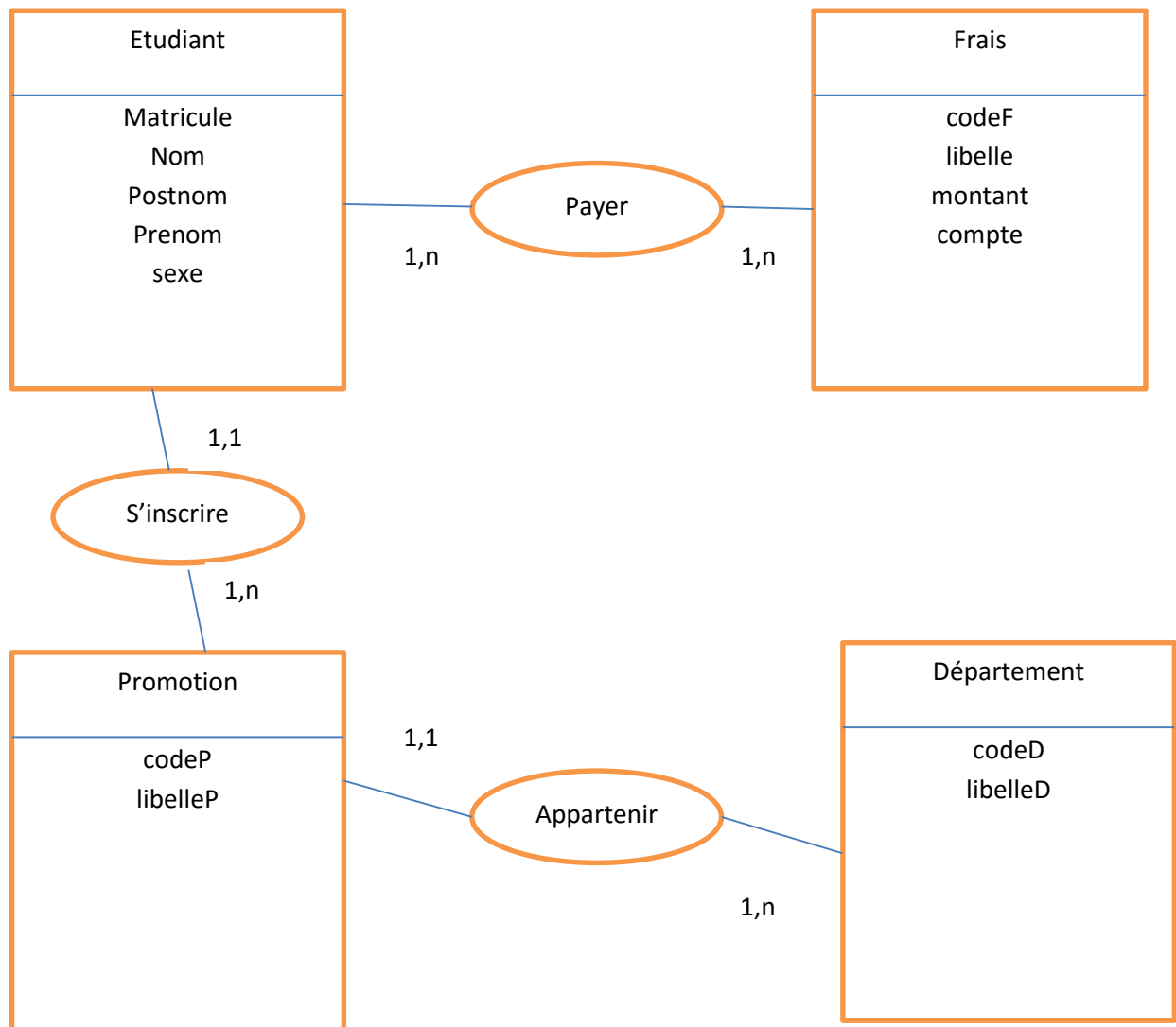
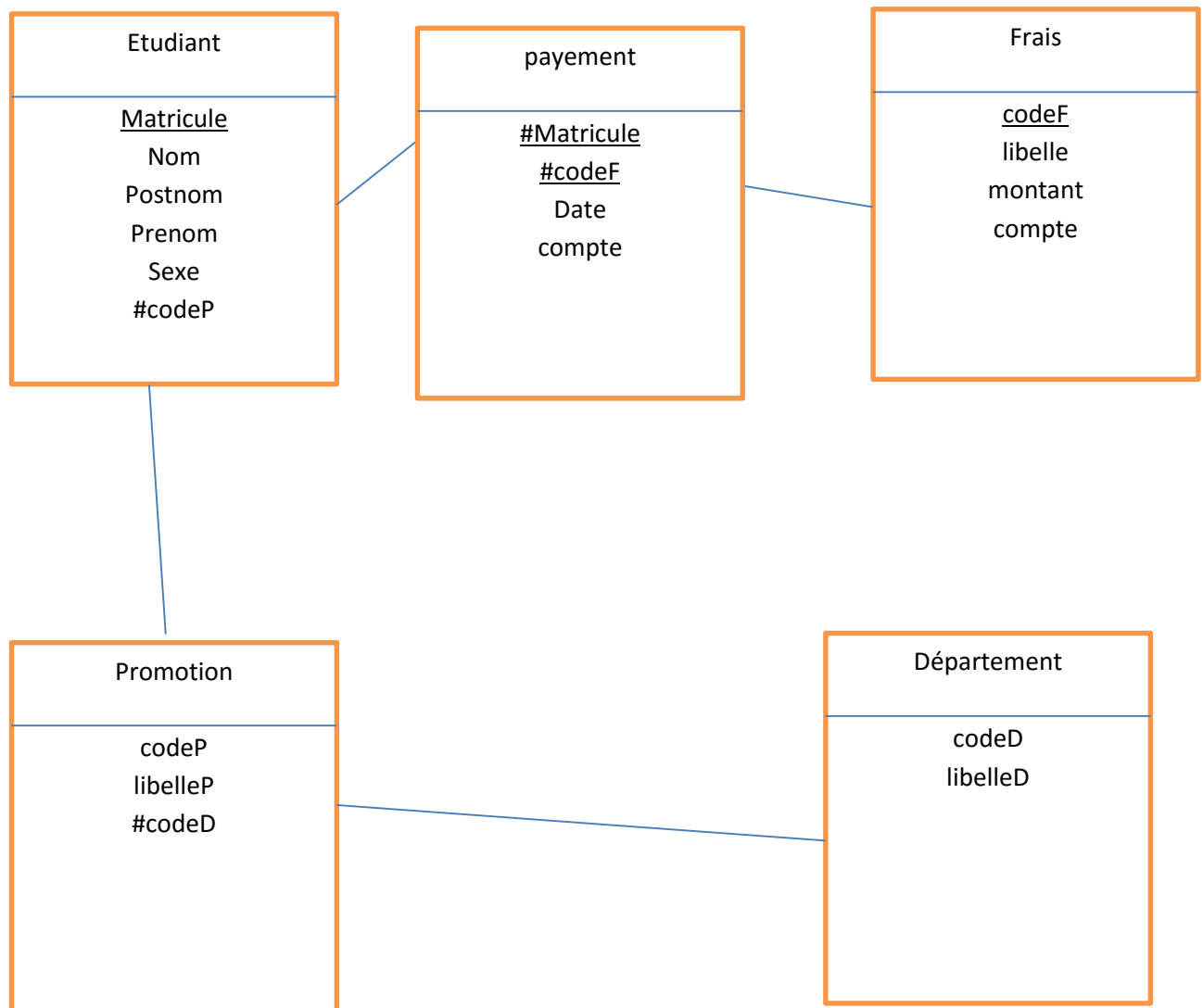


Le petit programme que nous allons essayer d'écrire ici servira à gérer le paiement des frais payés par les étudiants au sein de la faculté des lettres et sciences humaines.

Après Etudes, nous nous sommes mis d'accord sur le MCD suivant :



En traduisant en MLD, nous obtenons le schéma suivant :



Ce modèle a été implémenté sous le SGBD mysql dans une base des données du nom « gestfrais ».

Partie java :

Ce projet n'est pas du tout complet. Ceci n'est qu'une simple illustration de ce qui peut être fait.

1. Classe BaseDeDonnees : la classe qui permet de se connecter, d'interroger et de mettre à jour les informations de la base de données :

```
package gestionfrais;
import java.sql.*;
import javax.swing.*;
public class BaseDeDonnees {
    private static Statement st=null;

    private static void connecter()// la méthode permettant de se connecter à la base des
données
    try{
        Class.forName("com.mysql.jdbc.Driver");
        String url = "jdbc:mysql://localhost:3306/gestfrais";
        Connection cn=DriverManager.getConnection(url,"root","");
        st=cn.createStatement();
        ;
    }catch(Exception e){
        JOptionPane.showMessageDialog(null,
            "Echec de connexion à la base de données \n"+e,
            "Message de la classe DataBase", JOptionPane.WARNING_MESSAGE);
    }

    }

    public static int miseAJour(String requete)// la méthode permettant de mettre à jour les
donnée
    int i=0;
    connecter();
    try{
        i=st.executeUpdate(requete);
    }catch(Exception e){}
    return i;
    }
```

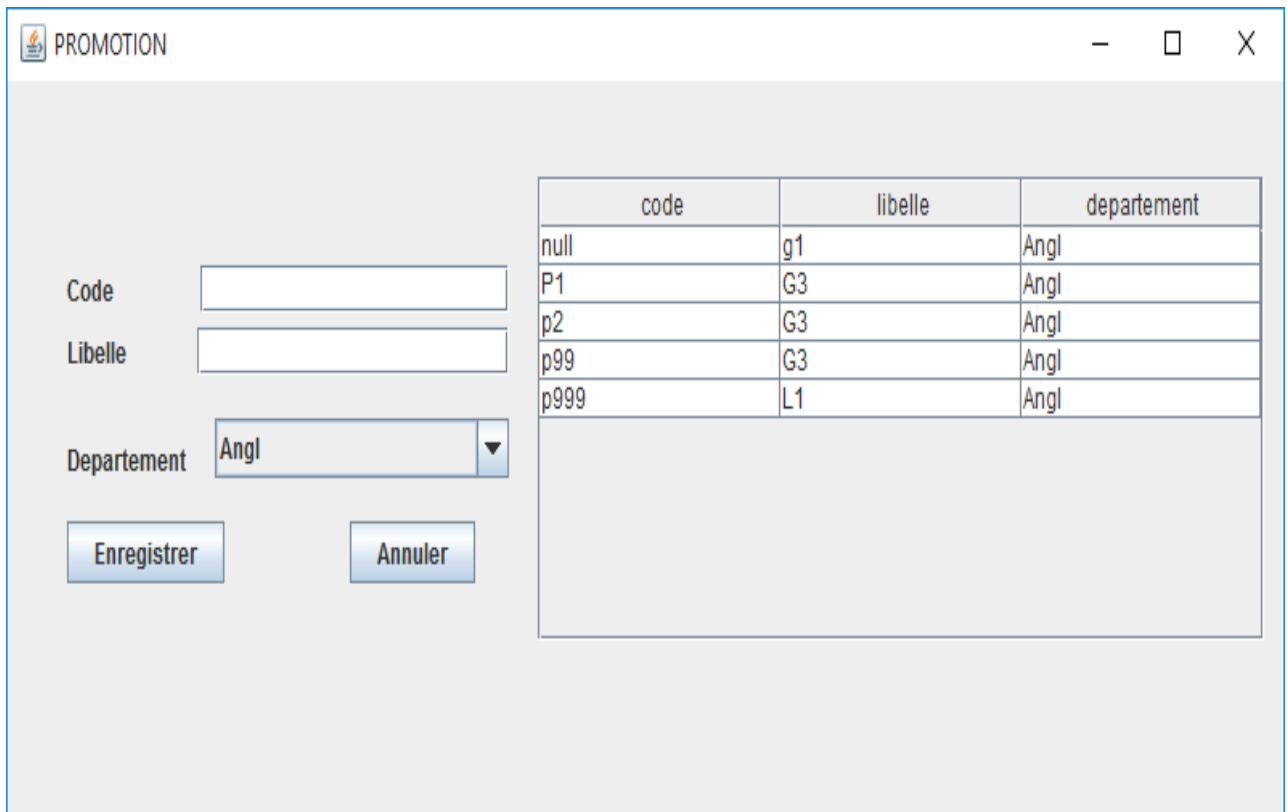
```

    public static ResultSet consulter(String requete){// la méthode permettant de consulter les
donnée
    ResultSet rs=null;
    connecter();
    try{
    rs=st.executeQuery(requete);
    }catch(Exception e){
    }
    return rs;
    }
}

```

Chacune des tables de notre base de données constitue une classe Java, nous allons ici illustrer le fonctionnement du programme en Interfaçant la promotion et en donnant la structure de sa classe et de la classe département qui y intervient car « le code du département » est une clé étrangère dans la table promotion :

Aspect graphique (Interface rapidement réalisée)



code	libelle	departement
null	g1	Angl
P1	G3	Angl
p2	G3	Angl
p99	G3	Angl
p999	L1	Angl

Dans cette petite illustration nous voyons l'existence des champs code et libelle ainsi qu'un combobox qui contient les libellés des départements. Rappelons que normalement c'est le code du département qui est la clé étrangère mais comme l'utilisateur ne peut retenir les codes, nous lui montrons les libellés. Lors de l'enregistrement nous récupéreront la clef.

Nous voyons aussi un Jtable qui contient toutes les promotions se trouvant dans la base des données et deux boutons (pour l'instant) enregistrer et annuler d'autres opérations comme la suppression, la modification, la recherche, le déplacement pouvant être ajoutées.

La classe promotion (uniquement avec le constructeur et la méthode d'enregistrement, seule opération que nous faisons pour l'instant)

```

public class Promotion {
    private String codeP;
    private String libelleP;
    private String codeD;

    public Promotion(String codeP,String libelleP,String codeD){
        this.codeD=codeD;
        this.libelleP=libelleP;
        this.codeP=codeP;
    }

    public int Enregistrer(){
        String req="insert into promotion
values("+"""+codeP+"""+",""+libelleP+"""+",""+codeD+"""+")";
        return BaseDeDonnees.miseAJour(req);
    }
}

```

Au démarrage, le combobox et le jtable sont rempli d'avance, ceci se fait grâce à une méthode se trouvant dans l'interface graphique que nous exécutons au démarrage. Comme plusieurs interfaces aurons à remplir le combo box et Jtable, nous avons écrit une classe pouvant

réaliser ces opérations dont les méthodes peuvent être appelées par toutes les interfaces. La classe s'appelle GestIntGrap. Voici sa structure :

```
package gestionfrais;

import java.sql.ResultSet;
import javax.swing.*;
import javax.swing.table.DefaultTableModel;

public class GesIntGrap {

    public static void remplirCombo(JComboBox b, ResultSet rs){
        b.removeAllItems();
        try{
            while(rs.next()){
                b.addItem(rs.getString(1));
            }
        }catch(Exception e){}
    }

    public static void remplirTable(JTable j, String req, String[] camp){
        DefaultTableModel d= new DefaultTableModel();

        for(int i=0;i<camp.length;i++){
            d.addColumn(camp[i]);
        }

        ResultSet r=BaseDeDonnees.consulter(req); // nous sommes directement passé
        exceptionnellement par la classe BaseDeDonnees.
        try{
            while(r.next()){
                String[] a=new String[camp.length];
                for(int i=0;i<a.length;i++){
                    a[i]=r.getString(i+1);
                }
                d.addRow(a);
            }
            j.setModel(d);
        }catch(Exception e){}
    }
}
```

Comme nous pouvons le voir, pour remplir le combo, nous avons besoin du combo lui-même et du ResultSet dont les éléments y seront placés. (voir méthode remplirCombo). Pour le Jtable, on aura besoin du Jtable en question, de la requête permettant de récupérer les éléments à y placer et d'un tableau contenant les champs qui seront les colonnes du Jtable.

Appelle de la classe GestIntGrap sur l'interface promotion, la méthode en question s'appelle **mise**:

```
public void mise(){
    ResultSet rs=Departement.listerLibelle();
    GesIntGrap.remplirCombo(dep, rs);
    String req="select promotion.codeP, promotion.libelleD,departement.libelleD from
    promotion, departement where promotion.codeD=departement.codeD";
    String[] camp={"code","libelle","departement"};
    GesIntGrap.remplirTable(tab, req, camp);
}
```

Comme nous pouvons nous rendre compte, le ResultSet que nous avons-nous donne les libelles des départements, occasion de voir la structure de la classe Département :

```
package gestionfrais;
import java.sql.*;

public class Departement {
    private String codeD;
    private String libelleD;

    public Departement(String codeD,String libelleD){
        this.codeD=codeD;
        this.libelleD=libelleD;
    }

    public int Enregistrer(){
        String req="insert into departement values('"+codeD+"','"+libelleD+"')";
        return BaseDeDonnees.miseAJour(req);
    }

    public static ResultSet listerLibelle(){
        ResultSet rs=BaseDeDonnees.consulter("select libelleD from departement");
        return rs;
    }

    public static int supprimer(String req){
```



```

int rep=BaseDeDonnees.miseAJour(req);
return rep;
}

public static String obtenirCode(String lib){
String co=null;
ResultSet rs=BaseDeDonnees.consulter("select codeD from departement where
libelleD='"+lib+"'");
try{
while(rs.next()){
co=rs.getString(1);
}
}catch(Exception e){}
return co;
}

}

```

Le bouton enregistrer :

```

String coded=Departement.obtenirCode(dep.getSelectedItem().toString());
Promotion p=new Promotion(code.getText(),lib.getText(),coded);
JOptionPane.showMessageDialog(null, p.Enregistrer()+" Enregistrement effectué");
mise();

```

Premièrement nous récupérons le code qui correspond au libellé sélectionné, en plus nous créons un objet de la classe « Promotion » et enfin nous invoquons la méthode **enregistrer**. Sans oublier le rappel de la méthode mise pour que le Jtable affiche le nouvel enregistrement.

NB : RESUME REALISE RAPIDEMENT POUVANT CONTENIR DES FAUTES D'ORTHOGRAPHE OU GRAMMATICALES