**LORANS-CANO Gurvan**

**LE BERRE Samuel**

**BOUDET Alexandre**

**MARTEL Benoît**

**Département INFORMATIQUE 1ère année – Groupe A**

**Année 2016-2017**

**M2107 - PROJET DE PROGRAMMATION**

**Cahier des charges et d’analyse**

Création d’un éditeur de base de données



**IUT de Vannes**

M. LEFEVRE, professeur référent

Table des matières

**Aucune entrée de table des matières n'a été trouvée.**

Cahier des Charges

# **Contexte du projet**

Le but de ce projet est de réaliser un éditeur de base de données accessible via Java (JDBC). Ce dernier doit comporter une interface texte ainsi qu’une interface graphique avec des fonctionnalités de base que l’on peut retrouver dans de nombreux logiciel de traitement de texte ou autres (Enregistrer, Enregistrer sous, Ouvrir, Exporter sous un autre format). Par ailleurs il faudra implémenter les fonctionnalités de base d’une base de données (BD, métadonnées, données : tables, champs, enregistrement) et d’éventuelles statistiques. Le logiciel doit assurer le fait de pouvoir éditer, insérer, supprimer des tables ou des enregistrements. L’utilisateur pourra rechercher une donnée pour pouvoir la remplacer. Dans la mesure où l’utilisateur manipulera des données nous devrons établir une sécurité en ajoutant un login et un mot de passe. Et enfin des fonctionnalités optionnelles pourront être ajoutés tels que : un jeu pour apprendre le SQL, mini-quiz, etc.)

**Bêtes à cornes du projet :**

UTILISATEUR DE BASES DE DONNEES

FICHIERS SQL

EDITEUR DE BASES DE DONNEES

MANIPULER UNE BASE DE DONNEES

A qui rend-il service ?

Sur quoi agit-il ?

Dans quel but ?

# **II – Fonctionnalités du logiciel**

## De base

* Accéder à une base de données depuis Java
* Posséder une interface texte (console où sera affiché le résultat)
* Posséder une interface graphique (menu, boutons …)
* Ouvrir, enregistrer, enregistrer sous, nouveau fichier (stocker dans des fichiers où sont enregistré les requêtes et permettre leur lecture etc.)
* Visualiser des informations (BD, métadonnées, données, table, champs, fichier d’enregistrement) et statistiques (pourcentage pour des valeurs selon les requêtes)
* Édition, insertion, suppression d’une table, d’un t-uple et enregistrement automatique
* Sécurité pour la connexion à la base de données (attribuer droits aux utilisateurs)
* Posséder un champ de saisie de texte (pour l’écriture des requêtes SQL)

## Optionnelles

* Mini quiz pour apprendre le SQL
* Pouvoir créer une table juste avec un enchaînement de clic sur des boutons et de saisie de nom d’attribut
* Onglet aide (nouvelle fenêtre qui s’ouvre avec des commandes de bases, des conseils et autres)
* Représentation graphique (schéma relationnel, diagramme de classes)
* Changer le fond/les couleurs (esthétisme)
* Exporter sous un autre format
* Imprimer le fichier SQL qui a été chargé

# **III – Diagramme de GANTT**

## Version de base

**Description :**

Le projet de programmation durera 62 jours ouvrés à partir du lundi 27 Mars qui marque le début du projet.

La première étape de ce projet est l’étude préalable qui nous est demandée de réaliser afin d’évaluer les besoins et les fonctionnalités à apporter à notre application. Cette étape nécessitera 2 rendus, le cahier des charges et le cahier d’analyse. Le cahier des charges définit ce qui sera produit dans la globalité tandis que le cahier d’analyse est beaucoup plus précis dans sa description du programme qui sera rendu. Cette période dure 35 jours et se finira le vendredi 12 Mai.

Suit alors la réalisation du logiciel qui commencera le lundi 15 Mai et finira le lundi 19 Juin pour une durée totale de 21 jours.

Cette étape commencera par des recherches sur le JDBC (Java DataBase Connectivity), qui est un langage de java spécifique aux bases de données, que nous ne connaissons pas mais qui nous est vital pour la poursuite du travail.

Après cette phase commencera la programmation du back c’est-à-dire tous le programme qui s’exécutera derrière l’interface graphique.

* On commencera par créer un accès à la base de données, nous devrons donc créer une base de données à laquelle nous fournirons un accès à l’aide d’un profil.
* Puis nous programmerons l’extraction des données, on va faire en sorte de pouvoir récupérer toutes les données stockées dans la table afin de les utilisés dans le reste de notre logiciel.
* Ensuite on implémentera les fonctions qui permettent de créer, d’insérer et de supprimer une table ou un t-uple : peut-être à l’aide d’une fonction de JDBC.
* On utilise maintenant l’extraction des données afin de calculer des statistiques suivant les colonnes d’une table ou autres : il s’agira d’une requête exercée sur la base.
* On finira la programmation du back par la sécurisation de la connexion : il faut donner des droits aux comptes existant depuis la création de l’accès à une base afin qu’ils puissent modifier ou non une table, la supprimer ou même extraire ses données.

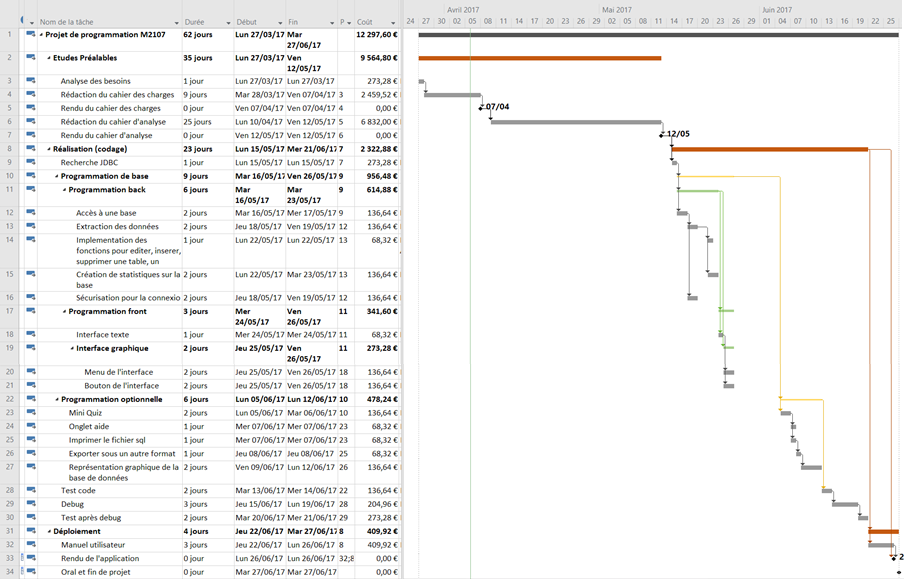
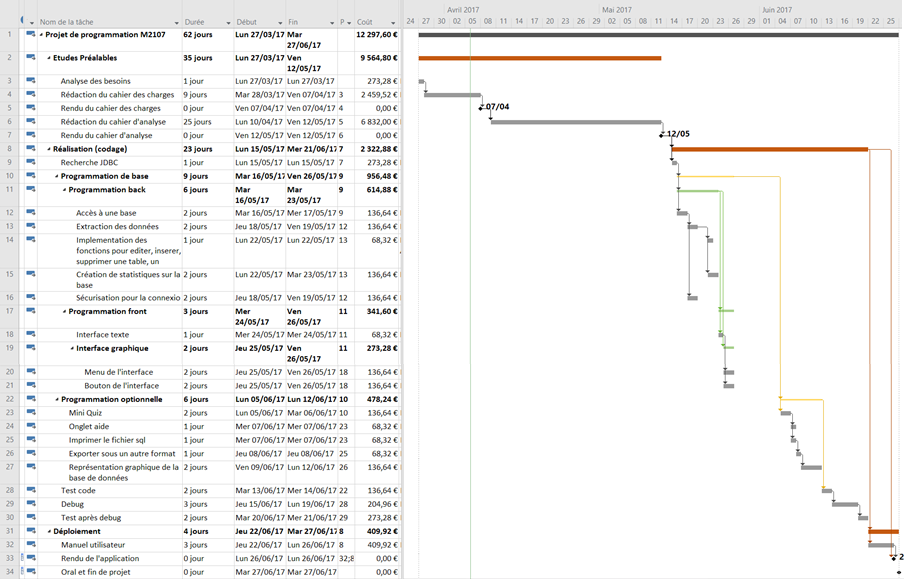
La programmation du font se déroulera juste après, le font représente ce que verra l’utilisateur du logiciel. Cette période de programmation commencera par :

* La réalisation d’une interface texte qui a pour but de nous renvoyer le résultat des requêtes dans un espace dédié à cet effet.
* La création d’une interface graphique :
* L’élaboration d’un menu regroupant les interactions les plus basiques de chaque logiciel comme nouveau, ouvrir, enregistrer ou encore enregistrer sous. Ces fonctions sont essentielles au bon fonctionnement du programme car si on ne peut sauvegarder nos requêtes alors la table elle-même devient inutile.
* Placement de boutons visant à améliorer la rapidité et la simplicité d’utilisation du logiciel.
* Prévoir le redimensionnement de la fenêtre que l’on peut modifier à l’aide d’un clic de souris afin qu’il n’y ait pas de problème en cas de changement de proportions.
* Maintenant que le logiciel est produit il faut :
* Faire des tests afin de trouver les fonctionnalités qui ne marche pas ou les erreurs qui se produisent lors de l’utilisation du logiciel.
* S’en suit une phase de débogage où l’on cherche à résoudre les bugs précédemment trouvés afin qu’il ne soit plus présent dans la version finale du logiciel.
* On procède à une seconde phase de tests qui n’est que la répétition de la première phase de tests après la correction des erreurs.

Le logiciel est maintenant prêt à être donné au client.

Vint l’étape du déploiement du logiciel, qui commence le 20 Juin. Cette étape consiste à la rédaction du manuel d’utilisation qui facilitera la prise en main et la compréhension de notre logiciel par l’utilisateur. Une fois le manuel rédigé, il faut rendre le logiciel et passer l’oral de fin de projet qui est une synthèse courte de 3 min sur notre expérience durant la conception d’un logiciel. Cette étape s’étend sur une durée de 6 jours

## Version optionnelle



**Description :**

Le diagramme de version optionnel reprend toutes les actions du diagramme de base mais diffère par leur temps d’exécution ainsi la programmation du back et du font dure 9 jours (12 jours de base) et le déploiement ne dure plus que 4 jours (6 jours de base). La phase de débogage (correction des problèmes) sera de 4 jours soit un jour de plus. Ce gain de temps sera utilisé pour ajouter une phase à l’étape de réalisation, il s’agit de la phase de programmation optionnelle qui se déroulera après la programmation du font.

L’étape durera 6 jours et durant ce temps on créera :

* Un mini quiz SQL qui vise à apprendre les bases du SQL à l’utilisateur tels que l’insertion, la création de table.
* L’ajout d’un bouton d’aide sur l’interface qui sera une sorte de tutoriel du logiciel avec des exemples, etc.
* La possibilité d’imprimer le fichier SQL qui est ouvert dans le logiciel.
* L’export sous un autre format : peut-être un fichier Excel ou autres formats inhabituel pour un fichier SQL.
* La représentation graphique de la base, il existe différentes façons de représenter une table (UML, etc.) il serait donc intéressant de les représenter pour l’utilisateur.

Cahier d’Analyse

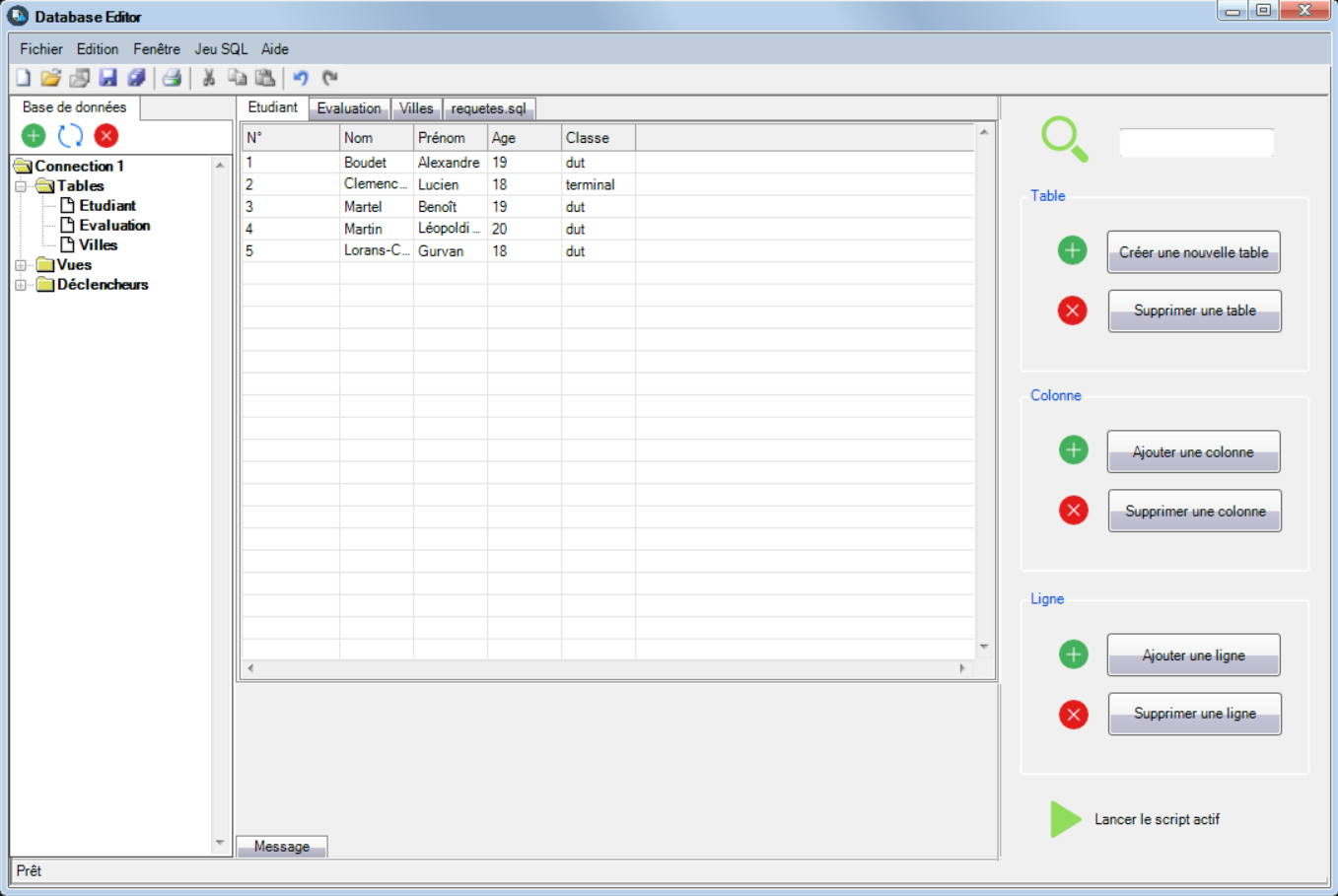
Cahier d’analyse

# **IHM et interactions avec l’utilisateur**

1. Interface de connexion a une base de données



1. Interface du programme général

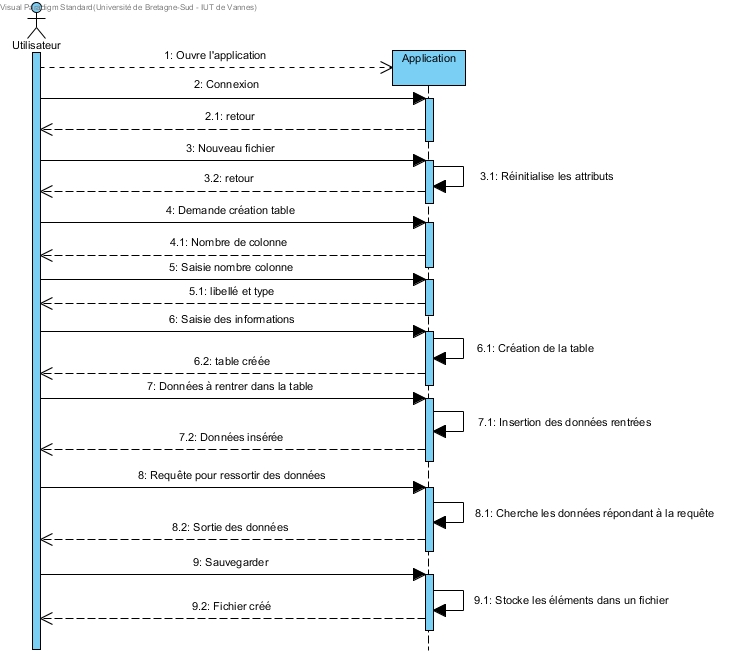


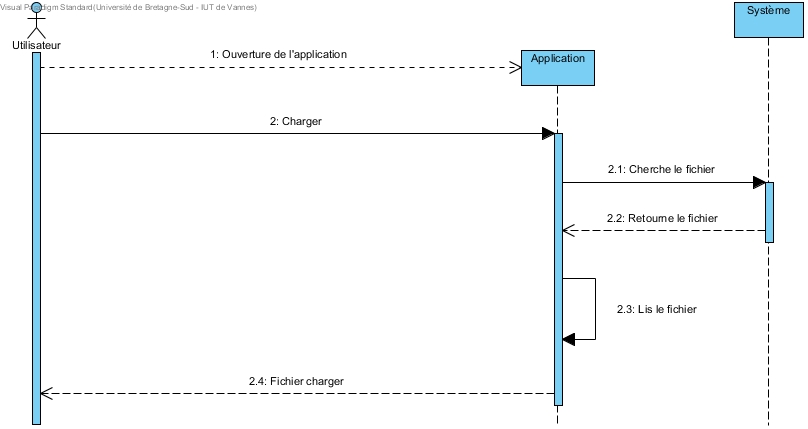
1. Interface du jeu SQL

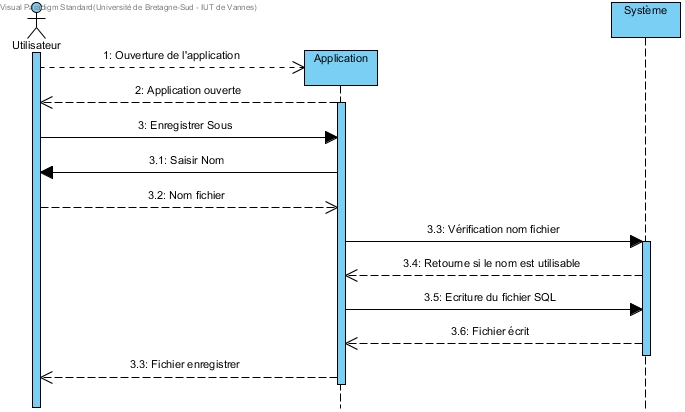


# **Diagrammes de séquence boite noire**

1. Diagramme de séquence général pour l’utilisation de l’application





1. Diagramme de séquence pour charger un fichier
2. Diagramme pour enregistrer un fichier

# **Plan de test fonctionnels**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonctionnalité** | **Scénario** | **Données utilisées** | **Résultat attendu** |
| accéder à la base de donnée | Appuyer sur un bouton qui demande des logins pour effectuer la connexion | le JDBC lance la connexion à la base de données suivant les logins rentrés par l’utilisateur | La connexion doit être effective |
| posséder une interface console | Quand l’utilisateur lance une requête par exemple la sortie doit s’afficher dans une zone de texte dédiée | Un JLabel ou un JTextField permettra de sortir le résultat de l’exécution | La console doit permettre d’afficher du texte suite à une requête ou à une interrogation de l’utilisateur ainsi qu’afficher les erreurs |
| posséder une interface graphique | Quand l’utilisateur lance le programme une interface graphique doit s’ouvrir | utilise des classes du javax.swing et des classes de java.awt pour afficher une interface graphique | L’interface s’ouvre avec les éléments bien intégré à leurs emplacement initiaux |
| avoir un menu : ouvrir  enregistrer, enregistrer-sous, nouveau, imprimer, quitter  fermer l’onglet | Quand l’utilisateur passe sa souris au dessus du menu, l’utilisateur doit pouvoir accéder aux options correspondantes | Utilise des onglets graphiques et des JLabel pour afficher le texte. | Quand la souris passe ou clique sur l’onglet, l’utilisateur doit avoir accès aux données qui correspondent |
| visualiser les informations et les données | Les informations des tables doivent être visible dans un panel principal | Utilise des éléments graphique de liste pour afficher de façon suffisamment claire les informations concernant les tables | Quand l’utilisateur double clique dans le JTreeMap ou autre éléments correspondant un nouvel onglet doit s’ouvrir pour afficher |
| édition, insertion, suppression | Quand l’application est ouverte, les éléments de modifications s’affiche en même temps que le reste de la Frame | Utilise des éléments graphique qui permette de modifier les tables | Les boutons s’affiche et chaque bouton possède la fonction adaptée |
| sécurité base de donnée (identifiant et mot de passe) | Quand l’utilisateur rentre son mot de passe la saisie doit être masquée | Utilise la saisie de l’utilisateur et ensuite envoie une demande à la base de donnée pour la vérifier | des astérisques s’affichent quand l’utilisateur rentre son mot de passe |
| posséder un champ de saisie SQL | Un panneau principal s’affiche quand le programme s’ouvre permettant | Utilise la saisie de l’utilisateur ainsi qu’une méthode de lecture pour afficher un fichier dans la zone dédiée qui sera éditable | Une zone de saisie s’ouvre et celle-ci est éditable |

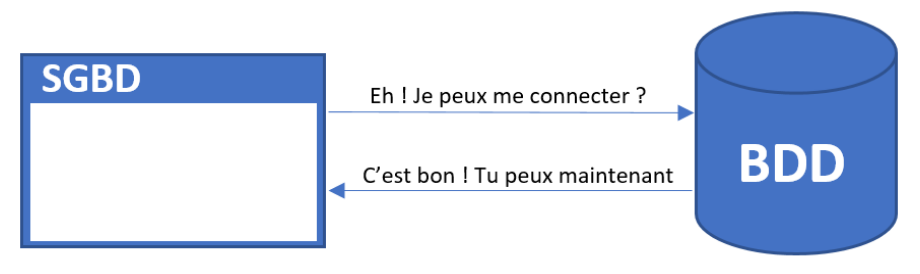
# **Diagramme de classe**

# **Utilisation de JDBC**

*Java DataBase Connectivity*, communément appelé **JDBC**

1. **Introduction aux JDBC**

Il s'agit en fait de classes Java permettant de se connecter et d'interagir avec des bases de données.

Le SGBD est un programme qui établit une connexion avec la BDD afin qu'ils puissent communiquer. Cela peut se schématiser par la figure suivante.

1. **Phase préliminaire :**

On a besoins de drivers pour se connecter à une base de données, il faut donc un fichier .jar qui correspond au fameux pilote et qui contient tout ce dont on a besoin pour se connecter à une base oracle.

Pour trouver le driver JDBC qu'il faut, une rapide recherche à l'aide du moteur de recherche répondra à nos attentes.

Placer l’archive téléchargé dans le dossier CLASSPATH inclus au projet.

1. **Connexion :**

Dans un premier temps, nous avons créé une instance de l'objet Driver présent dans le fichier .jar que nous avons téléchargé.

A ce stade, il existe comme un pont entre votre programme Java et votre BDD, mais le trafic routier n'y est pas encore autorisé : il faut qu'une connexion soit effective afin que le programme et la base de données puissent communiquer. Cela se réalise grâce à cette ligne de code :

→ Connection conn = DriverManager.getConnection(url, user, passwd);

L'URL de connexion est indispensable à Java pour se connecter à n'importe quelle BDD.

→  jdbc:oracle:thin:@localhost:1521:xe (par exemple)

Si une erreur survient pendant la connexion, une exception est levée.

1. **Interroger la base de données :**

* Le couple Statement - ResultSet

L'objet Statement permet d'exécuter des instructions SQL, il interroge la base de données et retourne les résultats. Ensuite, ces résultats sont stockés dans l'objet ResultSet, grâce auquel on peut parcourir les lignes de résultats et les afficher.

Ces dernières peuvent être de différents types :

* CREATE ;
* INSERT ;
* UPDATE ;
* SELECT ;
* DELETE.

L'objet Statement est fourni par l'objet Connection grâce à l'instruction conn.createStatement()

Un objet de type ResultSetMetaData permet de récupérer les métadonnées d’une requête. Cet objet de métadonnées permet de récupérer des informations très utiles, comme :

* le nombre de colonnes d'un résultat ;
* le nom des colonnes d'un résultat ;
* le type de données stocké dans chaque colonne ;
* le nom de la table à laquelle appartient la colonne (dans le cas d'une jointure de tables) ;
* etc.
* Modifier des données

Pendant la lecture, on peut utiliser des méthodes comme *res.updateInt().*

Ces méthodes de mise à jour des données prennent deux paramètres :

* le nom de la colonne (String) ;
* la valeur à attribuer à la place de la valeur existante (cela dépend de la méthode utilisée).

Changer la valeur d'un champ est donc très facile. Cependant, il faut, en plus de changer les valeurs, valider ces changements pour qu'ils soient effectifs : cela se fait par la méthode updateRow(). De la même manière, vous pouvez annuler des changements grâce à la méthode cancelRowUpdates(). Sachez que si vous devez annuler des modifications, vous devez le faire avant la méthode de validation, sinon l'annulation sera ignorée.