JDBC

JDBC (Java DataBase Connectivity — соединение с базами данных на Java) предназначен для взаимодействия Java-приложения с различными системами управления базами данных (СУБД). Всё движение в JDBC основано на драйверах которые указываются специально описанным URL.

Существует несколько видов СУБД по способу организации хранения данных:

Иерархические. Данные организованы в виде древовидной структуры. Пример — файловая система, которая начинается с корня диска и далее прирастает ветвями файлов разных типов и папок разной степени вложенности.

Сетевые. Видоизменение иерархической, у каждого узла может быть больше одного родителя.

Объектно-ориентированные. Данные организованы в виде классов/объектов с их атрибутами и принципами взаимодействия согласно ООП.

Реляционные. Данные этого вида СУБД организованы в таблицах. Таблицы могут быть связаны друг с другом, информация в них структурирована.

Графовые. Основываются на системе графом. Можно представить в виде схемы связанных между собой элементов.

SQL (https://biblprog.org.ua/ru/open_server/download/)

Внешние программы формируют запросы к СУБД на языке управления данными Structured Query Language. Что такое SQL и чем отличается от привычных языков программирования? Одна из особенностей SQL – декларативность. То есть, SQL — декларативный язык. Это значит, что, вбивая команды, то есть, создавая запросы к SQL-серверу, мы описываем, что именно хотим получить, а не каким способом. Посылая серверу запрос SELECT * FROM CUSTOMER (приблизительный перевод с SQL на русский: «сделать выборку из таблицы COSTUMER, выборка состоит из всех строк таблица»), мы получим данные по всем пользователям. Совершенно неважно, как и откуда сервер загрузит и сформирует интересующие нас данные. Главное – правильно сформулировать запрос.

Ссылки

Драйвер - http://www.java2s.com/Code/Jar/c/Downloadcommysqljdbc515jar.htm#google_vignette

Windows

Open Server - https://biblprog.org.ua/ru/open_server/download/

Mac OS

MAMP - https://www.mamp.info/en/downloads/

Основные компоненты JDBC Api включают:

DriverManager:

Это класс, использующийся для управления списком Driver (database drivers).

Driver:

Это интерфейс, использующийся для соединения коммуникации с базой данных, управления коммуникации с базой данных. Когда загружается Driver, программисту не нужно конкретно вызывать его.

Connection :

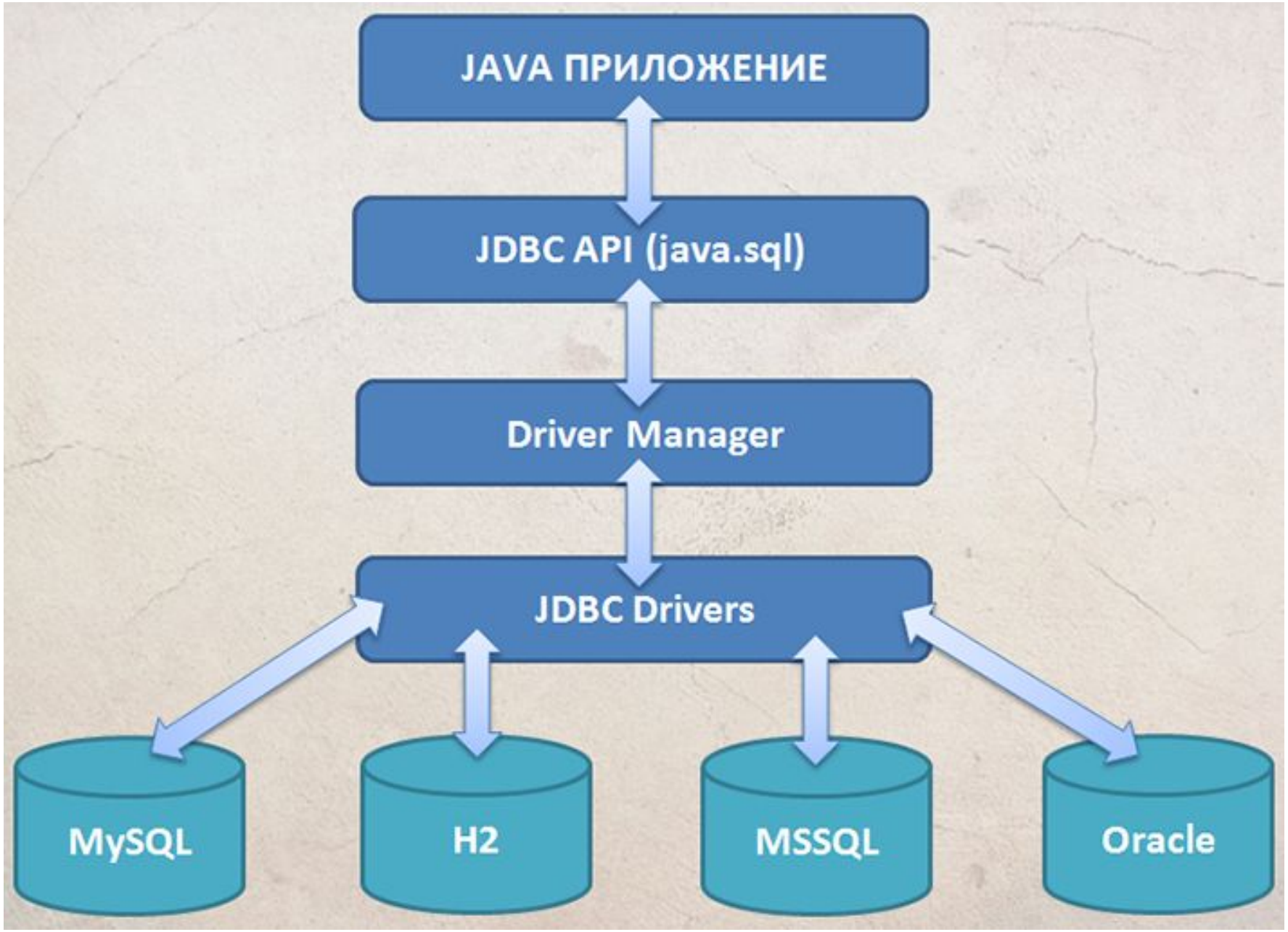
Интерфейс со всеми методами связи с базой данных. Он описывает коммуникационный контекст. Вся связь с базой данных осуществляется только через объект соединения (connection).

Statement :

Это интерфейс, включающий команду SQL отправленный в базу данных для анализа, обобщения, планирования и выполнения.

ResultSet :

ResultSet представляет набор записей, извлеченных из-за выполнения запроса.


JAVA ПРИЛОЖЕНИЕ
JDBC API (java.sql)
Driver Manager
JDBC Drivers
MySQL
H2
MSSQL
Oracle

```
package com.company;

import java.sql.*;

public class Main {

    public static void main(String[] args) throws ClassNotFoundException {

        String url = "jdbc:mysql://localhost/app";

        String username = "root";

        String password = "vertrigo";

        Class.forName("com.mysql.jdbc.Driver");

        try (Connection conn = DriverManager.getConnection(url, username, password)) {

            System.out.println("Connection to Store DB successful!");

            PreparedStatement stmt = conn.prepareStatement("UPDATE users SET login=?, password=? WHERE id=?");

//          PreparedStatement stmt = conn.prepareStatement("SELECT * FROM users");

//          PreparedStatement stmt = conn.prepareStatement("INSERT INTO users (login, password) VALUES (?,?)");

//          PreparedStatement stmt = conn.prepareStatement("DELETE FROM users WHERE login=? AND password=?");

            stmt.setString(1, "Roman");

            stmt.setString(2, "232323");

            stmt.setInt(3, 3);

            stmt.execute();

//          String name;

//          while (rs.next()) {

//              name = rs.getString("password");

//              System.out.println("----------------");

//              System.out.println(name + "\n");

//          }

        } catch (SQLException e) {

            e.printStackTrace();

        }


    }

}
```