Алгоритм

Для виконання лабораторної роботи була встановлена на комп’ютер і конфігурована СУБД. У даній лабораторній роботі була використана вільна СУБД MySQL. У встановленій СУБД створимо нову базу даних з таблицями для зберігання об’єктів відповідно до заданої предметної області, а саме для варіанта №1 було створено дві таблиці: таблицю країн і таблицю міст. Адміністрування СУБД MySQL здійснювалось через консольний додаток MySQL Command Line Client. Для створення нової бази даних була використана команда CREATE DATABASE і була створена нова база даних з назвою MAP. Створення таблиць здійснювалось за допомогою запиту CREATE TABLE і були створені таблиці COUNTRIES і CITIES. Для забезпечення унікальності ідентифікаторів, а також цілісності зв’язків між сутностями, були використані первинні та зовнішні ключі. За допомогою SQL-оператора INSERT було заповнено таблиці даними. Вибірка даних з таблиць здійснювалась за допомогою оператора SELECT.

Для доступу до бази даних MySQL через інтерфейс JDBC потрібна була бібліотека mysql-connector-java. Дана бібліотека була завантажена і в ній був знайдений jar-файл mysql-connector-java.5.6.7-bin.jar. Даний jar-файл містить драйвер com.mysql.jdbc.Driver, що забезпечує роботу з СУБД MySQL через інтерфейс JDBC. Даний jar-файл був підключений до мого Java-проекту. На початку роботи програми драйвер com.mysql.jdbc.Driver був завантажений через Class.forName.

Установка з’єднання з базою даних виконується на початку роботи програми. Отримане з’єднання використовувалось в процесі роботи програми при виконанні операцій над даними. Управління з’єднанням здійснювалося через клас java.sql.Сonnection. Для установки з’єднання використовувався метод getConnection класу java.sql.DriverManager. Метод приймає на вхід параметри з’єднання:

* url, який визначає протокол і ідентифікатор бази даних (для JDBC типу 4 вказується мережеве ім’я сервера, порт, ім’я БД)
* ім’я користувача (для зручності я працював під адміністратором: root)
* пароль

і в разі успіху повертає об’єкт класу Connection. У разі виникнення помилки метод getConnection генерував виняток SQLException. У даній лабораторній роботі у першому варіанті було встановлено з’єднання з базою даних map, що розміщується на локальному сервері (localhost, порт 3306), було використано ім’я користувача root і пароль password.

Виконання запитів SQL здійснювалося через клас java.sql.Statement. Об’єкт класу Statement створювався в рамках заданого з’єднання за допомогою методу createStatement класу Connection. Зміна даних здійснювалася через метод executeUpdate класу Statement. Метод приймає на вхід рядок із запитом SQL типу INSERT, UPDATE або DELETE і повертає кількість змінених записів в таблиці. Вибірка даних здійснювалася за допомогою методу executeQuery. Метод приймає на вхід SQL-запит виду SELECT і повертає об’єкт класу ResultSet, що дозволяє обробляти результуючі таблиці вибірок. Клас ResultSet містить наступні основні методи:

* next – переходить до нового запису у вибірці (перед роботою з вибіркою необхідно один раз викликати даний метод для переходу до першого запису);
* getXXXX (getInt, getString, і т.д.) – методи повертають значення заданого стовпчика вибірки для поточного запису вибірки;
* close – завершує роботу з вибіркою.

У разі виникнення помилки методи класу Statement генерували виняток SQLException.