



Laboratoire IBD *No 1*

Laboratoire «JDBC»

Eric Lefrançois – 5 Octobre 2011

Sommaire

1	LES OBJECTIFS DU LABO	2
2	L'EXERCICE	2
3	LES ELEMENTS DU TUTORIAL JDARMONT	3
4	ANNEXE – JTABLE	6

1 Les objectifs du labo

Il s'agit d'apprendre à utiliser l'API JDBC au travers d'un exercice relativement simple.

2 L'exercice

Jetez un coup d'œil au tutorial SQL très sympathique proposé par le professeur d'informatique Jérôme Darmont de l'Université Lumière de Lyon 2 (France).

Adresse http du tutorial:

<http://eric.univ-lyon2.fr/~jdarmont/tutoriel-sql/>

Ce que vous devez faire ?



Recréer ce didacticiel sur la base d'un applicatif Java-JDBC s'appuyant sur un gestionnaire de base de données MySQL.

Vous sont donnés :

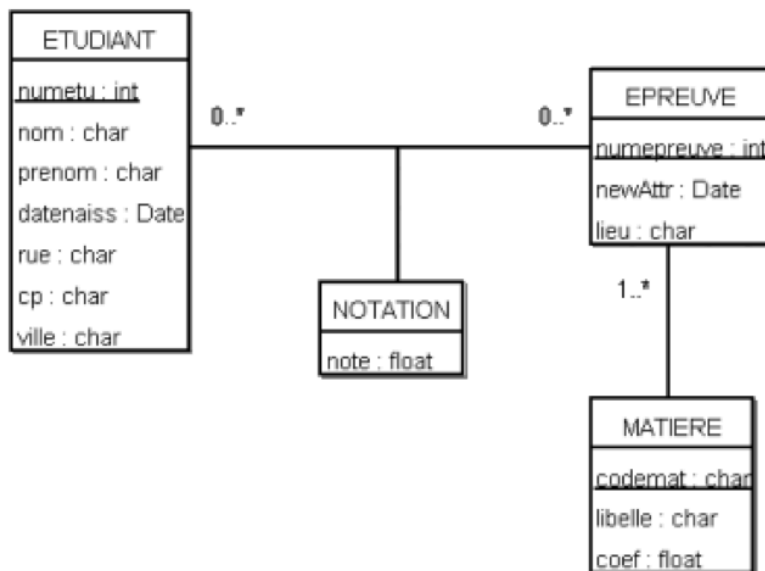
- Un dump de la base de données avec les 4 tables (structure + données)
- L'image du schéma UML de la base de données
- Les différents éléments de l'exercice (voir paragraphe suivant) :
 - Le schéma relationnel
 - La liste des questions
 - La liste des réponses correspondantes

3 Les éléments du tutorial JDarmont

Le schéma relationnel

etudiant (**numetu**, nom, prenom, datenaiss, rue, cp, ville)
 matiere (**codemat**, libelle, coef)
 epreuve (**numepreuve**, datepreuve, lieu, codemat#)
 notation (**numetu#**, **numepreuve#**, note)

Schéma UML



Questions SQL

1. Liste de tous les étudiants.
2. Liste de tous les étudiants, classée par ordre alphabétique inverse.
3. Libellé et coefficient (exprimé en pourcentage) de chaque matière.
4. Nom et prénom de chaque étudiant
5. Nom et prénom des étudiants domiciliés à Lyon
6. Liste des notes supérieures ou égales à 10
7. Liste des épreuves dont la date se situe entre le 1er janvier et le 30 juin 2004.
8. Nom, prénom et ville des étudiants dont la ville contient la chaîne "ll".
9. Prénoms des étudiants de nom Dupont, Durand ou Martin.
10. Somme des coefficients de toutes les matières

11. Nombre total d'épreuves
12. Nombre de notes indéterminées (NULL).
13. Liste des épreuves (numéro, date et lieu) incluant le libellé de la matière.
14. Liste des notes en précisant pour chacune le nom et le prénom de l'étudiant qui l'a obtenue.
15. Liste des notes en précisant pour chacune le nom et le prénom de l'étudiant qui l'a obtenue et le libellé de la matière concernée.
16. Nom et prénom des étudiants qui ont obtenu au moins une note égale à 20.
17. Moyennes des notes de chaque étudiant (indiquer le nom et le prénom).
18. Moyennes des notes de chaque étudiant (indiquer le nom et le prénom), classées de la meilleure à la moins bonne
19. Moyennes des notes pour les matières (indiquer le libellé) comportant plus d'une épreuve.
20. Moyennes des notes obtenues aux épreuves (indiquer le numéro d'épreuve) où moins de 6 étudiants ont été notés.

Les réponses

1. Liste de tous les étudiants.

```
SELECT * FROM etudiant
```

2. Liste de tous les étudiants, classée par ordre alphabétique inverse.

```
SELECT * FROM etudiant ORDER BY nom DESC
```

3. Libellé et coefficient (exprimé en pourcentage) de chaque matière.

```
SELECT libelle, coef*100 FROM matiere
```

4. Nom et prénom de chaque étudiant

```
SELECT nom, prenom FROM etudiant
```

5. Nom et prénom des étudiants domiciliés à Lyon

```
SELECT nom, prenom FROM etudiant WHERE ville='Lyon'
```

6. Liste des notes supérieures ou égales à 10

```
SELECT note FROM notation WHERE note>=10
```

7. Liste des épreuves dont la date se situe entre le 1er janvier et le 30 juin 2004.

```
SELECT * FROM epreuve WHERE datepreuve BETWEEN '2004-01-01' AND '2004-06-30'
```

8. Nom, prénom et ville des étudiants dont la ville contient la chaîne "ll".

```
SELECT nom, prenom, ville FROM etudiant WHERE ville LIKE '%ll%'
```

9. Prénoms des étudiants de nom Dupont, Durand ou Martin.

```
SELECT prenom FROM etudiant WHERE nom IN ('Dupont', 'Durand', 'Martin')
```

10.Somme des coefficients de toutes les matières

```
SELECT SUM(coef) FROM matiere
```

11.Nombre total d'épreuves

```
SELECT COUNT(*) FROM epreuve
```

12.Nombre de notes indéterminées (NULL).

```
SELECT COUNT(*) FROM notation WHERE note IS NULL
```

13.Liste des épreuves (numéro, date et lieu) incluant le libellé de la matière.

```
SELECT numepreuve, datepreuve, lieu, libelle
FROM epreuve, matiere
WHERE epreuve.codemat=matiere.codemat
```

14.Liste des notes en précisant pour chacune le nom et le prénom de l'étudiant qui l'a obtenue.

```
SELECT nom, prenom, note
FROM etudiant, notation
WHERE etudiant.numetu=notation.numetu
```

15.Liste des notes en précisant pour chacune le nom et le prénom de l'étudiant qui l'a obtenue et le libellé de la matière concernée.

```
SELECT nom, prenom, note, libelle
FROM etudiant, notation, epreuve, matiere
WHERE etudiant.numetu=notation.numetu AND
notation.numepreuve=epreuve.numepreuve AND
epreuve.codemat=matiere.codemat
```

16.Nom et prénom des étudiants qui ont obtenu au moins une note égale à 20.

```
SELECT DISTINCT nom, prenom
FROM etudiant, notation
WHERE etudiant.numetu=notation.numetu AND note=20
```

17.Moyennes des notes de chaque étudiant (indiquer le nom et le prénom).

```
SELECT nom, prenom, AVG(note)
FROM etudiant, notation
WHERE etudiant.numetu=notation.numetu
GROUP BY nom, prenom
```

18.Moyennes des notes de chaque étudiant (indiquer le nom et le prénom), classées de la meilleure à la moins bonne

```
SELECT nom, prenom, AVG(note) AS moyenne
FROM etudiant, notation
WHERE etudiant.numetu=notation.numetu
GROUP BY nom, prenom
```

```
ORDER BY moyenne DESC
```

19. Moyennes des notes pour les matières (indiquer le libellé) comportant plus d'une épreuve.

```
SELECT libelle, AVG(note) FROM matiere AS m, epreuve AS e, notation AS n
WHERE m.codemat=e.codemat AND e.numepreuve=n.numepreuve
GROUP BY libelle HAVING COUNT(DISTINCT e.numepreuve)>1
```

20. Moyennes des notes obtenues aux épreuves (indiquer le numéro d'épreuve) où moins de 6 étudiants ont été notés.

```
SELECT e.numepreuve, AVG(note)
FROM epreuve AS e, notation AS n
WHERE e.numepreuve=n.numepreuve AND note IS NOT NULL
GROUP BY e.numepreuve HAVING COUNT(*)<6
```

4 Annexe – JTable

Pour afficher le contenu d'une table – comme par exemple le résultat d'une requête SQL -, Java met à disposition un composant graphique relativement puissant : **JTable**.

Ce composant fonctionne selon le modèle MVC. Comme le montre l'exemple ci-dessous, ainsi que l'exemple no 3 (montré plus loin), on peut créer un composant JTable (la « vue ») à partir d'un modèle (la table à afficher). Les modifications apportées à table-modèle seront alors automatiquement mises à jour dans la vue correspondante.

```
import java.awt.*;
import javax.swing.*;
import javax.swing.table.*;

//-----
public class Amorce {
    public static void main(String [] args) {

        TableModel dataModel = new AbstractTableModel() {
            public int getColumnCount() { return 10; }
            public int getRowCount() { return 10;}
            public Object getValueAt(int row, int col) {
                return new Integer(row*col);
            }
        };

        JTable table = new JTable(dataModel);
        JScrollPane scrollpane = new JScrollPane(table);

        JFrame f = new JFrame ("Essai JTable");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```

        f.getContentPane().add(scrollpane, BorderLayout.CENTER);
        f.setSize(300, 300);
        f.setVisible(true);
    }
}

```

⇒ Qui donnera à l'affichage



A	B	C	D	E	F	G	H	I	J
0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9
0	2	4	6	8	10	12	14	16	18
0	3	6	9	12	15	18	21	24	27
0	4	8	12	16	20	24	28	32	36
0	5	10	15	20	25	30	35	40	45
0	6	12	18	24	30	36	42	48	54
0	7	14	21	28	35	42	49	56	63
0	8	16	24	32	40	48	56	64	72
0	9	18	27	36	45	54	63	72	81

Pour en savoir plus, consulter

<http://download.oracle.com/javase/6/docs/api/javax/swing/JTable.html>

Ou encore..

<http://www.java2s.com/Code/Java/Swing-JFC/Table.htm>

.. dont sont tirés les 3 exemples montrés ci-après.

Exemple 1 : Créer un objet `JTable` et l'afficher

```

import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;

public class Main {
    public static void main(String[] argv) throws Exception {

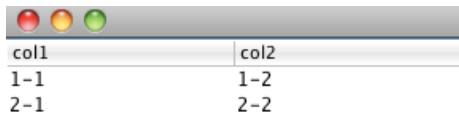
        Object[][] cellData = { { "1-1", "1-2" }, { "2-1", "2-2" } };
        String[] columnNames = { "col1", "col2" };

        JTable table = new JTable(cellData, columnNames);

        JFrame f = new JFrame();
        f.setSize(300,300);
        f.add(new JScrollPane(table));
        f.setVisible(true);
    }
}

```

⇒ Qui donnera à l’affichage



col1	col2
1-1	1-2
2-1	2-2

Exemple 2 : Construire une table à partir d’une liste de données et d’une liste de noms de colonnes

```
import java.util.Arrays;
import java.util.Vector;

import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;

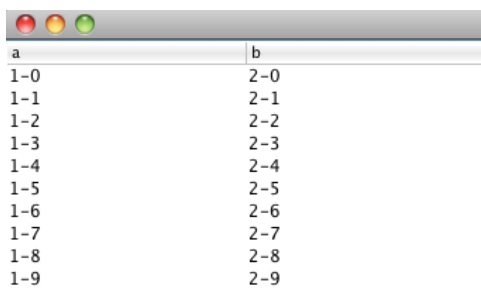
public class Amorce {
    public static void main(String[] argv) {
        Vector<Object> data = new Vector<Object>();
        for (int i = 0; i < 10; i++) {
            String[] values= {"1-"+i, "2-"+i};
            Vector<Object> rowData = new Vector<Object>(Arrays.asList(values));
            data.add(rowData);
        }

        String[] columnNames = {"a", "b"};

        Vector<Object> columnNamesV =
            new Vector<Object>(Arrays.asList(columnNames));

        JTable table = new JTable(data, columnNamesV);
        JFrame f = new JFrame();
        f.setSize(300, 300);
        f.add(new JScrollPane(table));
        f.setVisible(true);
    }
}
```

⇒ Qui donnera à l’affichage



a	b
1-0	2-0
1-1	2-1
1-2	2-2
1-3	2-3
1-4	2-4
1-5	2-5
1-6	2-6
1-7	2-7
1-8	2-8
1-9	2-9

Exemple 3 : Ajouter une ligne à un composant JTable

```
import javax.swing.JFrame;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

public class Main {
    public static void main(String[] argv) throws Exception {
        DefaultTableModel model = new DefaultTableModel();
        JTable table = new JTable(model);

        model.addColumn("Col1");
        model.addColumn("Col2");

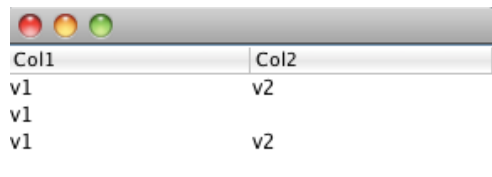
        model.addRow(new Object[] { "v1", "v2" });

        model.addRow(new Object[] { "v1" });

        model.addRow(new Object[] { "v1", "v2", "v3" });

        JFrame f = new JFrame();
        f.setSize(300, 300);
        f.add(new JScrollPane(table));
        f.setVisible(true);
    }
}
```

⇒ Qui donnera à l'affichage



Col1	Col2
v1	v2
v1	
v1	v2