# 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 обученным человеком. 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 нас 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.

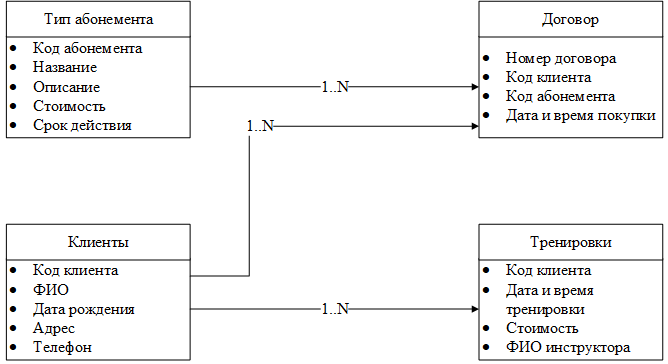


Рисунок 2.1 — Логическая модель данных

## 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 типа 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, 89101002010, 52-04-01, 121314.

## 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**. Это специальный случай комбинирования ограничений UNIQUE и NOT NULL. Простой первичный ключ задается как ограничение на конкретный столбец.
4. Ограничение внешнего ключа **FOREIGN KEY**. 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** (<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 | PRIMARY KEY, IDENTITY |
| Название | nvarchar | UNIQUE |
| Описание | nvarchar |  |
| Стоимость | smallmoney | NOT NULL |
| Срок действия | date | NOT NULL |

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

|  |  |  |
| --- | --- | --- |
| Клиенты | | |
| Код клиента | int | PRIMARY KEY, IDENTITY |
| ФИО | nvarchar | NOT NULL |
| Дата рождения | date | CHECK(<01.01.2016) |
| Адрес | nvarchar |  |
| Телефон | nvarchar | NOT NULL |

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

|  |  |  |
| --- | --- | --- |
| Договор | | |
| Номер договора | int | PRIMARY KEY, IDENTITY |
| Код клиента | int | FOREIGN KEY(Клиенты) |
| Код абонемента | int | FOREIGN KEY (Абонементы) |
| Дата и время покупки | datetime | NOT NULL |

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

|  |  |  |
| --- | --- | --- |
| Персональные тренировки | | |
| Код клиента | int | FOREIGN KEY(Клиенты) |
| Дата и время тренировки | datetime | UNIQUE(ФИО инструктор) |
| Стоимость | smallmoney | NOT NULL |
| ФИО инструктора | nvarchar | UNIQUE(Дата время тренировки) |

## 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.

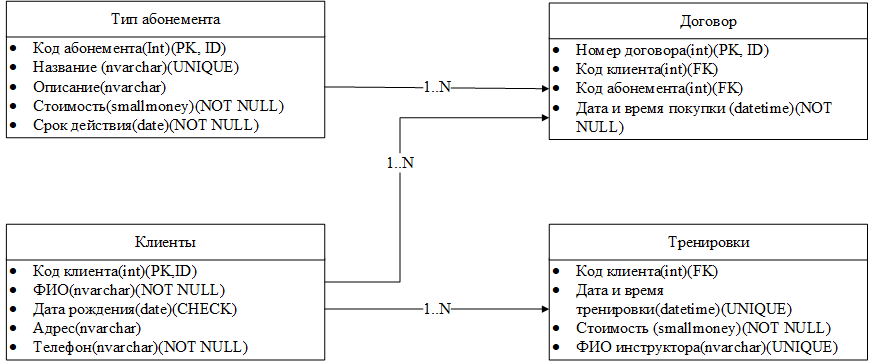


Рисунок 2.2 — Схема данных

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

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

|  |
| --- |
| USE MASTER  GO  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 PRIMARY KEY,  Название nvarchar(100) UNIQUE,  Описание nvarchar(100),  Стоимость smallmoney NOT NULL,  [Срок действия] date NOT NULL)    GO  use SwimmingPool  CREATE TABLE Клиенты  ([Код клиента] int PRIMARY KEY,  ФИО nvarchar(100) NOT NULL,  [Дата рождения] date, -- ТУТ ДОЛЖЕН БЫТЬ CHECK  Адрес nvarchar (100),  Телефон nvarchar (100) NOT NULL )    GO  use SwimmingPool  CREATE TABLE Договор  ([Номер договора] int PRIMARY KEY IDENTITY (1,1),  [Код клиента]int ,  [Код абонемента] int,  [Дата и время покупки] datetime NOT NULL,  CONSTRAINT ДОГОВОР\_ТИПАБОНЕМЕНТА\_FK  FOREIGN KEY ([Код клиента]) REFERENCES [Клиенты]([Код клиента]),  CONSTRAINT ДОГОВОР\_КЛИЕНТЫ\_FK  FOREIGN KEY ([Код клиента]) REFERENCES [Клиенты]([Код клиента]),  )  GO  use SwimmingPool  CREATE TABLE Тренировки  ([Код клиента]int,  [Дата и время] datetime,  Стоимость smallmoney NOT NULL,  [ФИО инструктора] nvarchar(100),  CONSTRAINT ТРЕНИРОВКИ\_КЛИЕНТЫ\_FK  FOREIGN KEY ([Код клиента]) REFERENCES [Клиенты]([Код клиента]),  CONSTRAINT ДАТА\_ФИО\_UNIQUE  UNIQUE ([Дата и время], [ФИО инструктора]))  GO |