\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ізмаїльський Фаховий Коледж Економіки і Права \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КУРСОВА РОБОТА**

з \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Баз даних\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему:\_\_\_\_\_\_\_\_\_\_Розробка БД і застосунку для роботи з БД\_\_\_\_\_\_\_\_

Студента \_\_\_\_3\_\_\_\_\_ курсу \_\_В31\_\_\_\_ групи

галузі знань

\_\_\_\_\_\_\_\_\_12. Інформаційні технології\_\_\_\_\_\_\_

спеціальності

\_\_121. Інженерія програмного забезпечення\_\_

\_\_\_\_\_\_\_\_\_\_\_Геока Євгенія Ілліча\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Керівник \_\_\_\_\_\_\_\_Матвійчук Т.М.\_\_\_\_\_\_\_\_\_\_

( прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

. Ізмаїл - 2022 рік

**Ізмаїльський Фаховий Коледж Економіки і Права**

**Рецензія**

**про якість курсової роботи**

**з дисципліни**

***« Бази даних »***

Тема курсової роботи\_\_«Розробка БД і застосунку для роботи з БД»\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Автор курсової роботи \_\_\_\_\_\_\_\_\_Геока Євгеній Ілліч\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Спеціальність 121. Інженерія програмного забезпечення\_курс\_\_\_3\_\_\_ група\_В31\_\_\_

Курсова робота виконана \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зауваження викладача \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Критерії оцінки курсової роботи

1. Ступінь розробки теми \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Повнота використання літератури \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
3. Використання спеціальної літератури \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
4. Творчий підхід до написання курсової роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
5. Оформлення курсової роботи -\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Попередня оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Захист курсової роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Загальна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище, ініціали)

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_р.

**Зміст**

[Вступ 4](#_Toc103063446)

[Опис обладнання та ПЗ 5](#_Toc103063447)

[**Конфігурація системи** 5](#_Toc103063448)

[**ПЗ, яке було застосовано** 5](#_Toc103063449)

[Постановка задачі 6](#_Toc103063450)

[Проектування БД 7](#_Toc103063451)

[Проектування застосунку 10](#_Toc103063452)

[**Загальний опис вигляду форм** 10](#_Toc103063453)

[**Система найменувань компонентів** 11](#_Toc103063454)

[**Заходи для стійкості програми** 11](#_Toc103063455)

[**Діаграма класів** 12](#_Toc103063456)

[**Опис методів** 15](#_Toc103063457)

[Тестування роботи додатка 16](#_Toc103063458)

[**Тестові дані** 16](#_Toc103063459)

[**Скріншоти роботи програми** 18](#_Toc103063460)

[Висновки 27](#_Toc103063461)

[Перелік використаних інформаційних джерел 28](#_Toc103063462)

[Додаток 1. Код програми 29](#_Toc103063463)

[**Info.cs** 29](#_Toc103063464)

[**Main.cs** 29](#_Toc103063465)

# Вступ

Використання різноманітних баз даних для збереження даних у наші часи становиться дедалі популярнішим. Бази даних дозволяють структуровано зберігати інформацію, що спрощує процес її обробки. Однак, БД не призначені для відображення даних у зручному вигляді. Тому існують такі застосунки, що звуться СУБД. Вони дозволяють створювати БД, здійснювати різноманітні операції з даними, які знаходяться в БД у вигляді таблиць; деякі дозволяють навіть створювати різноманітні засоби зручного відображення даних (тут прикладом є СУБД MS Access). Але у деяких випадках є потреба у застосунках, що створені спеціально для роботи із конкретною БД. Вони повинні повністю задовольняти функціональні потреби, але, так як вони створені спеціально для конкретної БД, можуть мати покращені показники комфорту роботи та деякі додаткові можливості, які неможливо чи недоцільно реалізувати в універсальних СУБД.

Завданням цієї курсової роботи як раз і є створення такого спеціалізованного застосунку.

Інколи мені доводилось звертатися за додатковою інформацією до різноманітних джерел для усвідомлення деяких теоретичних моментів або для пошуку цікавих практичних прийомів.

Я б виділив дві основні мети цієї курсової роботи:

* Виконати завдання у повному обсязі;
* Покращити мої навички та знання шляхом застосування якомога більшої кількості різноманітних прийомів з використанням пройдених протягом семестру теоретичних даних.

# Опис обладнання та ПЗ

## **Конфігурація системи**

Розробка й тестування застосунку здійснювалися на такій апаратній платформі:

* Процесор: AMD E-450 APU with Radeon(tm) HD Graphics 1.65 GHz;
* Оперативна пам’ять: 3,00 ГБ (+ 5,5ГБ файл підкачки);
* Відеоадаптер інтегрований у процесор.
* Операційна система: Windows 10 Pro

## **ПЗ, яке було застосовано**

* База даних була спроектована, створена та заповнена тестовими даними у СУБД MS Access 2013
* Застосунок був розроблений у середовищі MS Visual Studio 2013 Common v12.0.40629.0 на платформі .NET v4.5. Мова програмування C#.

# Постановка задачі

Завдання №5: БД Радіостанції.

Таблиці:

1. Працівники;
2. Посади;
3. Виконавці;
4. Жанри;
5. Записи;
6. Графік роботи.

Виконати такі запити:

1. Відділ кадрів (поєднує таблиці «Працівники» та «Посади» по полю «Код посади»);
2. Музичний архів (Поєднує таблиці «Записи», «Виконавці», «Жанри» по полям «Код виконавця» та «Код жанру»
3. Сітка мовлення (Поєднує таблиці «Графік роботи», «Працівники», «Записи» по полям «Код працівника», «Код запису», «Код запису1», «Код запису2», «Код запису3»).

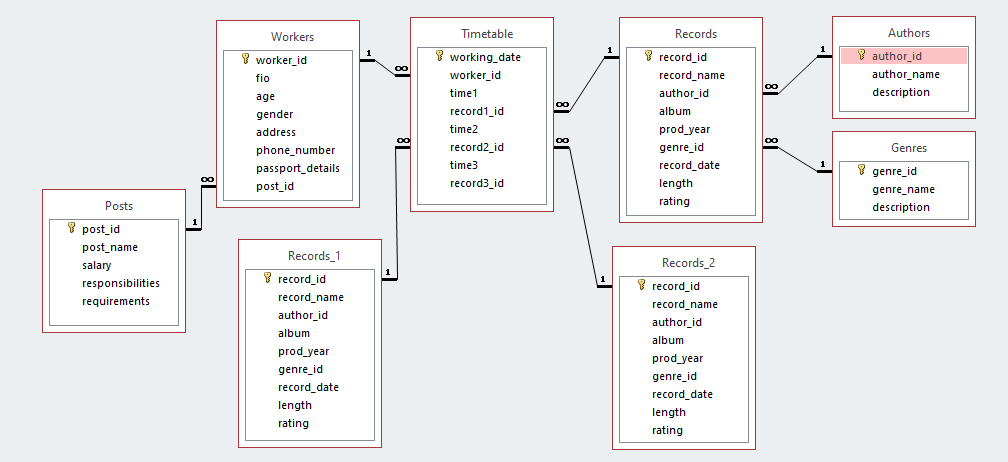
Передбачити наступні фільтри:

1. Фільтр для відображення працівників певних посад (На основі запиту «Відділ кадрів»);
2. Фільтр для відображення записів певного виконавця (На основі запиту «Музикальний архів»);
3. Фільтр для відображення записів певного жанру (На основі запиту «Музикальний архів»);
4. Фільтр сітки мовлення по окремим датам (На основі запиту «Сітка мовлення»);
5. Фільтр сітки мовлення по окремим працівникам (На основі запиту «Сітка мовлення»);

Розробити застосунок для роботи з цією БД.

# Проектування БД

Так як я вирішив обрати англійську мову у якості мови інтерфейсу, назви таблиць та запитів теж анлійськомовні.



Малюнок 1. Схема даних БД.

Records\_1 та Records\_2 є копіями Records. (Тільки шляхом створення цих копій зміг виконати запит «Сітка мовлення», де є досить складні зв’язки).

Працівники - Workers;

Посади - Posts;

Виконавці - Authors;

Жанри - Genres;

Записи - Records;

Графік роботи – Timetable;

Відділ кадрів – Personnel department;

Музикальний архів – Music archive;;

Сітка мовлення – Broadcast grid.

Таблиця Workers

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Код працівника | worker\_id | лічильник |  | - | + |
| ПІБ | fio | короткий текст | 60 | - | - |
| Вік | age | чисельний | байт | - | - |
| Стать | gender | короткий текст | 6 | - | - |
| Адрес | address | короткий текст | 255 | - | - |
| Номер телефону | phone\_number | короткий текст | 18 | - | - |
| Паспортні дані | passport\_details | короткий текст | 255 | - | - |
| Код посади | post\_id | чисельний | довге ціле | - | - |

Таблиця Posts

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Код посади | post\_id | лічильник |  | - | + |
| Назва | post\_name | короткий текст | 60 | - | - |
| Зарплата | salary | чисельний | довге ціле | - | - |
| Обов’язки | responsibilities | короткий текст | 255 | - | - |
| Вимоги | requirements | короткий текст | 255 | - | - |

Таблиця Authors

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Код виконавця | author\_id | лічильник |  | - | + |
| Назва | author\_name | короткий текст | 60 | - | - |
| Опис | description | короткий текст | 255 | - | - |

Таблиця Genres

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Код жанру | genre\_id | лічильник |  | - | + |
| Назва | genre\_name | короткий текст | 60 | - | - |
| Опис | description | короткий текст | 255 | - | - |

Таблиця Records

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Код запису | record\_id | лічильник |  | - | + |
| Назва | record\_name | короткий текст | 60 | - | - |
| Код автора | author\_id | чисельний | довге ціле | - | - |
| Альбом | album | короткий текст | 60 | - | - |
| Рік релізу | prod\_year | чисельний | ціле | - | - |
| Код жанру | genre\_id | чисельний | довге ціле | - | - |
| Дата релізу(запису) | record\_date | дата й час |  | - | - |
| Тривалість | length | дата й час |  | - | - |
| Рейтинг | rating | чисельний | double | - | - |

Таблиця Timetable

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування | Ідентифікатор | Тип даних | Довжина | Порожні рядки | Ключ |
| Робоча дата | working\_date | дата й час |  | - | + |
| Код працівника | worker\_id | чисельний | довге ціле | - | - |
| Час1 | time1 | дата й час |  | - | - |
| Код запису1 | record1\_id | чисельний | довге ціле | - | - |
| Час2 | time2 | дата й час |  | - | - |
| Код запису2 | record2\_id | чисельний | довге ціле | - | - |
| Час3 | time3 | дата й час |  | - | - |
| Код запису3 | record3\_id | чисельний | довге ціле | - | - |

# Проектування застосунку

## **Загальний опис вигляду форм**

Я вирішив реалізувати застосунок за допомогою якомога меншої кількості форм (вікон). В результаті маю форму Main, яка є основною формою додатку. Увесь функціонал реалізовано на ній, окрім відображення текстової інформації про додаток. Для цього була створена форма Info.Для відображення інформації про додаток використовуються компоненти Label та GroupBox. Більше нічого там немає.

Загальний вигляд головної формі під час перегляду таблиць має вигляд:

1. Заголовок таблиці; (один Label, текст якого відповідає назві поточної таблиці)
2. Компонент DataGridView; (Один на формі, заповнюється даними поточної таблицї)
3. Кнопки додавання/видалення/сбереження даних
4. Компонент GroupBox, в якому знаходяться компоненти, що потрібні для додавання нових записів для кожної таблиці. (Таких груп 6, одночасно доступна тільки одна, відповідно до поточної таблиці)

Загальний вигляд головної форми під час перегляду запитів має вігляд:

1. Заголовок таблиці; (той же Label)
2. Компонент DataGridView;
3. Компонент GroupBox для фільтрів (містить 3 RadioButton та 2 Combobox)

Також у верхній частині вікна розташований компонент MenuStrip, який дозволяє зручно здійстнювати навігацію між таблицями і запитами. Також там є можливість переглянути інформацію про додаток.

Під час перегляду запитів із активованим фільтром, пункт меню, в якому можна обрати запит для перегляду, заблокований. (Таке рішення зумовлене попередженням критичних та некритичних помилок під час виконання.) Для розблокування достатньо вимкнути фільтр.

## **Система найменувань компонентів**

Зважаючи на те, що компонентів на одній формі багато, потрібно було запровадити зрозумілу систему найменувань компонентів, щоб не заплутатися.

Спеціальні компоненти на головній формі, які виконують одразу декілька задач чи є універсальніми для різних таблиць і запитів, мають спеціальні назви. Вони є досить унікальними та легко запам’ятовуються.

Компоненти, що належать до груп додавання даних у певну таблию, мають таку систему назв, яка забезпечує легкий пошук та навігацію.

Загальний вигляд імені такого компонента має такий вигляд:

*<перша літера таблиці>\_<призначення>\_<скорочена назва компонента>*

Наприклад, A\_id\_LB – компонент Label, який використовується у якості надписа для іншого компонента, що відповідає за введення даних в поле id таблиці Authors

W - Workers

P - Posts

T - Timetable

A - Authors

G- Genres

R – Records

LB – Label.

TB - TextBox

CB - ComboBox

RB – RadioButton

NUD – Numeric Up Down

DTP - DateTimePicker

## **Заходи для стійкості програми**

Програма повинна бути стійкою, тому потрібна реалізація різноманітного захисту від речей, що призводять до критичних і некритичних помилок під час виконання.

З цією метою було реалізовано:

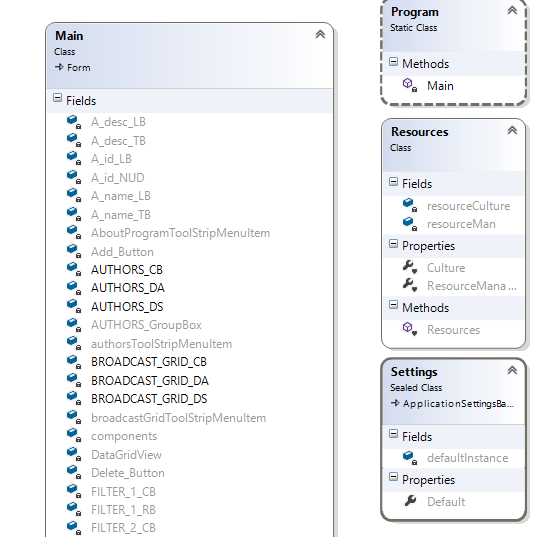
* Попередження, якщо файл БД не знаходиться за певним шляхом;
* Повідомлення про некоректність даних під час їх додавання у таблицю;
* Відключив можливість редагування тексту компонентів ComboBox в розділі фільтрів;
* Заборонив повторне відображення запитів, якщо программа зараз відображає запит із активованим фільтром. (Потрібно спочатку вимкнути фільтр);

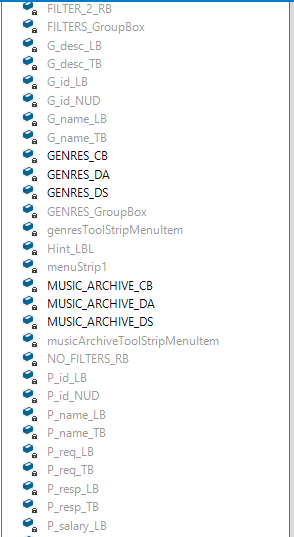
Додатково:

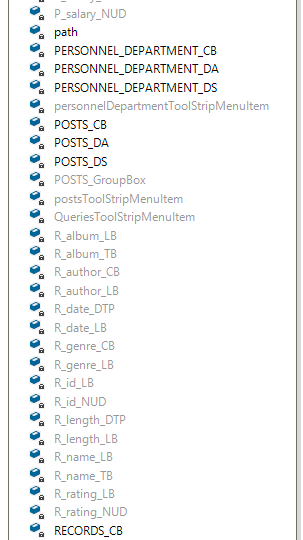
Редагування тексту компонентів ComboBox під час додавання нових записів у БД не призводить до критичної помилки (Відповідне повідомлення відображається);

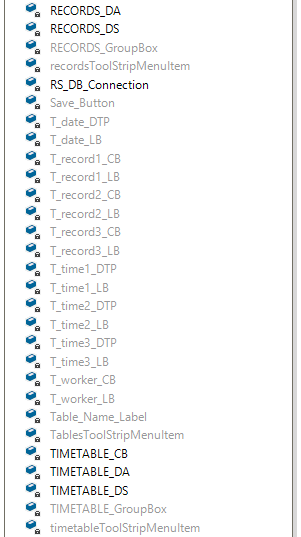
Якщо під час додавання нового запису в таблицю ключ буде збігатися із вже існуючим, то критичної помилки не виникне. Натомість, ключ самостійно отримає унікальне значення, завдяки типу даних «лічильник», який використовується в усіх таблицях для ключа.

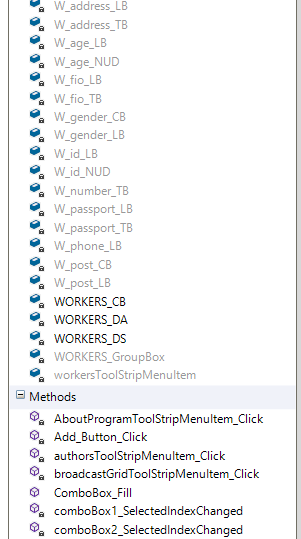
## **Діаграма класів**

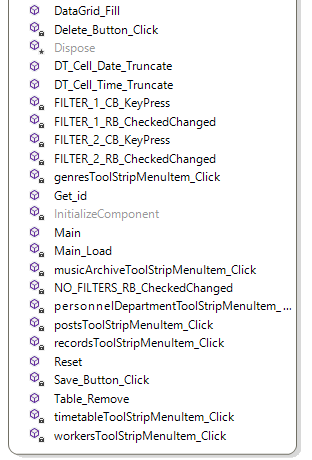


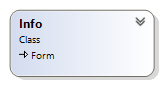












Малюнок 2. Діаграма класів.

На жаль, мій підхід до виконання на одній формі також має недоліки. ВЕЛИЧЕЗНА діаграма класів, яку ну просто неможливо вставити в документ не по частинах.

## **Опис методів**

Параметри пропущені.

Reset() – встановлює початкові значення компонентам і змінним.

DataGrid\_Fill() – дозволяє заповнити DataGridView даними із DataSet.

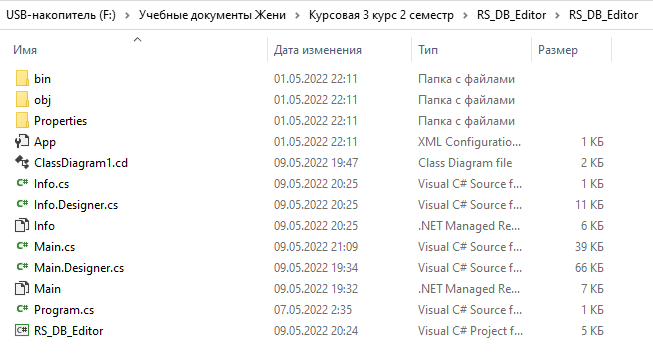
DT\_Cell\_Date\_Truncate() – дозволяє видалити непотрібну дату, залишивши тильки час (01.01.2022 17:00:00 -> 17:00:00).

DT\_Cell\_Time\_Truncate() – дозволяє видалити непотрібний час, залишивши тільки дату (01.01.2022 17:00:00 -> 01.01.2022).

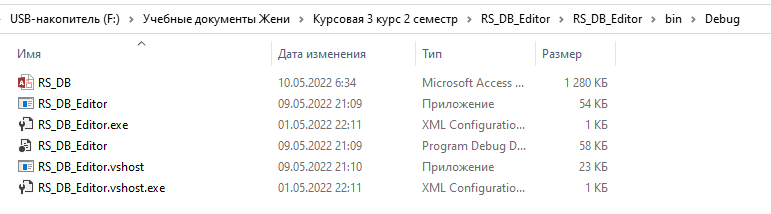
ComboBox\_Fill() – дозволяє заповнити ComboBox даними певного стовпчика певної таблиці із DataSet.

Get\_id() – дозволяє «витягти» id по значенню поля (назви чогось).

Table\_Remove() – дозволяє видалити всі виділені рядки із DataSet і DataGridView.



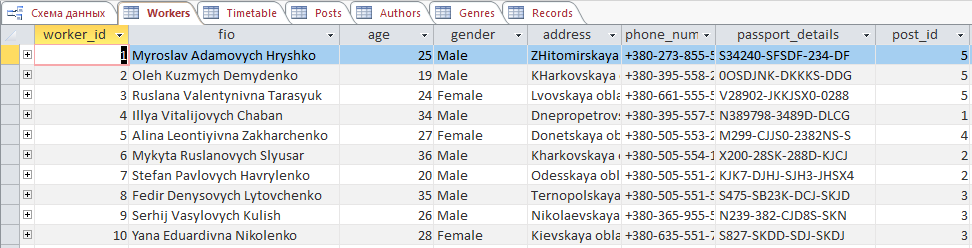
Малюнок 3. Каталог проекту.



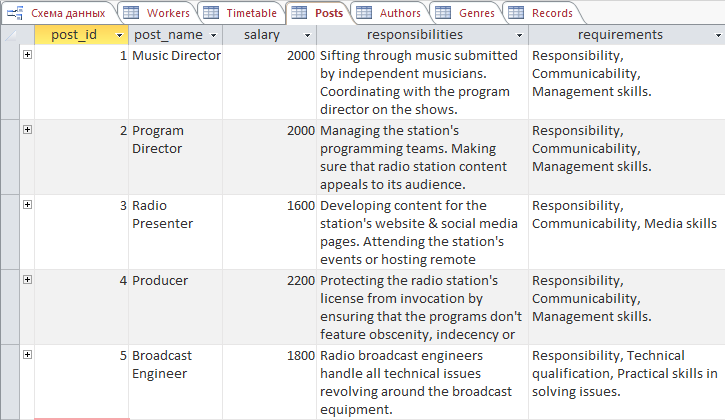
Малюнок 4. Місцезнаходження БД.

# Тестування роботи додатка

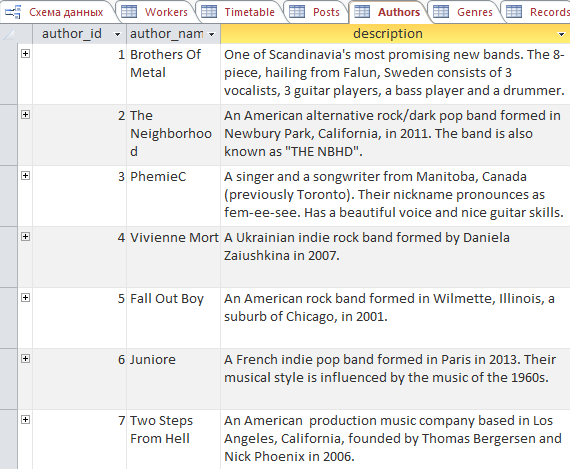
## **Тестові дані**



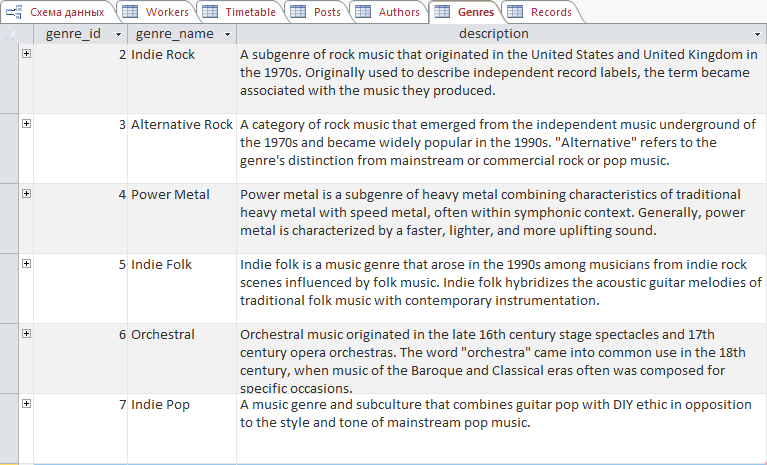
Малюнок 5. Таблиця Workers.



Малюнок 6. Таблиця Posts.



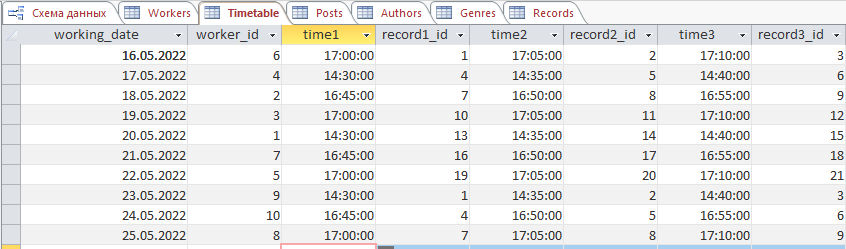
Малюнок 7. Таблиця Authors.



Малюнок 8. Таблиця Genres.

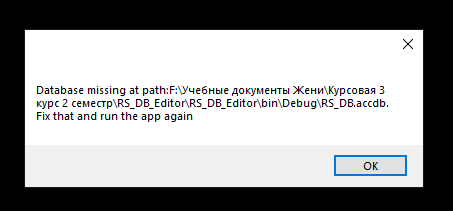


Малюнок 9. Таблиця Records.

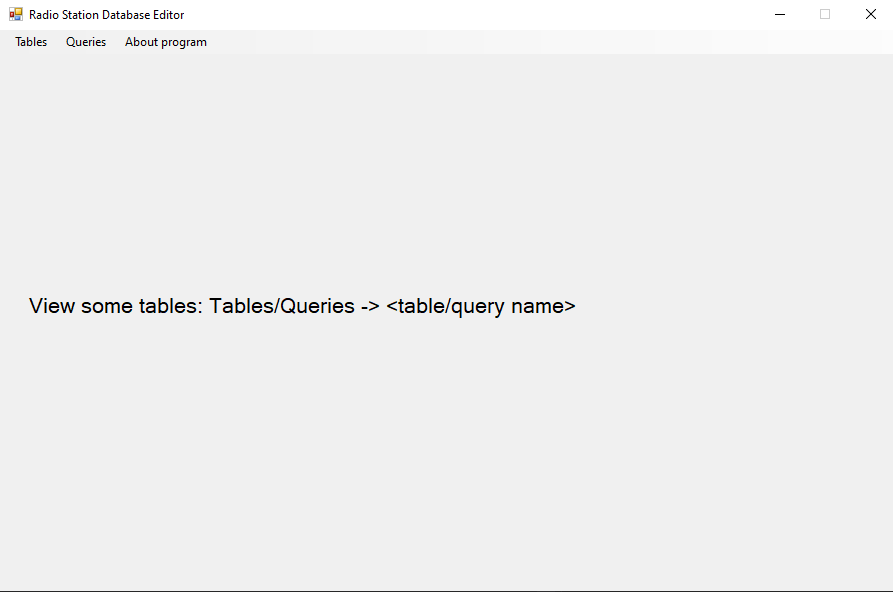


Малюнок 10. Таблиця Timetable.

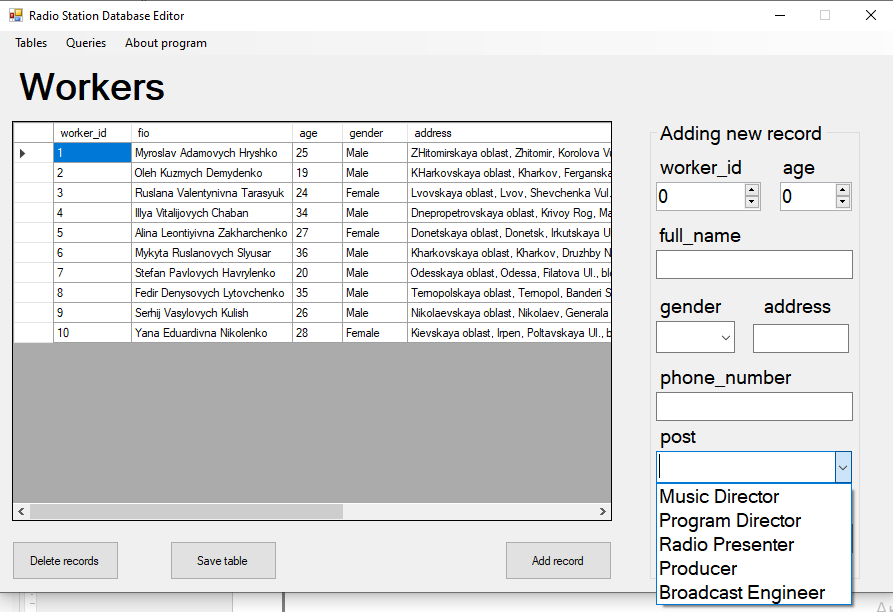
## **Скріншоти роботи програми**



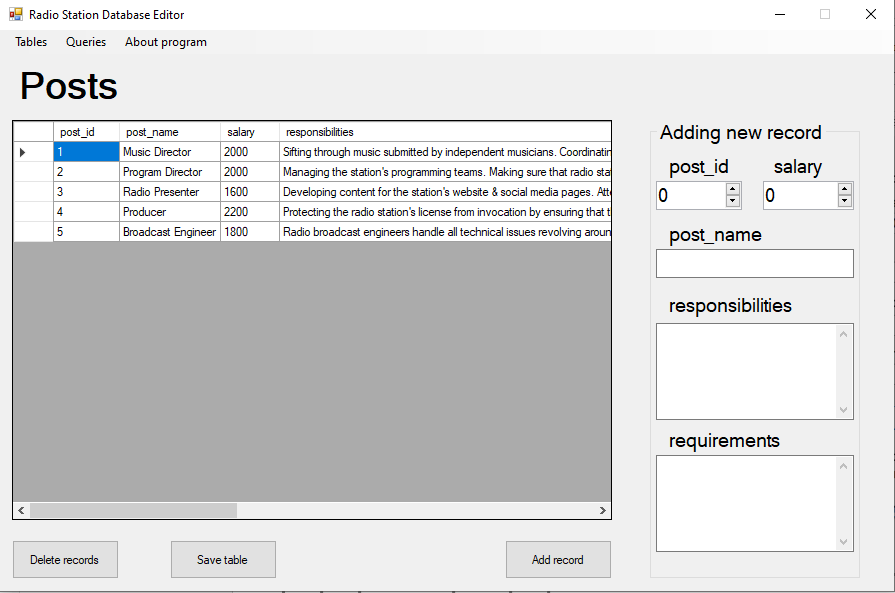
Малюнок 11. Повідомлення про відсутність БД.



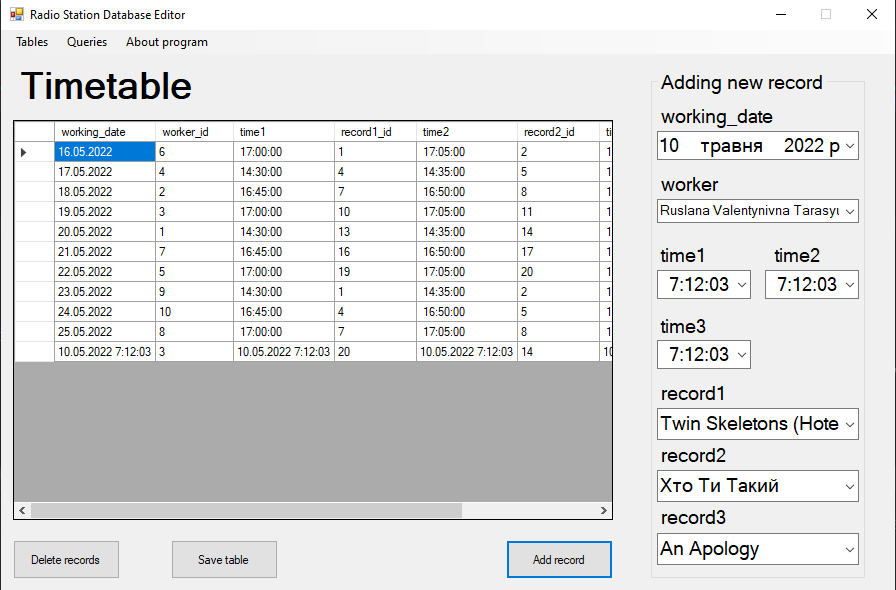
Малюнок 12.Старт.



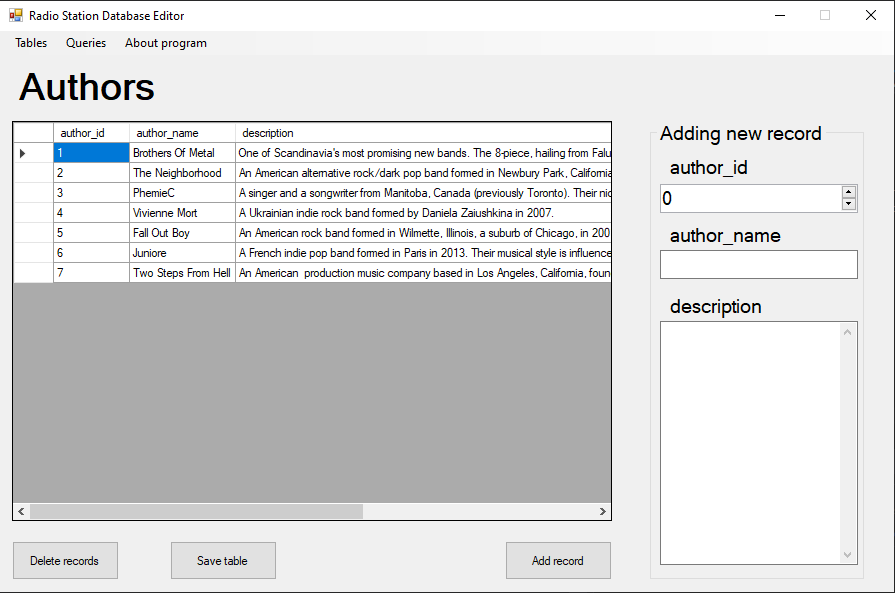
Малюнок 13.



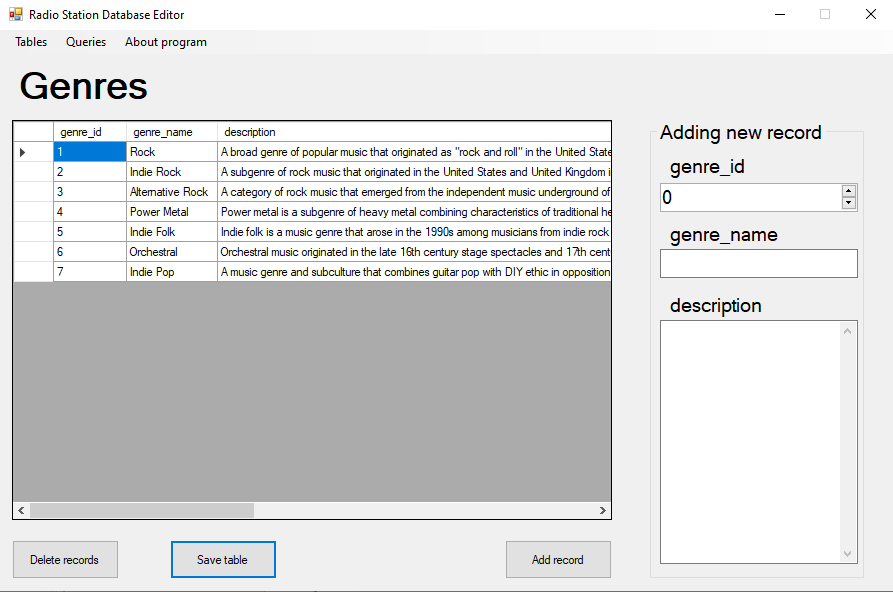
Малюнок 14.



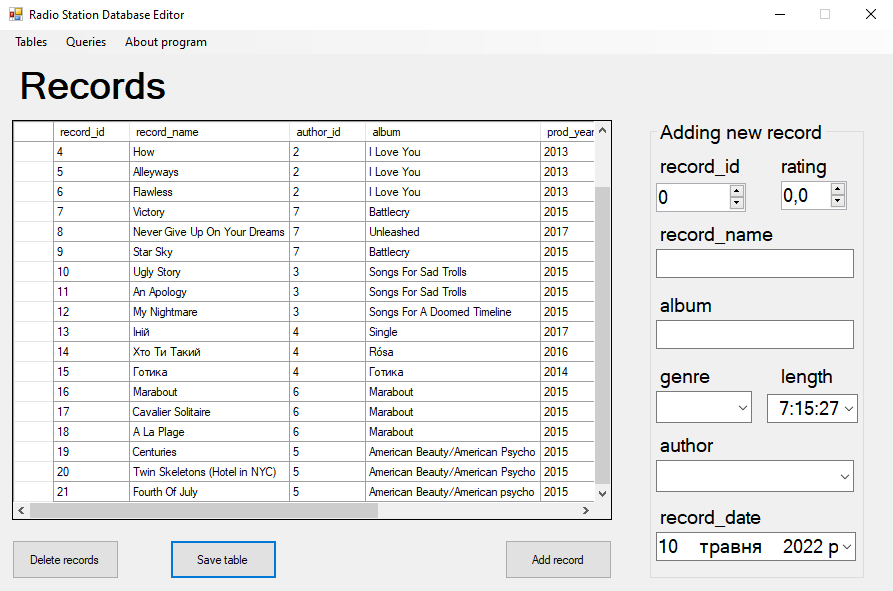
Малюнок 15. Додав новий запис.



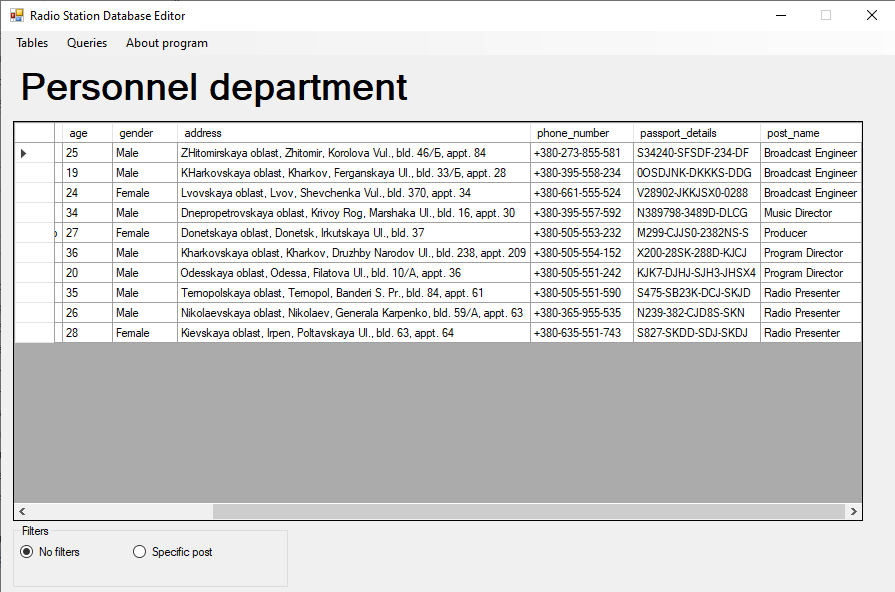
Малюнок 16.



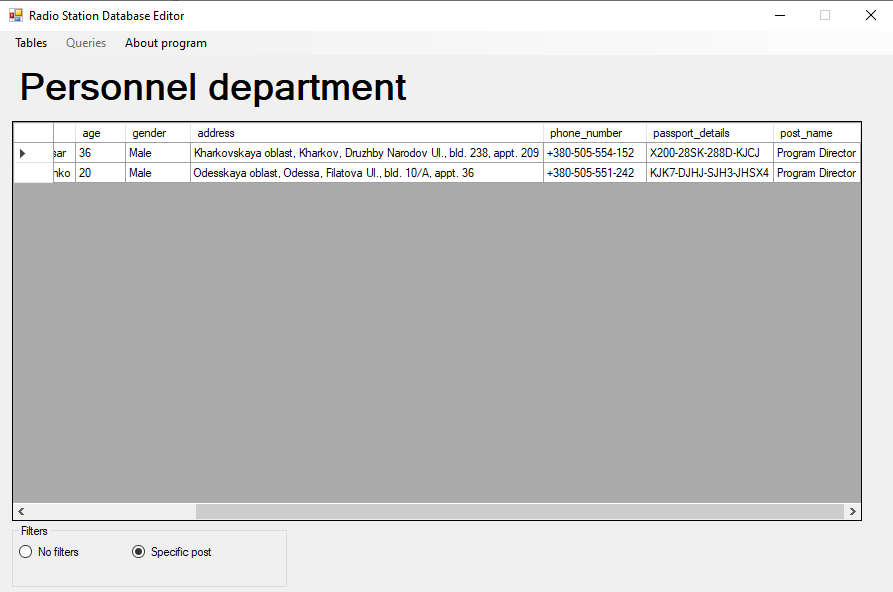
Малюнок 17.



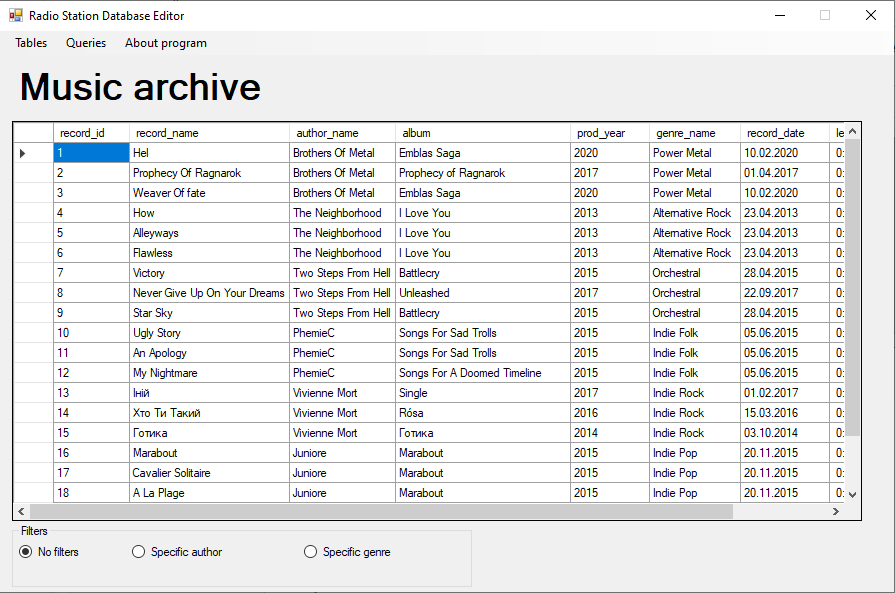
Малюнок 18.



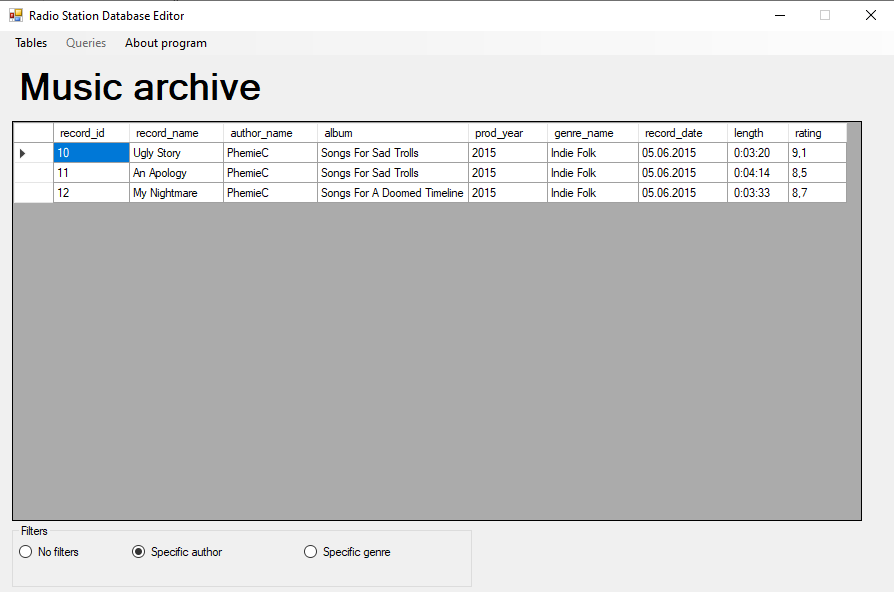
Малюнок 19. Запит Відділ кадрів



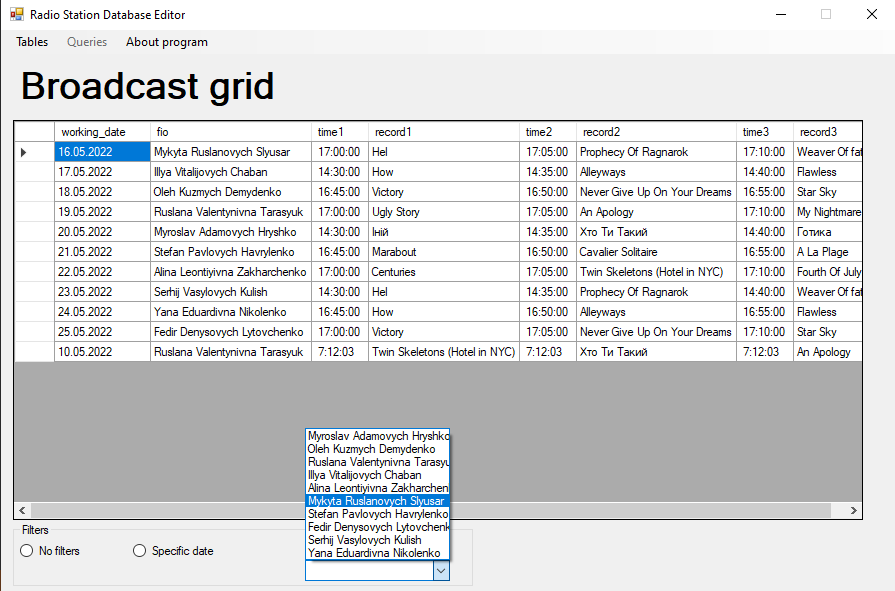
Малюнок 20. Відділ кадрів + фільтр по посаді (Program Director)



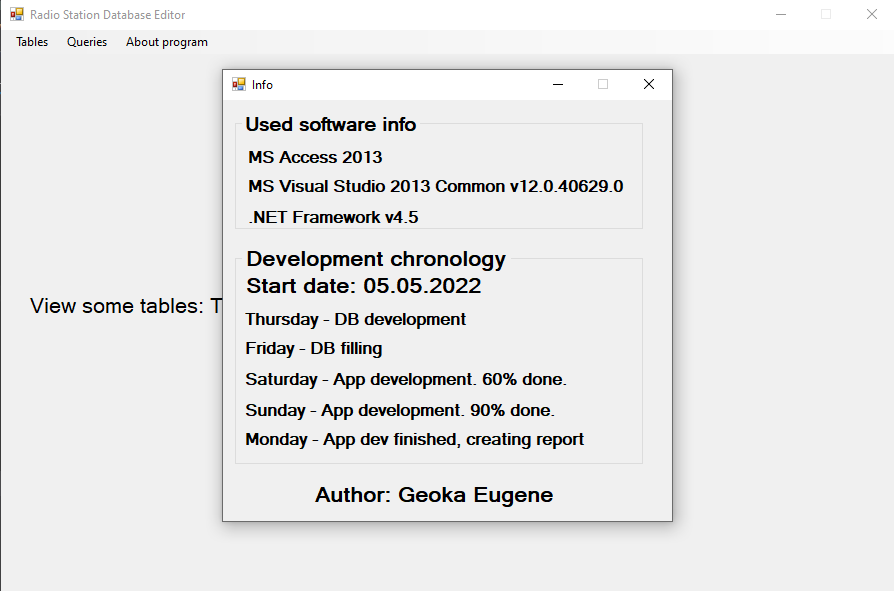
Малюнок 21. Запит Музичний архів



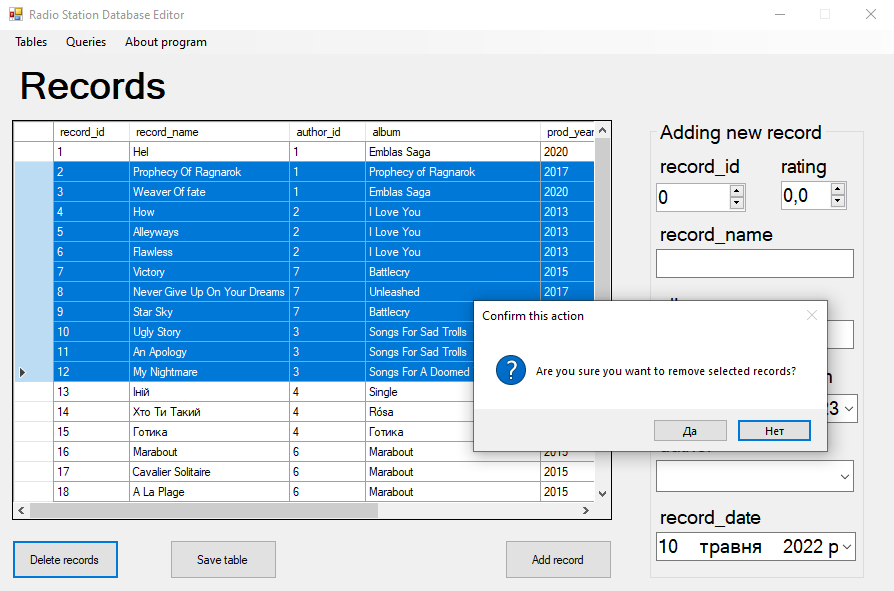
Малюнок 22. Запит Музичний архів + фільтр по виконавцю (PhemieC)



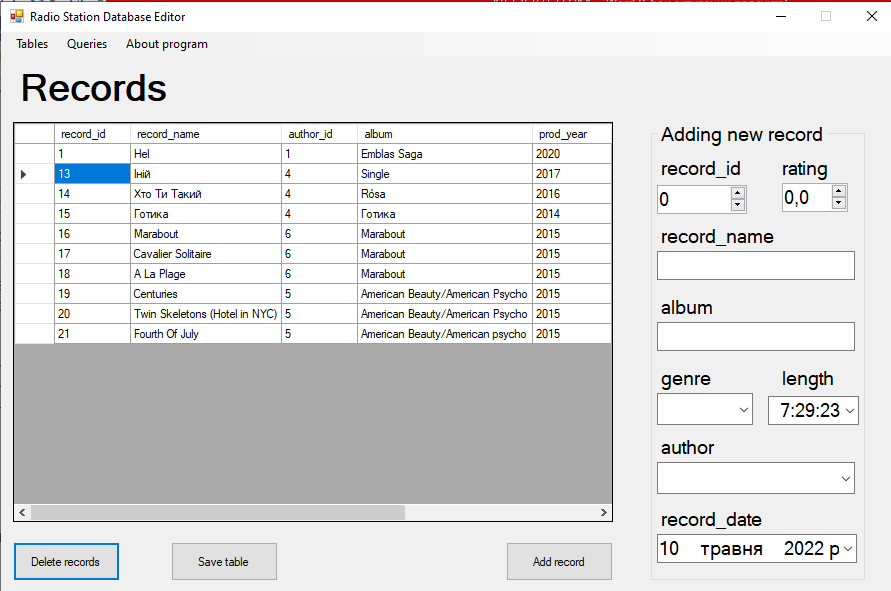
Малюнок 23. Запит Сітка мовлення + вибір ПІБ для фільтру по працівнику



Малюнок 24. Інформація про додаток



Малюнок 25. Повідомлення про підтвердження видалення записів



Малюнок 26. Записи видалені

# Висновки

Під час виконання завдання цієї курсової роботи мені довелося шукати додаткову інформацію, вивчати нові практичні прийоми, тому обидві мети були досягнуті:

* Я покращив свої навички і знання;
* Вимоги до завдань були виконані у повному обсязі і, навіть, були реалізовані додаткові функції та перевірки.

Але все ж таки моє виконання не є ідеальним. Можна додати функцію редагування даних, можна трохи змінити структуру таблиці Timetable, додавши ключове поле лічильника, що дозволить зняти ключове поле із working\_date, що зробить можливим мати різні записи на одну й ту ж дату.

Також в процесі створення цього документа я знайшов декілька нюансів, які можна виправити, щоб вдосконалити додаток:

* Якщо назва пісні містить символ **‘** то така назва не проходить перевірку на коректність даних.
* Якщо спочатку вибрати фільтр, відкрити ComboBox і нічого не вибрати, після чого зняти фокус з ComboBox, виникне критична помилка в процесі виконання програми. (Індекс обраного елемента не існує, а в подальшому він використовується).

Не зміг я залишити критичну помилку, про яку знаю. Виправив. Що стосується символів **‘** в назвах, то я просто їх видалив, і все працює.

# Перелік використаних інформаційних джерел

1. Герберт Шилдт – «C# 4.0. Повне керівництво».
2. https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/.
3. С. Макконнел – «Досконалий код». (рос. «Совершенный код»)
4. В. Пасічник «Організація баз даних та знань».

# Додаток 1. Код програми

## **Info.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace RS\_DB\_Editor

{

public partial class Info : Form

{

public Info()

{

InitializeComponent();

}

}

}

## **Main.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.OleDb;

using System.IO;

namespace RS\_DB\_Editor

{

public partial class Main : Form

{

static string path = @"F:\Учебные документы Жени\Курсовая 3 курс 2 семестр\RS\_DB\_Editor\RS\_DB\_Editor\bin\Debug\RS\_DB.accdb";

static OleDbConnection RS\_DB\_Connection = new OleDbConnection(@"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + path);

DataSet WORKERS\_DS = new DataSet();

DataSet POSTS\_DS = new DataSet();

DataSet TIMETABLE\_DS = new DataSet();

DataSet AUTHORS\_DS = new DataSet();

DataSet GENRES\_DS = new DataSet();

DataSet RECORDS\_DS = new DataSet();

DataSet PERSONNEL\_DEPARTMENT\_DS = new DataSet();

DataSet MUSIC\_ARCHIVE\_DS = new DataSet();

DataSet BROADCAST\_GRID\_DS = new DataSet();

OleDbDataAdapter WORKERS\_DA = new OleDbDataAdapter("SELECT \* FROM Workers", RS\_DB\_Connection);

OleDbDataAdapter POSTS\_DA = new OleDbDataAdapter("SELECT \* FROM Posts", RS\_DB\_Connection);

OleDbDataAdapter TIMETABLE\_DA = new OleDbDataAdapter("SELECT \* FROM Timetable", RS\_DB\_Connection);

OleDbDataAdapter AUTHORS\_DA = new OleDbDataAdapter("SELECT \* FROM Authors", RS\_DB\_Connection);

OleDbDataAdapter GENRES\_DA = new OleDbDataAdapter("SELECT \* FROM Genres", RS\_DB\_Connection);

OleDbDataAdapter RECORDS\_DA = new OleDbDataAdapter("SELECT \* FROM Records", RS\_DB\_Connection);

OleDbDataAdapter PERSONNEL\_DEPARTMENT\_DA = new OleDbDataAdapter("SELECT Workers.worker\_id, Workers.fio, Workers.age, Workers.gender, Workers.address, Workers.phone\_number, Workers.passport\_details, Posts.post\_name FROM Workers INNER JOIN Posts ON Workers.post\_id = Posts.post\_id", RS\_DB\_Connection);

OleDbDataAdapter MUSIC\_ARCHIVE\_DA = new OleDbDataAdapter("SELECT Records.record\_id, Records.record\_name, Authors.author\_name, Records.album, Records.prod\_year, Genres.genre\_name, Records.record\_date, Records.length, Records.rating FROM Genres INNER JOIN (Authors INNER JOIN Records ON Authors.author\_id = Records.author\_id) ON Genres.genre\_id = Records.genre\_id", RS\_DB\_Connection);

OleDbDataAdapter BROADCAST\_GRID\_DA = new OleDbDataAdapter("SELECT Timetable.working\_date, Workers.fio, Timetable.time1, Records.record\_name AS record1, Timetable.time2, Records\_1.record\_name AS record2, Timetable.time3, Records\_2.record\_name AS record3 FROM Records AS Records\_2 INNER JOIN (Records AS Records\_1 INNER JOIN (Records INNER JOIN (Workers INNER JOIN Timetable ON Workers.worker\_id = Timetable.worker\_id) ON Records.record\_id = Timetable.record1\_id) ON Records\_1.record\_id = Timetable.record2\_id) ON Records\_2.record\_id = Timetable.record3\_id", RS\_DB\_Connection);

OleDbCommandBuilder WORKERS\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder POSTS\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder TIMETABLE\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder AUTHORS\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder GENRES\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder RECORDS\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder PERSONNEL\_DEPARTMENT\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder MUSIC\_ARCHIVE\_CB = new OleDbCommandBuilder();

OleDbCommandBuilder BROADCAST\_GRID\_CB = new OleDbCommandBuilder();

public Main()

{

InitializeComponent();

}

// Reset method for application

public static void Reset(DataGridView DataGridView, Label Table\_Name\_Label, GroupBox WORKERS\_GroupBox, GroupBox POSTS\_GroupBox, GroupBox TIMETABLE\_GroupBox, GroupBox AUTHORS\_GroupBox, GroupBox GENRES\_GroupBox, GroupBox RECORDS\_GroupBox, GroupBox FILTERS\_GroupBox, Button Add\_Button, Button Delete\_Button, Button Save\_Button, RadioButton F0\_RB, RadioButton F1\_RB, RadioButton F2\_RB, ComboBox F1\_CB, ComboBox F2\_CB)

{

DataGridView.Visible = false;

Table\_Name\_Label.Visible = false;

DataGridView.Rows.Clear();

DataGridView.Columns.Clear();

DataGridView.Width = 600;

WORKERS\_GroupBox.Visible = false;

POSTS\_GroupBox.Visible = false;

TIMETABLE\_GroupBox.Visible = false;

AUTHORS\_GroupBox.Visible = false;

GENRES\_GroupBox.Visible = false;

RECORDS\_GroupBox.Visible = false;

FILTERS\_GroupBox.Visible = false;

FILTERS\_GroupBox.Width = 460;

F0\_RB.Text = "No filters";

F1\_RB.Text = "f1";

F2\_RB.Text = "f2";

F1\_CB.Items.Clear();

F2\_CB.Items.Clear();

F1\_CB.Visible = false;

F2\_CB.Visible = false;

Add\_Button.Visible = false;

Delete\_Button.Visible = false;

Save\_Button.Visible = false;

}

//Fill method for DataGridView (1 DataGridView for any DataSet.Tables[0])

public static void DataGrid\_Fill(DataGridView DT, DataSet DS)

{

for (int i = 0; i < DS.Tables[0].Columns.Count; i++)

{

DT.Columns.Add(DS.Tables[0].Columns[i].Caption, DS.Tables[0].Columns[i].Caption);

}

for (int m = 0; m < DS.Tables[0].Rows.Count; m++)

{

DT.Rows.Add();

for (int n = 0; n < DS.Tables[0].Columns.Count; n++)

{

DT.Rows[m].Cells[n].Value = DS.Tables[0].Rows[m].ItemArray[n];

}

}

}

//DateTime truncate method (01.01.2022 17:00:00 -> 17:00:00)

public static void DT\_Cell\_Date\_Truncate(DataGridView DT, int id)

{

string new\_value;

for (int i = 0; i < DT.Rows.Count; i++)

{

new\_value = DT.Rows[i].Cells[id].Value.ToString().Remove(0, 10);

DT.Rows[i].Cells[id].Value = new\_value;

}

}

//DateTime truncate method (01.01.2022 17:00:00 -> 01.01.2022)

public static void DT\_Cell\_Time\_Truncate(DataGridView DT, int id)

{

string new\_value;

for (int i = 0; i < DT.Rows.Count; i++)

{

new\_value = DT.Rows[i].Cells[id].Value.ToString().Remove(10);

DT.Rows[i].Cells[id].Value = new\_value;

}

}

//Fill method for ComboBox (1 ComboBox to display any Column of any DataSet.Tables[0])

public static void ComboBox\_Fill(ComboBox CB, DataSet DS, int id)

{

CB.Items.Clear();

for (int i = 0; i < DS.Tables[0].Rows.Count; i++)

{

CB.Items.Add(DS.Tables[0].Rows[i].ItemArray[id]);

}

}

//A quite tricky method to get the id of selected item in Combobox

public static string Get\_id(string input, string id\_column, string name\_column, string table)

{

RS\_DB\_Connection.Open();

OleDbCommand select\_id = RS\_DB\_Connection.CreateCommand();

select\_id.Parameters.Add("@ID\_COLUMN", OleDbType.VarChar);

select\_id.Parameters.Add("@TABLE", OleDbType.VarChar);

select\_id.Parameters.Add("@NAME\_COLUMN", OleDbType.VarChar);

select\_id.Parameters.Add("@INPUT", OleDbType.VarChar);

select\_id.Parameters["@ID\_COLUMN"].Value = id\_column;

select\_id.Parameters["@TABLE"].Value = table;

select\_id.Parameters["@NAME\_COLUMN"].Value = name\_column;

select\_id.Parameters["@INPUT"].Value = input;

select\_id.CommandText = "SELECT " + id\_column + " FROM " + table + " WHERE " + name\_column + " = '" + input + "';";

string result = select\_id.ExecuteScalar().ToString();

RS\_DB\_Connection.Close();

return result;

}

//A method that removes all selected records from DataGridView and current DataSet

public static void Table\_Remove(DataGridView DT, DataSet DS, OleDbDataAdapter DA)

{

if (MessageBox.Show("Are you sure you want to remove selected records?", "Confirm this action", MessageBoxButtons.YesNo, MessageBoxIcon.Question, MessageBoxDefaultButton.Button2) == DialogResult.Yes)

{

int temp;

for (int i = DT.SelectedRows.Count - 1; i >= 0; i--)

{

temp = DT.SelectedRows[i].Index;

DT.Rows.RemoveAt(temp);

DS.Tables[0].Rows[temp].Delete();

DA.Update(DS);

}

}

}

private void Main\_Load(object sender, EventArgs e)

{

Width = 910;

CenterToScreen();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

//Checking whether the database file exists

if (File.Exists(path) == false)

{

MessageBox.Show("Database missing at path:" + path + ". Fix that and run the app again");

Environment.Exit(0);

}

WORKERS\_CB.DataAdapter = WORKERS\_DA;

WORKERS\_DA.InsertCommand = WORKERS\_CB.GetInsertCommand();

WORKERS\_DA.DeleteCommand = WORKERS\_CB.GetDeleteCommand();

WORKERS\_DA.UpdateCommand = WORKERS\_CB.GetUpdateCommand();

WORKERS\_DA.Fill(WORKERS\_DS);

POSTS\_CB.DataAdapter = POSTS\_DA;

POSTS\_DA.InsertCommand = POSTS\_CB.GetInsertCommand();

POSTS\_DA.DeleteCommand = POSTS\_CB.GetDeleteCommand();

POSTS\_DA.UpdateCommand = POSTS\_CB.GetUpdateCommand();

POSTS\_DA.Fill(POSTS\_DS);

TIMETABLE\_CB.DataAdapter = TIMETABLE\_DA;

TIMETABLE\_DA.InsertCommand = TIMETABLE\_CB.GetInsertCommand();

TIMETABLE\_DA.DeleteCommand = TIMETABLE\_CB.GetDeleteCommand();

TIMETABLE\_DA.UpdateCommand = TIMETABLE\_CB.GetUpdateCommand();

TIMETABLE\_DA.Fill(TIMETABLE\_DS);

AUTHORS\_CB.DataAdapter = AUTHORS\_DA;

AUTHORS\_DA.InsertCommand = AUTHORS\_CB.GetInsertCommand();

AUTHORS\_DA.DeleteCommand = AUTHORS\_CB.GetDeleteCommand();

AUTHORS\_DA.UpdateCommand = AUTHORS\_CB.GetUpdateCommand();

AUTHORS\_DA.Fill(AUTHORS\_DS);

GENRES\_CB.DataAdapter = GENRES\_DA;

GENRES\_DA.InsertCommand = GENRES\_CB.GetInsertCommand();

GENRES\_DA.DeleteCommand = GENRES\_CB.GetDeleteCommand();

GENRES\_DA.UpdateCommand = GENRES\_CB.GetUpdateCommand();

GENRES\_DA.Fill(GENRES\_DS);

RECORDS\_CB.DataAdapter = RECORDS\_DA;

RECORDS\_DA.InsertCommand = RECORDS\_CB.GetInsertCommand();

RECORDS\_DA.DeleteCommand = RECORDS\_CB.GetDeleteCommand();

RECORDS\_DA.UpdateCommand = RECORDS\_CB.GetUpdateCommand();

RECORDS\_DA.Fill(RECORDS\_DS);

}

private void workersToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//workers table

WORKERS\_DS.Clear();

DataGridView.Rows.Clear();

DataGridView.Columns.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

WORKERS\_CB.DataAdapter = WORKERS\_DA;

WORKERS\_DA.Fill(WORKERS\_DS);

DataGrid\_Fill(DataGridView, WORKERS\_DS);

W\_gender\_CB.Items.Clear();

W\_gender\_CB.Items.Add("Male");

W\_gender\_CB.Items.Add("Female");

ComboBox\_Fill(W\_post\_CB, POSTS\_DS, 1);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Workers";

Table\_Name\_Label.Visible = true;

WORKERS\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void postsToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//posts table

POSTS\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

POSTS\_CB.DataAdapter = POSTS\_DA;

POSTS\_DA.Fill(POSTS\_DS);

DataGrid\_Fill(DataGridView, POSTS\_DS);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Posts";

Table\_Name\_Label.Visible = true;

POSTS\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void timetableToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//timetable table

TIMETABLE\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

TIMETABLE\_CB.DataAdapter = TIMETABLE\_DA;

TIMETABLE\_DA.Fill(TIMETABLE\_DS);

DataGrid\_Fill(DataGridView, TIMETABLE\_DS);

DT\_Cell\_Time\_Truncate(DataGridView, 0);

DT\_Cell\_Date\_Truncate(DataGridView, 2);

DT\_Cell\_Date\_Truncate(DataGridView, 4);

DT\_Cell\_Date\_Truncate(DataGridView, 6);

ComboBox\_Fill(T\_worker\_CB, WORKERS\_DS, 1);

ComboBox\_Fill(T\_record1\_CB, RECORDS\_DS, 1);

ComboBox\_Fill(T\_record2\_CB, RECORDS\_DS, 1);

ComboBox\_Fill(T\_record3\_CB, RECORDS\_DS, 1);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Timetable";

Table\_Name\_Label.Visible = true;

TIMETABLE\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void authorsToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//authors table

AUTHORS\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

AUTHORS\_CB.DataAdapter = AUTHORS\_DA;

AUTHORS\_DA.Fill(AUTHORS\_DS);

DataGrid\_Fill(DataGridView, AUTHORS\_DS);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Authors";

Table\_Name\_Label.Visible = true;

AUTHORS\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void genresToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//genres table

GENRES\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

GENRES\_CB.DataAdapter = GENRES\_DA;

GENRES\_DA.Fill(GENRES\_DS);

DataGrid\_Fill(DataGridView, GENRES\_DS);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Genres";

Table\_Name\_Label.Visible = true;

GENRES\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void recordsToolStripMenuItem\_Click(object sender, EventArgs e)

{

NO\_FILTERS\_RB.Checked = true;

QueriesToolStripMenuItem.Enabled = true;

//records table

RECORDS\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

RECORDS\_CB.DataAdapter = RECORDS\_DA;

RECORDS\_DA.Fill(RECORDS\_DS);

DataGrid\_Fill(DataGridView, RECORDS\_DS);

DT\_Cell\_Time\_Truncate(DataGridView, 6);

DT\_Cell\_Date\_Truncate(DataGridView, 7);

ComboBox\_Fill(R\_genre\_CB, GENRES\_DS, 1);

ComboBox\_Fill(R\_author\_CB, AUTHORS\_DS, 1);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Records";

Table\_Name\_Label.Visible = true;

RECORDS\_GroupBox.Visible = true;

Add\_Button.Visible = true;

Delete\_Button.Visible = true;

Save\_Button.Visible = true;

}

private void personnelDepartmentToolStripMenuItem\_Click(object sender, EventArgs e)

{

// query 1

PERSONNEL\_DEPARTMENT\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

PERSONNEL\_DEPARTMENT\_CB.DataAdapter = PERSONNEL\_DEPARTMENT\_DA;

PERSONNEL\_DEPARTMENT\_DA.Fill(PERSONNEL\_DEPARTMENT\_DS);

DataGrid\_Fill(DataGridView, PERSONNEL\_DEPARTMENT\_DS);

DataGridView.Width = 850;

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Personnel department";

Table\_Name\_Label.Visible = true;

FILTER\_1\_RB.Text = "Specific post";

FILTER\_2\_RB.Visible = false;

FILTERS\_GroupBox.Width = 275;

FILTERS\_GroupBox.Visible = true;

}

private void musicArchiveToolStripMenuItem\_Click(object sender, EventArgs e)

{

// query 2

MUSIC\_ARCHIVE\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

MUSIC\_ARCHIVE\_CB.DataAdapter = MUSIC\_ARCHIVE\_DA;

MUSIC\_ARCHIVE\_DA.Fill(MUSIC\_ARCHIVE\_DS);

DataGrid\_Fill(DataGridView, MUSIC\_ARCHIVE\_DS);

DataGridView.Width = 850;

DT\_Cell\_Time\_Truncate(DataGridView, 6);

DT\_Cell\_Date\_Truncate(DataGridView, 7);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Music archive";

Table\_Name\_Label.Visible = true;

FILTER\_1\_RB.Text = "Specific author";

FILTER\_2\_RB.Text = "Specific genre";

FILTER\_1\_RB.Visible = true;

FILTER\_2\_RB.Visible = true;

FILTERS\_GroupBox.Visible = true;

}

private void broadcastGridToolStripMenuItem\_Click(object sender, EventArgs e)

{

//query 3

BROADCAST\_GRID\_DS.Clear();

Reset(DataGridView, Table\_Name\_Label, WORKERS\_GroupBox, POSTS\_GroupBox, TIMETABLE\_GroupBox, AUTHORS\_GroupBox, GENRES\_GroupBox, RECORDS\_GroupBox, FILTERS\_GroupBox, Add\_Button, Delete\_Button, Save\_Button, NO\_FILTERS\_RB, FILTER\_1\_RB, FILTER\_2\_RB, FILTER\_1\_CB, FILTER\_2\_CB);

BROADCAST\_GRID\_CB.DataAdapter = BROADCAST\_GRID\_DA;

BROADCAST\_GRID\_DA.Fill(BROADCAST\_GRID\_DS);

DataGrid\_Fill(DataGridView, BROADCAST\_GRID\_DS);

DataGridView.Width = 850;

DT\_Cell\_Time\_Truncate(DataGridView, 0);

DT\_Cell\_Date\_Truncate(DataGridView, 2);

DT\_Cell\_Date\_Truncate(DataGridView, 4);

DT\_Cell\_Date\_Truncate(DataGridView, 6);

DataGridView.Visible = true;

Table\_Name\_Label.Text = "Broadcast grid";

Table\_Name\_Label.Visible = true;

FILTER\_1\_RB.Text = "Specific date";

FILTER\_2\_RB.Text = "Specific worker";

FILTER\_1\_RB.Visible = true;

FILTER\_2\_RB.Visible = true;

FILTERS\_GroupBox.Visible = true;

}

private void AboutProgramToolStripMenuItem\_Click(object sender, EventArgs e)

{

QueriesToolStripMenuItem.Enabled = true;

// about program

Info info = new Info();

info.ShowDialog();

}

private void Add\_Button\_Click(object sender, EventArgs e)

{

switch (Table\_Name\_Label.Text)

{

case "Workers":

{

try

{

WORKERS\_DS.Tables[0].Rows.Add(W\_id\_NUD.Value.ToString(), W\_fio\_TB.Text, W\_age\_NUD.Value.ToString(), W\_gender\_CB.SelectedItem.ToString(), W\_address\_TB.Text, W\_number\_TB.Text, W\_passport\_TB.Text, Get\_id(W\_post\_CB.SelectedItem.ToString(), "post\_id", "post\_name", "Posts"));

DataGridView.Rows.Add(W\_id\_NUD.Value.ToString(), W\_fio\_TB.Text, W\_age\_NUD.Value.ToString(), W\_gender\_CB.SelectedItem.ToString(), W\_address\_TB.Text, W\_number\_TB.Text, W\_passport\_TB.Text, Get\_id(W\_post\_CB.SelectedItem.ToString(), "post\_id", "post\_name", "Posts"));

WORKERS\_DA.Update(WORKERS\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

case "Posts":

{

try

{

POSTS\_DS.Tables[0].Rows.Add(P\_id\_NUD.Value.ToString(), P\_name\_TB.Text, P\_salary\_NUD.Value.ToString(), P\_resp\_TB.Text, P\_req\_TB.Text);

DataGridView.Rows.Add(P\_id\_NUD.Value.ToString(), P\_name\_TB.Text, P\_salary\_NUD.Value.ToString(), P\_resp\_TB.Text, P\_req\_TB.Text);

POSTS\_DA.Update(POSTS\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

case "Timetable":

{

try

{

TIMETABLE\_DS.Tables[0].Rows.Add(T\_date\_DTP.Value.ToString(), Get\_id(T\_worker\_CB.SelectedItem.ToString(), "worker\_id", "fio", "Workers"), T\_time1\_DTP.Value.ToString(), Get\_id(T\_record1\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"), T\_time2\_DTP.Value.ToString(), Get\_id(T\_record2\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"), T\_time3\_DTP.Value.ToString(), Get\_id(T\_record3\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"));

DataGridView.Rows.Add(T\_date\_DTP.Value.ToString(), Get\_id(T\_worker\_CB.SelectedItem.ToString(), "worker\_id", "fio", "Workers"), T\_time1\_DTP.Value.ToString(), Get\_id(T\_record1\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"), T\_time2\_DTP.Value.ToString(), Get\_id(T\_record2\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"), T\_time3\_DTP.Value.ToString(), Get\_id(T\_record3\_CB.SelectedItem.ToString(), "record\_id", "record\_name", "Records"));

TIMETABLE\_DA.Update(TIMETABLE\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

case "Authors":

{

try

{

AUTHORS\_DS.Tables[0].Rows.Add(A\_id\_NUD.Value.ToString(), A\_name\_TB.Text, A\_desc\_TB.Text);

DataGridView.Rows.Add(A\_id\_NUD.Value.ToString(), A\_name\_TB.Text, A\_desc\_TB.Text);

AUTHORS\_DA.Update(AUTHORS\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

case "Genres":

{

try

{

GENRES\_DS.Tables[0].Rows.Add(G\_id\_NUD.Value.ToString(), G\_name\_TB.Text, G\_desc\_TB.Text);

DataGridView.Rows.Add(G\_id\_NUD.Value.ToString(), G\_name\_TB.Text, G\_desc\_TB.Text);

GENRES\_DA.Update(GENRES\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

case "Records":

{

try

{

string prod\_year = R\_date\_DTP.Value.ToString().Remove(0, 6);

prod\_year = prod\_year.Remove(4);

RECORDS\_DS.Tables[0].Rows.Add(R\_id\_NUD.Value.ToString(), R\_name\_TB.Text, Get\_id(R\_author\_CB.SelectedItem.ToString(), "author\_id", "author\_name", "Authors"), R\_album\_TB.Text, prod\_year, Get\_id(R\_genre\_CB.SelectedItem.ToString(), "genre\_id", "genre\_name", "Genres"), R\_date\_DTP.Value.ToString(), R\_length\_DTP.Value.ToString(), R\_rating\_NUD.Value.ToString());

DataGridView.Rows.Add(R\_id\_NUD.Value.ToString(), R\_name\_TB.Text, Get\_id(R\_author\_CB.SelectedItem.ToString(), "author\_id", "author\_name", "Authors"), R\_album\_TB.Text, prod\_year, Get\_id(R\_genre\_CB.SelectedItem.ToString(), "genre\_id", "genre\_name", "Genres"), R\_date\_DTP.Value.ToString(), R\_length\_DTP.Value.ToString(), R\_rating\_NUD.Value.ToString());

RECORDS\_DA.Update(RECORDS\_DS);

}

catch

{

MessageBox.Show("Data input error! Make sure the data is correct and primary key is unique!");

}

}

break;

}

}

private void Delete\_Button\_Click(object sender, EventArgs e)

{

switch (Table\_Name\_Label.Text)

{

case "Workers":

{

Table\_Remove(DataGridView, WORKERS\_DS, WORKERS\_DA);

}

break;

case "Posts":

{

Table\_Remove(DataGridView, POSTS\_DS, POSTS\_DA);

}

break;

case "Timetable":

{

Table\_Remove(DataGridView, TIMETABLE\_DS, TIMETABLE\_DA);

}

break;

case "Authors":

{

Table\_Remove(DataGridView, AUTHORS\_DS, AUTHORS\_DA);

}

break;

case "Genres":

{

Table\_Remove(DataGridView, GENRES\_DS, GENRES\_DA);

}

break;

case "Records":

{

Table\_Remove(DataGridView, RECORDS\_DS, RECORDS\_DA);

}

break;

}

}

private void Save\_Button\_Click(object sender, EventArgs e)

{

WORKERS\_DA.Update(WORKERS\_DS);

POSTS\_DA.Update(POSTS\_DS);

TIMETABLE\_DA.Update(TIMETABLE\_DS);

AUTHORS\_DA.Update(AUTHORS\_DS);

GENRES\_DA.Update(GENRES\_DS);

RECORDS\_DA.Update(RECORDS\_DS);

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (FILTER\_1\_CB.SelectedIndex < 0)

{

return;

}

switch (Table\_Name\_Label.Text)

{

case "Personnel department":

{

PERSONNEL\_DEPARTMENT\_DA.SelectCommand.CommandText = "SELECT Workers.worker\_id, Workers.fio, Workers.age, Workers.gender, Workers.address, Workers.phone\_number, Workers.passport\_details, Posts.post\_name FROM Workers INNER JOIN Posts ON Workers.post\_id = Posts.post\_id WHERE Posts.post\_name = '"+FILTER\_1\_CB.SelectedItem.ToString()+"';";

personnelDepartmentToolStripMenuItem\_Click(sender, e);

}

break;

case "Music archive":

{

MUSIC\_ARCHIVE\_DA.SelectCommand.CommandText = "SELECT Records.record\_id, Records.record\_name, Authors.author\_name, Records.album, Records.prod\_year, Genres.genre\_name, Records.record\_date, Records.length, Records.rating FROM Genres INNER JOIN (Authors INNER JOIN Records ON Authors.author\_id = Records.author\_id) ON Genres.genre\_id = Records.genre\_id WHERE Authors.author\_name = '" + FILTER\_1\_CB.SelectedItem.ToString() + "';";

musicArchiveToolStripMenuItem\_Click(sender, e);

}

break;

case "Broadcast grid":

{

// "date" is a converted input of type 01.01.2022 -> #01/01/2022#

string date = "#" + FILTER\_1\_CB.SelectedItem.ToString().Remove(2) + "/" + FILTER\_1\_CB.SelectedItem.ToString().Remove(0, 3).Remove(2) + "/" + FILTER\_1\_CB.SelectedItem.ToString().Remove(0, 6).Remove(4) + "#";

BROADCAST\_GRID\_DA.SelectCommand.CommandText = "SELECT Timetable.working\_date, Workers.fio, Timetable.time1, Records.record\_name AS record1, Timetable.time2, Records\_1.record\_name AS record2, Timetable.time3, Records\_2.record\_name AS record3 FROM Records AS Records\_2 INNER JOIN (Records AS Records\_1 INNER JOIN (Records INNER JOIN (Workers INNER JOIN Timetable ON Workers.worker\_id = Timetable.worker\_id) ON Records.record\_id = Timetable.record1\_id) ON Records\_1.record\_id = Timetable.record2\_id) ON Records\_2.record\_id = Timetable.record3\_id WHERE Timetable.working\_date = " + date + ";";

broadcastGridToolStripMenuItem\_Click(sender, e);

}

break;

}

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

if (FILTER\_2\_CB.SelectedIndex < 0)

{

return;

}

switch (Table\_Name\_Label.Text)

{

case "Personnel department":

{

//nothing

}

break;

case "Music archive":

{

MUSIC\_ARCHIVE\_DA.SelectCommand.CommandText = "SELECT Records.record\_id, Records.record\_name, Authors.author\_name, Records.album, Records.prod\_year, Genres.genre\_name, Records.record\_date, Records.length, Records.rating FROM Genres INNER JOIN (Authors INNER JOIN Records ON Authors.author\_id = Records.author\_id) ON Genres.genre\_id = Records.genre\_id WHERE Genres.genre\_name = '" + FILTER\_2\_CB.SelectedItem.ToString() + "';";

musicArchiveToolStripMenuItem\_Click(sender, e);

}

break;

case "Broadcast grid":

{

BROADCAST\_GRID\_DA.SelectCommand.CommandText = "SELECT Timetable.working\_date, Workers.fio, Timetable.time1, Records.record\_name AS record1, Timetable.time2, Records\_1.record\_name AS record2, Timetable.time3, Records\_2.record\_name AS record3 FROM Records AS Records\_2 INNER JOIN (Records AS Records\_1 INNER JOIN (Records INNER JOIN (Workers INNER JOIN Timetable ON Workers.worker\_id = Timetable.worker\_id) ON Records.record\_id = Timetable.record1\_id) ON Records\_1.record\_id = Timetable.record2\_id) ON Records\_2.record\_id = Timetable.record3\_id WHERE Workers.fio = '" + FILTER\_2\_CB.SelectedItem.ToString() + "';";

broadcastGridToolStripMenuItem\_Click(sender, e);

}

break;

}

}

//preventing editing of ComboBoxes in unprotected area of code

private void FILTER\_1\_CB\_KeyPress(object sender, KeyPressEventArgs e)

{

e.Handled = true;

}

private void FILTER\_2\_CB\_KeyPress(object sender, KeyPressEventArgs e)

{

e.Handled = true;

}

private void NO\_FILTERS\_RB\_CheckedChanged(object sender, EventArgs e)

{

if (NO\_FILTERS\_RB.Checked)

{

switch (Table\_Name\_Label.Text)

{

case "Personnel department":

{

PERSONNEL\_DEPARTMENT\_DA.SelectCommand.CommandText = "SELECT Workers.worker\_id, Workers.fio, Workers.age, Workers.gender, Workers.address, Workers.phone\_number, Workers.passport\_details, Posts.post\_name FROM Workers INNER JOIN Posts ON Workers.post\_id = Posts.post\_id";

FILTER\_1\_CB.Visible = false;

FILTER\_2\_CB.Visible = false;

personnelDepartmentToolStripMenuItem\_Click(sender, e);

}

break;

case "Music archive":

{

MUSIC\_ARCHIVE\_DA.SelectCommand.CommandText = "SELECT Records.record\_id, Records.record\_name, Authors.author\_name, Records.album, Records.prod\_year, Genres.genre\_name, Records.record\_date, Records.length, Records.rating FROM Genres INNER JOIN (Authors INNER JOIN Records ON Authors.author\_id = Records.author\_id) ON Genres.genre\_id = Records.genre\_id";

FILTER\_1\_CB.Visible = false;

FILTER\_2\_CB.Visible = false;

musicArchiveToolStripMenuItem\_Click(sender, e);

}

break;

case "Broadcast grid":

{

BROADCAST\_GRID\_DA.SelectCommand.CommandText = "SELECT Timetable.working\_date, Workers.fio, Timetable.time1, Records.record\_name AS record1, Timetable.time2, Records\_1.record\_name AS record2, Timetable.time3, Records\_2.record\_name AS record3 FROM Records AS Records\_2 INNER JOIN (Records AS Records\_1 INNER JOIN (Records INNER JOIN (Workers INNER JOIN Timetable ON Workers.worker\_id = Timetable.worker\_id) ON Records.record\_id = Timetable.record1\_id) ON Records\_1.record\_id = Timetable.record2\_id) ON Records\_2.record\_id = Timetable.record3\_id";

FILTER\_1\_CB.Visible = false;

FILTER\_2\_CB.Visible = false;

broadcastGridToolStripMenuItem\_Click(sender, e);

}

break;

}

QueriesToolStripMenuItem.Enabled = true;

}

}

private void FILTER\_1\_RB\_CheckedChanged(object sender, EventArgs e)

{

if (FILTER\_1\_RB.Checked)

{

switch (Table\_Name\_Label.Text)

{

case "Personnel department":

{

FILTER\_1\_CB.Visible = true;

FILTER\_2\_CB.Visible = false;

ComboBox\_Fill(FILTER\_1\_CB, POSTS\_DS, 1);

}

break;

case "Music archive":

{

FILTER\_1\_CB.Visible = true;

FILTER\_2\_CB.Visible = false;

ComboBox\_Fill(FILTER\_1\_CB, AUTHORS\_DS, 1);

}

break;

case "Broadcast grid":

{

FILTER\_1\_CB.Visible = true;

FILTER\_2\_CB.Visible = false;

ComboBox\_Fill(FILTER\_1\_CB, TIMETABLE\_DS, 0);

}

break;

}

QueriesToolStripMenuItem.Enabled = false;

}

}

private void FILTER\_2\_RB\_CheckedChanged(object sender, EventArgs e)

{

if (FILTER\_2\_RB.Checked)

{

switch (Table\_Name\_Label.Text)

{

case "Personnel department":

{

// nothing

}

break;

case "Music archive":

{

FILTER\_1\_CB.Visible = false;

FILTER\_2\_CB.Visible = true;

ComboBox\_Fill(FILTER\_2\_CB, GENRES\_DS, 1);

}

break;

case "Broadcast grid":

{

FILTER\_1\_CB.Visible = false;

FILTER\_2\_CB.Visible = true;

ComboBox\_Fill(FILTER\_2\_CB, WORKERS\_DS, 1);

}

break;

}

QueriesToolStripMenuItem.Enabled = false;

}

}

}

}