Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Рязанский государственный радиотехнический университет

имени В.Ф. Уткина»

Кафедра «Электронных вычислительных машин»

К защите

Руководитель работы: ~~Тарасова В.Ю.~~

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

дата, подпись

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙ РАБОТЕ**

по дисциплине

**«Базы данных»**

на тему

«Разработка базы данных для предметной области:

«Бассейн»

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы 945  Леонов М.М. | | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  дата сдачи на проверку,  подпись |
| Руководитель работы  ассистент кафедры ЭВМ  Тарасова В.Ю. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  оценка | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  дата защиты, подпись |

Рязань 2020

**Содержание**

[Введение 3](#__RefHeading___Toc1665_3797659017)

[1 Семантическое описание предметной области 4](#__RefHeading___Toc1667_3797659017)

[2 Разработка базы данных 6](#__RefHeading___Toc1669_3797659017)

[2.1 Разработка логической модели данных 6](#__RefHeading___Toc1671_3797659017)

[2.2 Определение типов хранимых данных 6](#__RefHeading___Toc1673_3797659017)

[2.3 Задание ограничений целостности данных 8](#__RefHeading___Toc1675_3797659017)

[2.4 Построение схемы данных 11](#__RefHeading___Toc1677_3797659017)

[2.5 Разработка сценария для создания БД и основных объектов структуры 11](#__RefHeading___Toc1679_3797659017)

[2.6 Разработка сценария для заполнения таблиц БД 14](#__RefHeading___Toc6084_3797659017)

[3 Разработка основных операций реляционной алгебры 18](#__RefHeading___Toc6086_3797659017)

[4 Разработка запросов к БД для реализации информационных потребностей пользователя 20](#__RefHeading___Toc6117_3797659017)

[Список используемой литературы 43](#__RefHeading___Toc11682_3797659017)

Не хватает Заключения

# Введение

Современному человеку необходимо узнавать, изменять и обрабатывать взаимосвязанную информацию. В 21 веке количество информации выросло в разы, поэтому хранение информации доверили базам данных.

База данных (БД) — это организованная структура, предназначенная для хранения, изменения и обработки взаимосвязанной информации, преимущественно больших объемов.

Для изменения и управления данными существуют системы управления базами данных (СУБД). Это комплекс программно-языковых средств, позволяющих создать БД и управлять данными.

Целью выполнения данного курсового проекта является систематизация, закрепление и углубление теоретических знаний и практических навыков проектирования баз данных.

# 1 Семантическое описание предметной области

Требуется разработать информационную систему для учета и хранения информации о продаже абонементов в бассейн. Допускаем, что возможность использования бассейна дается только после приобретения определенного абонемента, следовательно база данных должна хранить информацию об абонементе, каждый из которых характеризуется следующими параметрами:

* уникальный код или номер;
* название;
* описание;
* стоимость;
* срок действия.

Чтобы получить данный абонемент необходимо заключить договор между клиентом и организацией, в которой находится бассейн. Следовательно, с каждым клиентом заключается договор, в который заносятся следующие сведения:

* номер договора;
* код клиента;
* код абонемента;
* дата и время покупки абонемента.

Чтобы идентифицировать клиента необходимо иметь следующую информацию о нем:

* код клиента;
* фамилия, имя, отчество;
* дата рождения;
* адрес;
* телефон.

В случае, если клиент не умеет плавать, должна иметься возможность получения индивидуальной тренировки с инструктором. На каждого человека не возможно дать личного тренера, следовательно необходимо хранить информацию о том, в какое время и с каким клиентом у тренера будет занятие. Поэтому персональная тренировка характеризуется:

* кодом клиента;
* датой и временем тренировки;
* стоимостью;
* фамилией, именем и отчеством инструктора, который проводит тренировку.

Таким образом, база данных «Бассейн» будет иметь четыре отношения, которые будут хранить необходимую информацию:

1. Абонементы (Код, Название, Описание, Стоимость, Срок действия);
2. Клиенты (Код клиента, ФИО, Дата рождения, Адрес, Телефон);
3. Договор (Номер договора, код клиента, код абонемента, Дата и время покупки);
4. Персональные тренировки (Код клиента, Дата и время тренировки, Стоимость, ФИО инструктора).

# 2 Разработка базы данных

## 2.1 Разработка логической модели данных

Абонемент на плавание выдается на основе договора, заключенного между клиентом и организацией. Договор заключается с конкретным клиентом. Персональная тренировка может пройти, если клиенту она понадобится и он договорится с тренером.

Таким образом, получим логическую модель данных, которая представлена графически на рисунке 2.1.

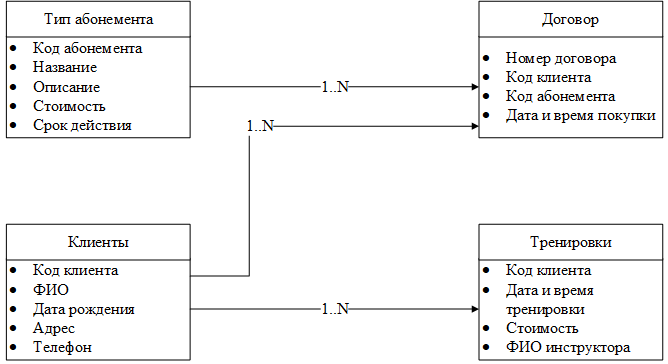


Рисунок 2.1 — Логическая модель данных

## 2.2 Определение типов хранимых данных

Чтобы определить типы хранимых данных, необходимо проанализировать, что будет представлять из себя каждый атрибут какого-либо отношения.

Рассмотрим отношение «Тип абонемента». В нем используются 5 атрибутов, такие как: «Код абонемента», «Название», «Описание», «Стоимость», «Срок действия».

«Код абонемента» будет хранить в себе некий набор цифр, поэтому будет использоваться тип данных int.

«Название абонемента» и «Описание» будут состоять из некого набора букв — слова или слов, поэтому эти атрибуты будут использовать тип данных nvarchar.

Атрибут «Стоимость» будет хранить в себе какую-то сумму денег. Для этого лучше всего подходит тип данных money. Однако, нет необходимости хранить огромные суммы, следовательно будем использовать smallmoney.

Атрибуту «Срок действия» будет хранить какую-то дату, поэтому лучше всего подойдет тип данных date.

Проанализировав отношение «Абонементы», выбрали для каждого атрибута тип данных. Представим это в виде таблицы (таблица 2.1).

Таблица 2.1 — Отношение «Абонементы» и типы данных

|  |  |
| --- | --- |
| Абонементы | |
| Код | int |
| Название | nvarchar() |
| Описание | nvarchar() |
| Стоимость | smallmoney |
| Срок действия | date |

Таким же образом, проанализировав остальные отношения, получим еще 3 таблицы (таблицы 2.2-2.4), в которых как и в таблице 2.1 запишем типы данных, сделав лишь некоторые пояснения.

Таблица 2.2 — Отношение «Клиенты» и типы данных

|  |  |
| --- | --- |
| Клиенты | |
| Код клиента | int |
| ФИО | nvarchar() |
| Дата рождения | date |
| Адрес | nvarchar() |
| Телефон | nvarchar() |

Таблица 2.3 — Отношение «Договор» и типы данных

|  |  |
| --- | --- |
| Договор | |
| Номер договора | int |
| Код клиента | int |
| Код абонемента | int |
| Дата и время покупки | datetime |

Таблица 2.4 — Отношение «Персональные тренировки» и типы данных

|  |  |
| --- | --- |
| Персональные тренировки | |
| Код клиента | int |
| Дата и время тренировки | datetime |
| Стоимость | smallmoney |
| ФИО инструктора | nvarchar() |

В таблице 2.2 в атрибуте «Телефон» тип данных указан nvarchar, хотя казалось бы, номер телефона состоит из цифр. Это сделано для того, чтобы избежать ошибок при попытке ввода номеров телефонов такого вида:

+7 (4912) 50-40-20.

## 2.3 Задание ограничений целостности данных

Чтобы задать ограничение целостности данных, необходимо рассмотреть, какие ограничения есть.

Существуют следующие типы ограничений:

1. Ограничение **NOT NULL** дает возможность запретить использование в указанном столбце NULL-значений. Такое ограничение может быть указано только как ограничение на столбец.
2. Ограничение **UNIQUE**. Ограничение разрешает использовать только уникальные значения для указанных столбцов.
3. Ограничение первичных ключей **PRIMARY KEY (PK).** Это специальный случай комбинирования ограничений UNIQUE и NOT NULL. Простой первичный ключ задается как ограничение на конкретный столбец.
4. Ограничение внешнего ключа **FOREIGN KEY(FK)**. Определяется в дочерней таблице и задает ее связь с родительской таблицей
5. Ограничение на проверку **CHECK**. Задает множество возможных значений атрибута. Оно записывается как ограничение на столбец или таблицу. Для одного столбца может быть задано несколько ограничений.
6. Ограничение **DEFAULT**. Позволяет задать значение по умолчанию. Ограничение задается только как ограничение на столбец.
7. При определении столбца в операторе CREATE TABLE можно определить столбец-счетчик, значения которого будут автоматически генерироваться сервером. Для этого нужно указать свойство **IDENTITY(ID)** (<начальное значение >,<шаг>). Свойство **IDENTITY** может быть указано только для одного столбца в таблице. Обычно оно указывается для атрибута, выбранного в качестве первичного ключа.

Используя таблицы 2.1-2.4 рассмотрим, к каким атрибутам можно применить ограничения. Запишем их в таблицах 2.5-2.8.

Таблица 2.5 — Отношение «Абонементы», типы данных, ограничения

|  |  |  |
| --- | --- | --- |
| Абонементы | | |
| Код | int | PK, ID |
| Название | nvarchar | UNIQUE, CHECK LIKE(‘[A-Я]%’) |
| Описание | nvarchar | CHECK LIKE(‘[A-Я]%’) |
| Стоимость | smallmoney | NOT NULL, CHECK >0 |
| Срок действия | date | NOT NULL |

Таблица 2.6 — Отношение «Клиенты», типы данных, ограничения

|  |  |  |
| --- | --- | --- |
| Клиенты | | |
| Код клиента | int | PK, ID |
| ФИО | nvarchar | NOT NULL, CHECK LIKE(‘[А-Я]%’) |
| Дата рождения | date | NOT NULL, CHECK (YEAR<2019) |
| Адрес | nvarchar |  |
| Телефон | nvarchar | NOT NULL, CHECK LIKE(‘+7 ([0-9][0-9][0-9]) [0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]’) |

Таблица 2.7 — Отношение «Договор», типы данных, ограничения

|  |  |  |
| --- | --- | --- |
| Договор | | |
| Номер договора | int | PK, ID |
| Код клиента | int | FK(Клиенты), NOT NULL |
| Код абонемента | int | FK(Абонементы), NOT NULL |
| Дата и время покупки | datetime | NOT NULL |

Таблица 2.8 — Отношение «Персональные тренировки», типы данных, ограничения

|  |  |  |
| --- | --- | --- |
| Персональные тренировки | | |
| Код клиента | int | FK(Клиенты) |
| Дата и время тренировки | datetime | UNIQUE(ФИО инструктор), NOT NULL |
| Стоимость | smallmoney | NOT NULL, CHECK>0 |
| ФИО инструктора | nvarchar | UNIQUE(Дата время тренировки), NOT NULL |

## 2.4 Построение схемы данных

На основе информации, полученной в пунктах 2.1-2.3 построим схему данных будущей БД.

Полученная схема представлена на рисунке 2.2.

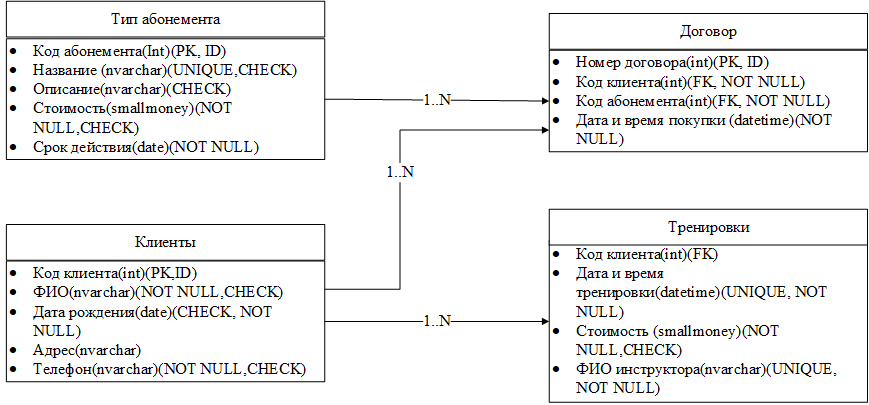


Рисунок 2.2 — Схема данных

## 2.5 Разработка сценария для создания БД и основных объектов структуры

После разработки сценария для создания БД получили код, представленный ниже.

|  |
| --- |
| use master  drop database SwimmingPool  --  USE MASTER  CREATE DATABASE SwimmingPool --имя БД  ON --описание файла БД  (NAME='Swiming\_Pool', --логическое имя файла БД  FILENAME='H:\university\3 семестр\Базы Данных\КР\Swimming\_Pool.mdf', --физическое имя файла БД  SIZE=2, --первоначальный размер в МБ  MAXSIZE=10, --максимальный размер в МБ  FILEGROWTH=2) --шаг для автоматического увеличения файла в МБ  LOG ON --описание файла БД  (NAME='SP\_Log', --логическое имя журнала транзакций  FILENAME='H:\university\3 семестр\Базы Данных\КР\Swimming\_Pool\_Log.mdf', --физическое имя файла БД  SIZE=2, --первоначальный размер в МБ  MAXSIZE=10, --максимальный размер в МБ  FILEGROWTH=2) --шаг для автоматического увеличения файла в МБ  GO  --  use SwimmingPool  CREATE TABLE [Тип Абонемента]  ([Код абонемента] int identity (1,1) NOT NULL,  Название nvarchar(100) UNIQUE check (Название like ('[А-Я][а-я]%')),  Описание nvarchar(100) check (Описание like ('[А-Я][а-я]%')),  Стоимость smallmoney NOT NULL CHECK(Стоимость>0),  [Срок действия] date NOT NULL)  GO  --  ALTER TABLE [Тип Абонемента]  ADD CONSTRAINT PK\_ТипАбонемента primary key ([Код абонемента])  go  --  use SwimmingPool  CREATE TABLE Клиенты  ([Код клиента] int identity (1,1) NOT NULL,  ФИО nvarchar(100) NOT NULL check (ФИО like ('[А-Я][а-я]%')),  [Дата рождения] date check (year([Дата рождения])<2019) NOT NULL,  Адрес nvarchar (100) ,  Телефон nvarchar (100) NOT NULL check ([Телефон] like '+7 ([0-9][0-9][0-9]) [0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]'))  GO  --  ALTER TABLE Клиенты  ADD CONSTRAINT PK\_Клиенты primary key ([Код клиента])  go  --  use SwimmingPool  CREATE TABLE Договор  ([Номер договора] int IDENTITY (1,1) NOT NULL,  [Код клиента]int NOT NULL,  [Код абонемента] int NOT NULL,  [Дата и время покупки] datetime NOT NULL)  GO  --  ALTER TABLE Договор  ADD CONSTRAINT PK\_Договор primary key ([Номер договора])  go  --  ALTER TABLE Договор  add CONSTRAINT ДОГОВОР\_ТИПАБОНЕМЕНТА\_FK  FOREIGN KEY ([Код абонемента]) REFERENCES [Тип Абонемента]([Код абонемента])  ON DELETE CASCADE  ON UPDATE CASCADE  go  --  ALTER TABLE Договор  add CONSTRAINT ДОГОВОР\_КЛИЕНТЫ\_FK  FOREIGN KEY ([Код клиента]) REFERENCES [Клиенты]([Код клиента])  ON DELETE CASCADE  ON UPDATE CASCADE  go  --  use SwimmingPool  CREATE TABLE Тренировки  ([Код клиента]int NOT NULL,  [Дата и время] datetime NOT NULL,  Стоимость smallmoney NOT NULL Check(Стоимость>0),  [ФИО инструктора] nvarchar(100) NOT NULL)  go  --  alter table Тренировки  add CONSTRAINT ТРЕНИРОВКИ\_КЛИЕНТЫ\_FK  FOREIGN KEY ([Код клиента]) REFERENCES [Клиенты]([Код клиента])  ON DELETE CASCADE  ON UPDATE CASCADE  go  --  alter table Тренировки  add CONSTRAINT ДАТА\_ФИО\_UNIQUE  UNIQUE ([Дата и время], [ФИО инструктора])  GO  -- |

## 2.6 Разработка сценария для заполнения таблиц БД

После разработки сценария для заполнения таблиц БД получили код, представленный ниже.

|  |
| --- |
| use master  use SwimmingPool  go  DBCC CHECKIDENT('Клиенты', RESEED, 1) --сброс счетчика  go  insert into Клиенты  values  ( 'Иванов Иван Иванович', '1975-10-25', 'Пушкина 8', '+7 (915) 558-65-69'),  ( 'Михайлов Дмитрий Владимирович', '1980-08-14', 'Грибоедова 19', '+7 (930) 667-56-58'),  ( 'Леонов Михаил Максимович', '2006-02-5', 'Васильевская 8', '+7 (800) 535-35-35'),  ( 'Чуваев Артем Александрович', '2000-09-23', 'Тимуровцев 15', '+7 (910) 999-53-53'),  ( 'Дмитриев Дмитрий Иванович', '1975-06-22', 'Новоселов 35', '+7 (992) 321-22-33'),  ( 'Смольский Иван Олегович', '2004-10-10', 'Московский 2', '+7 (920) 258-45-65'),  ( 'Егорова Елена Васильевна', '2008-12-22', 'Горького 16', '+7 (910) 456-69-62'),  ( 'Олегова Алина Евгеньевна', '1990-03-14', 'Гоголя 23', '+7 (932) 999-44-55'),  ( 'Вольская Ирина Чернова', '1999-03-25', 'Пушкина 25', '+7 (920) 356-44-25'),  ( 'Одинцова Ольга Владимировна', '2009-03-8', 'Горького 63', '+7 (920) 321-22-55')  go  insert into Клиенты(ФИО, [Дата рождения], Телефон)  values  ('Житков Ильдар Викторович','1980-05-02','+7 (920) 888-77-66')  go  DBCC CHECKIDENT('Тип Абонемента', RESEED, 1) --сброс счетчика  go  insert into [Тип Абонемента]  values  ('Разовый детский', 'Одно посещение ребенка', 200, '2020-12-31'),  ('Разовый взрослый', 'Одно посещение взрослого', 300, '2020-12-31'),  ('Разовый студенческий', 'Одно посещение студента', 100, '2020-12-31'),  ('Пятиразовый', 'Пять посещений', 800, '2020-12-31'),  ('Месячный взрослый', 'Месяц без ограничений', 1500, '2021-12-31'),  ('Месячный студенческий', 'Месяц без ограничений студентов', 1000, '2021-12-31'),  ('Пятиразовый студенческий', 'Пять посещений студента', 500, '2021-12-31'),  ('Трехмесячный', 'Пять месяцев без ограничений', 3000, '2021-12-31'),  ('Полугодовой', 'Полгода без ограничений', 5000, '2022-12-31'),  ('Трехмесячный студенческий', 'Три месяца студента без ограничений', 1900, '2021-12-31')  go  insert into Тренировки  values  (1, '2020-11-06T12:00:00', 300, 'Попов Юрий Александрович'),  (2, '2020-11-02T13:00:00', 300, 'Попов Юрий Александрович'),  (4, '2020-11-13T16:00:00', 100, 'Николаев Артем Викторович'),  (3, '2020-11-09T18:00:00', 200, 'Филлипов Евгений Геннадиевич'),  (6, '2020-11-08T12:00:00', 300, 'Филлипов Евгений Геннадиевич'),  (7, '2020-11-13T11:00:00', 200, 'Попов Юрий Александрович'),  (5, '2020-11-26T09:00:00', 300, 'Попов Юрий Александрович'),  (8, '2020-11-20T20:00:00', 300, 'Филлипов Евгений Геннадиевич'),  (10, '2020-11-19T14:00:00', 200, 'Попов Юрий Александрович'),  (9, '2020-11-15T12:00:00', 100, 'Николаев Артем Викторович'),  (7, '2020-11-16T20:00:00', 200, 'Попов Юрий Александрович'),  (1, '2020-11-26T15:00:00', 300, 'Попов Юрий Александрович'),  (10, '2020-11-29T13:00:00', 200, 'Попов Юрий Александрович'),  (2, '2020-11-20T21:00:00', 300, 'Попов Юрий Александрович'),  (4, '2020-11-23T10:00:00', 100, 'Николаев Артем Викторович'),  (3, '2020-11-16T13:00:00', 200, 'Филлипов Евгений Геннадиевич'),  (5, '2020-11-26T18:00:00', 300, 'Попов Юрий Александрович'),  (6, '2020-11-30T19:00:00', 200, 'Филлипов Евгений Геннадиевич'),  (8, '2020-11-12T19:00:00', 300, 'Филлипов Евгений Геннадиевич'),  (1, '2020-11-11T18:00:00', 300, 'Попов Юрий Александрович'),  (4, '2020-11-16T11:00:00', 100, 'Николаев Артем Викторович'),  (8, '2020-11-16T12:00:00', 300, 'Филлипов Евгений Геннадиевич')  go  DBCC CHECKIDENT('Договор', RESEED, 1) --сброс счетчика  go  insert into Договор  values  (1,2,'2020-01-20T12:31:00'),  (2,4,'2020-10-30T13:18:00'),  (4,3,'2020-10-22T10:35:00'),  (5,2,'2020-10-23T09:45:00'),  (6,1,'2020-10-11T14:58:00'),  (3,1,'2020-10-18T19:23:00'),  (10,1,'2020-10-23T20:20:00'),  (9,3,'2020-10-24T20:00:00'),  (7,1,'2020-10-13T19:05:00'),  (8,2,'2020-10-09T18:13:00'),  (10,1,'2020-10-27T21:14:00'),  (9,3,'2020-10-11T20:54:00'),  (8,2,'2020-10-30T11:11:00'),  (6,1,'2020-10-21T10:21:00'),  (7,1,'2020-10-14T21:17:00'),  (2,1,'2020-10-04T22:36:00'),  (1,2,'2020-10-10T13:32:00'),  (2,2,'2020-10-10T13:32:00'),  (3,1,'2020-10-10T13:32:00'),  (1,2,'2020-10-20T12:31:00'),  (1,2,'2020-10-20T12:32:00'),  (1,2,'2020-10-20T12:33:00')  go |

# 3 Разработка основных операций реляционной алгебры

1. Вывести ФИО и дату рождения клиентов.

R=Клиенты[ФИО][Дата рождения]

2. Вывести ФИО и номер клиентов, у которых отсутствует адрес.

R=(Клиенты WHERE Адрес IS NULL) [ФИО][Телефон]

3. Вывести названия абонементов у клиентов с кодами 1-5.

R1= (Договор JOIN Тип абонемента)

R2=(R1 WHERE Код клиента BETWEEN 1 and 5)[Код клиента][Название]

4. Вывести несовершеннолетних клиентов.

R=Клиенты WHERE YEAR(Дата рождения)>2002

5. Вывести клиентов тренера Филипова

R1=Тренировки JOIN Клиенты

R2= (R1 WHERE ФИО инструктора=Филипов Евгений Геннадиевич )[ФИО]

6. Клиенты со студенческим абонементом.

R1=(Тип абонемента JOIN Договор JOIN Клиенты)

R2= (R1 WHERE Название LIKE(%студенческий))[ФИО]

7.Вывести клиентов, у которых нет персональных тренировок.

R1=(Клиенты JOIN Тренировки)

R2=R1[Код клиента][ФИО][Дата рождения][Адрес][Телефон]

R3=Клиенты EXCEPT R2

8. Вывести клиентов, у которых тренер Попов.

R1=(Тренировки WHERE ФИО инструктора=Попов Юрий Александрович)[Код клиента]

R2=(Клиенты DIVIDE BY R1)[ФИО]

9. Вывести код клиентов, у которых месячный взрослый абонемент

R1=Договор JOIN Тип абонемента

R2=(R1 WHERE Название=“Месячный взрослый“)[Код клиента]

10.Вывести код взрослых клиентов

R1=Клиенты WHERE Дата рождения>=2002-01-01

11.Вывести клиентов, у которых номер договора начинается с 2

R1=Клиенты JOIN Договор

R2=(R1 WHERE Номер договора like (‘2%’))[ФИО]

12. Вывести ФИО клиентов у которых стоимость тренировки >100

R1=Клиенты JOIN Тренировки

R2=(R1 WHERE Стоимость>100) [ФИО]

13.Вывести информацию о клиентах, у которых указан адрес

R=Клиенты WHERE Адрес IS NOT NULL

14. Вывести всю информацию об абонементе, начинающегося с 3

R=Тип абонемента WHERE Код абонемента like (‘3%’)

15.ФИО клиентов, у которых разовое занятие

R1=Клиенты JOIN Договор JOIN Тип Абонемента

R2= (R1 WHERE Название LIKE (‘Разовый%’))[ФИО]

# 4 Разработка запросов к БД для реализации информационных потребностей пользователя

Для созданной базы данных разработаем следующие типы запросов:

1. Запросы с выбором строк по условию:

- запрос с использованием операторов сравнения;

|  |
| --- |
| --Вывести всю информацию об абонементах, с номером >3  select \*  from [Тип Абонемента]  where [Код абонемента]>3 |
| Рисунок 4.1.1 — Запрос с использованием операторов сравнения |

- запрос с использованием логических операторов AND, OR и NOT;

|  |
| --- |
| --вывести всю информацию об абонементах, в которых нет слова Разовый  select \*  from [Тип Абонемента]  where Название not like ('Разовый%') |
| Рисунок 4.1.2 — Запрос c использованием логических операторов AND, OR и NOT |

- запрос на использование комбинации логических операторов;

|  |
| --- |
| --вывести всю информацию о клиентах, с кодом оператора 910 или не 920  select \*  from Клиенты  where Телефон like ('%910%') or not Телефон like ('%920%') |
| Рисунок 4.1.3 — Запрос на использование комбинации логических операторов |

- запрос на использование выражений над столбцами;

|  |
| --- |
| --вывести все студенческие абонементы  select [Код абонемента], Название AS [Студенческие абонементы]  from [Тип Абонемента]  where Название like ('% студенческий') |
| Рисунок 4.1.4 — Запрос на использование выражений над столбцами |

- запрос с проверкой на принадлежность множеству;

|  |
| --- |
| --вывести всю информацию о тренировках, стоимостью 300  select \*  from Тренировки  where Стоимость in (300) |
| Рисунок 4.1.5 — Запрос с проверкой на принадлежность множеству |

- запрос с проверкой на принадлежность диапазону значений;

|  |
| --- |
| --вывести названия абонементов, цена которых в диапазоне 500 и 1000  select Название  from [Тип Абонемента]  where Стоимость between 500 and 1000 |
| Рисунок 4.1.6 — Запрос с проверкой на принадлежность диапазону значений |

- запрос с проверкой на соответствие шаблону;

|  |
| --- |
| --вывести дату и время тренировок Попова Ю.А.  select [ФИО инструктора], [Дата и время]  from Тренировки  where [ФИО инструктора] like ('Попов%') |
| Рисунок 4.1.7 — Запрос с проверкой на соответствие шаблону |

- запрос с проверкой на неопределенное значение.

|  |
| --- |
| --вывести ФИО и номер клиента, у которого не указан адрес  select ФИО, Телефон  from Клиенты  where Адрес is null |
| Рисунок 4.1.8 — Запрос с проверкой на неопределенное значение |

2. Запросы на теоретико-множественные реляционные операции:

- запрос на объединение таблиц;

|  |
| --- |
| --вывести ФИО клиентов и названия их абонементов  select distinct ФИО, Название as Абонемент  from [Тип Абонемента] join Договор on [Тип Абонемента].[Код абонемента]=Договор.[Код абонемента] join Клиенты on Договор.[Код клиента]=Клиенты.[Код клиента] |
| Рисунок 4.2.1 — Запрос на объединение таблиц |

- запрос на пересечение таблиц;

|  |
| --- |
| --вывести коды клиентов, у которых студенческий абонемент и есть персональные тренировки  select [Код клиента]  from Тренировки  intersect  select [Код клиента]  from Договор  where [Код абонемента] in (select [Код абонемента] from [Тип Абонемента] where Название like ('% студенческий')) |
| Рисунок 4.2.2 — Запрос на пересечение таблиц |

- запрос на вычитание таблиц;

|  |
| --- |
| --вывести коды клиентов, у которых нет персональных тренировок  select [Код клиента]  from Клиенты  except  select [Код клиента]  from Тренировки |
| Рисунок 4.2.3 — Запрос на вычитание таблиц |

- запрос с использованием декартового произведения таблиц;

|  |
| --- |
| --декартово произведение Тренировок и Клиентов  select \*  from Тренировки cross join Клиенты |
| Рисунок 4.2.3 — Запрос с использованием декартового произведения таблиц |

3. Многотабличные запросы:

- запрос с использованием соединения двух таблиц по равенству и условием отбора;

|  |
| --- |
| --вывести ФИО клиентов, ФИО тренеров и дату и время их тренировки  select ФИО, [ФИО инструктора], [Дата и время]  from Клиенты, Тренировки  where Клиенты.[Код клиента]=Тренировки.[Код клиента] |
| Рисунок 4.3.1 — Запрос с использованием соединения двух таблиц по равенству и условием отбора |

- запрос с использованием соединения трех таблиц по равенству и условием отбора;

|  |
| --- |
| --вывести ФИО клиентов и названия их абонементов  select distinct ФИО, Название  from Клиенты, Договор, [Тип Абонемента]  where Клиенты.[Код клиента]=Договор.[Код клиента] and Договор.[Код абонемента]=[Тип Абонемента].[Код абонемента] |
| Рисунок 4.3.2 — Запрос с использованием соединения трех таблиц по равенству и условием отбора |

- запрос с использованием соединения по четырем таблицам;

|  |
| --- |
| --вывести ФИО инструктора, ФИО клиента, номер договора, название абонемента  select distinct [ФИО инструктора], ФИО, [Номер договора], Название  from Тренировки, Договор, Клиенты, [Тип Абонемента]  where Тренировки.[Код клиента]=Договор.[Код клиента] and Договор.[Код клиента]=Клиенты.[Код клиента] and Договор.[Код абонемента]=[Тип Абонемента].[Код абонемента] |
| Рисунок 4.3.3 — Запрос с использованием соединения по четырем таблицам |

- запрос с использованием левого внешнего соединения;

|  |
| --- |
| --вывести ФИО всех клиентов и их тренеров, если имеется  select distinct ФИО as Клиент, [ФИО инструктора] as Тренер  from Клиенты left join Тренировки on Клиенты.[Код клиента]=Тренировки.[Код клиента] |
| Рисунок 4.3.4 — Запрос с использованием левого внешнего соединения |

- запрос на использование правого внешнего соединения;

|  |
| --- |
| --вывести ФИО всех клиентов и их тренеров, если имеется  select distinct ФИО as Клиент, [ФИО инструктора] as Тренер  from Тренировки right join Клиенты on Клиенты.[Код клиента]=Тренировки.[Код клиента] |
| Рисунок 4.3.5 – Запрос на использование правого внешнего соединения |

- запрос с использованием функции COUNT;

|  |
| --- |
| --вывести ФИО клиента и количество его тренировок  select ФИО, COUNT (Тренировки.[Код клиента]) as [Кол-во тренировок]  from Клиенты, Тренировки  where Клиенты.[Код клиента]=Тренировки.[Код клиента]  group by Клиенты.ФИО |
| Рисунок 4.3.6 – запрос с использованием функции COUNT |

- запрос с использованием функции SUM;

|  |
| --- |
| --Вывести фио клиента и сколько он потратил на тренировки  select ФИО, SUm (Тренировки.[Стоимость]) as [Стоимость тренировок]  from Клиенты, Тренировки  where Клиенты.[Код клиента]=Тренировки.[Код клиента]  group by Клиенты.ФИО |
| Рисунок 4.3.7 – Запрос с использованием функции SUM |

- 3 запроса с использованием временных функций;

|  |
| --- |
| --вывести несовершеннолетних клиентов  select \*  from Клиенты  where Year([Дата рождения]) > 2002 |
| Рисунок 4.3.8 – Запрос с функцией year() |
| --вывести номера договоров, заключенных в январе  select [Номер договора]  from Договор  where Month( [Дата и время покупки])=1 |
| Рисунок 4.3.9 – Запрос с функцией month() |
| select DAY([Дата и время покупки]) as [день заключения договора]  from Договор  where [Код клиента]=(select Клиенты.[Код клиента] from Клиенты Where ФИО ='Леонов Михаил Максимович') |
| Рисунок 4.3.10 – Запрос с функцией day() |

- запрос с использованием группировки по одному столбцу;

|  |
| --- |
| --вывести тренеров, кол-во клиентов  select [ФИО инструктора], COUNT ([Код клиента]) as [Кол-во клиентов]  from Тренировки  group by [ФИО инструктора] |
| Рисунок 4.3.11 — Запрос с использование по одному столбцу |

- запрос на использование группировки по нескольким столбцам;

|  |
| --- |
| select [ФИО инструктора], ФИО, COUNT(\*) as [Кол-во занятий]  from Тренировки inner join Клиенты on Тренировки.[Код клиента]=Клиенты.[Код клиента]  group by [ФИО инструктора], ФИО |
| Рисунок 4.3.12 — Запрос на использование группировки по нескольким столбцам |

- запрос с использованием условия отбора групп HAVING;

|  |
| --- |
| -- вывести клиентов, которые потратили на тренировки не менее 500  select Клиенты.ФИО, SUM (Тренировки.Стоимость) As Потратили  from Клиенты, Тренировки  where Клиенты.[Код клиента]=Тренировки.[Код клиента]  group by Клиенты.ФИО  having SUM(Тренировки.Стоимость)>=500 |
| Рисунок 4.3.13 — Запрос с использованием условия отбора групп HAVING |

- запрос с использованием сортировки по столбцу;

|  |
| --- |
| --Вывести фио клиентов в порядке убывания потраченного на тренировки  select Клиенты.ФИО, SUM (Тренировки.Стоимость) As Потратили  from Клиенты, Тренировки  where Клиенты.[Код клиента]=Тренировки.[Код клиента]  group by Клиенты.ФИО  order by Потратили DESC |
| Рисунок 4.3.14 — Запрос с использованием сортировки по столбцу |

4. Запросы на модификацию таблиц:

- запрос на добавление новых данных в таблицу;

|  |
| --- |
| --добавить новый абонемент  insert into [Тип Абонемента]  values  ('Разовый школьный', 'Одно посещение школьника', 150, '2020-12-31') |
| Рисунок 4.4.1 — Запрос на добавление новых данных в таблицу |

- запрос на добавление новых данных по результатам выполненного запроса;

|  |
| --- |
| insert into Договор  select [Код клиента], [Код абонемента]+1, [Дата и время покупки]  from Договор  where [Номер договора]=22 |
| Рисунок 4.4.2 — Запрос на добавление новых данных по результатам выполненного запроса |

- запрос на обновление существующих данных в таблице;

|  |
| --- |
| --увеличить стоимость всех абонементов  update [Тип Абонемента]  set Стоимость = Стоимость + 100 |
| Рисунок 4.4.3 — Запрос на обновление существующих данных в таблице |

- запрос на обновление существующих данных по результатам подзапроса во фразе WHERE;

|  |
| --- |
| --скидка 30% на взрослый абонемент  update [Тип Абонемента]  set Стоимость = Стоимость \* 0.7  where [Код абонемента] = (select [Код абонемента]  From [Тип Абонемента]  where Название='Разовый взрослый') |
| Рисунок 4.4.4 — Запрос на обновление существующих данных по результатам подзапроса во фразе WHERE |

- запрос на удаление существующих данных;

|  |
| --- |
| delete  from [Тип Абонемента] |
| Рисунок 4.4.5 — Запрос на удаление существующих данных |

- запрос на удаление существующих данных из одной таблицы на основе связанных с ней таблиц.

|  |
| --- |
| delete [Тип Абонемента]  from [Тип Абонемента] join Договор on [Тип Абонемента].[Код абонемента]=Договор.[Код абонемента]  where [Тип Абонемента].[Код абонемента]=Договор.[Код абонемента] |
| Рисунок 4.4.6 — Запрос на удаление существующих данных из одной таблицы на основе связанных с ней таблиц |

- запрос на удаление существующих данных по результатам подзапроса во фразе WHERE;

|  |
| --- |
| --удалить абонемент Разовый школьный  ddelete  from [Тип Абонемента]  where [Код абонемента]=(Select [Код абонемента]  from [Тип Абонемента]  Where Название= 'Разовый школьный') |
| Рисунок 4.4.6 — Запрос на удаление существующих данных по результатам подзапроса во фразе WHERE |

5. Многотабличные запросы с использованием подзапросов:

- 3 запроса с использованием операций сравнения;

|  |
| --- |
| --самый взрослый клиент  select \*  from Клиенты  where [Дата рождения] = (select MIN(([Дата рождения]))  from Клиенты)  --самый молодой клиент  select \*  from Клиенты  where [Дата рождения] = (select MAX(([Дата рождения]))  from Клиенты)  --Вывести тренеров, у которых оплата выше среднего  select distinct [ФИО инструктора]  from Тренировки  where Стоимость>= (Select avg(Стоимость) as Оплата  from Тренировки ) |
| Рисунок 4.5.1 — 3 запроса с использованием операций сравнения |

- запрос с использованием операции ANY;

|  |
| --- |
| --вывести информацию о клиентах, у которых есть тренировки  select \*  from Клиенты  where [Код клиента] = any (select [Код клиента] from Тренировки) |
| Рисунок 4.5.2 — Запрос с использованием операции ANY |

- запрос с использованием операции ALL;

|  |
| --- |
| -- Вывести информацию о клиентах, у которых нет персональных тренировок  select \*  from Клиенты  where [Код клиента] != all (select [Код клиента] from Тренировки) |
| Рисунок 4.5.3 — Запрос с использованием операции ALL |

- запрос с использованием операции IN;

|  |
| --- |
| --вывести клиентов у которых тренер Попов  select \*  from Клиенты  where [Код клиента] in (select [Код клиента] from Тренировки where [ФИО инструктора] like ('Попов%')) |
| Рисунок 4.5.4 — Запрос с использованием операции IN |

- запрос с использованием операции EXISTS;

|  |
| --- |
| -- определить клиентов, у которых есть тренировки  select \*  from Клиенты  where Exists (select \* from Тренировки where Клиенты.[Код клиента]=Тренировки.[Код клиента]) |
| Рисунок 4.5.5 — Запрос с использованием операции EXISTS |

- запрос с использованием двух вложенных друг в друга подзапросов;

|  |
| --- |
| --минимальное количество договоров у клиента  select ФИО  from Клиенты  where [Код клиента] = (select MIN(a) from  (select count(\*) as a, [Код клиента]  from Договор  group by [Код клиента]) as a) |
| Рисунок 4.5.6 — Запрос с использованием двух вложенных друг в друга подзапросов |

- запрос с использованием трех вложенных друг в друга подзапросов;

|  |
| --- |
| select distinct [ФИО инструктора]  from Тренировки  where [Код клиента]in(select [Код клиента]  from Клиенты  where [Код клиента]in(select [Код клиента]  from Договор  where [Код абонемента]in  (select [Код абонемента]  from [Тип Абонемента] where Название='Разовый детский'))) |
| Рисунок 4.5.7 — Запрос с использованием трех вложенных друг в друга подзапросов |

6. Запросы на изменение структуры существующей таблицы:

- запрос на добавление нового столбца к таблице;

|  |
| --- |
| alter table Клиенты  add [Постоянная скидка] int |
| Рисунок 4.6.1 — Запрос на добавление нового столбца к таблице |

- запрос на добавление нового ограничения к таблице;

|  |
| --- |
| alter table Клиенты  add Constraint uniq\_phone UNIQUE (Телефон) |
| Рисунок 4.6.2 — Запрос на добавление нового ограничения к таблице |

- запрос на удаление нового столбца к таблице;

|  |
| --- |
| alter table Клиенты  drop column [Постоянная скидка] |
| Рисунок 4.6.3 — Запрос на удаление нового столбца к таблице |

- запрос на удаление нового ограничения к таблице;

|  |
| --- |
| alter table Клиенты  drop Constraint uniq\_phone |
| Рисунок 4.6.4 — Запрос на удаление нового ограничения к таблице |

- запрос на изменение типа данных в таблице;

|  |
| --- |
| alter table [Тип абонемента] drop constraint [CK\_\_Тип Абоне\_\_Описа\_\_25869641]  alter table [Тип абонемента]  alter column Описание text |
| Рисунок 4.6.5 — Запрос на изменение типа данных в таблице |

- запрос на добавление нового первичного ключа к таблице с сохранением целостности таблицы;

|  |
| --- |
| alter table Тренировки  add constraint PK\_Тренировки  Primary key ([Код клиента],[Дата и время]) |
| Рисунок 4.6.6 — Запрос на добавление нового первичного ключа к таблице с сохранением целостности таблицы |

# Список используемой литературы

1. Р. Виейра. Программирование баз данных Microsoft SQL Server 2005. Базовый курс — М.: «Диалектика», 2007. —832 c.
2. Клайн К. SQL. Справочник. 2-е издание / Пер. с англ. – М.: КУДИЦ-ОБРАЗ, 2006 – 832 с.
3. Дж. Грофф, П. Вайнберг. SQL: Полное руководство: Пер. с англ. – 2-е изд., перераб. и доп. – К.: Издательская группа BHV, 2001. – 816 с., ил.
4. Дунаев В.В. Базы данных. Язык SQL. – СПб.: БХВ-Петербург, 2006. – 288 с.: ил.
5. Андон Ф., Резниченко В. Язык запросов SQL. Учебный курс. – СПб.: Питер; Киев: Издательская группа BHV, 2006. – 416 c.: ил.
6. Глушаков С.В.,Ломотько Д.В. Базы данных:Учебный курс / Харьков, Ростов-на-Дону, Киев,2002.
7. Гарбус Дж.,Паскузи Д.,Чанг Э. Database design on SQL Server 7.Сертификационный экзамен – экстерном (экзамен 70-029) СПб., 2000.
8. Конолли Томас, Бегг Каролин. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е изд.:Уч.пос. М.,2000.
9. Мамаев Е., Вишневский А. Microsoft SQL Server 7.0 для профессионалов. СПб. ., 2001.
10. Кренке Д. Теория и практика построения баз данных, - СПб.: Питер, 2005.
11. Черноусова А.М. Создание и использование баз данных, - ГОУ ОГУ, 2009. – 244 с.
12. Токмаков Г.П. Базы данных. Концепция баз данных, реляционная модель данных, языки SQL и XML. – УлГТУ, 2010. - 193 с.