

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»
ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ
**Кафедра системного програмування та спеціалізованих комп'ютерних
систем**

Лабораторна робота №2
з дисципліни
«Бази даних і засоби управління»
Тема: «Створення додатку бази даних, орієнтованого на взаємодію з СУБД
PostgreSQL»

Виконав: студент III курсу
ФПМ групи КВ-82
Любич І. Д.
Перевірів: Павловський В. І.

Метою роботи є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

Загальне завдання роботи полягає у наступному:

1. Реалізувати функції внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з 2-х та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

Деталізоване завдання:

1. Забезпечити можливість введення/редагування/вилучення даних у таблицях бази даних з можливістю контролю відповідності типів даних атрибутів таблиць (рядків, чисел, дати/часу). Для контролю пропонується два варіанти:

а) контроль при введенні - валідація даних;

б) перехоплення помилок (**try...except**) від сервера PostgreSQL при виконанні відповідної команди SQL.

Особливу увагу варто звернути на дані таблиць, що мають зв'язок 1:N.

З боку батьківської таблиці необхідно контролювати **вилучення (ON DELETE)** рядків за умови наявності даних у підлеглий таблиці.

З боку підлеглої таблиці варто контролювати наявність відповідного рядка у батьківській таблиці при виконанні **внесення** до неї нових даних.

Унеможливити виведення програмою на екрані системних помилок PostgreSQL шляхом їх перехоплення і адекватної обробки.

Внесення даних виконується користувачем у консольному вікні програми.

2. Забезпечити можливість автоматичної генерації великої кількості даних у таблицях за допомогою вбудованих у PostgreSQL функцій роботи з псевдовипадковими числами. Дані мають бути згенерованими **не програмою, а відповідним SQL-запитом!** Кількість даних для генерування має вводити користувач з клавіатури.

3. Для реалізації багатокритеріального пошуку необхідно підготувати 3 запити, що включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість введення конкретних значень констант для фільтрації з клавіатури користувачем. Після виведення даних вивести час виконання запиту у мілісекундах. Перевірити швидкодію роботи запитів на попередньо згенерованих даних.

4. Програмний код організувати згідно шаблону Model-View-Controller (MVC). Приклад організації коду згідно шаблону доступний [за даним посиланням](#). Модель, подання (представлення) та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** (без ORM).

Нормалізована модель даних

На рис. 1 наведено структуру нормалізованої бази даних.

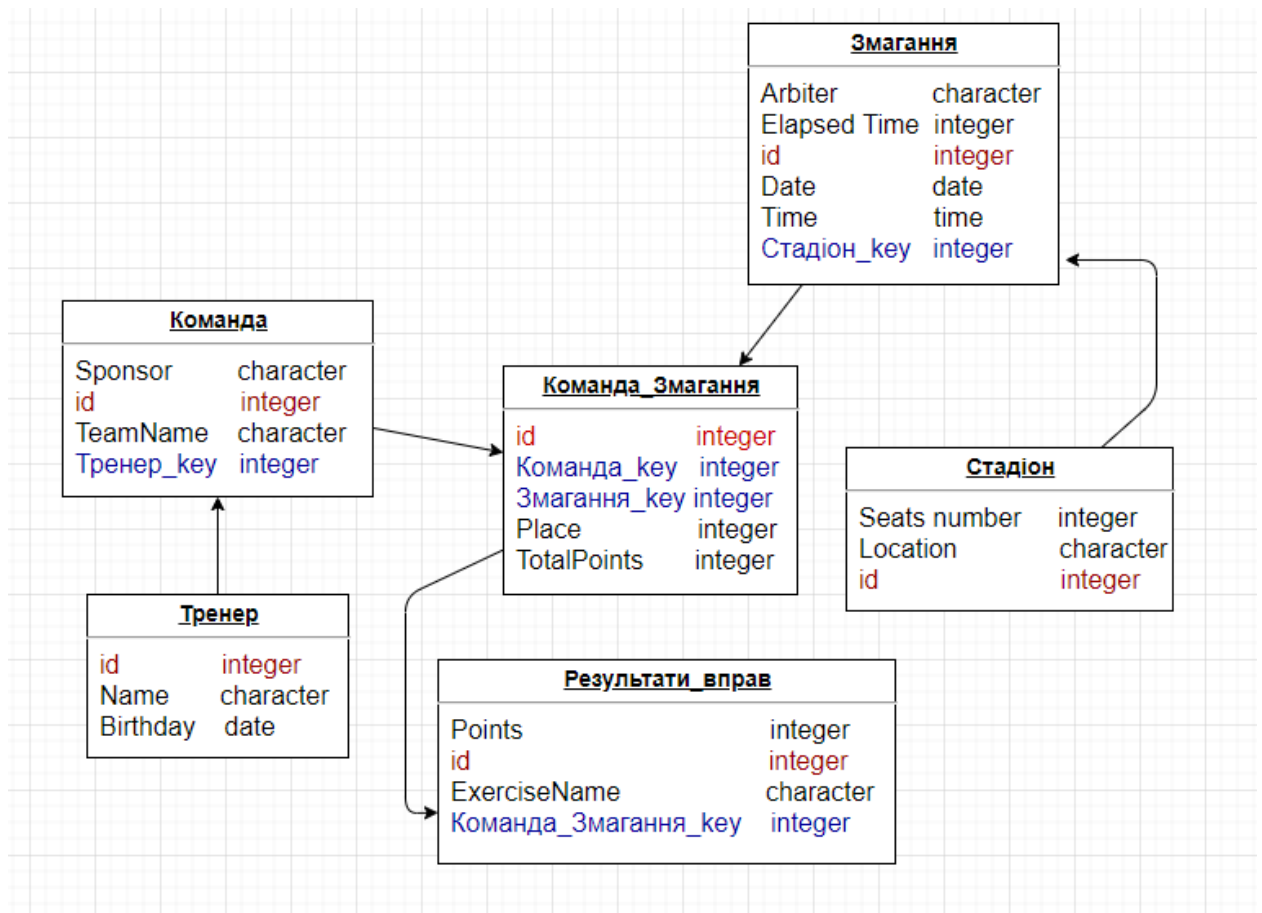


Рис. 1 – Нормалізована модель БД

Зміни у порівнянні з першою лабораторною роботою:

1. Змінений зв'язок між сутностями Стадіон і Змагання з N:M до 1:N.
2. Атрибут Date and Time таблиці Змагання перероблено на 2 атрибути типів.
3. Два зовнішні ключі таблиці Результати_вправ (Команди та Змагання) замінено на один зовнішній ключ, який містить первинний ключ таблиці Команда_Змагання.

Середовище розробки та налаштування підключення до бази даних

Для виконання лабораторної роботи використовувалась мова програмування C# та IDE Visual Studio 2019.

Для підключення до серверу бази даних PostgreSQL використовувався пакет Npgsql. Для цього створений метод, який з'єднується з базою даних.

```
private void connectDB()
{
    var cs = $"Host=localhost;Username=postgres;Password={password};Database=DB_Lab1";
    con = new NpgsqlConnection(cs);
    con.Open();
    cmd = new NpgsqlCommand();
    cmd.Connection = con;
}
```

Надалі керування БД відбуватиметься за допомогою змінної «cmd».

Опис структури програми

Програма містить 10 модулів, серед яких основні: **Program** (початок програми), **Model**, **View**, **Controller** – відповідають за дані, інтерфейс та обробку інформації відповідно. Інші 6 модулів виконують лише одну визначену задачу: ShowTable – вивід таблиць, Insert – внесення даних, Update – оновлення даних, Delete – видалення записів з таблиць, SQL Tool – параметризовані запити до БД, RandomInsert – рандомізоване заповнення таблиці «Тренер».

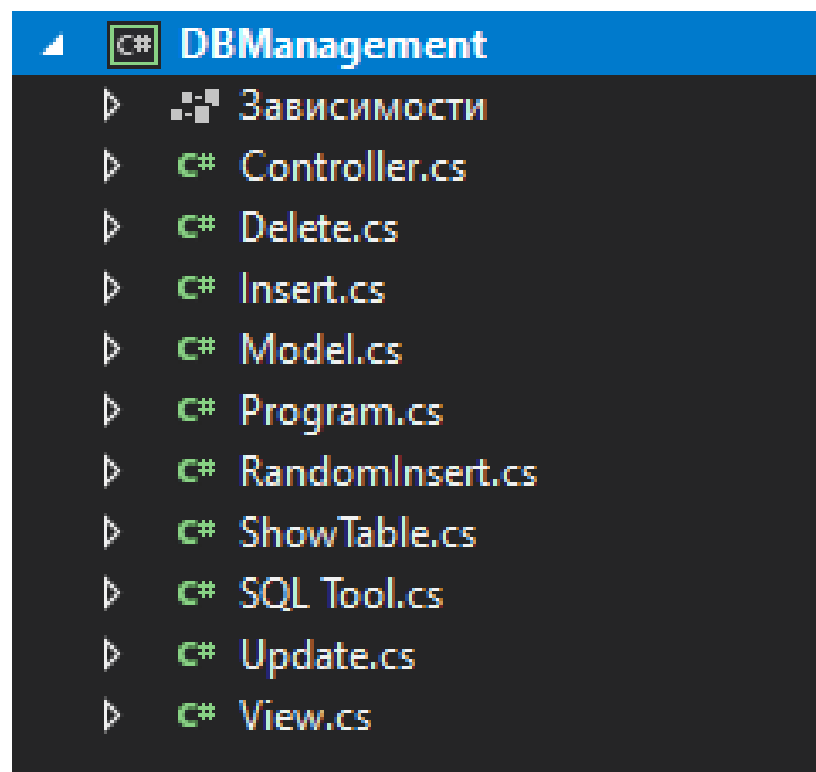


Рис. 2 – Структура програмного коду

Структура MVC у файлах

На рис. 3 наведена відповідність модулів програми до компонентів логічної побудови MVC.

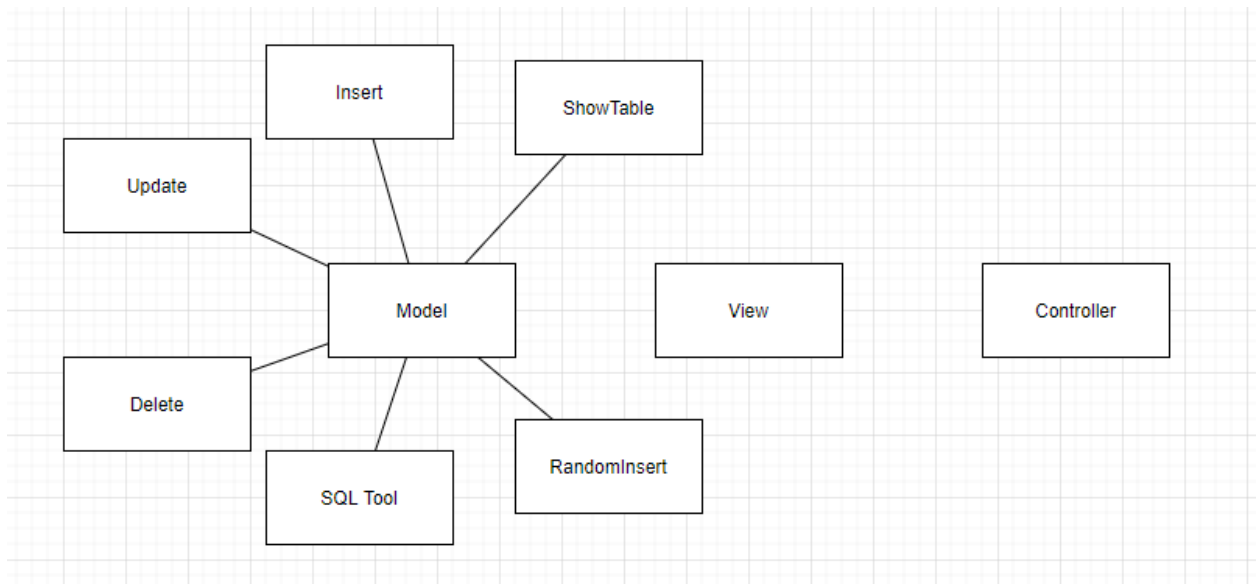


Рис. 3 – Відповідність файлів програми до MVC

Структура меню програми

Головне меню

```
1. View Db content
2. Insert
3. Update
4. Delete
5. SQL tool
6. Random insert to "Тренер"
7. Exit
```

Меню вибору таблиці

```
1. Команда
2. Тренер
3. Стадіон
4. Змагання
5. Результати_вправ
6. Команда_Змагання
0. Back
```

Меню вибору запиту

```
1. Пошук команд та їх тренерів за назвою команди, спонсором та датою народження тренера команди
2. Пошук стадіонів та змагань за кількістю місць, на яких заплановані змагання у заданий проміжок дати та часу
3. Пошук команд і змагань за загальною кількістю очок, тривалістю змагань і арбітром
0. Back
```

Меню вибору кількості рандомізованих рядків

```
Number of randomized records:
```

Посилання для навігації

1. [Лістинги програми з директивами внесення, редагування та вилучення даних у базі даних та результати виконання цих директив](#)
 - a. [Функції внесення](#)
 - b. [Функції редагування](#)
 - c. [Функції видалення](#)
2. [Лістинг програми з директивами рандомізованого внесення даних до таблиці «Тренер» та результати](#)
3. [Лістинги програми з директивами динамічних запитів у базі даних та результати виконання цих директив](#)
4. [Обробка виняткових ситуацій \(помилки\) при введенні/вилученні та валідації даних](#)
5. [Дослідження режимів обмеження ON DELETE](#)
6. [Ілюстрації програмного коду на Github](#)

Лістинги програми з директивами внесення, редагування та вилучення даних у базі даних та результати виконання цих директив

Функції для внесення

Унесення відбувається через метод `ExecuteInserting`, який приймає номер таблиці як параметр і отримує частину запиту від відповідного методу:

```
public void ExecuteInserting(int dbPart)
{
    switch (dbPart)
    {
        case 1:
            cmd.CommandText += insert.InsertTeam();
            break;
        case 2:
            cmd.CommandText += insert.InsertCoach();
            break;
        case 3:
            cmd.CommandText += insert.InsertStadium();
            break;
        case 4:
            cmd.CommandText += insert.InsertCompetition();
            break;
        case 5:
            cmd.CommandText += insert.InsertResults();
            break;
        case 6:
            cmd.CommandText += insert.InsertTeam_Competition();
            break;
    }
    ExecuteQuery(cmd);
}
```

Додавання запису в таблицю «Тренер»

```
public string InsertTeam()
{
    string TeamName, Sponsor; int Coach;
    Console.Write("TeamName: ");
    TeamName = Console.ReadLine();
    Console.Write("Sponsor: ");
    Sponsor = Console.ReadLine();
    Console.Write("Coach ID: ");
    Coach = int.Parse(Console.ReadLine());
    return $"(\"Sponsor\", \"TeamName\", \"Тренер_key\") VALUES('{Sponsor}', '{TeamName}', {Coach});";
}
```

```
Name: Jack
$ Birthday:
$ Day: 10
$ Month: 12
$ Year: 1991
```

Результат:

36	90	stuvwxyz	2014-01-14
37	91	WXYZabcdef	2014-01-17
38	92	Jack	2000-03-03
39	93	Jack	1991-12-10

Додавання запису в таблицю «Команда»

```
public string InsertTeam()
{
    string TeamName, Sponsor; int Coach;
    Console.Write("TeamName: ");
    TeamName = Console.ReadLine();
    Console.Write("Sponsor: ");
    Sponsor = Console.ReadLine();
    Console.Write("Coach ID: ");
    Coach = int.Parse(Console.ReadLine());
    return $"(\"Sponsor\", \"TeamName\", \"Тренер_key\") VALUES('{Sponsor}', '{TeamName}', {Coach});";
}
```

```
TeamName: TeamKyiv
Sponsor: Coca-Cola
Coach ID: 89
```

Результат:

3	Sponsor2	16	TeamTwo	2
4	Sponsor3	17	TeamThree	3
5	Coca-Cola	53	TeamKyiv	89

Додавання запису в таблицю «Стадіон»

```
public string InsertStadium()
{
    string Location; int SeatsNumber;
    Console.Write("Location: ");
    Location = Console.ReadLine();
    Console.Write("Seats number: ");
    SeatsNumber = int.Parse(Console.ReadLine());
    return $"(\"Seats number\", \"Location\") VALUES({SeatsNumber}, '{Location}');";
}
```

```
Location: Moscow
Seats number: 4000
```

Результат:

5722	JKLMNOPQRS	337
6153	YZabcdefgh	338
4000	Moscow	339

Додавання запису в таблицю «Змагання»

```
public string InsertCompetition()
{
    string Arbiter, Date, Time; int Duration, StadiumFkey;
    Console.WriteLine("Arbiter: ");
    Arbiter = Console.ReadLine();
    Console.WriteLine("Duration: ");
    Duration = int.Parse(Console.ReadLine());
    Console.WriteLine("Stadium key: ");
    StadiumFkey = int.Parse(Console.ReadLine());
    Console.WriteLine("Date: ");
    Date = Console.ReadLine();
    Console.WriteLine("Time: ");
    Time = Console.ReadLine();
    return $"(\"Arbiter\", \"Elapsed Time\", \"Date\", \"Time\", \"Стадіон_key\") VALUES('{Arbiter}', " +
        $"{Duration}', '{Date}', '{Time}', {StadiumFkey});";
}
```

```
Arbiter: Arb1
Duration: 100
Stadium key: 2
Date: 23-10-2020
Time: 15:40
```

Результат:

Arbiter character varying	Elapsed Time integer	id [PK] integer	Date date	Time time without time zone	Стадіон_key integer
Arbiter1	60	72	2020-10-10	15:40:00	2
Arbiter2	90	73	2020-12-11	18:00:00	3
aesf	234	74	2005-04-23	23:40:00	2
sdf	234	75	2020-12-20	20:30:00	3
Arb1	100	76	2019-10-19	19:30:00	2
Arb1	100	77	2020-10-23	15:40:00	2

Додавання запису в таблицю «Результати_вправ»

```
public string InsertResults()
{
    string ExName; int Points, Team_Comp_key;
    Console.WriteLine("Exercise name: ");
    ExName = Console.ReadLine();
    Console.WriteLine("Team_Competition key: ");
    Team_Comp_key = int.Parse(Console.ReadLine());
    Console.WriteLine("Points: ");
    Points = int.Parse(Console.ReadLine());
    return $"(\"Points\", \"ExerciseName\", \"Команда_Змагання_key\") VALUES({Points}, '{ExName}', {Team_Comp_key});";
}
```

```
Exercise name: Pull ups
Team_Competition key: 15
Points: 23
```

Результат:

Points integer	id [PK] integer	ExerciseName character varying	Команда_Змагання_key integer
143	31	Ex1	14
123	32	Ex1	15
23	33	Pull ups	15

Додавання запису в таблицю «Команда_Змагання»

```
public string InsertTeam_Competition()
{
    int TeamKey, CompKey, Place, TotalPoints;
    Console.Write("Team key: ");
    TeamKey = int.Parse(Console.ReadLine());
    Console.Write("Competition key: ");
    CompKey = int.Parse(Console.ReadLine());
    Console.Write("Place: ");
    Place = int.Parse(Console.ReadLine());
    Console.Write("Total Points: ");
    TotalPoints = int.Parse(Console.ReadLine());
    return $"(\\"Команда_key\\", \\"Змагання_key\\", \\"Place\\", \\"TotalPoints\\") VALUES({TeamKey}, {CompKey}, {Place}, {TotalPoints});";
}
```

```
Team key: 17
Competition key: 72
Place: 4
Total Points: 200
```

Результат:

id [PK] integer	Команда_key integer	Змагання_key integer	Place integer	TotalPoints integer
16	15	73	1	155
17	16	73	2	143
18	17	73	3	123
20	17	72	4	200

Функції для редагування

Редагування відбувається через метод ExecuteUpdating, який приймає номер таблиці як параметр і отримує частину запиту від відповідного методу:

```
public void ExecuteUpdating(int dbPart)
{
    switch (dbPart)
    {
        case 1:
            cmd.CommandText += update.UpdateTeam();
            break;
        case 2:
            cmd.CommandText += update.UpdateCoach();
            break;
        case 3:
            cmd.CommandText += update.UpdateStadium();
            break;
        case 4:
            cmd.CommandText += update.UpdateCompetition();
            break;
        case 5:
            cmd.CommandText += update.UpdateResults();
            break;
        case 6:
            cmd.CommandText += update.UpdateTeam_Competition();
            break;
    }
    ExecuteQuery(cmd);
}
```

Редагування запису в таблиці «Тренер»

```
public string UpdateCoach()
{
    string Name, Birthday; int day, month, year;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("Name: ");
    Name = Console.ReadLine();
    Console.WriteLine("Birthday: ");
    Console.Write("Day: ");
    day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    year = int.Parse(Console.ReadLine());
    Birthday = year.ToString() + '-' + month.ToString() + '-' + day.ToString();
    return $" SET \"Name\"='{Name}', \"Birthday\"='{Birthday}' where id = {id}";
}
```

```
2
Record ID to change: 2
Name: NewName
Birthday:
Day: 20
Month: 12
Year: 1970
```

Результат:

id [PK] integer	Name character varying	Birthday date
1	Denis Denisov	1977-06-20
3	Vlad Nosko	1977-07-30
0	Ivan Gromov	1979-05-10
5	123	0232-02-02
2	NewName	1970-12-20

Редагування запису в таблиці «Команда»

```
public string UpdateTeam()
{
    string TeamName, Sponsor; int Coach;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("TeamName: ");
    TeamName = Console.ReadLine();
    Console.Write("Sponsor: ");
    Sponsor = Console.ReadLine();
    Console.Write("Coach ID: ");
    Coach = int.Parse(Console.ReadLine());
    return $" SET \"Sponsor\"='{Sponsor}', \"TeamName\"='{TeamName}', \"Тренер_key\"={Coach} where id = {id}";
}
```

```

1
Record ID to change: 14
TeamName: TeamPepsi
Sponsor: Pepsi
Coach ID: 80

```

Результат:

Sponsor character varying	id [PK] integer	TeamName character varying	Тренер_key integer
Pepsi	14	TeamPepsi	80
Sponsor1	15	TeamOne	1
Sponsor2	16	TeamTwo	2

Редагування запису в таблиці «Стадіон»

```

public string UpdateStadium()
{
    string Location; int SeatsNumber;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("Location: ");
    Location = Console.ReadLine();
    Console.Write("Seats number: ");
    SeatsNumber = int.Parse(Console.ReadLine());
    return $" SET \"Location\"='{Location}', \"Seats number\"={SeatsNumber} where id = {id};";
}

```

```

Record ID to change: 2
Location: New York
Seats number: 30000

```

Результат:

Seats number integer	Location character varying	id [PK] integer
400	Moscow	3
30000	New York	2

Редагування запису в таблиці «Змагання»

```

public string UpdateCompetition()
{
    string Arbiter, Date, Time; int Duration, StadiumFkey, _day, _month, _year;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("Arbiter: ");
    Arbiter = Console.ReadLine();
    Console.Write("Duration: ");
    Duration = int.Parse(Console.ReadLine());
    Console.Write("Stadium key: ");
    StadiumFkey = int.Parse(Console.ReadLine());
    Console.WriteLine("Date: ");
    Console.Write("Day: ");
    _day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    _month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    _year = int.Parse(Console.ReadLine());
    Date = _year.ToString() + '-' + _month.ToString() + '-' + _day.ToString();
    Console.Write("Time: ");
    Time = Console.ReadLine();
    return $" SET \"Arbiter\"='{Arbiter}', \"Date\"='{Date}', \"Time\"='{Time}', " +
        $"\"Elapsed Time\"={Duration}, \"Стадіон_key\"={StadiumFkey} where id = {id};";
}

```

```
Record ID to change: 74
Arbiter: NewArb
Duration: 100
Stadium key: 2
Date:
Day: 30
Month: 10
Year: 2010
Time: 15:30
```

Результат:

Arbiter character varying	Elapsed Time integer	id [PK] integer	Date date	Time time without time zone	Стадіон_key integer
sdf	234	75	2020-12...	20:30:00	3
Arb1	100	76	2019-10...	19:30:00	2
Arb1	100	77	2020-10...	15:40:00	2
NewArb	100	74	2010-10...	15:30:00	2

Редагування запису в таблиці «Результати_вправ»

```
public string UpdateResults()
{
    string ExName; int Points, Team_Comp_key;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("Exercise name: ");
    ExName = Console.ReadLine();
    Console.Write("Team_Competition key: ");
    Team_Comp_key = int.Parse(Console.ReadLine());
    Console.Write("Points: ");
    Points = int.Parse(Console.ReadLine());
    return $" SET \"ExerciseName\"='{ExName}', \"Points\"={Points}, " +
        $"\"Команда_Змагання_key\"={Team_Comp_key} where id = {id};";
}
```

```
Record ID to change: 32
Exercise name: Push ups
Team_Competition key: 14
Points: 60
```

Результат:

Points integer	id [PK] integer	ExerciseName character varying	Команда_Змагання_key integer
143	31	Ex1	14
60	32	Push ups	14
23	33	Pull ups	15

Редагування запису в таблиці «Команда_Змагання»

```
public string UpdateTeam_Competition()
{
    int TeamKey, CompKey, Place, TotalPoints;
    Console.Write("Record ID to change: ");
    id = int.Parse(Console.ReadLine());
    Console.Write("Team key: ");
    TeamKey = int.Parse(Console.ReadLine());
    Console.Write("Competition key: ");
    CompKey = int.Parse(Console.ReadLine());
    Console.Write("Place: ");
    Place = int.Parse(Console.ReadLine());
    Console.Write("Total Points: ");
    TotalPoints = int.Parse(Console.ReadLine());
    return $" SET \"TotalPoints\"={TotalPoints}, \"Place\"={Place}, " +
        $"\"Команда_key\"={TeamKey}, \"Змагання_key\"={CompKey} where id = {id};;";
}
```

```
Record ID to change: 18
Team key: 16
Competition key: 72
Place: 2
Total Points: 85
```

Результат:

id [PK] integer	Команда_key integer	Змагання_key integer	Place integer	TotalPoints integer
16	15	73	1	155
17	16	73	2	143
18	16	72	2	85
20	17	72	4	200

Функції для видалення

Видалення реалізується у методі DeleteItem, який приймає як параметр назву таблиці та записує запит видалення за полем id, який передається і виконується контролером.

Метод DeleteItem:

```
public static string DeleteItem(string Table)
{
    Console.WriteLine("Record ID to delete: ");
    int id = int.Parse(Console.ReadLine());
    return $"delete from competitions.{Table} where id = {id}";
}
```

Метод видалення в контролері:

```
public void ExecuteDeleting()
{
    ExecuteQuery(cmd);
}
```

Видалення запису з кожної таблиці відбувається за id. Приклад: видалення запису з таблиці «Стадіон»:

Seats number integer	Location character varying	id [PK] integer
400	Moscow	3
30000	New York	2
50000	Lviv	340

```
1. Команда
2. Тренер
3. Стадіон
4. Змагання
5. Результати_вправ
6. Команда_Змагання
0. Back
3
Record ID to delete: 340
```

Результат:

Data Output

	Seats number integer	Location character varying	id [PK] integer
1	400	Moscow	3
2	30000	New York	2

Лістинги програми з директивами внесення рандомізованих даних і виконання динамічних запитів у базі даних та результати виконання цих директив

Рандомізоване внесення даних до таблиці «Тренер»

Метод RandomCoachInsert приймає кількість рандомізованих даних для внесення до таблиці «Тренер» та повертає запит до контролера, де відбувається виконання запиту.

Метод RandomCoachInsert:

```
public string RandomCoachInsert()
{
    Console.WriteLine("Number of randomized records: ");
    int number = int.Parse(Console.ReadLine());
    return $"insert into competitions.\"Тренер\" (\"Name\", \"Birthday\") " +
        "select substr(characters, (random() * length(characters) + 1)::integer, 10), " +
        "timestamp '2014-01-10' + random() * (timestamp '2014-01-20' - timestamp '2014-01-10') " +
        $"from(values('ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz') as symbols(characters), generate_series(1, {number}));";
}
```

Виконання запиту контролером:

```
public void ExecuteRandomCoachInsert()
{
    cmd.CommandText = randomInsert.RandomCoachInsert();
    ExecuteQuery(cmd);
}
```





Таблиця «Тренер» до рандомізованого додавання даних:

Data Output	Explain	Messages	Notifications
id [PK] integer	Name character varying	Birthday date	
1	0 Ivan Gromov	1979-05-10	
2	1 Denis Denisov	1977-06-20	
3	2 Alexander Ivanyuk	1965-06-20	
4	3 Vlad Nosko	1977-07-30	
5	4 ibs	2001-12-20	
6	5 123	0232-02-02	

Уведення користувачем кількості записів:

Number of randomized records:
32

Таблиця «Тренер» після рандомізованого додавання 32 рядків:

	 id [PK] integer	 Name character varying	 Birthday date	
1	0	Ivan Gromov	1979-05-10	
2	1	Denis Denisov	1977-06-20	
3	2	Alexander Ivanyuk	1965-06-20	
4	3	Vlad Nosko	1977-07-30	
5	4	ibs	2001-12-20	
6	5	123	0232-02-02	
7	60	nopqrstuvwxyz	2014-01-12	
8	61	ghijklmnop	2014-01-10	
9	62	WXYZabcdef	2014-01-19	
10	63	nopqrstuvwxyz	2014-01-11	
11	64	QRSTUVWXYZ	2014-01-17	
12	65	z	2014-01-17	
13	66	stuvwxyz	2014-01-10	
14	67	TUVWXYZabc	2014-01-18	
15	68	fghijklmno	2014-01-14	
16	69	DEFGHIJKLM	2014-01-10	
17	70	wxyz	2014-01-11	
18	71	abcdefghijkl	2014-01-12	
19	72	UVWXYZabcd	2014-01-19	
20	73	z	2014-01-10	
21	74	CDEFGHIJKL	2014-01-14	
22	75	pqrstuvwxyz	2014-01-19	
23	76	XYZabcdefg	2014-01-18	
24	77	BCDEFGHIJK	2014-01-17	
25	78	JKLMNOPQRS	2014-01-17	
26	79	FGHIJKLMNO	2014-01-18	
27	80	WXYZabcdef	2014-01-11	
28	81	FGHIJKLMNO	2014-01-17	
29	82	stuvwxyz	2014-01-15	
30	83	EFGHIJKLMN	2014-01-11	
31	84	NOPQRSTUVWXYZ	2014-01-13	
32	85	RSTUVWXYZa	2014-01-18	
33	86	VWXYZabcde	2014-01-15	
34	87	ijklmnopqrs	2014-01-17	
35	88	vWXYZ	2014-01-10	
36	89	defghijklm	2014-01-11	
37	90	stuvwxyz	2014-01-14	
38	91	WXYZabcdef	2014-01-17	

Виконання динамічних запитів бази даних

Меню вибору запиту:

1. Пошук команд та їх тренерів за назвою команди, спонсором та датою народження тренера команди
2. Пошук стадіонів та змагань за кількістю місць, на яких заплановані змагання у заданий проміжок дати та часу
3. Пошук команд і змагань за загальною кількістю очок, тривалістю змагань і арбітром
0. Back

Виконання запитів та замірювання часу виконання виконується в контролері:

```
public List<string> ExecuteSQL(int choice, ref long ms)
{
    switch(choice)
    {
        case 1:
            cmd.CommandText = sqlTool.DoSQL1();
            break;
        case 2:
            cmd.CommandText = sqlTool.DoSQL2();
            break;
        case 3:
            cmd.CommandText = sqlTool.DoSQL3();
            break;
    }
    System.Diagnostics.Stopwatch ExecutingTime = System.Diagnostics.Stopwatch.StartNew();
    rdr = cmd.ExecuteReader();
    ExecutingTime.Stop();
    ms = ExecutingTime.ElapsedMilliseconds;
}
```

Складання запитів відбувається в модулі «SQL Tool».

Перший запит:

```
public string DoSQL1()
{
    string NameLike, SponsorLike; Date BirthB, BirthT;
    Console.WriteLine("Пошук команд та їх тренерів за назвою команди, спонсором та датою народження тренера команди");
    Console.Write("TeamName like: ");
    NameLike = Console.ReadLine();
    Console.Write("Sponsor like: ");
    SponsorLike = Console.ReadLine();
    Console.WriteLine("Проміжок дати народження");
    Console.WriteLine("Нижня границя");
    Console.Write("Day: ");
    BirthB.day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    BirthB.month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    BirthB.year = int.Parse(Console.ReadLine());

    Console.WriteLine("Верхня границя");
    Console.Write("Day: ");
    BirthT.day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    BirthT.month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    BirthT.year = int.Parse(Console.ReadLine());
    return $"select \"TeamName\", \"Sponsor\", \"Name\" as \"CoachName\", \"Birthday\" as \"CoachBirthday\" from " +
        $"competitions.\"Команда\" inner join competitions.\"Тренер\" as tr " +
        $"on tr.id = \"Тренер_key\" where \"TeamName\" like '{NameLike}' and \"Sponsor\" like '{SponsorLike}' and " +
        $"tr.\"Birthday\" >= '{BirthB.StrDate()}' and tr.\"Birthday\" <= '{BirthT.StrDate()}'; ";
}
```

Повна таблиця:

Data Output					Messages	Explain	Notifications
	TeamName character varying	Sponsor character varying	CoachName character varying	Birthday date			
1	TeamOne	Sponsor1	Denis Denisov	1977-06-20			
2	TeamTwo	Sponsor2	Alexander Ivanyuk	1965-06-20			
3	TeamThree	Sponsor3	Vlad Nosko	1977-07-30			
4	randomized	hehe	123	0232-02-02			

Результат:

```
Пошук команд та їх тренерів за назвою команди, спонсором та датою народження тренера команди
TeamName like: %
Sponsor like: %
Проміжок дати народження
Нижня границя
Day: 03
Month: 03
Year: 1964
Верхня границя
Day: 03
Month: 03
Year: 2012
Team = TeamOne, Sponsor = Sponsor1, CoachName = Denis Denisov, CoachBirthday = 1977-06-20
Team = TeamTwo, Sponsor = Sponsor2, CoachName = Alexander Ivanyuk, CoachBirthday = 1965-06-20
Team = TeamThree, Sponsor = Sponsor3, CoachName = Vlad Nosko, CoachBirthday = 1977-07-30
Execution time: 3
```






Другий запит:

```
public string DoSQL2()
{
    Console.WriteLine("Пошук стадіонів та змагань за кількістю місць, на яких заплановані змагання у заданий проміжок дати та часу");
    Date dateB, dateT; int numberMin, numberMax; string timeMin, timeMax;
    Console.Write("Мінімальна кількість місць: ");
    numberMin = int.Parse(Console.ReadLine());
    Console.Write("Максимальна кількість місць: ");
    numberMax = int.Parse(Console.ReadLine());
    Console.WriteLine("Проміжок дати");

    Console.WriteLine("Нижня границя");
    Console.Write("Day: ");
    dateB.day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    dateB.month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    dateB.year = int.Parse(Console.ReadLine());

    Console.WriteLine("Верхня границя");
    Console.Write("Day: ");
    dateT.day = int.Parse(Console.ReadLine());
    Console.Write("Month: ");
    dateT.month = int.Parse(Console.ReadLine());
    Console.Write("Year: ");
    dateT.year = int.Parse(Console.ReadLine());
    Console.Write("Мінімальний час початку: ");
    timeMin = Console.ReadLine();
    Console.Write("Максимальний час початку: ");
    timeMax = Console.ReadLine();
    return "select \"Стадіон\".id, \"Seats number\", \"Date\", \"Time\" from competitions.\"Стадіон\" " +
        $"inner join competitions.\"Змагання\" as Zm on Zm.\"Стадіон_key\" = \"Стадіон\".id where \"Date\" <= '{dateT.StrDate()}' " +
        $"and \"Date\" >= '{dateB.StrDate()}' and \"Time\" <= '{timeMax}' and \"Time\" >= '{timeMin}' and " +
        $"\"Seats number\" <= {numberMax} and \"Seats number\" >= {numberMin}";
}
```

Повна таблиця:

Data Output		Messages	Explain	Notifications
	id integer 	Seats number integer 	Date date 	Time time without time zone 
1	2	200	2020-10-10	15:40:00
2	3	400	2020-12-11	18:00:00
3	2	200	2005-04-23	23:40:00
4	3	400	2020-12-20	20:30:00

Результат:

```
Пошук стадіонів та змагань за кількістю місць, на яких заплановані змагання у заданий проміжок дати та часу
Мінімальна кількість місць: 10
Максимальна кількість місць: 100000
Проміжок дати
Нижня границя
Day: 1
Month: 1
Year: 1900
Верхня границя
Day: 1
Month: 1
Year: 2025
Мінімальний час початку: 8:30
Максимальний час початку: 19:00
id = 2, Seats number = 200, Date = 2020-10-10, Time = 15:40:00
id = 3, Seats number = 400, Date = 2020-12-11, Time = 18:00:00
Execution time: 3
```

Третій запит:

```
public string DoSQL3()
{
    Console.WriteLine("Пошук команд і змагань за загальною кількістю очок, тривалістю змагань і арбітром");
    int minPoints, maxPoints, minDur, maxDur; string Arb;
    Console.Write("Мінімальна кількість очок: ");
    minPoints = int.Parse(Console.ReadLine());
    Console.Write("Максимальна кількість очок: ");
    maxPoints = int.Parse(Console.ReadLine());
    Console.Write("Мінімальна тривалість змагання: ");
    minDur = int.Parse(Console.ReadLine());
    Console.Write("Максимальна тривалість змагання: ");
    maxDur = int.Parse(Console.ReadLine());
    Console.Write("Arbiter like: ");
    Arb = Console.ReadLine();
    return $"select \"Команда\".\"TeamName\", Kz.\"Змагання_key\" as \"Змагання\", \"TotalPoints\", \"Elapsed Time\" " +
        $"as \"Duration\", \"Arbiter\" from competitions.\"Команда\" inner join competitions.\"Команда_Змагання\" " +
        $"as Kz on Kz.\"Команда_key\" = \"Команда\".id inner join competitions.\"Змагання\" as Zm on " +
        $"Zm.id = Kz.\"Змагання_key\" where \"Arbiter\" like '{Arb}' and \"Elapsed Time\" >= {minDur} and " +
        $"\"Elapsed Time\" <= {maxDur} and \"TotalPoints\" >= {minPoints} and \"TotalPoints\" <= {maxPoints}";
}
```

Повна таблиця:

Data Output		Messages	Explain	Notifications	
	TeamName character varying	Competition ID integer	TotalPoints integer	duration integer	Arbiter character varying
1	TeamOne	72	111	60	Arbiter1
2	TeamTwo	72	77	60	Arbiter1
3	TeamThree	72	100	60	Arbiter1
4	TeamOne	73	155	90	Arbiter2
5	TeamTwo	73	143	90	Arbiter2
6	TeamThree	73	123	90	Arbiter2

Результат:

```
Пошук команд і змагань за загальною кількістю очок, тривалістю змагань і арбітром
Мінімальна кількість очок: 110
Максимальна кількість очок: 160
Мінімальна тривалість змагання: 20
Максимальна тривалість змагання: 95
Arbiter like: %
Team = TeamOne, Competition ID = 72, Total Points = 111, Duration = 60, Arbiter = Arbiter1
Team = TeamOne, Competition ID = 73, Total Points = 155, Duration = 90, Arbiter = Arbiter2
Team = TeamTwo, Competition ID = 73, Total Points = 143, Duration = 90, Arbiter = Arbiter2
Team = TeamThree, Competition ID = 73, Total Points = 123, Duration = 90, Arbiter = Arbiter2
Execution time: 1
```

Обробка виняткових ситуацій (помилки) при введенні/вилученні та валідації даних

Обробка виняткових ситуацій при введенні (insert) та вилученні даних (delete) виконується за допомогою блоку try-catch. При введенні помилкових даних помилка серверу SQL не зупинить роботу програми, а в меню користувача з'явиться повідомлення про помилку.

```
public void ExecuteQuery(NpgsqlCommand _cmd)
{
    try
    {
        _cmd.ExecuteNonQuery();
    }
    catch(Exception ex)
    {
        Console.WriteLine("Помилка перехоплена");
    }
}
```

Рис. 4 – Обробка помилок з сервера PostgreSQL

Приклад додавання рядка з неіснуючим зовнішнім ключем:

```
1. Команда
2. Тренер
3. Стадіон
4. Змагання
5. Результати_вправ
6. Команда_Змагання
0. Back
1
TeamName: T1
Sponsor: Sp1
Coach ID: 100500
Помилка перехоплена
```

Приклад введення рядка з полем, тип якого не відповідає дійсному:

```
1. Команда
2. Тренер
3. Стадіон
4. Змагання
5. Результати_вправ
6. Команда_Змагання
0. Back
2
Name: Jack
Birthday:
Day: 32
Month: 5
Year: 1991
Помилка перехоплена
```

Приклад видалення рядка, ключ якого є зовнішнім ключем іншої таблиці:

```
1. Команда
2. Тренер
3. Стадіон
4. Змагання
5. Результати_вправ
6. Команда_Змагання
0. Back
2
Record ID to delete: 1
Помилка перехоплена
```

Дослідження режимів обмеження ON DELETE

Дослідження режимів будемо проводити на таблиці «Тренер» (батьківська) та «Команда» (дочірня). Використаємо команду TeamThree з тренером id = 3 (Name = Vlad Nosko) .

Таблиця «Тренер»

	id [PK] integer	Name character varying	Birthday date
1	1	Denis Denisov	1977-06-20
2	3	Vlad Nosko	1977-07-30
3	0	Ivan Gromov	1979-05-10
4	5	123	0232-02-02

Таблиця «Команда»

Sponsor character varying	id [PK] integer	TeamName character varying	Тренер_key integer
Pepsi	14	TeamPepsi	80
Sponsor1	15	TeamOne	1
Sponsor2	16	TeamTwo	2
Sponsor3	17	TeamThree	3

Режим NO ACTION

При видаленні запису з таблиці «Тренер», id якого присутній в записі «Команда» отримуємо повідомлення про помилку.

```
Record ID to delete: 3
23503: UPDATE или DELETE в таблице "Тренер" нарушает ограничение внешнего ключа "Тренер_fkey" таблицы "Команда"
```

Режим SET NULL (за умови що Тренер_key має обмеження NOT NULL)

```
Record ID to delete: 3
23502: нулевое значение в столбце "Тренер_key" нарушает ограничение NOT NULL
```

Якщо поле Тренер_key не має обмеженості NOT NULL, після видалення отримаємо:

Sponsor character varying	id [PK] integer	TeamName character varying	Тренер_key integer
Pepsi	14	TeamPepsi	80
Sponsor1	15	TeamOne	1
Sponsor2	16	TeamTwo	2
Sponsor3	17	TeamThree	[null]
Coca-Cola	53	TeamKyiv	89

Запис з таблиці «Тренер» було видалено, а Тренер_key перейшов у NULL.

Використаємо команду TeamOne з тренером id = 1 (Name = Denis Denisov)

Режим SET DEFAULT

```
Record ID to delete: 1
23502: нулевое значение в столбце "Тренер_key" нарушает ограничение NOT NULL
```

Оскільки в налаштуваннях Тренер_key поле Default не заповнено, SET DEFAULT намагатиметься його і встановити як значення Тренер_key. Якщо в DEFAULT встановити якесь числове значення, то при видаленні двох і більше записів таблиці «Тренер» відповідні записи дочірньої таблиці матимуть однакове значення(DEFAULT), що порушує зв'язок 1:1. Якби ми мали тренера, який приймає усі команди, від яких відмовляються інші тренери, його id можна було б встановити як DEFAULT, але це також порушує реляційну модель нашої БД.

Режим RESTRICT

При видаленні запису з таблиці «Тренер», id якого присутній в записі «Команда» отримуємо таке ж повідомлення про помилку, як і в режимі NO ACTION.

```
Record ID to delete: 1
23503: UPDATE или DELETE в таблице "Тренер" нарушает ограничение внешнего ключа "Тренер_fkey" таблицы "Команда"
```

Режим CASCADE

При видаленні запису з таблиці «Тренер», id якого присутній в записі «Команда» отримуємо таке ж повідомлення про помилку, оскільки id команди тренера знаходиться в записі таблиці «Команда_Змагання».

```
Record ID to delete: 1
23503: UPDATE или DELETE в таблице "Команда" нарушает ограничение внешнего ключа "Команда_fkey" таблицы "Команда_Змагання"
```

Використаємо команду, id якої не записаний до інших таблиць, наприклад TeamPepsi з тренером id = 80. Повідомлення про помилку не виявлено, вміст таблиць «Тренер» (рис. 5) і «Команда» (рис. 6):

```
select * from competitions."Тренер"
where id < 85
```

Data Output				
	id [PK] integer	Name character varying	Birthday date	
1	1	Denis Denisov	1977-06-20	
2	0	Ivan Gromov	1979-05-10	
3	5	123	0232-02-02	
4	2	NewName	1970-12-20	

Рис. 5


```
select * from competitions."Команда"
```

Data Output

	Sponsor character varying	id [PK] integer	TeamName character varying	Тренер_key integer
1	Sponsor3	17	TeamThree	90
2	Sponsor1	15	TeamOne	1
3	Sponsor2	16	TeamTwo	2
4	Coca-Cola	53	TeamKyiv	89

Рис. 6

Отже, запис дочірньої і батьківської таблиць було видалено.

Ілюстрації програмного коду на Github

Search or jump to...

Pull requests

Issues

Marketplace

Explore

A1xarT / Database

Unwatch

1

Star

0

Fork

0

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

master
1 branch
0 tags

Go to file
Add file
Code

Database and management system labs

Readme

About

Releases

Packages

A1xarT Lab 2

f870e57
1 minute ago
9 commits

Lab1	Lab1 report	27 days ago
Lab2	Lab 2	1 minute ago
README.md	Update README.md	2 months ago

README.md

Database

Database & Management tools labs

Liubchych Ilyia KV-82

Search or jump to... Pull requests Issues Marketplace Explore A1xarT / Database Unwatch 1 Star 0 Fork 0

< Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master
Database / Lab2 /

Go to file
Add file

Avatar
AtxarT report added
8907c3a now
History

Controller.cs	Lab 2	7 minutes ago
Delete.cs	Lab 2	3 hours ago
Insert.cs	Lab 2	3 hours ago
Model.cs	Lab 2	3 hours ago
Program.cs	Lab 2	3 hours ago
README.md	Lab 2	7 minutes ago
RandomInsert.cs	Lab 2	3 hours ago
SQL Tool.cs	Lab 2	3 hours ago
ShowTable.cs	Lab 2	3 hours ago
Update.cs	Lab 2	3 hours ago
View.cs	Lab 2	3 hours ago
Лібочки іллі KB82 Ла62.pdf	report added	now

README.md

Змігання

Лабораторна робота №2 Тема: Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL Структура бази даних: