Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОННИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Лабораторная работа №4

Создание контроллера для работы с BerkeleyDB

Студент: К.В. Горбачевский

Преподователь: А.И. Крюков

МИНСК 2024

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc181962152)

[1 СОЗДАНИЕ UML-ДИАГРАММЫ 5](#_Toc181962153)

[1.1 Предметная область 5](#_Toc181962154)

[1.2 Типы объектов 5](#_Toc181962155)

[1.3 Атрибуты объектов 6](#_Toc181962156)

[1.4 Типы связей 6](#_Toc181962157)

[2 КОНТРОЛЛЕР 7](#_Toc181962158)

[ЗАКЛЮЧЕНИЕ 11](#_Toc181962159)

# ВВЕДЕНИЕ

В лабораторной работе выполняется концептуального проектирование БД с использованием UML-модели представления данных (модели «сущность- связь»). Требуется разработать UML-модель данных с учетом семантических ограничений заданной предметной области и представить модель в виде UML- диаграммы.

Темой данной лабораторной работы является разработка UML-диаграммы сущностей и связей в организации «Студия звукозаписи».

«Студия звукозаписи» - это стандартная модель, работающая по принципу клиент, услуга, продавец. В качестве продавца выступает владелец студии и напрямую с клиентом он не взаимодействует. В качестве клиента выступает музыкальный исполнитель или группа, которая хочет арендовать помещение на короткий срок (не более суток), в определенном жанре.

# СОЗДАНИЕ UML-ДИАГРАММЫ

Исходное задание: Создать концептуальную модель организации «Студия звукозаписи»и представить сущности и связи в виде ER-диаграммы.

Концептуальная UML-диаграмма представлена на рисунке 1.

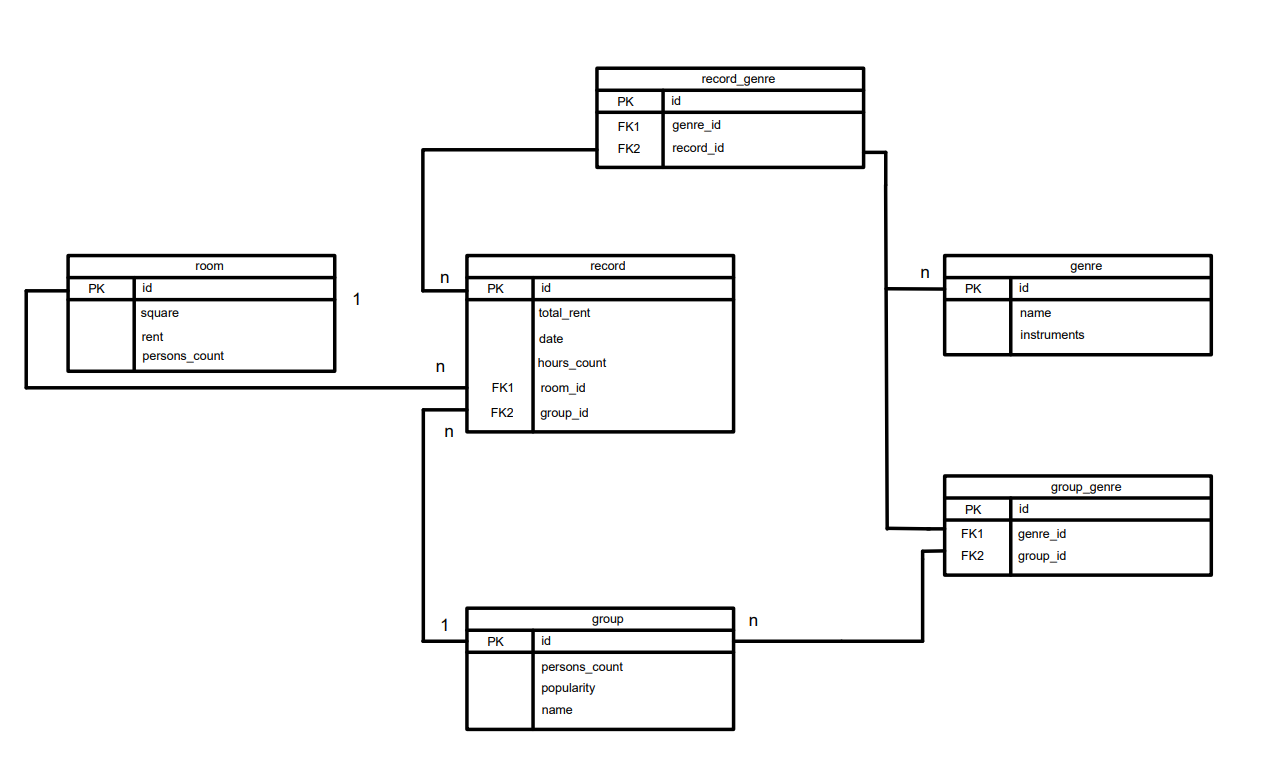


Рисунок 1 – UML-диаграмма

## Предметная область

Предметная область – «Студия звукозаписи». Модели по типу «клиент-продавец». Предоставляемая услуга – запись голоса или инструментов, аренда инструментов и комнат.

## Типы объектов

Для модели «Студия звукозаписи» было выделено 6 типов объектов

1) «Комната» – физическое пространство, в котором будет происходить запись.

2) «Группа» – представляет клиента студии.

3) «Жанр» – представляет справочную информацию о жанрах музыки для записи.

4) «Запись» - представляет исчерпывающую информацию о конкретной записи (дата, время, комната, стоимость и т.д.)

Созданы связи типа многие ко многим для таблиц жанры и группы. Так как одна и так же группа может играть сразу в нескольких жанрах. Например рок и блюз. Для данной связи была создана отдельная таблица. Также созданы связи многим для таблицы записи и жанры. Так как запись может быть одновремененно в нескольких жанрах.

## Атрибуты объектов

Атрибуты объекта «Комната» – «Площадь», «Стоимость аренды» и «Допустимое кол-во человек».

Для объекта «Группа» – «Название», «Популярность» и «Кол-во участников».

Для объекта «Жанр» – «Название», «Инструменты».

Атрибуты объекта «Запись» – собственные атрибуты «Дата», «Кол-во часов» и «Общая стоимость». Также у данного объекты имеются атрибуты, являющиеся ссылками на все другие объекты.

## Типы связей

Самым главным объектом в данной организации является «Запись». Этот объект имеет прямые связи с остальными пятью объектами, так объединяет всю информацию о используемый инструментах, сотруднике, который будет осуществлять запись, комнате, в которой будет проходить запись и другие аспекты записи.

Остальные объекты независимы, что в дальнейшем поможет разработать модульное ПО для данного типа организации.

# КОНТРОЛЛЕР

Разработан новый контроллер для базы данных BerkeleyDB.

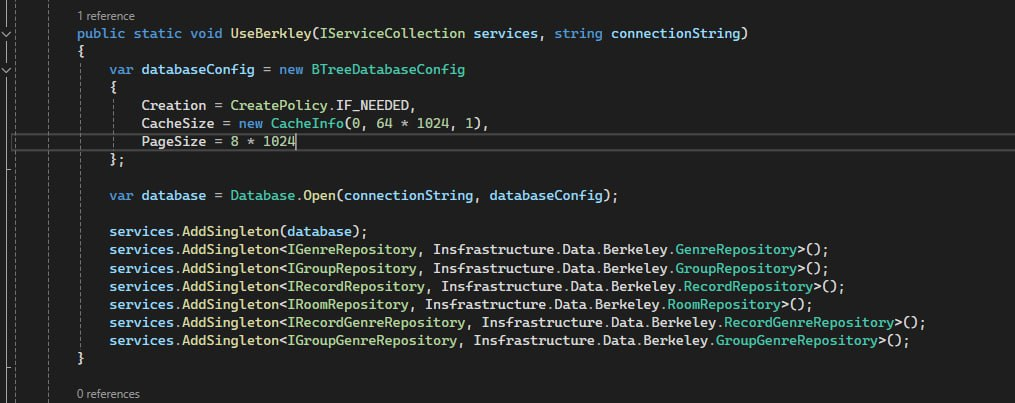


Рисунок 2.1 – Скриншот кода контроллера

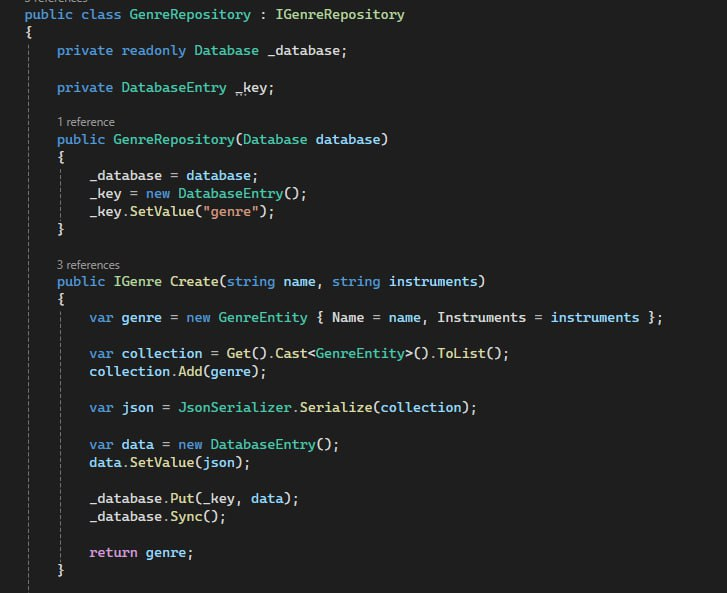


Рисунок 2.2 – Скриншот кода создания записи Жанр

Для работы контроллера были добавлены реализации интерфейсов для BerkeleyDB. Это такие же по функционалу интерфейсы, что и для работы с Postgres, но переопределены по специфику BerkeleyDB. Ниже приведен код репозитория жанр:

using BerkeleyDB;

using Domain.Core.Berkeley;

using Domain.Interfaces.Entity;

using Domain.Interfaces.Repository;

using Insfrastructure.Data.Berkeley.Extensions;

using System.Text.Json;

namespace Insfrastructure.Data.Berkeley

{

public class GenreRepository : IGenreRepository

{

private readonly Database \_database;

private DatabaseEntry \_key;

public GenreRepository(Database database)

{

\_database = database;

\_key = new DatabaseEntry();

\_key.SetValue("genre");

}

public IGenre Create(string name, string instruments)

{

var genre = new GenreEntity { Name = name, Instruments = instruments };

var collection = Get().Cast<GenreEntity>().ToList();

collection.Add(genre);

var json = JsonSerializer.Serialize(collection);

var data = new DatabaseEntry();

data.SetValue(json);

\_database.Put(\_key, data);

\_database.Sync();

return genre;

}

public void Delete(IGenre genre)

{

var collection = Get().Cast<GenreEntity>().ToList();

collection.RemoveAll(g => g.Name.Equals(genre.Name));

var json = JsonSerializer.Serialize(collection);

var data = new DatabaseEntry();

data.SetValue(json);

\_database.Put(\_key, data);

\_database.Sync();

}

public IGenre Get(string name)

{

return Get().FirstOrDefault(g => g.Name.Equals(name));

}

public IEnumerable<IGenre> Get()

{

IEnumerable<IGenre> result = null;

if (\_database.Exists(\_key))

{

var json = \_database.Get(\_key).Value.AsString();

result = JsonSerializer.Deserialize<IEnumerable<GenreEntity>>(json);

}

return result ?? Enumerable.Empty<IGenre>();

}

}

}

# ЗАКЛЮЧЕНИЕ

В результате работы над лабораторной работой был разработан контроллер для работы с базой данных BerkeleyDB. Основные методы создания, удаления и чтения были переопределены в соответствии с спецификой базы данных.