JDBC

JDBC – технологія, яка дозволяє Java програмам працювати з базами даних.

Доволі гнучка. Можна почати розробляти програму з використанням однієї СУБД, потім замінити її на іншу, і все буде працювати.

Основні поняття

- **POJO** – **P**lain **O**ld **J**ava **O**bject. Клас, який зберігає лише дані, але не містить методів для їх обробки.
- **Entity** – сутність (*сущность*). Окремий об’єкт, який являє собою окремий елемент предметної області. Зазвичай це POJO об’єкт. Наприклад, Developer – це Entity, що представляє розробника.
- **CRUD** – типовий набір операцій для роботи з Entity. **C**reate, **R**ead, **U**pdate, **D**elete

Підготовка – створення класів

- Створити POJO клас Developer з полями ID, firstName, lastName, specialty
- Створити клас Storage. В цьому класі буде інкапсульована вся робота з БД

Клас Storage

- В конструкторі налаштувати з’єднання з БД, використовуючи **DriverManager.getConnection(url, user, pass)**
- Отримати екземпляр Statement
- Написати метод, який створює таблицю в БД (**CREATE TABLE IF NOT EXISTS**...)

Клас Storage – перехід до CRUD

- Створити метод **Developer createDeveloper(Developer developer)**, який додає нового розробника в БД;
- Створити метод Developer getDeveloper(long id), який повертатиме розробника по ID
- Створити метод void **updateDeveloper(Developer developer)**, який оновлює розробника.
- Створити метод **void deleteDeveloper(Developer developer)**, який видаляє розробника з БД.

Storage - розширення

- Створити метод **List<Developer> listDevelopers()**, який повертає всіх розробників з БД;
- Створити метод **void addDevelopers(List<Developer> developers)**, який додає групу розробників. Використати batch update
- Створити метод **void clearDatabase()**, який очищує БД. Використати **TRUNCATE** sql команду.

Управління БД

Написати примітивну консоль управління, яка підтримує наступні команди:

- **create** <first_name, last_name, specialty>
- **read** <developer_id>
- **list**
- **delete** <developer_id>
- **clear**
- **update** <developer_id, first_name, last_name, specialty>

Все управління відбувається через клас Scanner, вивід йде в консоль методом System.out.println

Storage - рефакторинг

- Перевести CRUD методи (**createDeveloper, getDeveloper, updateDeveloper, deleteDeveloper**) на використання PreparedStatement
- Використати транзакції у методі **void addDevelopers(List<Developer> developers)**. Додаються або всі розробники, або ніякий.

Розширення моделі - Project

- Створити POJO клас Project по аналогії з класом Developer. Поля класу – id, developerId, name, description, List<Developer> developers.
- До класу Developer додати поле List<Project> projects
- Створити таблиці PROJECT, PROJECT_DEVELOPER із зв’язками
- В клас Storage додати методи для роботи з Project, по аналогії з методами для роботи з Developer. Врахувати, що коли ми додаємо Project, нам потрібно зберегти і всіх зв’язаних з ним Developers.
- Модифікувати CRUD методи для Developer (коли ми робимо якісь дії з Developer, потрібно оновлювати/видаляти/додавати всі пов’язані з ним Project)