МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ

Государственное бюджетное профессиональное образовательное учреждение Московской области «Люберецкий техникум имени Героя Советского Союза, летчика-космонавта Ю.А.Гагарина»

КУРСОВАЯ РАБОТА

Фомин Игорь Андреевич

по профессиональному модулю (дисциплине)

ОП.08 Основы проектирования баз данных

Курс 2 Группа № ИС-21

тема: спроектировать БД автоматизированной регистрации документов, сопровождающих управленческую деятельность и (или) кадровый учёт на некотором предприятии

Выполнил студент _____ Фомин Игорь Андреевич
(подпись) (ФИО полностью)

Руководитель _____ Тарджиманян Лия Николаевна
(подпись) (ФИО полностью)

Оценка _____

Дзержинский 2023

Содержание

Введение……..…….3
Постановка задачи…….……..4
Глава 1 (Анализ литературы).....7
Глава 2 (Практическая часть).....15
Заключение……19
Список источников…….……...……27

Введение

Актуальность:

Автоматизация бизнес-процессов является неотъемлемой частью успешной деятельности любого предприятия. В настоящее время многие компании сталкиваются с необходимостью регистрации и обработки большого количества документов, связанных с управленческой деятельностью и кадровым учетом. Это может быть вызвано как ростом бизнеса, так и улучшением контроля над процессами на предприятии. В связи с этим, создание эффективной системы автоматизированной регистрации документов является важной задачей.

Цель курсового проектирования:

Закрепление теоретических знаний, а также навыков проектирования БД, полученных при изучении дисциплины «Базы данных».

Постановка задачи:

В данной курсовой работе будет рассмотрен весь жизненный цикл разработки базы данных: начиная от анализа требований и проектирования структуры базы данных, заканчивая созданием схемы базы данных и написанием запросов на языке SQL.

Окончательный результат данной работы будет представлять собой готовую базу данных, которая будет соответствовать требованиям предприятия и позволит эффективно регистрировать и обрабатывать документы, связанные с управленческой деятельностью и (или) кадровым учетом.

Техническое задание:

Общее описание проекта: на предприятии необходима система автоматизации регистрации документов, сопровождающих управленческую деятельность и/или кадровый учет. В рамках проекта необходимо разработать базу данных для хранения информации о документах, сотрудниках, отделах, должностях и т.д.

Требования к базе данных:

- База данных должна быть реляционной и хранить информацию о документах, сотрудниках, отделах, должностях и т.д.
- База данных должна обеспечивать возможность добавления, удаления и редактирования записей в таблицах.
- Должна быть предусмотрена возможность связывания записей в разных таблицах.
- Для каждой таблицы должна быть определена первичная ключевая колонка.
- База данных должна обеспечивать возможность поиска записей по заданным критериям.
- База данных должна обеспечивать защиту данных от несанкционированного доступа.

Описание функциональности:

В интерфейсе должно быть реализовано добавление новых данных в базу данных, возможность редактировать уже существующие данные. Также пользователь может удалять ненужные записи из базы данных. Должна быть реализована возможность просмотра полных таблиц, а также возможность выводить данные по определенным значениям. Интерфейс должен быть защищен от стороннего доступа к изменению целостности данных в БД.

Задачи:

- Создать Er-диаграмму по техническому заданию.
- Создать базу данных по Er-диаграмме.
- Заполнить базу данных тестовыми данными.
- Создать авторизацию.
- Разработать формы для автоматизированного заполнения данными БД.
- Тестирование созданных форм на работоспособность.

Инструменты:

- Draw SQL - сайт, на котором можно разработать ER-диаграмму.
- Sqlite - СУБД, для создание базы данных.
- Qt5 дизайнер - инструмент, позволяющий визуально разрабатывать графический интерфейс.
- visual studio code - IDE, которая необходима для написания кода.

Выбор СУБД.

Для разработки базы данных мною была выбрана СУБД SQLite.

Причины выбора данной СУБД:

1. Экономичность: SQLite бесплатен и не требует больших затрат на обслуживание и хранение данных.
2. Надежность: SQLite обеспечивает безопасность данных с помощью механизмов транзакций и снимков состояния базы данных.
3. Портативность: SQLite может быть использован как на серверах, так и на мобильных устройствах, что делает его удобным для разработки приложений, работающих на разных платформах.

Глава 1 (Анализ литературы)

Традиционные файловые системы предназначены для хранения небольших объемов данных и обеспечивают простоту в использовании. Однако, при увеличении объема данных, эти системы становятся неэффективными и не могут обеспечить необходимый уровень безопасности и устойчивости к сбоям. Именно поэтому информационные системы с базами данных (БД) стали более популярными. Базы данных обеспечивают более эффективное управление большими объемами данных и сложными приложениями, однако могут быть более сложными в использовании и затратными. Для работы с БД существуют различные системы управления базами данных (СУБД)

Архитектура СУБД определяет, как они организуют, хранят, управляют и обрабатывают данные в базе данных через три уровня: внешние схемы, концептуальная схема и физическая схема. Это позволяет определить логическую модель данных, структуру хранения данных и управлять транзакциями, обеспечивая безопасность и оптимизацию запросов. Цель архитектуры СУБД - обеспечить эффективное управление данными и защиту их конфиденциальности, целостности и доступности.

С развитием технологий базы данных, появились новые перспективы развития. Например, расширение облачных баз данных для масштабирования и уменьшения нагрузки на локальные серверы, использование искусственного интеллекта и машинного обучения для улучшения производительности.

Реляционная модель данных и объектно-ориентированная модель данных - это два разных подхода к хранению и управлению данными. В реляционной модели данные представлены в виде таблиц, а связи между данными описываются отношениями между таблицами. В объектно-ориентированной модели данных данные представлены в виде объектов, а связи между данными описываются ссылками между объектами.

Объектно-ориентированный подход к проектированию баз данных (ООП

БД) использует концепции объектно-ориентированного программирования для проектирования баз данных. Классы объектов представляют собой структуры данных, а методы классов определяют операции, которые можно выполнять с этими данными. Наследование позволяет создавать более сложные структуры данных. ООП БД более гибок и расширяем, что делает его популярным для обработки больших объемов данных.

Объектно-ориентированный подход к проектированию ПО (ООП) также использует объекты для инкапсуляции данных и функций. ООП предоставляет множество механизмов для управления данными и моделями, включая наследование и полиморфизм. Инкапсуляция скрывает детали реализации от пользователей, что делает код более безопасным и легко поддерживаемым. ООП - популярный подход для разработки ПО в различных областях, включая создание приложений для работы с данными.

Глава 2 (Практическая часть)

1. Выбор методологии проектирования

Для проектирования базы данных автоматизированной регистрации документов и кадрового учета на предприятии можно использовать методологию Entity-Relationship (ER). Эта методология позволяет описать сущности, атрибуты и связи между ними.

2. Построение инфологической (концептуальной) модели предметной области.

На основе технического задания в базе данных должны присутствовать следующие сущности: сотрудники, документы, должности, отделы, связанные документы. У которых должны присутствовать атрибуты и связи, точная схема базы данных находится ниже рис.1 .

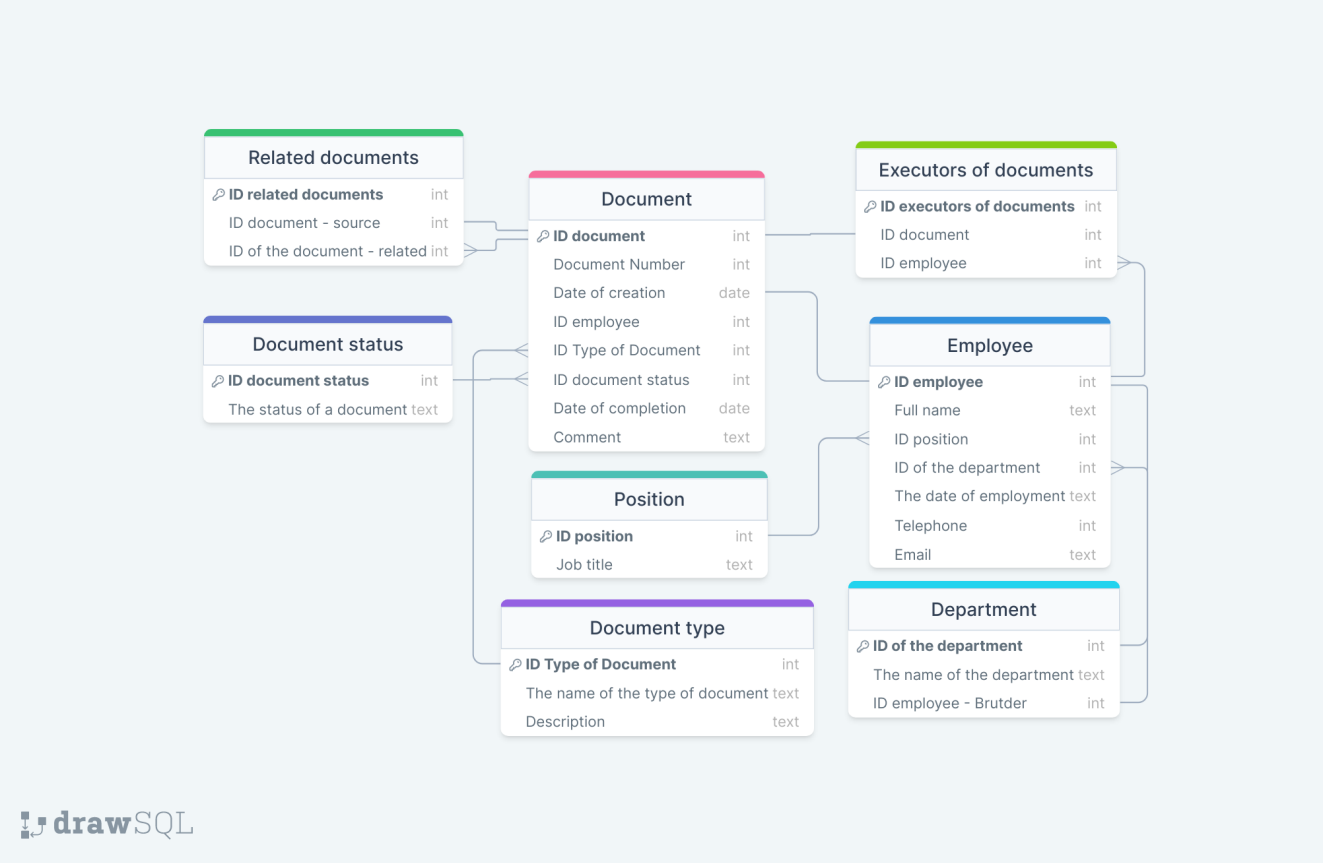


Рисунок 1. ER-диаграмма базы данных.

3. Проектирование логической структуры базы данных.

На базе er-диаграммы создаём базу данных в Sqlite. У нас есть восемь сущностей: документы, сотрудники, должности, статус документа, тип документ, связанные документы, отделы, исполнители документов.

Таблица “Документы” имеет следующие атрибуты:

IdДокумента
Номер документа
Дата создания
IdСотрудника (автор документа)
IdТип документа

IСтату документа
Дата завершения
Комментарий

Таблица “Сотрудники” имеет следующие атрибуты:
IdСотрудника
ФИО
IDДолжности
IdОтдела
Дата трудоустройства
Номер телефона
Электронная почта

Таблица “Должности” имеет следующие атрибуты:
IdДолжности
Название должности

Таблица “СтатусДокумента” имеет следующие атрибуты:
IdСтатусДокумента
Название

Таблица “ТипДокумент” имеет следующие атрибуты:
IdТипДокумент
Название
Описание

Таблица “СвязанныеДокументы” имеет следующие атрибуты:
IdСвязанныеДокументы
IdДокументы (главный документ)
IdДокументы (связанный документ)

Таблица “Отделы” имеет следующие атрибуты:
IdОтделы
Название
IdСотрудника (руководитель)

Таблица “ИсполнителиДокументов” имеет следующие атрибуты:
IdИсполнителиДокументов

IdСотрудники
IdДокументы

4. Выявление полного перечня ограничений целостности, присущего данной предметной области.

В предметной области автоматизированной регистрации документов и кадрового учета могут быть следующие ограничения целостности:

1. Уникальность номера документа
2. Обязательность заполнения поля "номер документа", "дата создания", "автор"

Для контроля уникальности номера документа можно использовать ограничение PRIMARY KEY на соответствующем поле таблицы "документы". Для контроля обязательности заполнения полей можно использовать ограничение NOT NULL на соответствующих полях.

5. Проектирование физической структуры базы данных.

Для проектирования физической структуры базы данных была выбрана СУБД SQLite. В качестве инструмента для создания и редактирования базы данных была использована программа DB Browser for SQLite.

Схема базы данных включает в себя три таблицы: "Employees", "Documents" и "Departments". Каждая таблица имеет свой уникальный идентификатор ("ID"), который используется для связывания записей в разных таблицах.

Таблица "Employees" содержит информацию о сотрудниках предприятия: их ID, ФИО, дату рождения, адрес, должность, дату приема на работу и ID отдела, в котором они работают.

Таблица "Documents" содержит информацию о документах, связанных с управленческой деятельностью и кадровым учетом. Каждый документ имеет свой уникальный ID, дату создания, тип документа, название, содержание и ID автора документа.

Таблица "Departments" содержит информацию об отделах предприятия: их ID, название и описание.

Для обеспечения целостности данных были заданы следующие ограничения целостности:

- В таблице "Employees" поле "DepartmentID" является внешним ключом, который связывает сотрудника с отделом, в котором он работает.
- В таблице "Documents" поле "AuthorID" является внешним ключом, который связывает документ с автором, который является сотрудником предприятия.

Для реализации контроля целостности были использованы встроенные средства СУБД SQLite, такие как ограничения NOT NULL, UNIQUE и FOREIGN KEY.

В качестве типов данных были выбраны наиболее подходящие для каждого поля: INTEGER для идентификаторов, TEXT для строковых значений, REAL для дат и времени.

6. Организация ввода данных в БД.

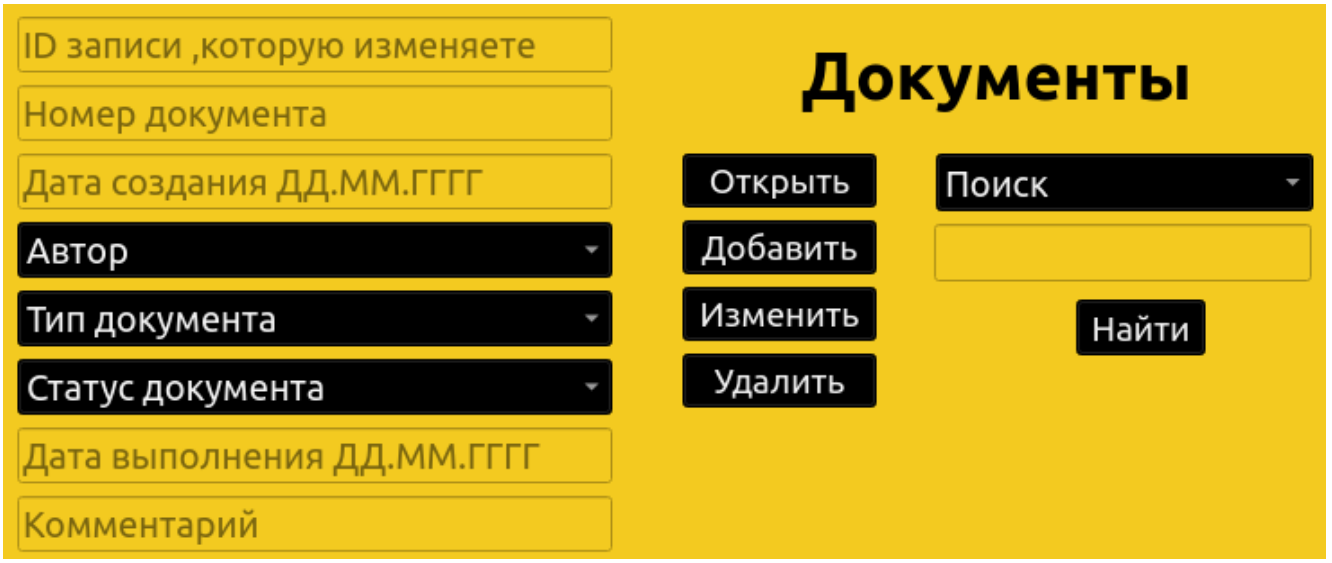
Первичный ввод записей в базу данных будет производится при помощи импортирования данных из Excel см. (рис. 2) . Далее ввод данных будет производится непосредственно с формы.

A	B	C	D	E	F	G	H
ID document	Document Number	Date of creation	ID author	ID Type of Documen	ID document status	Date of completion	A comment
1	1	15.02.2022	10	2	1	22.02.2022	Нет комментариев
2	2	22.03.2022	14	3	2	01.04.2022	Нужно проверить дополнительно
3	3	07.04.2022	8	1	3		Отсутствует подпись
4	4	10.05.2022	15	4	1	17.05.2022	Необходимо уточнение по срокам
5	5	20.06.2022	12	2	2		Ожидает подтверждение
6	6	02.07.2022	5	3	3	10.07.2022	Согласовано с руководством

Рисунок 2. Ввод данных в Excel

7. Организация корректировки БД.

Корректироваться данных производится с формы, или непосредственно в Sqlite.



8. Описание информационных потребностей пользователей и выбор способов их реализации.

У пользователя могут возникнуть следующие потребности при пользовании интерфейсом:

- Авторизация на форме
- Добавление данных в базу данных
- Изменение данных в таблице
- Удаление данных из базы данных
- Поиск информации по определенным значениям

Авторизация:

Пароли и логины для персонала выдается руководством.

При запуске интерфейса все пользователи попадают на первую страницу страницу авторизации, чтобы продолжить работу с формами все пользователи должны авторизоваться.

Если пользователь вводит в ячейку логин “0000” и в ячейку пароля “0000”, то он успешно переходит на следующие формы для редактирования записей в базе данных. Еси данные не соответствуют значением, которые указанны выше то в терминал выводится “Нет такого пользователя”.

Работа с формами и редактирование данных в БД.

После авторизации пользователь попадает на форму “Документы”, с которой пользователь может добавлять новые данные в базу данных, редактировать уже существующие записи в таблице и удалять ненужную информацию.

Перед работой с интерфейсом нужно запустить ее, для этого необходимо нажать на кнопк “Открыть”, после чего в нижнем окне должна отобразиться таблица с данными.

Чтобы добавить новую запись в базу данных необходимо заполнить все необходимые ячейки, после чего нажать на кнопку добавить, данные будут успешно добавлены в таблицу.

Для изменения уже существующих данных необходимо заполнить все поля и указать id записи из таблицы, в которой будут производится изменения.

Чтобы удалить ненужную информацию из Бд необходимо в окне, в котором отображается таблица, выбрать запись и нажать на нее, после чего кликнуть на кнопку “Удалить”.

Для вывода данных по определенным значением нужно переместится в блок поиска. Нажав на выпадающий список нужно выбрать столбец и таблицы по которому будет производиться поиск, после чего нужно ввести необходимые значения для поиска в строку ниже и нажать на кнопку “Найти”. В окне, где отображалась таблица появятся данные, которые Вы искали.

Для переключения с одной формы на другую внизу страницы предусмотрены кнопки, при нажатии на необходимую кнопку будет осуществляться переход на соответствующую страницу интерфейса.

Все фирмы работают аналогично форме - “Документы”.

9. Разработка интерфейса.

Для начала в QT 5 дизайнере необходимо создать внешний вид форм, перенести на окно необходимые кнопки, поля для ввода данных, текстовые значения см. (рис. 3).

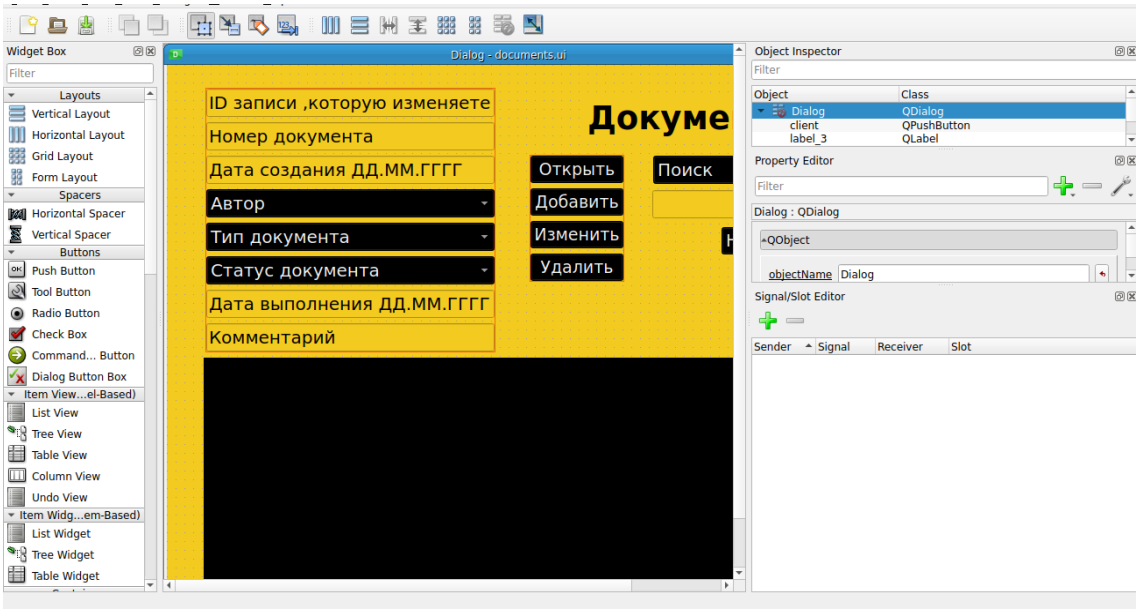


Рисунок 3. Создание формы в QT дизайнере

После создания внешнего вида формы нам необходимо добавить функции на кнопки и поля ввода, чтобы наш интерес исправно функционировал (рис. 4).

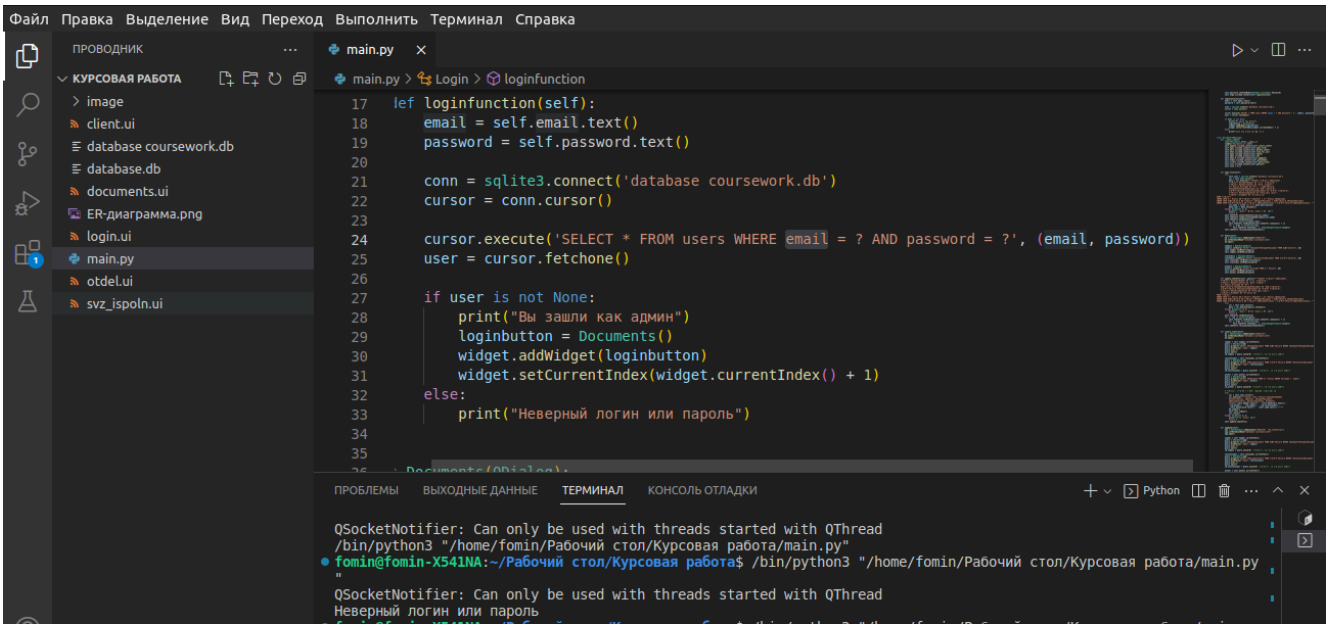


Рисунок 4. Написание кода в visual studio code

Разбор кода

Создание класса и вызов необходимых функций (рис. 5).

```
class Login(QDialog):
    def __init__(self):
        super(Login,self).__init__()
        loadUi("login.ui",self)


        self.password.setEchoMode(QtWidgets.QLineEdit.Password)
        self.logo.clicked.connect(self.loginfunction)
```

Рисунок 5. Создание класса.

Логирование пользователя, проверка введенных данных (рис. 6).

```
lef loginfunction(self):
    email = self.email.text()
    password = self.password.text()

    conn = sqlite3.connect('database coursework.db')
    cursor = conn.cursor()

    cursor.execute('SELECT * FROM users WHERE email = ? AND password = ?', (email, password))
    user = cursor.fetchone()

    if user is not None:
        print("Вы зашли как админ")
        loginbutton = Documents()
        widget.addWidget(loginbutton)
        widget.setCurrentIndex(widget.currentIndex() + 1)
    else:
        print("Неверный логин или пароль")
```

Рисунок 6. Логирование

Открытие Данных на форме, SQL запрос на отображение таблицы на форме, подключение к базе данных (рис. 7).

```
    def open_file(self):
        try:
            self.conn = sqlite3.connect('database coursework.db')
            cur = self.conn.cursor()
            data = cur.execute(f"""SELECT Документы.IDdocument ,
            Документы.DocumentNumber AS "Номер документа" ,
            Документы.Dateofcreation AS "Дата создания",
            Сотрудники.Fullname AS "Автор", ТипДокумента.
            Thenameofthetypeofdocument AS "Тип документа",
            СтатусДокумента.Thestatusofadocument AS "Статус документа",
            Документы.Dateofcompletion AS "Дата завершения",
            Документы.Acomment AS "Комментарий"
FROM Документы
INNER JOIN Сотрудники ON Документы.IDauthor = Сотрудники.IDemployee
INNER JOIN ТипДокумента ON Документы.IDTypeofDocument = ТипДокумента.IDTypeofDocument
INNER JOIN СтатусДокумента ON Документы.IDdocumentstatus = СтатусДокумента.IDdocumentstatus; "
            col_name = [i[0] for i in data.description]
            data_rows = data.fetchall()
        except Exception as e:
            print(f"Проблемы с подключением к БД. {e}")
            return e
        self.twStaffs.setColumnCount(len(col_name))
        self.twStaffs.setHorizontalHeaderLabels(col_name)
        self.twStaffs.setRowCount(0)
        for i, row in enumerate(data_rows):
            self.twStaffs.setRowCount(self.twStaffs.rowCount() + 1)
            for j, elem in enumerate(row):
                self.twStaffs.setItem(i, j, QTableWidgetItem(str(elem)))
        self.twStaffs.resizeColumnsToContents()
```

Рисунок 7. Запуск формы, вывод данных

Запрос на отображение в выпадающем списке (ComboBox) данных из другой таблицы (рис. 8).

```
typdocs = QSqlQueryModel()
typdocs.setQuery("SELECT Thenameofthetypeofdocument FROM ТипДокумента", db)
self.typdoc.setModel(typdocs)
self.typdoc.setModelColumn(0)
```

Рисунок 8. Вывод данных в ComboBox

Получение выбранной записи в ComboBox, которая будет вноситься в таблицу базы данных (рис. 9).

```
author = self.author.currentText()
query = QSqlQuery(db)
query.prepare("SELECT IDemployee FROM Сотрудники WHERE Fullname = :name")
query.bindValue(":name", author)
query.exec_()
query.next()
id_author = query.value(0)  # получить id выбранной записи
```

Рисунок 9. Получение id записи

Функция, которая изменяет данные в окошке с таблицей и мгновенно выводит изменения (рис. 10).

```
    def update_twStaffs(self, query=f"""SELECT Документы.IDdocument ,
    Документы.DocumentNumber AS "Номер документа" ,
    Документы.Dateofcreation AS "Дата создания",
    Сотрудники.Fullname AS "Автор",
    ТипДокумента.Thenameofthetypeofdocument AS "Тип документа",
    СтатусДокумента.Thestatusofadocument AS "Статус документа",
    Документы.Dateofcompletion AS "Дата завершения",
    Документы.Acomment AS "Комментарий"
FROM Документы
INNER JOIN Сотрудники ON Документы.IDauthor = Сотрудники.IDemployee
INNER JOIN ТипДокумента ON Документы.IDTypeofDocument = ТипДокумента.IDTypeofDocument
INNER JOIN СтатусДокумента ON Документы.IDdocumentstatus = СтатусДокумента.IDdocumentstatus;
        try:
            cur = self.conn.cursor()
            data = cur.execute(query).fetchall()
        except Exception as e:
            print(f"Проблемы с подключением к БД. {e}")
            return e
        self.twStaffs.setRowCount(0)
        for i, row in enumerate(data):
            self.twStaffs.setRowCount(self.twStaffs.rowCount() + 1)
            for j, elem in enumerate(row):
                self.twStaffs.setItem(i, j, QTableWidgetItem(str(elem)))
        self.twStaffs.resizeColumnsToContents()
```

Рисунок 10. Вывод измененных данных на форму

Добавление новых данных в таблицу, используя полученный id на (рис 9), (см. рис. 11).

```
try:
    cur = self.conn.cursor()
    cur.execute(f"""insert into Документы(DocumentNumber,
    Dateofcreation, IDauthor, IDTypeofDocument,
    IDdocumentstatus, Dateofcompletion, Acomment)
    values('{self.namber.text()}', '{self.datestart.text()}',
    '{id_author}', '{id_typdoc}', '{id_tytstatusdoc}',
    '{self.datefinish.text()}', '{self.comm.text()}')""")
    db.commit()
    self.conn.commit()
    cur.close()
except Exception as e:
    print(f"Исключение1: {e}")
    return e
self.update_twStaffs()
```

Рисунок 11. Добавление новых данных

Изменение уже существующих данных в таблице (рис. 12).

```
query.prepare("""UPDATE Документы SET DocumentNumber=:number,
Dateofcreation=:datestart, IDauthor=:id_author,
IDTypeofDocument=:id_typdoc, IDdocumentstatus=:id_tytstatusdoc,
Dateofcompletion=:datefinish, Acomment=:comm WHERE IDdocument=:id""")
query.bindValue(":number", self.namber.text())
query.bindValue(":datestart", self.datestart.text())
query.bindValue(":id_author", id_author)
query.bindValue(":id_typdoc", id_typdoc)
query.bindValue(":id_tytstatusdoc", id_tytstatusdoc)
query.bindValue(":datefinish", self.datefinish.text())
query.bindValue(":comm", self.comm.text())
query.bindValue(":id", self.id.text())
query.exec_()
app.commit()
app.close()
self.update_twStaffs()
```

Рисунок 12. Изменение данных

Удаление данных из таблицы (рис. 13).

```
def delete_staff(self):
    row = self.twStaffs.currentRow()
    num = self.twStaffs.item(row, 0).text()
    try:
        cur = self.conn.cursor()
        cur.execute(f"delete from Документы where IDdocument = {num}")
        self.conn.commit()
        cur.close()
    except Exception as e:
        print(f"Исключение: {e}")
        return e
    self.update_twStaffs()
```

Рисунок 13. Удаление данных

Поиск данных по заданным значениям (рис. 14).

```
def find_for_val(self):
    val = self.whattext.text()
    col = self.typdoc_2.itemText(self.typdoc_2.currentIndex())
    self.update_twStaffs(f"""SELECT Документы.IDdocument ,
    Документы.DocumentNumber AS "Номер документа" ,
    Документы.Dateofcreation AS "Дата создания",
    Сотрудники.Fullname AS "Автор",
    ТипДокумента.Thenameofthetypeofdocument AS "Тип документа",
    СтатусДокумента.Thestatusofadocument AS "Статус документа",
    Документы.Dateofcompletion AS "Дата завершения",
    Документы.Acomment AS "Комментарий"
FROM Документы
INNER JOIN Сотрудники ON Документы.IDauthor = Сотрудники.IDemployee
INNER JOIN ТипДокумента ON Документы.IDTypeofDocument = ТипДокумента.IDTypeofDocument
INNER JOIN СтатусДокумента ON Документы.IDdocumentstatus = СтатусДокумента.IDdocumentstatus
where {val} like {col};""")
```

Рисунок 14. Поиск данных

Вывод данных в ComboBox в разделе поиска с ограничением вывода на один раз (рис. 15).

```
test = False
def TD(self):

    if Documents.test == False:
        self.conn = sqlite3.connect('database coursework.db')
        cur = self.conn.cursor()
        data = cur.execute(f"SELECT * FROM Документы;")
        col_name = [i[0] for i in data.description]
        self.typdoc_2.addItems(col_name)
        Documents.test = True
```

Рисунок 15. Вывод данных в ComboBox поиска.

Функция для перехода между формами (рис. 16).

```
def gotoclient(self):
    client=Clients()
    widget.addWidget(client)
    widget.setCurrentIndex(widget.currentIndex()+1)
```

Рисунок 16. Переход между формами

Заключение

В ходе написания курсовой работы были закреплены теоретические знания, а также навыки проектирования БД, полученных при изучении дисциплины “Базы данных”.

Были выполнены все поставленные задачи. Создана Er-диаграмму по техническому заданию. Разработана база данных по Er-диаграмме. База данных была заполнена тестовыми данными. Была создана авторизация. Разработаны формы для автоматизированного заполнения БД. Интерфейс был успешно протестирован на работоспособность.

Список источников

Журналы и статьи:

- "Системы управления базами данных"
- "Информационные технологии"
- "Журнал Радиоэлектроники"
- "Журнал Научно-технической информации".

Книги:

- "Базы данных: проектирование, реализация, использование" К. Джеймс, Г. Хеннеси;
- "Системы управления базами данных" К. Дейт;
- "Базы данных. Концептуальное проектирование и моделирование" А. Николаев, А. Николаева;
- "Реляционные базы данных: теория и практика" М. Элмасри, Ш. Наватхе.

Образовательные ресурсы: курсы на STEPIK.