**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математической Кибернетики и Информационных Технологий

****

**Отчет по курсовой работе**

по предмету «Технологии баз данных»

на тему:

«Разработка БД для АСУ «Тренажёрный зал»»

Выполнил:

студент группы БВТ1802

Руководитель:

Вовик А. Г.

Москва 2021

Содержание

[**Введение** 3](#_Toc72774476)

[**Глава 1. Анализ предметной области** 3](#_Toc72774477)

[**1. Анализ проблемы учета продаж абонементов** 3](#_Toc72774478)

[**1.1. Проблемы учета продаж спортивных абонементов** 4](#_Toc72774479)

[**1.2. Формирование цели и задач проекта, определение стратегии достижения поставленной цели** 4](#_Toc72774480)

[**Глава 2. Проектирование базы данных «Тренажёрный зал»** 5](#_Toc72774481)

[**2. Разработка концепции автоматизации учета продаж спортивных абонементов в тренажёрном зале** 5](#_Toc72774482)

[**2.1. Разработка структуры данных** 6](#_Toc72774483)

[**2.2. Структура таблиц** 7](#_Toc72774484)

[**Глава 3. Программная реализация** 10](#_Toc72774485)

[**3.1. Разработка контрольного примера** 10](#_Toc72774486)

[**3.2. Разработка алгоритмов и текстов программ** 13](#_Toc72774487)

[**3.3 Разработка оконных форм взаимодействия с конечным пользователем** 22](#_Toc72774488)

[**4. Выводы и результаты** 24](#_Toc72774489)

[**Список литературы** 26](#_Toc72774490)

[**Приложение 1 Описание объекта автоматизации, договор покупки спортивного абонемента** 29](#_Toc72774491)

# **Введение**

В последнее время из-за популяризации здорового образа жизни увеличился спрос на оздоровительные центры. Люди предпочитают больше заниматься спортом, чем проводить время впустую. Это всё очень полезно для поддержания здоровья и сохранения иммунитета человека. Поэтому люди всё чаще стали посещать тренажёрные залы.

Сотрудникам трудно выполнять бумажную работу с множеством документов. Необходимо улучшить и ускорить работу в компаниях, занимающихся продажей абонементов посещений в тренажёрный зал.

Современному человеку трудно представить работу компаний без компьютера, управление экономикой на всех ее уровнях, научные исследования. Целью данной проектной работы будет автоматизация системы управления и создания базы данных для тренажёрного зала.

В итоге данной научно-исследовательской работы была спроектирована база данных для автоматизированной системы управления продажами абонементов в тренажёрный зал. Построена логическая и физическая структура данных и база данных. Внедрена автоматизированная система по учету продаж абонементов, что существенно ускорило работу по продажам абонементов в меньшие сроки.

# **Глава 1. Анализ предметной области**

## **1. Анализ проблемы учета продаж абонементов**

Тренажёрные залы должны постоянно контролировать все процессы в оздоровительном центре и эффективно обновлять списки своих клиентов, сотрудников и транзакций.

Классический способ мониторинга с помощью бумажных форм и отчётов неэффективен и небезопасен для тренажёрного зала. Могут возникать различные проблемы, из-за которых тренажёрный зал может нести убытки.

Было принято решение разработать АСУ БД для тренажёрного зала, чтобы эффективно управлять финансовыми, материальными и информационными потоками.

Продажа абонементов достаточно сложный процесс, с которым связано много документов и лиц, разработка АСУ БД для данной отрасли необходима. Разработанная система позволит фирме снизить свои риски и нарастить объём продаж абонементов, многие процессы станут автоматизированы, что позитивно отразится на работе тренажёрного зала.

## **1.1. Проблемы учета продаж спортивных абонементов**

Анализ учета продаж спортивных абонементов выявил следующие проблемы:

– отсутствуют требований к ведению истории о продажах. При продаже абонемента отсутствует полная история продажи абонемента, что небезопасно для тренажёрного зала. Нет контроля за финансовым потоков, что может приводить к убыткам;

– большое количество бумажных документов в процессе продажи спортивного абонемента. В процессе работы возникает большое количество бумажных документов, что усложняет поиск нужных бумажных документов и увеличивает время продажи спортивного абонемента. Такое нельзя допускать, тренажёрный зал должен быть готов всегда предоставить обслуживание клиенту, иначе он рискует уменьшить свою клиентскую базу, что нежелательно;

– сложность контроля за ходом выполнения продаж абонементов сотрудниками. Руководители должны следить за ходом выполнения работ сотрудниками. Управленцам должна быть доступна информация о всех финансовых, материальных и информационных потоках, иначе качество предоставляемых тренажёрным залом услуг может снизиться, что клиенты не оценят. Это приводит к финансовым убыткам;

– ручная работа. Много необоснованных трудозатрат с бумажными документами. Сотрудники тратят много времени на необоснованную работу, которую может проделывать АСУ. Если мы избавим сотрудников от ненужной работы, мы получим прирост в скорости работы, что приведёт к повышению дохода.

## **1.2. Формирование цели и задач проекта, определение стратегии достижения поставленной цели**

Целью проекта является разработка БД для автовокзала для автоматизации продаж спортивных абонементов, что позволит тренажёрному залу увеличить количество продаж абонементов и прибыль.

Для достижения поставленной цели определим следующие задачи проекта:

– провести системный анализ предметной области тренажёрного зала, изучить полностью процесс работы данного тренажёрного зала и разработать соответствующую концепцию будущей АСУ;

– разработать логическую и физическую модель БД, модель, которая призвана облегчить работу по созданию БД, что позволит уменьшить время разработки АСУ БД;

– осуществить реализацию БД на выбранной СУБД, реализация будет осуществляться с помощью СУБД Microsoft Server SQL и языка программирования C#, а также будет представлен код необходимых объектов на языке SQL, что поможет портировать АСУ на другие СУБД.

Разрабатываемая автоматизированная система управления продажами спортивных абонементов для тренажёрного зала является актуальной в связи с высокой потребностью граждан в спорте.

# **Глава 2. Проектирование базы данных «Тренажёрный зал»**

## **2. Разработка концепции автоматизации учета продаж спортивных абонементов в тренажёрном зале**

Разработаем следующую концепцию АСУ тренажёрного зала.

Опираясь на проблемы учёта продаж спортивных абонементов и на конкретный анализ процесса работы данной области, представим в АСУ тренажёрного зала основные таблицы:

– продажи, хранит информацию о клиентах, группах, сотрудниках и доп. услугах.

– клиенты, хранит информацию о клиентах.

– должности клиентов, хранит информацию о должностях клиентов.

– тренера, хранит информацию о тренерах, которые работают в тренажёрном зале.

– должности тренеров, хранит информацию о должностях тренеров.

– сотрудники, хранит информацию о сотрудниках, которые работают в тренажёрном зале.

– должности сотрудников, хранит информацию о должностях сотрудников.

– занятия, хранит информацию о занятиях, которые проводятся в тренажёрном зале.

– типы занятий, хранит информацию о типах занятий.

– группы, хранит информацию о группах, которые сформированы в тренажёрном зале.

– дополнительные услуги, хранит информацию о дополнительных услугах.

– сумма, хранит информацию о всех поступлениях денежных средств от клиентов.

## **2.1. Разработка структуры данных**

Разработаем структуру хранения информации. На основании предложенной структуры будет проектироваться база данных. С этой целью выделим объекты и связи между ними. Анализ показывает, что в разрабатываемой системе выделяются объекты, описанные в концепции БД. Отобразим на Рисунке 1 объекты.

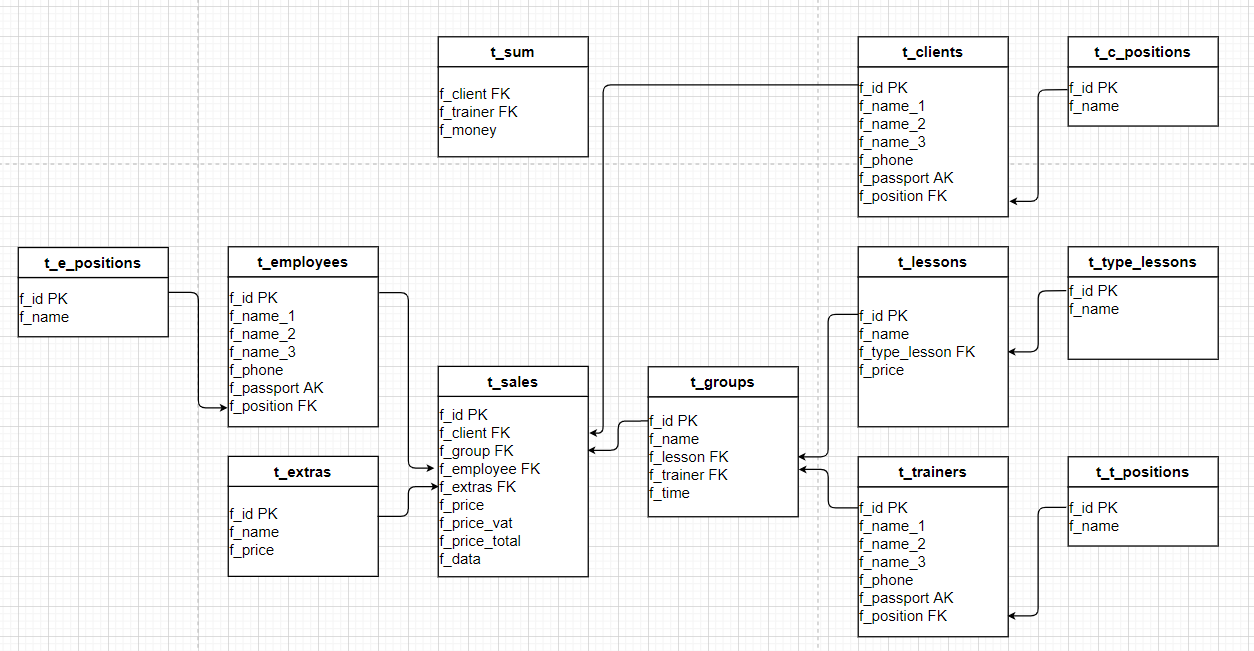


Рисунок 1 – Структура БД

На данном рисунке показаны основные объекты, без которых невозможны продажи спортивных абонементов.

В данной главе была разработана логическая модель данных. В следующей главе будет рассмотрено, на основании логической модели данных, разработка физической модели.

На данном рисунке показаны таблицы и взаимосвязи (один ко многим) между ними. Например, таблица «Продажи», т.е. договор продажи может включать много значений таблицы «Клиенты», т.е. все посетители данного тренажёрного зала. Это означает, что акты продаж могут составляться для нескольких клиентов. Аналогично этому примеру построены другие таблицы и связи.

## **2.2. Структура таблиц**

Структура таблицы включает, что каждый объект должен содержать определенный тип данных. Опишем структуру таблиц.

Сотрудники

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Фамилия | NVARCHAR(50) |  |  |
| Имя | NVARCHAR(50) |  |  |
| Отчество | NVARCHAR(50) |  |  |
| Телефон | NVARCHAR(10) |  |  |
| Паспорт | NVARCHAR(10) | Уникальный ключ | Unique key |
| Должность | INT | Внешний ключ | Foreign key |

Должности сотрудников

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Тренеры

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Фамилия | NVARCHAR(50) |  |  |
| Имя | NVARCHAR(50) |  |  |
| Отчество | NVARCHAR(50) |  |  |
| Телефон | NVARCHAR(10) |  |  |
| Паспорт | NVARCHAR(10) | Уникальный ключ | Unique key |
| Должность | INT | Внешний ключ | Foreign key |

Должности тренеров

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Клиенты

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Фамилия | NVARCHAR(50) |  |  |
| Имя | NVARCHAR(50) |  |  |
| Отчество | NVARCHAR(50) |  |  |
| Телефон | NVARCHAR(10) |  |  |
| Паспорт | NVARCHAR(10) | Уникальный ключ | Unique key |
| Должность | INT | Внешний ключ | Foreign key |

Должности клиентов

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Занятия

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) |  |  |
| Тип занятия | INT |  |  |
| Цена | MONEY |  |  |

Типы занятий

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Группы

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) |  |  |
| Занятие | INT | Внешний ключ | Foreign key |
| Тренер | INT | Внешний ключ | Foreign key |
| Время | INT |  |  |

Дополнительные услуги

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Название | NVARCHAR(50) | Уникальный ключ | Unique key |

Сумма

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Клиент | INT | Внешний ключ | Foreign key |
| Сотрудник | INT | Внешний ключ | Foreign key |
| Количество | MONEY |  |  |

Продажи

|  |  |  |  |
| --- | --- | --- | --- |
| Название поля | Тип данных | Пояснение | Ключи |
| Ключ | INT | Первичный ключ | Primary key |
| Клиент | INT | Внешний ключ | Foreign key |
| Группа | INT | Внешний ключ | Foreign key |
| Сотрудник | INT | Внешний ключ | Foreign key |
| Доп. услуги | INT | Внешний ключ | Foreign key |
| Цена | MONEY |  |  |
| Цена с НДС | MONEY |  |  |
| Цена сумма | MONEY |  |  |
| Дата | DATE |  |  |

Были разработаны все основные и дополнительных таблицы, которые используется в разрабатываемой АСУ БД тренажёрного зала. Далее представлен контрольный пример проекта и пользовательский интерфейс, с которым взаимодействуют сотрудники.

# **Глава 3. Программная реализация**

## **3.1. Разработка контрольного примера**

Для проверки возможности представлять информацию о продажах транспортных билетов в спроектированной структуре данных, разработаем контрольный пример.

Данный пример содержит: информацию о всех основных таблицах.

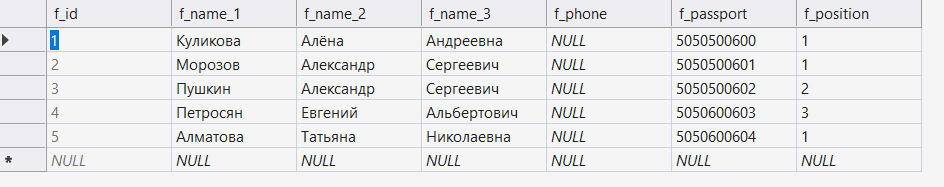


Рисунок 1 – Таблица «Сотрудники»

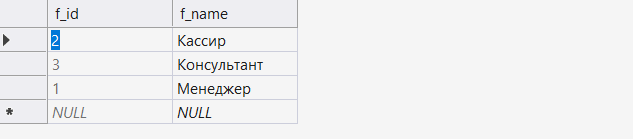


Рисунок 2 – Таблица «Должности сотрудников»

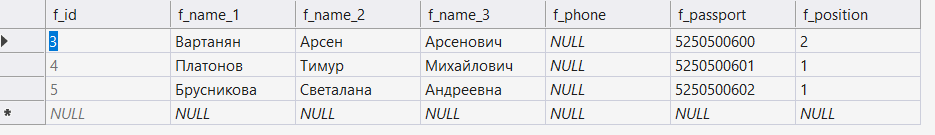


Рисунок 3 – Таблица «Тренеры»

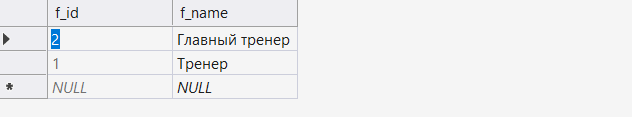


Рисунок 4 – Таблица «Должности тренеров»



Рисунок 5 – Таблица «Клиенты»

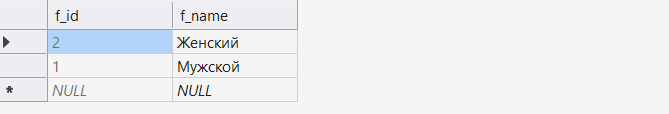


Рисунок 6 – Таблица «Должности клиентов»

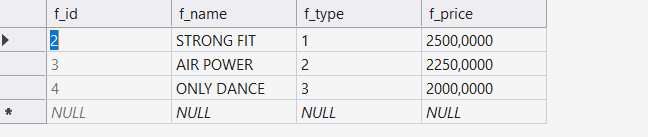


Рисунок 7 – Таблица «Занятия»

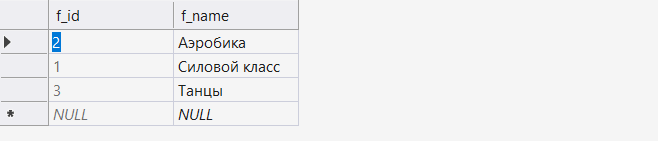


Рисунок 8 – Таблица «Типы занятий»

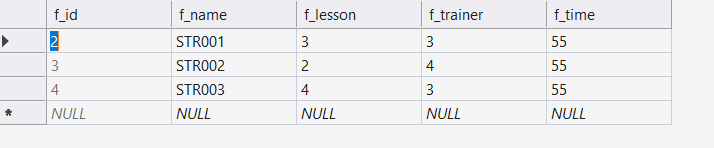


Рисунок 9 – Таблица «Группы»



Рисунок 10 – Таблица «Дополнительные услуги»

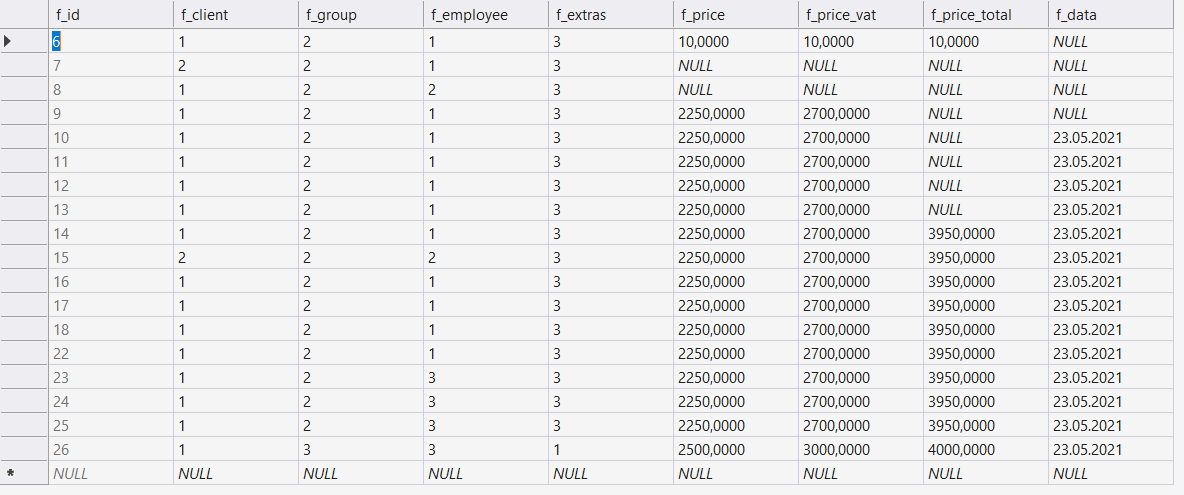


Рисунок 11 – Таблица «Продажи»



Рисунок 12 – Таблица «Сумма»

Этот контрольный пример был заполнен в программе Visual Studio 2019 c помощью Microsoft Server SQL.

## **3.2. Разработка алгоритмов и текстов программ**

Разработка базы данных должна начинаться с построения структуры таблиц и определения первичных, внешних и альтернативных ключей.

Все ключи в таблицах именуются следующим образом: Первичный ключ = Primary key, Внешний ключ = Foreign key, уникальный ключ = Unique key.

Создадим БД «Database\_GYM»

**Создание таблиц**

Создадим таблицу «Сотрудники».

CREATE TABLE [dbo].[t\_employees] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name\_1] NVARCHAR (50) NOT NULL,

[f\_name\_2] NVARCHAR (50) NOT NULL,

[f\_name\_3] NVARCHAR (50) NULL,

[f\_phone] NVARCHAR (50) NULL,

[f\_passport] NVARCHAR (10) NOT NULL,

[f\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_employees\_f\_position] FOREIGN KEY ([f\_position]) REFERENCES [dbo].[t\_e\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_employees\_passport]

ON [dbo].[t\_employees]([f\_passport] ASC);

Создадим таблицу «Должности сотрудников».

CREATE TABLE [dbo].[t\_e\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_e\_positions\_name]

ON [dbo].[t\_e\_positions]([f\_name] ASC);

Создадим таблицу «Тренеры».

CREATE TABLE [dbo].[t\_trainers] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name\_1] NVARCHAR (50) NOT NULL,

[f\_name\_2] NVARCHAR (50) NOT NULL,

[f\_name\_3] NVARCHAR (50) NULL,

[f\_phone] NVARCHAR (50) NULL,

[f\_passport] NVARCHAR (10) NOT NULL,

[f\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_trainers\_f\_position] FOREIGN KEY ([f\_position]) REFERENCES [dbo].[t\_t\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_trainers\_passport]

ON [dbo].[t\_trainers]([f\_passport] ASC);

Создадим таблицу «Должности тренеров».

CREATE TABLE [dbo].[t\_t\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_positions\_name]

ON [dbo].[t\_t\_positions]([f\_name] ASC);

Создадим таблицу «Клиенты».

CREATE TABLE [dbo].[t\_clients] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name\_1] NVARCHAR (50) NOT NULL,

[f\_name\_2] NVARCHAR (50) NOT NULL,

[f\_name\_3] NVARCHAR (50) NULL,

[f\_phone] NVARCHAR (50) NULL,

[f\_passport] NVARCHAR (10) NOT NULL,

[f\_position] INT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_clients\_f\_position] FOREIGN KEY ([f\_position]) REFERENCES [dbo].[t\_c\_positions] ([f\_id])

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_clients\_passport]

ON [dbo].[t\_clients]([f\_passport] ASC);

Создадим таблицу «Должности клиентов».

CREATE TABLE [dbo].[t\_c\_positions] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_c\_positions\_name]

ON [dbo].[t\_c\_positions]([f\_name] ASC);

Создадим таблицу «Занятия».

CREATE TABLE [dbo].[t\_lessons] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_type] INT NOT NULL,

[f\_price] MONEY NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_lessons\_f\_type] FOREIGN KEY ([f\_type]) REFERENCES [dbo].[t\_type\_lessons] ([f\_id])

);

Создадим таблицу «Типы занятий».

CREATE TABLE [dbo].[t\_type\_lessons] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_type\_lessons]

ON [dbo].[t\_type\_lessons]([f\_name] ASC);

Создадим таблицу «Группы».

CREATE TABLE [dbo].[t\_groups] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_lesson] INT NOT NULL,

[f\_trainer] INT NOT NULL,

[f\_time] INT NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_groups\_f\_lesson] FOREIGN KEY ([f\_lesson]) REFERENCES [dbo].[t\_lessons] ([f\_id]),

CONSTRAINT [c\_fk\_t\_groups\_f\_trainer] FOREIGN KEY ([f\_trainer]) REFERENCES [dbo].[t\_trainers] ([f\_id])

);

Создадим таблицу «Дополнительные услуги».

CREATE TABLE [dbo].[t\_extras] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_name] NVARCHAR (50) NOT NULL,

[f\_price] MONEY NOT NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC)

);

GO

CREATE UNIQUE NONCLUSTERED INDEX [idx\_t\_extras]

ON [dbo].[t\_extras]([f\_name] ASC);

Создадим таблицу «Продажи» и зададим три триггера: 1 – для подсчёта цены, 2 – для подсчёта общей цены, 3 – для формирования списка средств в таблице Сумма.

CREATE TABLE [dbo].[t\_sales] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_client] INT NOT NULL,

[f\_group] INT NOT NULL,

[f\_employee] INT NOT NULL,

[f\_extras] INT NULL,

[f\_price] MONEY NULL,

[f\_price\_vat] MONEY NULL,

[f\_price\_total] MONEY NULL,

[f\_data] DATE NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_sales\_f\_client] FOREIGN KEY ([f\_client]) REFERENCES [dbo].[t\_clients] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sales\_f\_group] FOREIGN KEY ([f\_group]) REFERENCES [dbo].[t\_groups] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sales\_f\_employee] FOREIGN KEY ([f\_employee]) REFERENCES [dbo].[t\_employees] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sales\_f\_extras] FOREIGN KEY ([f\_extras]) REFERENCES [dbo].[t\_extras] ([f\_id])

);

GO

CREATE TRIGGER tr\_sales

ON t\_sales

FOR INSERT

AS

DECLARE

@row\_id int = 0,

@row\_count int = 0,

@row\_client int = 0,

@row\_employee int = 0

BEGIN

SET @row\_id = (select f\_id from inserted);

SET @row\_client = (select f\_client from inserted);

SET @row\_employee = (select f\_employee from inserted);

SET @row\_count = (SELECT count(f\_id) from t\_sum where f\_client = (select f\_client from inserted) and f\_employee = (select f\_employee from inserted));

IF @row\_count = 0

BEGIN

INSERT INTO

t\_sum(f\_client, f\_employee, f\_price)

VALUES

(@row\_client, @row\_employee, (select f\_price\_total from t\_sales where f\_client = @row\_client and f\_employee = @row\_employee and f\_id = @row\_id));

END

ELSE

BEGIN

UPDATE

t\_sum

SET

t\_sum.f\_price = t\_sum.f\_price + (select f\_price\_total from t\_sales where f\_client = @row\_client and f\_employee = @row\_employee and f\_id = @row\_id)

WHERE

t\_sum.f\_client = @row\_client and t\_sum.f\_employee = @row\_employee;

END

END

GO

CREATE TRIGGER tr\_ticket\_price\_1

ON t\_sales

FOR INSERT

AS

DECLARE

@row\_group int = 0,

@row\_lesson int = 0

BEGIN

SET @row\_group = (select f\_group from inserted);

SET @row\_lesson = (select f\_lesson from t\_groups where t\_groups.f\_id = @row\_group);

UPDATE

t\_sales

SET

f\_price = (select f\_price from t\_lessons where t\_lessons.f\_id = @row\_lesson)

WHERE

t\_sales.f\_id = (select f\_id from inserted);

END

GO

CREATE TRIGGER tr\_ticket\_price\_2

ON t\_sales

FOR INSERT

AS

DECLARE

@row\_extras int = 0

BEGIN

SET @row\_extras = (select f\_extras from inserted);

UPDATE

t\_sales

SET

f\_price\_vat = f\_price / 100 \* 120

WHERE

t\_sales.f\_id = (select f\_id from inserted);

UPDATE

t\_sales

SET

f\_price\_total = f\_price\_vat + (select f\_price from t\_extras where f\_id = @row\_extras)

WHERE

t\_sales.f\_id = (select f\_id from inserted);

END

Создадим таблицу «Сумма».

CREATE TABLE [dbo].[t\_sum] (

[f\_id] INT IDENTITY (1, 1) NOT NULL,

[f\_client] INT NOT NULL,

[f\_employee] INT NOT NULL,

[f\_price] MONEY NULL,

PRIMARY KEY CLUSTERED ([f\_id] ASC),

CONSTRAINT [c\_fk\_t\_sum\_f\_client] FOREIGN KEY ([f\_client]) REFERENCES [dbo].[t\_clients] ([f\_id]),

CONSTRAINT [c\_fk\_t\_sum\_f\_employee] FOREIGN KEY ([f\_employee]) REFERENCES [dbo].[t\_employees] ([f\_id])

);

**Создание представлений**

Создадим представление «Сотрудники».

CREATE VIEW [dbo].[v\_employees]

AS SELECT

t\_employees.f\_id ID,

t\_employees.f\_name\_1 Фамилия,

t\_employees.f\_name\_2 Имя,

t\_employees.f\_name\_3 Отчество,

t\_employees.f\_phone Телефон,

t\_employees.f\_passport Паспорт,

t\_e\_positions.f\_name Должность

FROM [t\_employees],[t\_e\_positions]

WHERE t\_employees.f\_position = t\_e\_positions.f\_id;

Создадим представление «Тренеры».

CREATE VIEW [dbo].[v\_trainers]

AS SELECT

t\_trainers.f\_id ID,

t\_trainers.f\_name\_1 Фамилия,

t\_trainers.f\_name\_2 Имя,

t\_trainers.f\_name\_3 Отчество,

t\_trainers.f\_phone Телефон,

t\_trainers.f\_passport Паспорт,

t\_t\_positions.f\_name Должность

FROM [t\_trainers],[t\_t\_positions]

WHERE t\_trainers.f\_position = t\_t\_positions.f\_id;

Создадим представление «Клиенты».

CREATE VIEW [dbo].[v\_clients]

AS SELECT

t\_clients.f\_id ID,

t\_clients.f\_name\_1 Фамилия,

t\_clients.f\_name\_2 Имя,

t\_clients.f\_name\_3 Отчество,

t\_clients.f\_phone Телефон,

t\_clients.f\_passport Паспорт,

t\_c\_positions.f\_name Должность

FROM [t\_clients],[t\_c\_positions]

WHERE t\_clients.f\_position = t\_c\_positions.f\_id;

Создадим представление «Должности клиентов».

CREATE VIEW [dbo].[v\_c\_positions]

AS SELECT

t\_c\_positions.f\_id ID,

t\_c\_positions.f\_name Название

FROM [t\_c\_positions]

Создадим представление «Занятия».

CREATE VIEW [dbo].[v\_lessons]

AS SELECT

t\_lessons.f\_id ID,

t\_lessons.f\_name Название,

t\_type\_lessons.f\_name Тип,

t\_lessons.f\_price Цена

FROM [t\_lessons],[t\_type\_lessons]

WHERE t\_lessons.f\_type = t\_type\_lessons.f\_id;

Создадим представление «Группы».

CREATE VIEW [dbo].[v\_groups]

AS SELECT

t\_groups.f\_id ID,

t\_groups.f\_name Название,

t\_lessons.f\_name Тренировка,

t\_trainers.f\_name\_1 Фамилия,

t\_trainers.f\_name\_2 Имя,

t\_trainers.f\_name\_3 Отчество,

t\_groups.f\_time Время

FROM [t\_groups],[t\_lessons],[t\_trainers]

WHERE t\_groups.f\_lesson = t\_lessons.f\_id and

t\_groups.f\_trainer = t\_trainers.f\_id;

Создадим представление «Дополнительные услуги».

CREATE VIEW [dbo].[v\_extras]

AS SELECT

t\_extras.f\_id ID,

t\_extras.f\_name Название,

t\_extras.f\_price Цена

FROM [t\_extras];

Создадим представление «Продажи».

CREATE VIEW [dbo].[v\_sales]

AS SELECT

t\_sales.f\_id ID,

t\_clients.f\_name\_1 Фамилия,

t\_clients.f\_name\_2 Имя,

t\_clients.f\_name\_3 Отчество,

t\_groups.f\_name Группа,

t\_employees.f\_name\_1 Работник,

t\_extras.f\_name ДопУслуги,

t\_sales.f\_price ЦенаТренировки,

t\_sales.f\_price\_vat ЦенаТренировкиНДС,

t\_sales.f\_price\_total ЦенаВсего,

t\_sales.f\_data Дата

FROM [t\_clients],[t\_groups],[t\_employees],[t\_extras],[t\_sales]

WHERE t\_sales.f\_client = t\_clients.f\_id and

t\_sales.f\_group = t\_groups.f\_id and

t\_sales.f\_employee = t\_employees.f\_id and

t\_sales.f\_extras = t\_extras.f\_id;

Создадим представление «Сумма».

CREATE VIEW [dbo].[v\_sum]

AS SELECT

t\_sum.f\_id ID,

t\_clients.f\_name\_1 КлиентФамилия,

t\_clients.f\_name\_2 Имя,

t\_clients.f\_name\_3 Отчество,

t\_employees.f\_name\_1 СотрудникФамилия,

t\_sum.f\_price Сумма

FROM [t\_sum],[t\_clients],[t\_employees]

WHERE t\_sum.f\_client = t\_clients.f\_id and

t\_sum.f\_employee = t\_employees.f\_id;

**Основной код интерфейса приложения**

using System;

using System.Windows.Forms;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

namespace TBD\_GYM

{

public partial class Form1 : Form

{

private SqlConnection sqlConnection = null;

private SqlDataAdapter dataAdapter = null;

private DataSet dataSet = null;

private SqlCommand command = null;

private void view(string str)

{

dataAdapter = new SqlDataAdapter(str, sqlConnection);

dataSet = new DataSet();

dataAdapter.Fill(dataSet);

dataGridView1.DataSource = dataSet.Tables[0];

dataAdapter.Dispose();

dataSet.Dispose();

}

private void procedure(string str)

{

command = new SqlCommand(str, sqlConnection);

if (command.ExecuteNonQuery().ToString() == "-1")

{

MessageBox.Show("Ошибка!");

}

else

{

MessageBox.Show("Успешно!");

}

command.Dispose();

}

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

sqlConnection = new SqlConnection(ConfigurationManager.ConnectionStrings["GYMDB"].ConnectionString);

sqlConnection.Open();

if (sqlConnection.State == ConnectionState.Open)

{

MessageBox.Show("Подключение к БД установлено");

}

}

private void button4\_Click(object sender, EventArgs e)

{

view("select \* from v\_clients");

}

private void button5\_Click(object sender, EventArgs e)

{

view("select \* from v\_groups");

}

private void button6\_Click(object sender, EventArgs e)

{

view("select \* from v\_employees");

}

private void button7\_Click(object sender, EventArgs e)

{

view("select \* from v\_extras");

}

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

{

view("select \* from v\_trainers");

}

private void button9\_Click(object sender, EventArgs e)

{

view("select \* from v\_lessons");

}

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

{

view("select \* from v\_c\_positions");

}

private void button11\_Click(object sender, EventArgs e)

{

view("select \* from v\_sales");

}

private void button12\_Click(object sender, EventArgs e)

{

view("select \* from v\_sum");

}

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

{

procedure($"EXECUTE p\_sales @new\_client = N'{textBox1.Text}', @new\_group = N'{textBox2.Text}', @new\_employee = N'{textBox3.Text}', @new\_extras = N'{textBox4.Text}';");

}

private void button2\_Click(object sender, EventArgs e)

{

procedure($"EXECUTE p\_clients @new\_name\_1 = N'{textBox5.Text}', @new\_name\_2 = N'{textBox6.Text}', @new\_name\_3 = N'{textBox7.Text}', @new\_phone = N'{textBox8.Text}', @new\_passport = N'{textBox9.Text}', @new\_position = N'{textBox10.Text}';");

}

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

{

procedure($"EXECUTE p\_extras @new\_name = N'{textBox11.Text}', @new\_price = N'{textBox12.Text}';");

}

}

}

## **3.3 Разработка оконных форм взаимодействия с конечным пользователем**

Ведение учёта продаж абонементов описывается при помощи оконных форм взаимодействия с конечным пользователем.

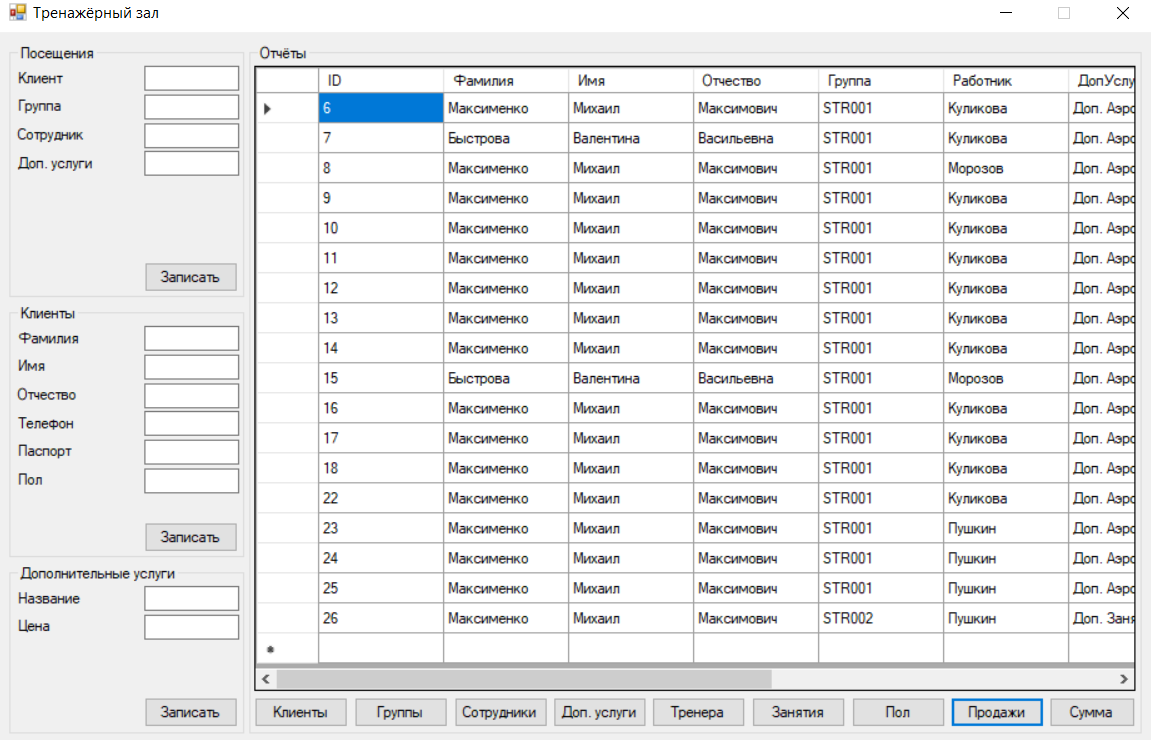


Рисунок 1 – Форма «Тренажёрный зал»

Для работы с формой или формирования отчётов, следует нажать на одну из кнопок формы тренажёрный зал.

Заполнение данных о продажах абонементов или добавлении новых клиентов или услуг происходит при помощи главной формы, представленная на Рисунке 2.

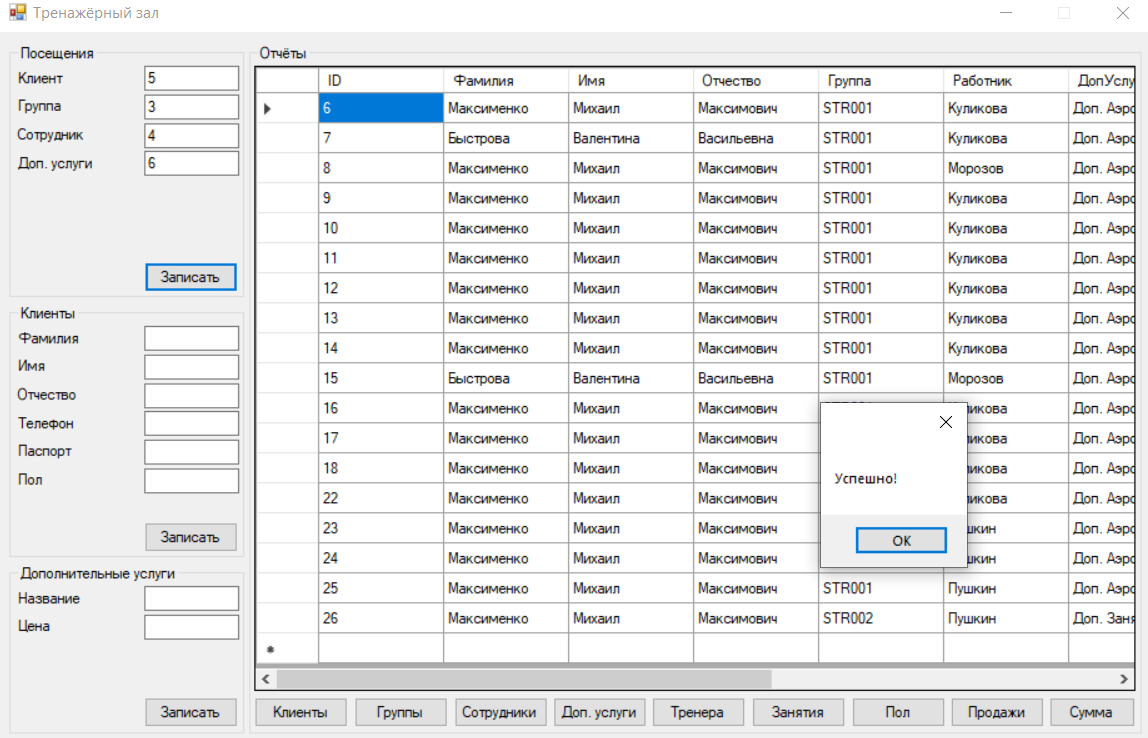
****

Рисунок 2 – Формы «Продажа абонемента или добавление клиента или услуги»

Вывод данных БД представлен на Рисунке 3.

При нажатии на одну из кнопок в отчёт будут выведены данные из выбранного представления.

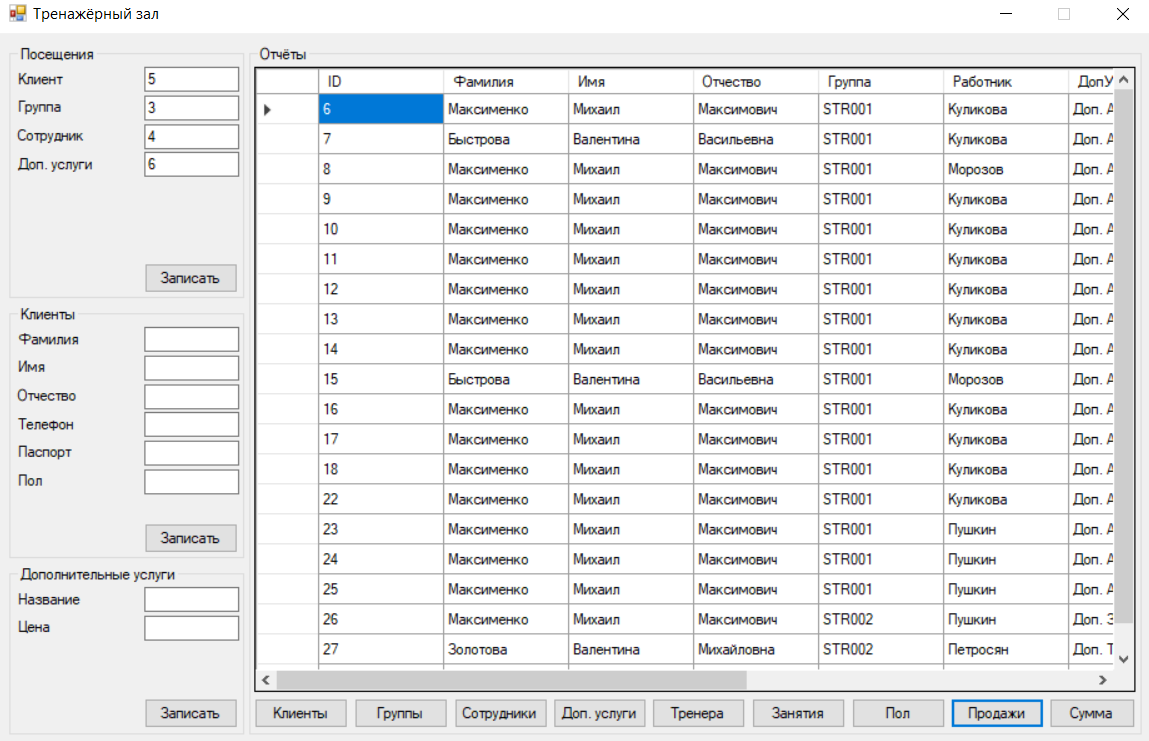
****

Рисунок 3 – Формы «Вывод информации»

Эти оконные формы помогают усовершенствовать процесс ввода, передачи и хранения информации тренажёрного зала.

# **4. Выводы и результаты**

При выполнении проектной работы по проектированию и разработке автоматизированной системы учета продаж спортивных абонементов выявлены следующие результаты:

– проанализирован порядок системы учета продаж абонементов. Выявлены слабые места в данном учете продаж абонементов. Предложен вариант наиболее рационального решения по автоматизации учета продаж абонементов;

– проанализировано и выбрано, соответствующее программное обеспечение, СУБД. Построена структура данных по автоматизации учета продаж абонементов. Спроектирована и разработана база данных по учету продаж абонементов.

Выводы:

– оконные формы лучше разрабатывать в средствах Visual Studio 2019 с использованием языка программирования C#, которые следует создать на основании учёта продаж абонементов;

– при анализе и выборе программного обеспечения следует учитывать плюсы и минусы современных антивирусных программ, операционных систем и баз данных;

– представлять программный код построения таблиц и запросов следует в виде отчёта.

– разработанная АСУ БД позволит повысить объём продаж абонементов и прибыль для данного тренажёрного зала;

– взаимодействие с ФНС целесообразно осуществлять в электронном формате.

# **Список литературы**

1. Давиденко К.Я. Технология программирования АСУ ТП. - М.: Энергоатомиздат, 1986.-184 с.: ил.

2. Олейник С.У. и др. Автоматизированные системы управления машиностроительными предприятиями: Учеб. для вузов./ - М.: Высш. шк., 1991. - 222 с.: ил. ISBN 5-06-001859-8

3. Баранюк В.А. и др. Основы создания больших АСУ. М.: Сов. радио. 1979.

4. Мамиконов А.Г. Проектирование АСУ: Учебник для специальности «АСУ» вузов. - М.: Высш. шк., 1987. -303 с.: ил.

5. Грекул М.И. Проектирование информационных систем. [www.intuit.ru](http://www.intuit.ru).

6. Черемных С.В. и др. Моделирование и анализ систем. IDEF-Технологии : практикум.,- М.: «ФиС», 2002 г.-188 с.

7. Черемных С.В. и др.Структурный анализ систем. IDEF-Технологии.- М.:«ФиС»,2002 г.-207 с.

8. Мейер Д. Теория реляционных баз данных: Пер. с англ. - М.: Мир, 1987. - 608 с., ил.

9. Мартин Дж. Организация баз данных в вычислительных системах. Издательство "Мир" 1980.

10. Кодд Е.Ф. «Реляционная модель данных для больших совместно используемых банков данных» с. 145-160 в журнале «СУБД» N1/1995. Москва. ул.Авиамоторная, 55.

11. Гилула М.М. Множественная модель данных в информационных системах. - М.: Наука, 1992. - 208с. ISBN 5-02-006938-8

12. Ковтун И.И. Кузьмин И.В. «Матрично-реляционная модель данных в автоматизированных системах управления потоками процессуальных документов и органами предварительного следствия» в материалах научно-практического семинара ВНИИ МВД «Персональный компьютер на службе криминальной милиции и следствия. Возможности и перспективы». 1997 год. Стр. 54-61.

13. Ковтун И.И. «Матрично-реляционная модель данных в автоматизированных производственных системах». В журнале «Информатика-машиностроение». N3 1997 год. Стр. 2-11. Москва. «ВиражЦентр».

14. Ковтун И.И. «Матрично-реляционный подход к проектированию автоматизированных систем управления предприятием». В журнале «Информатика-машиностроение». N4 1997 год. Стр. 5-15. Москва. «ВиражЦентр».

15. I.I.Kovtun. «A Matrix-Relational Data Model for Information Representation and Processing in Computer-Aided Monitoring and Management Systems». In journal «Programming and Computer Software», Vol. 23, No. 6, 1997, pp. 331- 341. Moskow. «Nauka». 69

16. Oracle 8. Энциклопедия пользователя.: Пер. с англ./ Компания Advannced Information Systems и др. – К.: Издательство «ДиаСофт», 1998. – 864 с.

17. Мартин Грабер. Введение в SQL. Перевод В.А. Ястребов. Издательство «ЛОРИ». 1996. (Martin Gruber. Understunding SQL. Copyright 1990. SYBEX Inc., 2021. Challenger Drive. Alameda, CA 94501).

ГОСТ 34.003-90 Основные понятия.

ГОСТ 34.201-89. Виды, комплектность и обозначение документов при создании автоматизированных систем.

ГОСТ 34.601-90. Автоматизированные системы. Стадии создания.

ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы.

ГОСТ 34.603-92. Виды испытаний автоматизированных систем.

ГОСТ 24.103-84. Основные положения.

ГОСТ 24.104-85. Общие требования к разработке АСУ.

ГОСТ 24.602-86. Состав и содержание работ по стадиям создания.

ГОСТ 19.101-77. Виды программ и программных документов.

ГОСТ 24.202-80. Требования к содержанию документа «Техникоэкономическое обоснование создания АСУ».

ГОСТ 24.203-80. Требования к содержанию общесистемных документов.

ГОСТ 24.204-80. Требования к содержанию документа «Описание постановки задачи».

ГОСТ 24.205-80. Требования к содержанию документов по информационному обеспечению.

ГОСТ 24.206-80. Требования к содержанию документов по техническому обеспечению.

ГОСТ 24.207-80. Требования к содержанию документов по программному обеспечению.

ГОСТ 24.208-80. Требования к содержанию документов стадии «Ввод в эксплуатацию».

ГОСТ 24.209-80. Требования к содержанию документов по организационному обеспечению.

ГОСТ 24.210-82. Требования к содержанию документов по функциональной части.

ГОСТ 24.211-82. Требования к содержанию документа «Описание алгоритма».

ГОСТ 24.301-80. Общие требования к выполнению текстовых документов.

ГОСТ 24.302-80. Общие требования к выполнению схем.

ГОСТ 24.304-82. Требования к выполнению чертежей.

ГОСТ 24.401-80. Внесение изменений в техническую документацию.

ГОСТ 24.402-80. Учет, хранение и обращение документации.

# **Приложение 1 Описание объекта автоматизации, договор покупки спортивного абонемента**

