МИНИСТЕРСТВО ОБРАЗОВАНИЯ СТАВРОПОЛЬСКОГО КРАЯ

ГБПОУ «СТАВРОПОЛЬСКИЙ РЕГИОНАЛЬНЫЙ МНОГОПРОФИЛЬНЫЙ КОЛЛЕДЖ»

КАФЕДРА: Программного обеспечения и информационных технологий

ОТЧЕТ О ПРОХОЖДЕНИИ ПРОИЗВОДСТВЕННОЙ ПРАКТИКИ

по МДК 11.01. «ТЕХНОЛОГИЯ РАЗРАБОТКИ И ЗАЩИТЫ БАЗ ДАННЫХ»

Место прохождения практики: ООО «Зодчий»

Специальность 09.02.07 Информационные системы и программирование

Выполнил:

Студент 3 курса, группы П-31 \_\_\_\_\_\_\_\_\_\_\_\_/ Фролов Александр Витальевич/

(подпись)

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_.2024 г.

Итоговая оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики от СРМК \_\_\_\_\_\_\_\_\_\_\_ /Зимин Павел Васильевич / (подпись)

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.2024 г.

Ставрополь, 2024 г.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc153461430)

[1 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ «АРХИТЕКТУРНЫЕ ОБЪЕКТЫ» 4](#_Toc153461431)

[1.1 Общая характеристика организации ООО «Зодчий» 4](#_Toc153461432)

[1.2 Постановка задачи 6](#_Toc153461433)

[1.3 Описание и обоснование выбора программных средств 7](#_Toc153461434)

[1.4 Проектирование базы данных «Архитектурные объекты» 8](#_Toc153461435)

[2 РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ «АРХИТЕКТУРНЫЕ ОБЪЕКТЫ» 14](#_Toc153461436)

[2.1 Разработка базы данных «Архитектурные объекты» 14](#_Toc153461437)

[2.2 Описание программного продукта 19](#_Toc153461438)

[2.3 Информационная безопасность 23](#_Toc153461439)

[ЗАКЛЮЧЕНИЕ 24](#_Toc153461440)

[СПИСОК ИСТОЧНИКОВ И ЛИТЕРАТУРЫ 25](#_Toc153461441)

[ПРИЛОЖЕНИЯ 26](#_Toc153461442)

# ВВЕДЕНИЕ

Производственная практика является необходимым этапом формирования у обучающихся требуемых компетенций. Ее ценность заключается в приобретении новых и закреплении уже полученных знаний. В период прохождения производственной практики, обучающиеся должны закрепить теоретический материал, приобрести практические навыки.

Целью производственной практики является формирование соответствующих компетенций.

Достижению поставленной цели способствует решение следующих задач:

* изучить специфику предметной области архитектурного бюро, выявлены особенности;
* создать базу данных по заданной предметной области;
* разработать инфологическую модель базы данных;
* освоить возможности СУБД Access по созданию запросов на выборку данных;
* реализовать приложение для учета архитектурных объектов в среде программирования Visual Studio.

Практику проходил в ООО «Зодчий», архитектурно – строительное бюро, город Ставрополь, улица ​Пирогова, 18а.

Структура и объем курсового проекта. Проект состоит из введения, двух глав, заключения, списка литературы и источников, приложения написанного в среде программирования.

# 1 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ «АРХИТЕКТУРНЫЕ ОБЪЕКТЫ»

## Общая характеристика организации ООО «Зодчий»

Общество с ограниченной ответственностью «Зодчий» представляет собой бюро, берущее на себя обязательства по проведению архитектурно – строительных работ.

Согласно учредительным документам, основными видами деятельности общества является:

- проектирование и разработка проекта архитектурного объекта;

- дизайнерские работы на уже готовых объектах;

- отделочные работы в частных и коммерческих помещениях;

- сопровождение и проведение строительных работ;

Все виды деятельности Общество осуществляет в строгом соответствии с

действующим законодательством РФ и существующими нормами, и правилами.

Архитектурное бюро «Зодчий» работает для повышения благосостояния своих клиентов, предлагая им качественные архитектурно – строительные работы. Бюро ориентировано на клиентов с различным уровнем доходов и поэтому предлагает различные пакеты своих услуг, такие как простые архитектурно – строительные работы, частичное или полное сопровождение объекта. Такое решение помогает как обеспечить достойное качество независимо от уровня дохода клиента, так и позволяет доплатить клиентам за полный пакет услуг.

Основной коллектив располагается в главном офисе, в него входят архитекторы, дизайнеры, бухгалтер и технический инженер. Так же организация имеет различных подрядчиков и инженерно – строительную бригаду, которые берут на себя строительно – отделочные работы на обьекте.

Эффективный процесс проектирования, сопровождения и реализации архитектурных объектов возможен благодаря мощной логистической системе.

1.1.1 Перечень входных данных

В результате обследования предметной области определены входные данные, необходимые для решения комплекса задач: клиенты, сотрудники, информация об объектах, продажи, администраторы.

Поэтому при разработке базы данных, необходимо создать формы для ввода этой информации.

Входная информация может быть представлена следующими документами:

Таблица 1.1 – Входные данные клиентов

|  |  |  |  |
| --- | --- | --- | --- |
| Код | Фамилия | Имя | Отчество |
| 1 | Фомин | Дмитрий | Владимирович |
| 2 | Дроздов | Василий | Дмитриевич |
| 3 | Гуров | Леонид | Николаевич |
| 4 | Сеничкин | Александр | Борисович |
| 5 | Давыдов | Олег | Георгиевич |

- контактные данные и информация о клиентах бюро

Таблица 1.2 – Входные данные дизайнеры

|  |  |  |  |
| --- | --- | --- | --- |
| Код | Фамилия | Имя | Отчество |
| 1 | Новиков | Леонид | Владимирович |
| 2 | Жиганов | Илья | Витальевич |
| 3 | Косолапов | Алексей | Олегович |
| 4 | Курчатов | Владислав | Дмитриевич |
| 5 | Леонтьев | Ильдар | Александрович |

- контактные данные и информация о сотрудниках бюро

Таблица 1.3 – Входные данные объекты

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код | Название объекта | Клиент | Сотрудник | Компания | Улица |
| 1 | 2х ярусная квартира | 1 | 5 | 3 | 11.06.2024 |
| 2 | Пространство Лофт | 4 | 5 | 1 | 20.06.2024 |
| 3 | 3х комнатная квартира | 8 | 1 | 2 | 15.06.2024 |
| 4 | Дом Особняк | 2 | 2 | 2 | 21.06.2024 |
| 5 | Флэтхаус на 4х | 3 | 6 | 1 | 23.06.2024 |

- информация об архитектурных объектах в бюро

* + 1. Перечень выходных данных

Разрабатываемый программный продукт должен помочь сотрудникам бюро повысить эффективность работы, облегчая регистрацию объектов.

Программный продукт должен выдавать отчет об объектах и клиентах бюро.

Выходная информация представляется в виде отчётов:

* компании сотрудничающие с бюро;
* информация по оплате конкретного объекта
* клиенты и сотрудники бюро;

Выходную информацию представим в виде форм в среде программирования Visual Studio.

## 1.2 Постановка задачи

Разрабатываемая база данных предназначена для структурированного хранения данных и вывода информации об архитектурных объектах, клиентах и сотрудниках бюро.

Разрабатываемая база данных должна выполнять следующие задачи:

* добавление, удаление, редактирование информации о новых клиентах;
* ведение учёта о каждом клиенте бюро;
* осуществление поиска клиентов по номеру телефона, их заказу, фамилии;
* оформление отчетов по клиентам и сотрудникам;
* просмотр самого старого(ой) клиента(ки) бюро;
* мониторинг начала и конца работ на объекте;
* просмотр объектов.

## 1.3 Описание и обоснование выбора программных средств

При создании базы данных для архитектурного бюро была использована реляционная система управления базами данных Ms. Access. Данная СУБД, как и другие продукты этой категории предназначена для хранения, поиска, обработки данных и представления информации в удобном виде с автоматизацией часто повторяющихся операций. С помощью Access можно разрабатывать простые и удобные формы для ввода данных, а также проводить обработку данных и выдачу сложных отчетов.

Данное приложение хорошо сочетает в себе большую производительность и удобства, которые имеются в распоряжение пользователя Microsoft Windows. Оба продукта разработаны одной компанией, так что при работе с данным продуктом пользователю доступны все преимущества операционной системы. СУБД имеет полный набор основных функций и хорошо сочетается с остальными продуктами Microsoft, такими как Excel и Word.

В Access в полной мере реализовано управление реляционными базами данных. Система поддерживает первичные, внешние ключи и обеспечивает целостность данных на уровне ядра. В приложение присутствует полный набор стандартных типов полей, в том числе текстовый, числовой, счетчик, поле даты и времени, поле МЕМО, гиперссылка и поле объектов OLE.

Подводя итоги можно сказать что Microsoft Access это мощное, а главное удобное программное обеспечение, которое полностью удовлетворяет нашим требованиям. Функционала данного продукта с избытком хватает для реализации на его основе удобной, производительной и понятной базы данных с последующим подключением к приложению Visual Studio.

## 1.4 Проектирование базы данных «Архитектурные объекты»

Для ведения учета клиентов, объектов и сотрудников бюро необходимо иметь следующую информацию (таблицы 1.4 – 1.7):

Таблица 1.4 – Атрибуты отношения «Клиенты»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Код клиента |
| Фамилия | Фамилия клиента |
| Имя | Имя клиента |
| Отчество | Отчество клиента |

Таблица 1.5 – Атрибуты отношения «Сотрудники»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Код сотрудника |
| Фамилия | Фамилия сотрудника |
| Имя | Имя сотрудника |
| Отчество | Отчество сотрудника |

Таблица 1.6 – Атрибуты отношения «Архитектурные объекты»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Код номера объекта |
| Название объекта | Название архитектурного объекта |
| Код клиента | Код из таблицы «Клиенты» |
| Код исполнителя | Код из таблицы «Сотрудники» |
| Компания | Компания сотрудничество |

Таблица администраторы не связывается с прочими таблицами и их отношениями. Её наличие строго обусловлено нуждой в обеспечение безопасности базы данных и реализации приложения для работы с ней.

Таблица 1.7 – Атрибуты отношения «Администраторы»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Код администратора |
| Логин | Логин администратора |
| Пароль | Пароль администратора |

1.4.1 Проектирование ER-модели

В соответствии с первым этапом проектирования, в базе можно выделить следующие сущности:

* архитектурные объекты;
* оплата;
* сотрудники;
* клиенты;
* имена;
* фамилии;
* отчества;
* администраторы

Таким образом, получаем следующие сущности (таблицы 1.8 – 1.14).

Таблица 1.8 – Атрибуты отношения «Фамилии»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Фамилия | Короткий текст |

Таблица 1.9 – Атрибуты отношения «Имена»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Имя | Короткий текст |

Таблица 1.10 – Атрибуты отношения «Отчества»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Отчества | Короткий текст |

Таблица 1.11 – Атрибуты отношения «Клиенты»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Код фамилии | Числовой |
| Код имени | Числовой |
| Код отчества | Числовой |

Таблица 1.12 – Атрибуты отношения «Сотрудники»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Код фамилии | Числовой |
| Код имени | Числовой |
| Код отчества | Числовой |

Таблица 1.13 – Атрибуты отношения «Архитектурные объекты»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Название объекта | Короткий текст |
| Код клиента | Числовой |
| Код исполнителя | Числовой |
| Компания | Короткий текст |
| Улица | Короткий текст |

Таблица 1.14 – Атрибуты отношения «Администраторы»

|  |  |
| --- | --- |
| Наименование атрибута | Описание |
| Код | Счетчик |
| Логин | Короткий текст |
| Пароль | Короткий текст |
| Код сотрудника | Числовой |

ER-модель базы данных представлена на рисунке 1.

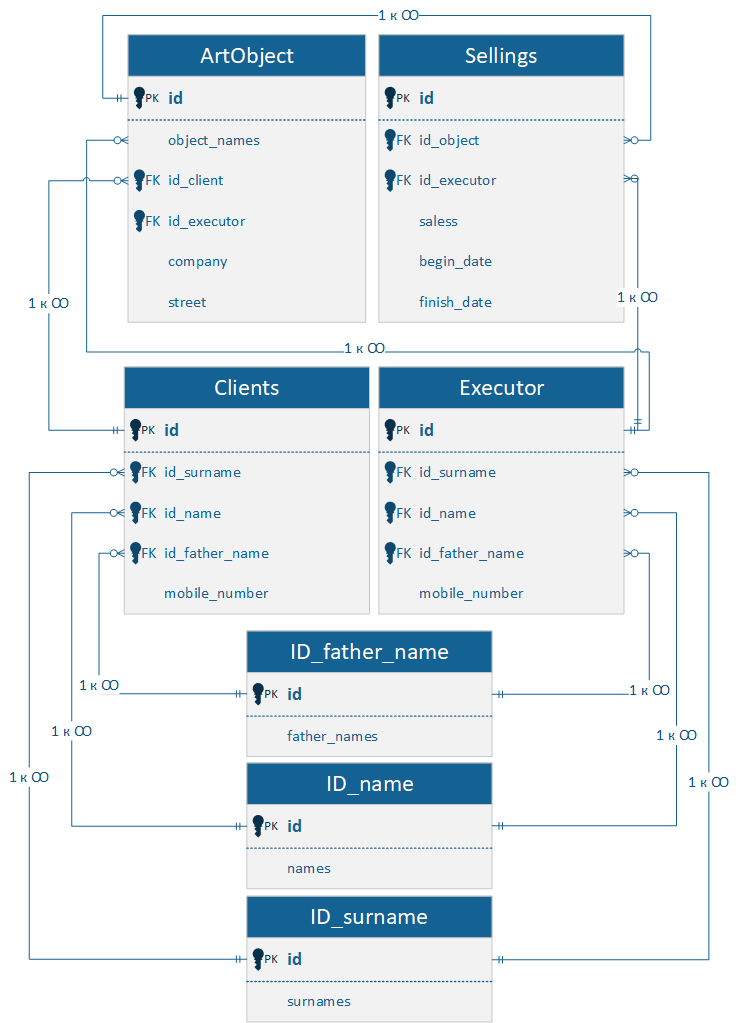


Рисунок 1 – ER-модель базы данных

Таким образом, в базе данных определены 8 взаимосвязанных таблиц, связи между таблицами – один-ко-многим, для каждой таблицы определены ключевые поля, таблицы нормализованы. Связь типа один-ко-многим означает, что один экземпляр первой сущности (левой) связан с несколькими экземплярами второй сущности (правой). Это наиболее часто используемый тип связи. Левая сущность (со стороны "один") называется родительской, правая (со стороны "много") - дочерней.

2 РЕАЛИЗАЦИЯ БАЗЫ ДАННЫХ «АРХИТЕКТУРНЫЕ ОБЪЕКТЫ»

2.1 Разработка базы данных «Архитектурные объекты»

2.1.1 Создание таблиц

Таблица «ArtObject» имеет ключевое поле «id», тип данных в котором – «Счетчик», который автоматически увеличивается на единицу при каждой новой записи. Для полей «id\_client», «id\_executor» установлена подстановка из таблицы «Clients», «Executor». Для реализации этого для таблицы подстановки построены запросы, структура таблицы показана на рисунке 2.

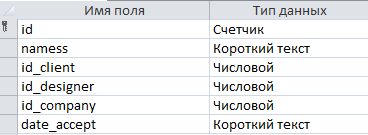


Рисунок 2 – Структура таблицы «ArtObject»

В результате данные таблицы при заполнении выглядят следующим образом (Рисунок 3).

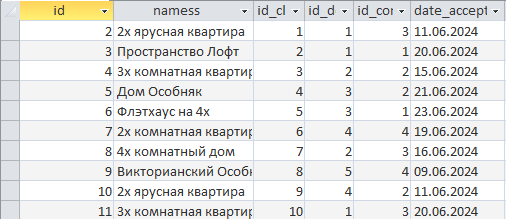


Рисунок 3 – Заполненная таблица «ArtObject»

Таблица «Clients» имеет ключевое поле «id», тип данных в котором – «Счетчик», который автоматически увеличивается на единицу при каждой новой записи. Для полей «id\_surname», «id\_name», «id\_father\_name» установлена подстановка из таблиц «Surnames», «Namess», «Father\_names». Для реализации этого для таблицы подстановки построены запросы, структура таблицы показана на рисунке 4.

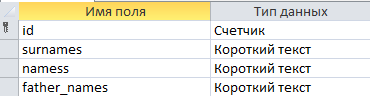


Рисунок 4 – Структура таблицы «Clients»

В результате данные таблицы при заполнении выглядят следующим образом (Рисунок 5).

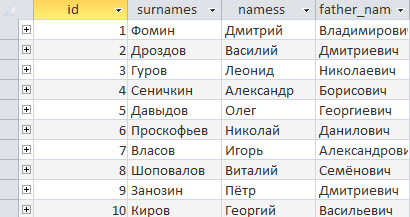


Рисунок 5 – Заполненная таблица «Clients»

Таблица «Executer» имеет ключевое поле «id», тип данных в котором – «Счетчик», который автоматически увеличивается на единицу при каждой новой записи. Для полей «id\_surname», «id\_name», «id\_father\_name» установлена подстановка из таблиц «Surnames», «Namess», «Father\_names». Для реализации этого для таблицы подстановки построены запросы, структура таблицы показана на рисунке 6.

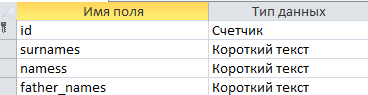


Рисунок 6 – Структура таблицы «Executer»

В результате данные таблицы при заполнении выглядят следующим образом (Рисунок 7).

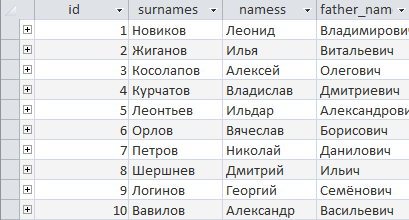


Рисунок 7 – Заполненная таблица «Executer»

2.1.2 Реализация запросов

В базе данных созданы следующие запросы:

1. Запрос на выборку информации о клиентах. Запрос имеет выборку из смежных таблиц, таких параметров как «Имя», «Фамилия» и «Отчество». В запросах участвуют таблицы «Clients», «Namess», «Surnames», «Father\_names»(Рисунок 10).

SELECT [Clients].[id], [Clients].[mobile\_number], [Surnames].[surnames], [Namess].[names], [Father\_names].[father\_names]

FROM Clients

INNER JOIN [Surnames] ON [Surnames].[id] = [Clients].[id\_surname]

INNER JOIN [Namess] ON [Namess].[id] = [Clients].[id\_name]

INNER JOIN [Father\_names] ON [Father\_names].[id] = [Clients].[id\_ father\_name]

ORDER BY [Clients].[id\_surname]

Рисунок 10 – Структура запроса информации о клиентах

1. Запрос на выборку информации о сотрудниках. Запрос имеет выборку из смежных таблиц, таких параметров как «Имя», «Фамилия» и «Отчество». В запросах участвуют таблицы «Executor», «Namess», «Surnames», «Father\_names»(Рисунок 11).

SELECT [Executor].[id], [Executor].[mobile\_number], [Surnames].[surnames], [Namess].[names], [Father\_names].[father\_names]

FROM Clients

INNER JOIN [Surnames] ON [Surnames].[id] = [Executor].[id\_surname]

INNER JOIN [Namess] ON [Namess].[id] = [Executor].[id\_name]

INNER JOIN [Father\_names] ON [Father\_names].[id] = [Executor].[id\_ father\_name]

ORDER BY [Clients].[id\_surname]

Рисунок 11 – Структура запроса информации о сотрудниках

1. Запрос на выборку информации об оплате. Запрос имеет такой

параметр как код объекта, который произвольно вводится и по которому выполняется выборка. В запросе участвует таблица «Sellings» (Рисунок 12).

SELECT [Sellings].[id], [Sellings].[id\_object], [Operation].[ id\_executor], [Operation].[saless], [Operation].[begin\_date], [Operation].[finish\_date]

FROM Sellings

WHERE ((([Sellings].[id\_object])=[Выбранный код]));

Рисунок 12 – Структура запроса на оплату по объекту

1. Запрос на добавление нового клиента, имеет такие параметры как

INSERT INTO «Clients» [(id\_surname, id\_name, id\_father\_name, mobile\_number)]

VALUES ([Введите данные], [Введите данные], [Введите данные], [Введите данные]);

Фамилия, Имя, Отчество и Номер телефона(Рисунок 13).

Рисунок 13 – Структура запроса на добавление клиента

1. Запрос на добавление нового сотрудника, имеет такие параметры как

INSERT INTO «Executer» [(id\_surname, id\_name, id\_father\_name, mobile\_number)]

VALUES ([Введите данные], [Введите данные], [Введите данные], [Введите данные]);

Фамилия, Имя, Отчество и Номер телефона(Рисунок 14).

Рисунок 14 – Структура запроса на добавление сотрудника

1. Запрос на добавления нового зарегистрированного администратора.

Запрос имеет параметры логин «login» и пароль «password». Оба параметра вводятся в поля вручную при регистрации нового администратора, что в дальнейшем помогает обеспечить вход в приложения по введенным данным (Рисунок 15).

INSERT INTO «Admin» [(login, password)]

VALUES ([Введите данные], [ Введите данные]);

Рисунок 15 – Структура запроса на добавление администратора

## 2.2 Описание программного продукта

2.2.1 Описание форм

Приложения базы данных архитектурного бюро имеет главную «стартовую» форму, которая открывается в первую очередь при запуске программы. На главной форме находится кнопка «Вход в базу» для дальнейшей работы приложения (Рисунок 16).

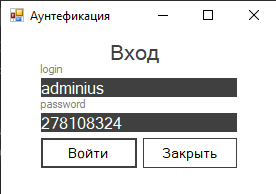


Рисунок 16 – Главная форма базы данных

При нажатии на кнопку Вход в базу, осуществляется вход в базу самого хостела, путём введения пароля, вводите пароль - входите в базу.

При правильном вводе пароля, пользователь переходит на форму аналогичную главной, но компонент «Меню» имеет другую структуру, в данном «Меню» управления осуществляется посредством кнопок с переходом на другие формы, содержащие таблицы, поиск данных, отчеты и кнопка (Рисунок 17).



Рисунок 17 – Форма «Меню»

Таблицы и запросы имеют компоненты для визуального отображения таблиц из базы данных для добавления, удаления, редактирования записей в таблице и перехода по записям таблицы, кнопка с надписью: «Выход» (для закрытия меню), нажимая на эту кнопку мы закрываем форму.

Чтобы добавить или удалить клиента можно нажать на соответствующую кнопку в блоке Добавить что вызывает форму для добавления или удаления клиента (Рисунок 18 - 19)

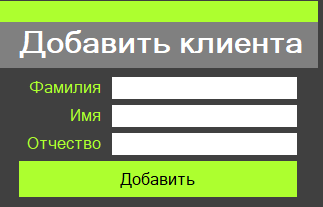


Рисунок 18 – Форма «Добавить клиента»

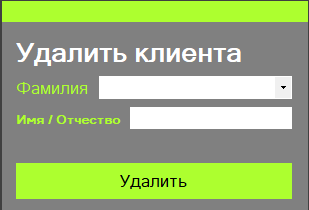


Рисунок 19 – Форма «Удалить клиента»

Для добавления или удаления архитектурного объекта применяется такой же метод, как и для клиента, выбрать соответствующее действие в блоке Администратор (Рисунок 20 - 21)

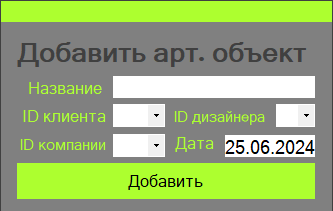


Рисунок 20 – Форма «Добавить объект»

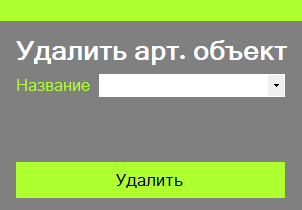


Рисунок 21 – Форма «Удалить объект»

Запросы по выборке из таблиц выполняются прямо на форме меню, а получаемые в результате запросов данные выводятся в соответствующие поля (Рисунок 20 - 22)

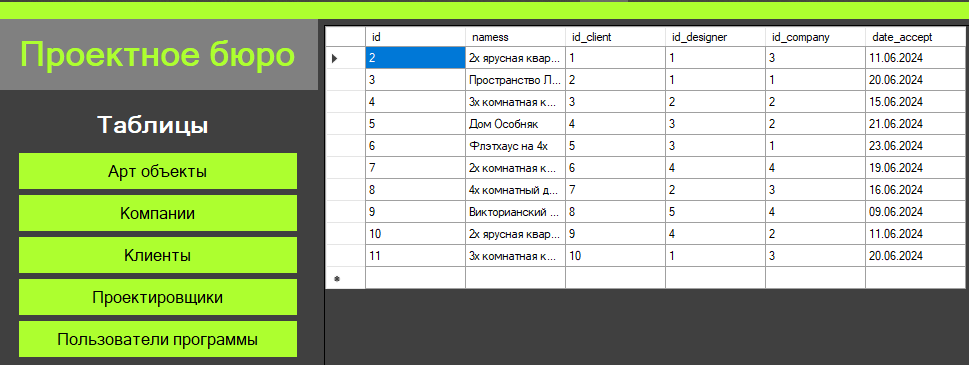


Рисунок 20 – Итог работы команды «Таблица объекты»

## 2.3 Информационная безопасность

Основные методы защиты базы данных:

Аутентификация и авторизация реализованы в базе данных посредством приложения с визуальным отображением формы для входа в меню управления базой данных и регистрацией новой учетной записи. Что не позволяет управлять базой данных не авторизированным пользователям.

Защита от SQL-инъекций, не допускающая введение слишком большого значения параметра и передачу посредством этого каких-либо SQL команд в базу данных.

Обеспечение целостности данных. Контроль целостности данных в базе реализован посредством наличия у каждой таблицы «Ключевого поля» и зависимостей от других таблиц. Что не позволяет одновременно находится двум одинаковым записям в таблицы. У каждой записи есть свой уникальный код, по которому её можно легко отследить.

Проверка изменения и достоверности данных. Изначально реализована с помощью программных средств Microsoft Access. Защищает базу данных от некорректно введенных данных, таких как слишком длинное поле или неправильный тип введенных данных.

# ЗАКЛЮЧЕНИЕ

Целью настоящего курсового проекта являлось повышение эффективности деятельности архитектурного бюро за счет создания единой базы данных объектов, автоматизированного формирования отчетов и отслеживания каждого клиента.

Во время выполнения курсовой работы, были проанализированы все задачи, а именно:

* изучена специфика предметной области учета архитектурных объектов, выявлены особенности;
* создана база данных по заданной предметной области;
* разработана инфологическая модель базы данных;
* освоены возможности СУБД Access по созданию запросов на выборку данных;
* реализовано приложение для учета архитектурных объектов в среде программирования Visual Studio.

Был получен положительный опыт по разработке базы данных в сфере архитектурно – строительного бизнеса. Разработана база данных для учета клиентов в Microsoft Access, выполнено приложение в среде программирования Visual Studio 17.7.6. В итоге полученное приложение работает и является основной системой учета архитектурных объектов бюро.

СПИСОК ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Агальцов, В. П. Базы данных (+ CD-ROM) / В.П. Агальцов. - М.: Мир, 2022. - 376 c.

2. Агальцов, В. П. Базы данных. В 2 книгах. Книга 1. Локальные базы данных / В.П. Агальцов. - М.: Инфра-М, Форум, 2017. - 352 c.

3. Арсеньев, Б. П. Интеграция распределенных баз данных / Б.П. Арсеньев, С. А Яковлев. - Москва: Гостехиздат, 2019. - 464 c.

4. Голицына, О. Л. Базы данных / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, Инфра-М, 2020. - 400 c.

5. Голицына, О. Л. Базы данных. Учебное пособие / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Дрофа, 2019. - 400 c.

6. Илюшечкин, В. М. Основы использования и проектирования баз данных / В.М. Илюшечкин. - М.: Юрайт, Юрайт, 2020. - 224 c.

7. Козырев, Ю.Г. Промышленные роботы. Основные типы и технические характеристики. Учебное пособие / Ю.Г. Козырев. - Москва: ИЛ, 2021. - 346 c.

8. Корсункий, В. А. Выбор критериев и классификация мобильных робототехнических систем на колесном и гусеничном ходу. Учебное пособие / В.А. Корсункий, К.Ю. Машков, В.Н. Наумов. - М.: МГТУ им. Н. Э. Баумана, 2022. - 46 c.

9. Кузин, А. В. Базы данных / А.В. Кузин, С.В. Левонисова. - М.: Академия, 2019. - 320 c.

10. Кузин, А.В. Базы данных / А.В. Кузин. - М.: Академия (Academia), 2017. - 545 c.

ПРИЛОЖЕНИЯ

Листинг программы

using Aspose.Cells;

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 static System.Net.Mime.MediaTypeNames;

namespace Loggin

{

public partial class Desk : Form

{

Call\_DB Call\_DB = new Call\_DB();

public string query;

public Desk()

{

InitializeComponent();

}

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

{

dataGridView1.AllowUserToAddRows = false;

dataGridView1.AllowUserToDeleteRows = false;

dataGridView1.ReadOnly = true;

dataGridView2.AllowUserToAddRows = false;

dataGridView2.AllowUserToDeleteRows = false;

dataGridView2.ReadOnly = true;

}

private void button1\_Click(object sender, EventArgs e)

{

}

//ВЫХОД с формы

private void button13\_Click(object sender, EventArgs e)

{

this.Close();

}

//ДОБАВЛЕНИЕ записей

private void button3\_Click(object sender, EventArgs e)

{

Insert\_P ins = new Insert\_P();

ins.ShowDialog();

}

private void button10\_Click(object sender, EventArgs e)

{

Insert\_A ins = new Insert\_A();

ins.ShowDialog();

}

//ЗАНЯТОСТЬ комнат

private void button8\_Click(object sender, EventArgs e)

{

dataGridView2.DataSource = "";

dataGridView2.Columns.Clear();

Call\_DB.Open();

//Выбираем комнаты

query = "SELECT " +

"Room.id AS Код, " +

"Room.capacity AS Вместимость " +

"FROM Room ";

DataSet room = Call\_DB.Request(query);

dataGridView2.DataSource = room.Tables[0].DefaultView;

//Выбираем их вместимость

query = "SELECT " +

" Room.capacity " +

" FROM Room";

DataSet cap = Call\_DB.Request(query);

//Создаём поля

var free = new DataGridViewColumn();

free.HeaderText = "Свободно";

free.Name = "name";

free.CellTemplate = new DataGridViewTextBoxCell();

dataGridView2.Columns.Add(free);

int a = 1;

//Перебор каждой комнаты

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

{

//Выбираем посетителей по нужной комнате

query = $"SELECT \* FROM Visitor WHERE id\_room = {a} ";

DataSet visitor = Call\_DB.Request(query);

int indexx = Convert.ToInt32(visitor.Tables[0].Rows.Count);

int capacity = Convert.ToInt32(cap.Tables[0].Rows[i][0]);

int result = capacity - indexx;

dataGridView2.Rows[i].Cells[2].Value = result;

if (a < 6)

{

if (result > 4) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.LightGreen; }

if (result < 4 && result != 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.Gold; }

if (result == 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.IndianRed; }

}

if (a > 5 && a < 9)

{

if (result > 3) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.LightGreen; }

if (result < 4 && result != 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.Gold; }

if (result == 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.IndianRed; }

}

if (a > 8)

{

if (result > 2) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.LightGreen; }

if (result < 3 && result != 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.Gold; }

if (result == 0) { dataGridView2.Rows[i].Cells[2].Style.BackColor = Color.IndianRed; }

}

}

Call\_DB.Close();

}

//ПОИСК ПО ДАТЕ

public string times;

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

times = dateTimePicker1.Value.ToShortDateString();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

dataGridView1.DataSource = "";

dataGridView1.Columns.Clear();

try

{

query = $"SELECT \* FROM Operation WHERE ent\_date = '{times}';";

Call\_DB.Open();

DataSet tables = Call\_DB.Request(query);

Call\_DB.Close();

dataGridView1.DataSource = tables.Tables[0].DefaultView;

}

catch { MessageBox.Show($"Дата {times} не найдена"); }

}

}

}