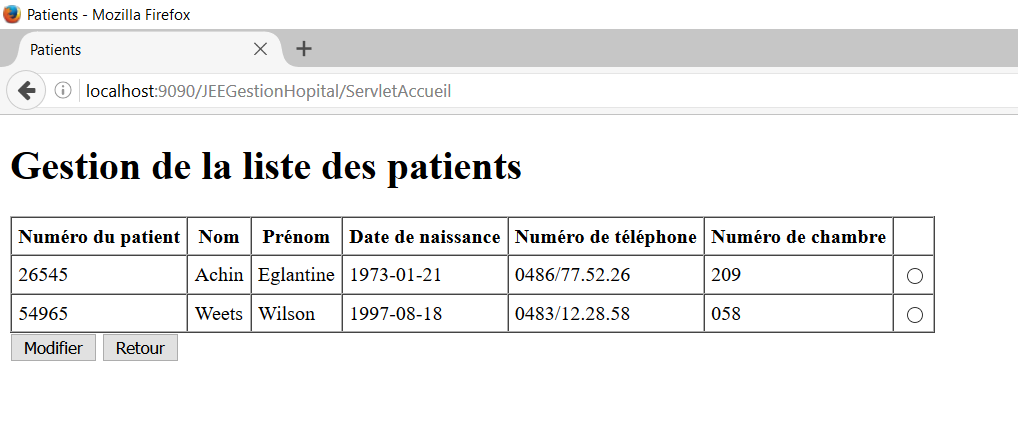
## Ecran Notification



## Ecran Information



## Ecran Patient

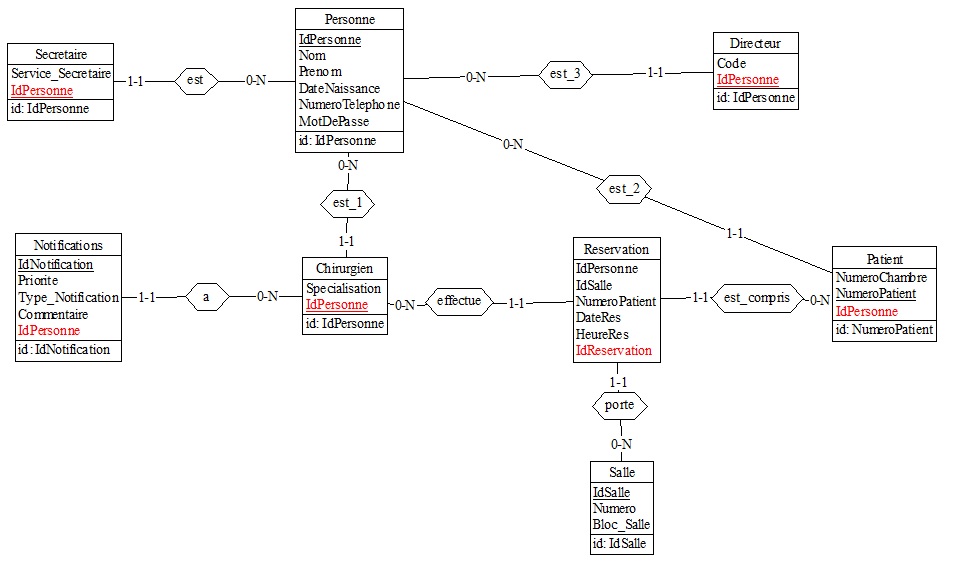


## Ecran Personnel



# Base de données

## Schéma conceptuel de la base de données



## Compte Oracle

Nom d’utilisateur : ora31

Mot de passe : oracle3

## Scripts

### Création de la base de données et des tables (CreationTable.sql)

drop table Personne;

drop table Notifications;

drop table Salle;

drop table Reservation;

create table Personne (

IdPersonne number not null,

Nom varchar2(50),

Prenom varchar2(50),

DateNaissance varchar2(10),

NumeroTelephone varchar2(20),

MotDePasse varchar2(30),

constraint PK\_Personne primary key (IdPersonne)

);

create table Secretaire (

IdPersonne number not null,

Service\_Secr varchar2(30),

constraint PK\_Secretaire primary key (IdPersonne),

constraint FK\_Secretaire\_Personne foreign key (IdPersonne)

references Personne (IdPersonne)

);

create table Chirurgien (

IdPersonne number not null,

Specialisation varchar2(30),

constraint PK\_Chirurgien primary key (IdPersonne),

constraint FK\_Chirurgien\_Personne foreign key (IdPersonne)

references Personne (IdPersonne)

);

create table Directeur (

IdPersonne number not null,

Code varchar2(4),

constraint PK\_Directeur primary key (IdPersonne),

constraint FK\_Directeur\_Personne foreign key (IdPersonne)

references Personne (IdPersonne)

);

create table Patient (

IdPersonne number not null,

NumeroChambre varchar2(3),

NumeroPatient varchar2(5) not null,

constraint PK\_Patient primary key (NumeroPatient),

constraint FK\_Patient\_Personne foreign key (IdPersonne)

references Personne (IdPersonne)

);

create table Notifications (

IdNotification number not null,

Priorite integer,

Type\_Notif integer,

Commentaire varchar2(300),

IdPersonne number not null,

constraint PK\_Notification primary key (IdNotification),

constraint FK\_Notification\_Chirurgien foreign key (IdPersonne)

references Chirurgien (IdPersonne)

);

create table Salle (

IdSalle number not null,

Numero varchar2(2),

Bloc\_Salle char,

constraint PK\_Salle primary key (IdSalle)

);

create table Reservation (

IdPersonne number not null,

IdSalle number not null,

NumeroPatient varchar2(5) not null,

DateRes date,

HeureRes varchar2(4),

constraint PK\_Reservation primary key (IdPersonne,IdSalle,NumeroPatient),

constraint FK\_Reservation\_Chirurgien foreign key (IdPersonne)

references Chirurgien (IdPersonne),

constraint FK\_Reservation\_Salle foreign key (IdSalle)

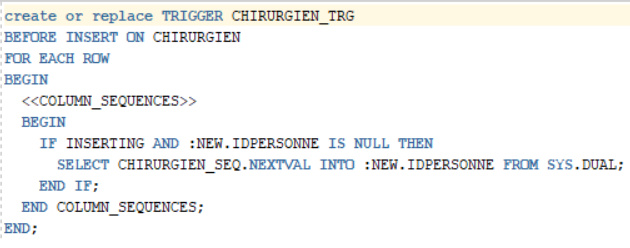
references Salle (IdSalle),

constraint FK\_Reservation\_Patient foreign key (NumeroPatient)

references Patient (NumeroPatient)

);

Les clés primaires de nos tables subissent une incrémentation automatique grâce à des déclencheurs :



### Insertion des premières données (InsertionDonnees.sql)

Ce script insert des données qui vont nous permettre de nous connecter à notre application et d’y effectuer des tests.

insert into Personne (Nom,Prenom,DateNaissance,NumeroTelephone,MotDePasse) values ('Ventura','Manfredi','5/2/1965','0495/15.54.89','Manfrediv65');

insert into Directeur values (1,1234);

insert into Personne (Nom,Prenom,DateNaissance,NumeroTelephone,MotDePasse) values ('Hervieux','Thierry','6/3/1973','0489/89.77.31','Thierryh73');

insert into Chirurgien values (2,'Digestive');

insert into Personne (Nom,Prenom,DateNaissance,NumeroTelephone,MotDePasse) values ('Marquis','Carine','20/8/1972','0487/31.76.98','Carinem72');

insert into Chirurgien values (3,'Obstétrique');

insert into Personne (Nom,Prenom,DateNaissance,NumeroTelephone,MotDePasse) values ('Brasseur','Maurice','11/2/1967','0483/64.62.14','Mauriceb67');

insert into Secretaire values (4,'Ophtalmologie');

insert into Personne (Nom,Prenom,DateNaissance,NumeroTelephone,MotDePasse) values ('Renaud','Francoise','13/7/1987','0472/83.42.68','Francoiser87');

insert into Secretaire values (5,'Génétique');

### Ajout du delete on cascade (AjoutCascades.sql)

Ce script nous permet de rajouter le delete on cascade sur les tables Secrétaire, Chirurgien et Patient.

ALTER TABLE Secretaire

DROP CONSTRAINT FK\_Secretaire\_Personne;

ALTER TABLE Chirurgien

DROP CONSTRAINT FK\_Chirurgien\_Personne;

ALTER TABLE Patient

DROP CONSTRAINT FK\_Patient\_Personne;

----------------------------------------------------------------------------

ALTER TABLE Secretaire

ADD CONSTRAINT FK\_Secretaire\_Personne\_Cascade

FOREIGN KEY (IdPersonne) REFERENCES Personne(IdPersonne) ON DELETE CASCADE;

ALTER TABLE Chirurgien

ADD CONSTRAINT FK\_Chirurgien\_Personne\_Cascade

FOREIGN KEY (IdPersonne) REFERENCES Personne(IdPersonne) ON DELETE CASCADE;

ALTER TABLE Patient

ADD CONSTRAINT FK\_Patient\_Personne\_Cascade

FOREIGN KEY (IdPersonne) REFERENCES Personne(IdPersonne) ON DELETE CASCADE;

### Package SelectAll (SelectAll.sql)

Ce script crée le package SelectAll qui contient des fonctions qui nous permette d’obtenir toutes les lignes des tables.

create or replace package SelectAll

is

TYPE select\_all\_cursor IS REF CURSOR;

function getChirurgiens return select\_all\_cursor;

function getSecretaires return select\_all\_cursor;

function getDirecteurs return select\_all\_cursor;

function getPatients return select\_all\_cursor;

function getNotifications return select\_all\_cursor;

function getReservations return select\_all\_cursor;

end SelectAll;

create or replace package body SelectAll

is

function getChirurgiens return select\_all\_cursor

as

chir\_cur select\_all\_cursor;

begin

OPEN chir\_cur FOR select \* from Chirurgien A inner join Personne B on A.IdPersonne = B.IdPersonne;

RETURN chir\_cur;

end getChirurgiens;

function getSecretaires return select\_all\_cursor

as

secr\_cur select\_all\_cursor;

begin

OPEN secr\_cur FOR select \* from Secretaire A inner join Personne B on A.IdPersonne = B.IdPersonne;

RETURN secr\_cur;

end getSecretaires;

function getDirecteurs return select\_all\_cursor

as

dire\_cur select\_all\_cursor;

begin

OPEN dire\_cur FOR select \* from Directeur A inner join Personne B on A.IdPersonne = B.IdPersonne;

RETURN dire\_cur;

end getDirecteurs;

function getPatients return select\_all\_cursor

as

pati\_cur select\_all\_cursor;

begin

OPEN pati\_cur FOR select \* from Patient A inner join Personne B on A.IdPersonne = B.IdPersonne;

RETURN pati\_cur;

end getPatients;

function getNotifications return select\_all\_cursor

as

noti\_cur select\_all\_cursor;

begin

OPEN noti\_cur FOR select \* from Notifications;

RETURN noti\_cur;

end getNotifications;

function getReservations return select\_all\_cursor

as

res\_cur select\_all\_cursor;

begin

OPEN res\_cur FOR select \* from Reservation;

RETURN res\_cur;

end getReservations;

end SelectAll;

## Package SelectOne (SelectOne.sql)

Ce script crée le package SelectOne qui contient des fonctions qui nous permette d’obtenir une ligne selon l’id passé en paramètre.

create or replace package SelectOne

is

TYPE select\_one\_cursor IS REF CURSOR;

function getChirurgien(idChir Chirurgien.IdPersonne%type) return select\_one\_cursor;

function getSecretaire(idSecr Secretaire.IdPersonne%type) return select\_one\_cursor;

function getPatient(idPati Patient.IdPersonne%type) return select\_one\_cursor;

function getSalle(idSall Salle.IdSalle%type) return select\_one\_cursor;

end SelectOne;

create or replace package body SelectOne

is

function getChirurgien(idChir Chirurgien.IdPersonne%type) return select\_one\_cursor

as

chir\_cur select\_one\_cursor;

begin

OPEN chir\_cur FOR select \* from Chirurgien A inner join Personne B on A.IdPersonne = B.IdPersonne

where A.IdPersonne = idChir;

RETURN chir\_cur;

end getChirurgien;

function getSecretaire(idSecr Secretaire.IdPersonne%type) return select\_one\_cursor

as

secr\_cur select\_one\_cursor;

begin

OPEN secr\_cur FOR select \* from Secretaire A inner join Personne B on A.IdPersonne = B.IdPersonne

where A.IdPersonne = idSecr;

RETURN secr\_cur;

end getSecretaire;

function getPatient(idPati Patient.IdPersonne%type) return select\_one\_cursor

as

pati\_cur select\_one\_cursor;

begin

OPEN pati\_cur FOR select \* from Patient A inner join Personne B on A.IdPersonne = B.IdPersonne

where A.NumeroPatient = idPati;

RETURN pati\_cur;

end getPatient;

function getSalle(idSall Salle.IdSalle%type) return select\_one\_cursor

as

salle\_cur select\_one\_cursor;

begin

OPEN salle\_cur FOR select \* from Salle where IdSalle = idSall;

RETURN salle\_cur;

end getSalle;

end SelectOne;

## Package Inserts (Inserts.sql)

Ce script crée le package Inserts qui contient des fonctions qui nous permette d’ajouter dans les tables.

create or replace package Inserts

is

function insertChirurgien(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,specialisation Chirurgien.Specialisation%type) return number;

function insertSecretaire(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,service\_secr Secretaire.Service\_secr%type) return number;

function insertPatient(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,numeroChambre Patient.NumeroChambre%type,numeroPatient Patient.NumeroPatient%type) return number;

function insertNotification(priorite Notifications.priorite%type,type\_notif Notifications.type\_notif%type,commentaire Notifications.commentaire%type,idPersonne Notifications.idPersonne%type) return number;

procedure insertReservation(idPersonne Reservation.IdPersonne%type,idSalle Reservation.IdSalle%type,numeroPatient Reservation.NumeroPatient%type,dateRes Reservation.DateRes%type,heureRes Reservation.HeureRes%type);

function insertSalle(numero\_salle Salle.numero%type,bloc\_sall Salle.bloc\_salle%type) return number;

id\_pers number;

id\_notif number;

id\_salle number;

cursor cur\_pers (nom\_var Personne.Nom%type, prenom\_var Personne.Prenom%type, date\_naiss\_var Personne.DateNaissance%type, num\_tel\_var Personne.NumeroTelephone%type, mot\_de\_passe\_var Personne.MotDePasse%type)

is

select \* from Personne

where Nom=nom\_var and Prenom=prenom\_var and DateNaissance=date\_naiss\_var and NumeroTelephone=num\_tel\_var and MotDePasse=mot\_de\_passe\_var;

var\_pers Personne%rowtype;

cursor cur\_salle (numero\_var Salle.numero%type,bloc\_salle\_var Salle.bloc\_salle%type) is SELECT \* FROM Salle where Numero=numero\_var and bloc\_salle=bloc\_salle\_var;

var\_salle Salle%rowtype;

end Inserts;

cursor cur\_salle (numero\_var Salle.numero%type,bloc\_salle\_var Salle.bloc\_salle%type) is SELECT \* FROM Salle where Numero=numero\_var and bloc\_salle=bloc\_salle\_var;

var\_salle Salle%rowtype;

end Inserts;

create or replace package body Inserts

is

function insertChirurgien(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,specialisation Chirurgien.Specialisation%type) return number

is begin

insert into Personne (Nom, Prenom, DateNaissance, NumeroTelephone, MotDePasse) values (nom, prenom, dateNaissance, numTelephone, motDePasse) returning IdPersonne INTO id\_pers;

insert into Chirurgien (IdPersonne, Specialisation) values (id\_pers, specialisation);

return id\_pers;

end insertChirurgien;

function insertSecretaire(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,service\_secr Secretaire.Service\_secr%type) return number

is begin

insert into Personne (Nom, Prenom, DateNaissance, NumeroTelephone, MotDePasse) values (nom, prenom, dateNaissance, numTelephone, motDePasse) returning IdPersonne INTO id\_pers;

insert into Secretaire (IdPersonne, service\_secr) values (id\_pers, service\_secr);

return id\_pers;

end insertSecretaire;

function insertPatient(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,numeroChambre Patient.NumeroChambre%type,numeroPatient Patient.NumeroPatient%type) return number

is begin

open cur\_pers(nom\_var => nom, prenom\_var => prenom, date\_naiss\_var => dateNaissance, num\_tel\_var => numTelephone, mot\_de\_passe\_var => motDePasse);

fetch cur\_pers into var\_pers;

if cur\_pers%notfound then

insert into Personne (Nom, Prenom, DateNaissance, NumeroTelephone, MotDePasse) values (nom, prenom, dateNaissance, numTelephone, motDePasse) returning IdPersonne INTO id\_pers;

insert into Patient (IdPersonne, numeroChambre,numeroPatient) values (id\_pers, numeroChambre,numeroPatient);

else

id\_pers := var\_pers.IdPersonne;

end if;

return id\_pers;

close cur\_pers;

end insertPatient;

function insertNotification(priorite Notifications.priorite%type,type\_notif Notifications.type\_notif%type,commentaire Notifications.commentaire%type,idPersonne Notifications.idPersonne%type) return number

is begin

insert into Notifications (priorite,type\_notif,commentaire,idPersonne) values (priorite,type\_notif,commentaire,idPersonne) returning IdNotification INTO id\_notif;

return id\_notif;

end insertNotification;

procedure insertReservation(idPersonne Reservation.IdPersonne%type,idSalle Reservation.IdSalle%type,numeroPatient Reservation.NumeroPatient%type,dateRes Reservation.DateRes%type,heureRes Reservation.HeureRes%type)

is begin

insert into Reservation (idPersonne,idSalle,numeroPatient,dateRes,heureRes) values (idPersonne,idSalle,numeroPatient,dateRes,heureRes);

end insertReservation;

function insertSalle(numero\_salle Salle.numero%type,bloc\_sall Salle.bloc\_salle%type) return number

is

begin

open cur\_salle(numero\_var => numero\_salle,bloc\_salle\_var => bloc\_sall);

fetch cur\_salle into var\_salle;

if cur\_salle%notfound then

insert into Salle (numero,bloc\_salle) values (numero\_salle,bloc\_sall) returning IdSalle INTO id\_salle;

else

id\_salle := var\_salle.idSalle;

end if;

return id\_salle;

close cur\_salle;

end insertSalle;

end Inserts;

## Package Updates (Updates.sql)

Ce script crée le package Updates qui contient des procédures qui nous permette de modifier des lignes dans les tables.

create or replace package Updates

is

procedure updateChirurgien(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,specialisation Chirurgien.Specialisation%type);

procedure updateSecretaire(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,service\_secr Secretaire.Service\_secr%type);

procedure updatePatient(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,numeroChambre Patient.NumeroChambre%type,numeroPatient Patient.NumeroPatient%type);

procedure updateReservation(idPersonne Reservation.IdPersonne%type,idSalle Reservation.IdSalle%type,numeroPatient Reservation.NumeroPatient%type,dateRes Reservation.DateRes%type,heureRes Reservation.HeureRes%type);

procedure updateSalle(numero Salle.numero%type,bloc\_salle Salle.bloc\_salle%type);

id\_pers number;

end Updates;

create or replace package body Updates

is

procedure updateChirurgien(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,specialisation Chirurgien.Specialisation%type)

is begin

update Personne set Nom = nom, Prenom = prenom, DateNaissance = dateNaissance, NumeroTelephone = numTelephone, MotDePasse = motDePasse returning IdPersonne INTO id\_pers;

update Chirurgien set IdPersonne = id\_pers, specialisation = specialisation;

end updateChirurgien;

procedure updateSecretaire(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,service\_secr Secretaire.Service\_secr%type)

is begin

update Personne set Nom = nom, Prenom = prenom, DateNaissance = dateNaissance, NumeroTelephone = numTelephone, MotDePasse = motDePasse returning IdPersonne INTO id\_pers;

update Secretaire set IdPersonne = id\_pers, service\_secr = service\_secr;

end updateSecretaire;

procedure updatePatient(nom Personne.nom%type,prenom Personne.prenom%type,dateNaissance Personne.DateNaissance%type,numTelephone Personne.NumeroTelephone%type,motDePasse Personne.MotDePasse%type,numeroChambre Patient.NumeroChambre%type,numeroPatient Patient.NumeroPatient%type)

is begin

update Personne set Nom = nom, Prenom = prenom, DateNaissance = dateNaissance, NumeroTelephone = numTelephone, MotDePasse = motDePasse returning IdPersonne INTO id\_pers;

update Patient set IdPersonne = id\_pers, numeroChambre = numeroChambre,numeroPatient = numeroPatient;

end updatePatient;

procedure updateReservation(idPersonne Reservation.IdPersonne%type,idSalle Reservation.IdSalle%type,numeroPatient Reservation.NumeroPatient%type,dateRes Reservation.DateRes%type,heureRes Reservation.HeureRes%type)

is begin

update Reservation set idPersonne = idPersonne,idSalle = idSalle,numeroPatient = numeroPatient,dateRes = dateRes,heureRes = heureRes;

end updateReservation;

procedure updateSalle(numero Salle.numero%type,bloc\_salle Salle.bloc\_salle%type)

is begin

update Salle set numero = numero,bloc\_salle = bloc\_salle;

end updateSalle;

end Updates;

## Package Deletes (Deletes.sql)

Ce script crée le package Deletes qui contient des procédures qui nous permette de supprimer des lignes dans les tables.

create or replace package Deletes

is

procedure deleteChirurgien(idChir Chirurgien.IdPersonne%type);

procedure deleteSecretaire(idSecr Secretaire.IdPersonne%type);

procedure deletePatient(idPati Patient.IdPersonne%type);

procedure deleteNotification(idNoti Notifications.IdNotification%type);

procedure deleteAllNotification;

procedure deleteReservation(idPers Reservation.IdPersonne%type,idSall Reservation.IdSalle%type,numPati Reservation.NumeroPatient%type);

procedure deleteSalle(idSall Salle.IdSalle%type);

end Deletes;

create or replace package body Deletes

is

procedure deleteChirurgien(idChir Chirurgien.IdPersonne%type)

is begin

delete from Chirurgien where IdPersonne = idChir;

end deleteChirurgien;

procedure deleteSecretaire(idSecr Secretaire.IdPersonne%type)

is begin

delete from Secretaire where IdPersonne = idSecr;

end deleteSecretaire;

procedure deletePatient(idPati Patient.IdPersonne%type)

is begin

delete from Patient where IdPersonne = idPati;

end deletePatient;

procedure deleteAllNotification

is begin

delete from Notifications;

end deleteAllNotification;

procedure deleteNotification(idNoti Notifications.IdNotification%type)

is begin

delete from Notifications where IdNotification = idNoti;

end deleteNotification;

procedure deleteReservation(idPers Reservation.IdPersonne%type,idSall Reservation.IdSalle%type,numPati Reservation.NumeroPatient%type)

is begin

delete from Reservation where IdPersonne = idPers and IdSalle = idSall and numeroPatient = numPati;

end deleteReservation;

procedure deleteSalle(idSall Salle.IdSalle%type)

is begin

delete from Salle where IdSalle = idSall;

end deleteSalle;

end Deletes;

# Problèmes rencontrés

## Crash du serveur GlassFish

Lors des tests de requêtes vers le service Web, notre serveur GlassFish subissait des crashs. Une exception était alors lancée. Celle-ci nous indiquait que la classe de driver Oracle permettant la connexion à la base de données était introuvable

Pour régler ce problème, nous avons dû copier le fichier contenant ces classes (classes12.jar) dans le répertoire /lib de GlassFish.

# Accès au programme

Pour vous connecter au programme vous pourrez utiliser soit le compte du Directeur, soit un compte de Secrétaire, soit un compte de Chirurgien.

## Comptes

### Directeur

Nom : Manfredi Ventura

Mot de passe : Manfrediv65

### Secrétaires

* Maurice Brasseur => Mauriceb67
* Françoise Renaud => Françoiser87

### Chirurgiens

* Thierry Hervieux => Thierryh73
* Carine Marquis => Carinem72