2 Лабораторная

## Подготовка базы данных.

Необходимо создать таблицы БД по примерам, указанным в приложении А. Вариант выбирайте, беря по модулю 4 свою фамилию в списке группы.

СУБД можно выбрать любую.

Нужно подготовить скрипт DDL, который описывает создание базы данных и таблиц, используемых в этой лабораторной.

## JDBC

Реализовать подключение к БД посредством JDBC (Java DataBase Connectivity).

Согласно интерфейсу, реализовать методы ниже для обеих таблиц базы данных (где T – это имя вашей сущности):   
public T save(T entity);

public void deleteById(long id);

public void deleteByEntity(T entity);

public void deleteAll();

public T update(T entity);

public T getById(long id);

public List<T> getAll();

Также необходимо реализовать метод, который возвращает все дочерние сущности по идентификатору родительской:

public List<T> getAllByVId(); Здесь T – это дочерняя сущность, V – родительская.

## Hibernate

Реализовать подключение к БД посредством Hibernate - популярное ORM – решение для Java.

Согласно интерфейсу, реализовать методы ниже для обеих таблиц базы данных (где T – это имя вашей сущности):   
public T save(T entity);

public void deleteById(long id);

public void deleteByEntity(T entity);

public void deleteAll();

public T update(T entity);

public T getById(long id);

public List<T> getAll();

Также необходимо реализовать метод, который возвращает все дочерние сущности по идентификатору родительской:

public List<T> getAllByVId(); Здесь T – это дочерняя сущность, V – родительская.

## MyBatis

Реализовать подключение к БД посредством MyBatis – Java фреймворк для работы с сущностями БД.

Согласно интерфейсу, реализовать методы ниже для обеих таблиц базы данных (где T – это имя вашей сущности):   
public T save(T entity);

public void deleteById(long id);

public void deleteByEntity(T entity);

public void deleteAll();

public T update(T entity);

public T getById(long id);

public List<T> getAll();

Также необходимо реализовать метод, который возвращает все дочерние сущности по идентификатору родительской:

public List<T> getAllByVId(); Здесь T – это дочерняя сущность, V – родительская.

Для последнего метода необходимо описать условие: если дочерних сущностей больше 5, то нужно вернуть первые 5.

## Сравнение

Необходимо отправить в бд запрос на добавление 100 сущностей и сравнить время, за которое это будет выполнено всеми тремя способами. Затем отправить запрос на получение этих же самых 100 сущностей, (можно использовать метод getAll()), и также сравнить время, за которое это будет сделано тремя различными способами.

## Приложение А.

Ниже представлены названия таблиц для БД с атрибутами каждой таблицы (название, тип данных).

Вариант 0.

Хозяин

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Имя | Varchar |
| Дата рождения | Date |

Котик

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Имя | Varchar |
| Дата рождения | Date |
| Порода | Varchar |
| Цвет | Varchar |
| Владелец | Long |

* В таблице «Котик» атрибут «Хозяин» является вторичным ключом для атрибута «Идентификатор» из таблицы «Хозяин»;
* Атрибуты «Идентификатор» в обеих таблицах должны быть первичными ключами;
* Атрибут «Цвет» таблицы «Котик» должен представлять из себя ограниченное количество значений: белый, рыжий, коричневый, серый, черный.

Вариант 1.

Улица

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Название | Varchar |
| Почтовый индекс | Int |

Дом

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Название | Varchar |
| Дата постройки | Date |
| Количество этажей | Int |
| Тип здания | Varchar |
| Улица | Long |

* В таблице «Дом» атрибут «Улица» является вторичным ключом для атрибута «Идентификатор» из таблицы «Улица»;
* Атрибуты «Идентификатор» в обеих таблицах должны быть первичными ключами;
* Атрибут «Тип здания» таблицы «Дом» должен представлять из себя ограниченное количество значений: жилой помещение, коммерческое помещение, гараж, подсобное помещение.

Вариант 2.

Марка автомобиля

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Название | Varchar |
| Дата основания | Date |

Модель автомобиля

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Название | Varchar |
| Длина | Int |
| Ширина | Int |
| Тип кузова | Varchar |
| Марка автомобиля | Long |

* В таблице «Модель автомобиля» атрибут «Марка автомобиля» является вторичным ключом для атрибута «Идентификатор» из таблицы «Марка автомобиля»;
* Атрибуты «Идентификатор» в обеих таблицах должны быть первичными ключами;
* Атрибут «Тип кузова» таблицы «Модель автомобиля» должен представлять из себя ограниченное количество значений: седан, хэтчбек, универсал, купе, пикап, родстер.

Вариант 3.

Сотрудник

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Имя | Varchar |
| Дата рождения | Int |

Задача

|  |  |
| --- | --- |
| Название поля | Тип данных поля |
| Идентификатор | Long |
| Название | Varchar |
| Дедлайн | Date |
| Описание | Int |
| Тип задачи | Varchar |
| Человек | Long |

* В таблице «Задача» атрибут «Сотрудник» является вторичным ключом для атрибута «Идентификатор» из таблицы «Сотрудник»;
* Атрибуты «Идентификатор» в обеих таблицах должны быть первичными ключами;
* Атрибут «Тип задачи» таблицы «Задача» должен представлять из себя ограниченное количество значений: новый функционал, ошибка, улучшение, аналитика.