МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ П. О. СУХОГО»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

дисциплина «Разработка приложений баз данных для информационных систем»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
«Разработка серверной части информационной системы в СУБД *MS SQL Server*»

ВАРИАНТ №12
Брачное агентство

Выполнил:
студент группы ИТИ-31, Зеленский К.А.
Принял:
доцент Асенчик О.Д.

Гомель 2024

Цель работы: разработать серверную часть клиент-серверной информационной системы, основанной на базе данных в заданной предметной области средствами СУБД *MS SQL Server*.

Ход работы и результаты.

На рисунке 1 представлена логическая модель реляционной базы данных, моделирующую предметную область.

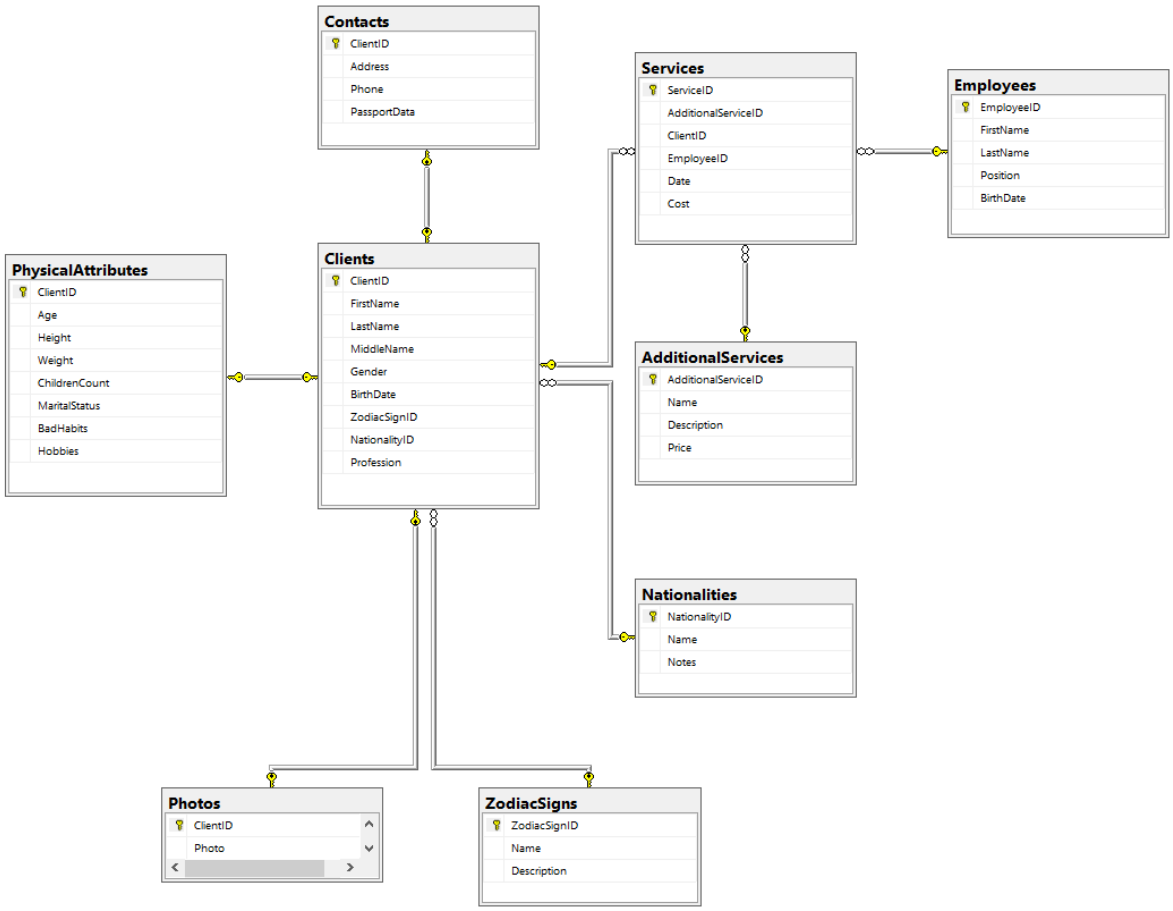

Рисунок 1 – Разработанная БД

На рисунках 2, 3, 4 представлен код на языке *Transact SQL* созданной базы данных *MarriageAgency* и всех её таблиц.

```
-- Создание базы данных
CREATE DATABASE MarriageAgency;
GO

-- Использование базы данных
USE MarriageAgency;
GO

-- Таблица Знаки зодиака
CREATE TABLE ZodiacSigns (
    ZodiacSignID INT PRIMARY KEY IDENTITY(1,1),
    Name NVARCHAR(50) NOT NULL UNIQUE, -- уникальные названия знаков
    Description NVARCHAR(MAX) DEFAULT 'Нет описания' -- значение по умолчанию
);
GO

-- Таблица Национальности
CREATE TABLE Nationalities (
    NationalityID INT PRIMARY KEY IDENTITY(1,1),
    Name NVARCHAR(50) NOT NULL UNIQUE, -- уникальные названия национальностей
    Notes NVARCHAR(MAX) DEFAULT 'Нет доступных заметок' -- значение по умолчанию
);
GO

-- Таблица личных данных клиентов
CREATE TABLE Clients (
    ClientID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    MiddleName NVARCHAR(50),
    Gender NVARCHAR(10) NOT NULL CHECK (Gender IN ('Женский', 'Мужской')), -- допустимы только Женский или Мужской
    BirthDate DATE CHECK (BirthDate <= GETDATE()), -- дата рождения не может быть в будущем
    ZodiacSignID INT FOREIGN KEY REFERENCES ZodiacSigns(ZodiacSignID) ON DELETE SET NULL ON UPDATE CASCADE,
    NationalityID INT FOREIGN KEY REFERENCES Nationalities(NationalityID) ON DELETE SET NULL ON UPDATE CASCADE,
    Profession NVARCHAR(100) DEFAULT 'Безработный' -- значение по умолчанию
);
GO
```

Рисунок 2 – Созданные таблицы 1

```
-- Таблица личных данных клиентов
CREATE TABLE Clients (
    ClientID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    MiddleName NVARCHAR(50),
    Gender NVARCHAR(10) NOT NULL CHECK (Gender IN ('Женский', 'Мужской')), -- допустимы только Женский или Мужской
    BirthDate DATE CHECK (BirthDate <= GETDATE()), -- дата рождения не может быть в будущем
    ZodiacSignID INT FOREIGN KEY REFERENCES ZodiacSigns(ZodiacSignID) ON DELETE SET NULL ON UPDATE CASCADE,
    NationalityID INT FOREIGN KEY REFERENCES Nationalities(NationalityID) ON DELETE SET NULL ON UPDATE CASCADE,
    Profession NVARCHAR(100) DEFAULT 'Безработный' -- значение по умолчанию
);
GO

-- Таблица контактов клиентов
CREATE TABLE Contacts (
    ClientID INT PRIMARY KEY FOREIGN KEY REFERENCES Clients(ClientID) ON DELETE CASCADE ON UPDATE CASCADE,
    Address NVARCHAR(255),
    Phone NVARCHAR(20) UNIQUE, -- уникальный номер телефона
    PassportData NVARCHAR(100) UNIQUE -- уникальные паспортные данные
);
GO

-- Таблица физической информации о клиентах
CREATE TABLE PhysicalAttributes (
    ClientID INT PRIMARY KEY FOREIGN KEY REFERENCES Clients(ClientID) ON DELETE CASCADE ON UPDATE CASCADE,
    Age INT CHECK (Age >= 18), -- минимальный возраст 18
    Height DECIMAL(5,2) CHECK (Height >= 0), -- рост не может быть отрицательным
    Weight DECIMAL(5,2) CHECK (Weight >= 0), -- вес не может быть отрицательным
    ChildrenCount INT CHECK (ChildrenCount >= 0), -- количество детей не может быть отрицательным
    MaritalStatus NVARCHAR(50) DEFAULT 'Одинокий', -- значение по умолчанию
    BadHabits NVARCHAR(MAX),
    Hobbies NVARCHAR(MAX)
);
GO
```

Рисунок 3 – Созданные таблицы 2

```sql
-- Таблица фотографий клиентов (один-к-одному с таблицей клиентов)
CREATE TABLE Photos (
    ClientID INT PRIMARY KEY FOREIGN KEY REFERENCES Clients(ClientID) ON DELETE CASCADE ON UPDATE CASCADE, -- связь один к одному
    Photo NVARCHAR(MAX) NOT NULL -- Путь к фото или двоичные данные
);
GO

-- Таблица дополнительных услуг
CREATE TABLE AdditionalServices (
    AdditionalServiceID INT PRIMARY KEY IDENTITY(1,1),
    Name NVARCHAR(100) NOT NULL UNIQUE, -- уникальные названия услуг
    Description NVARCHAR(MAX) DEFAULT 'Нет описания', -- значение по умолчанию
    Price DECIMAL(10, 2) NOT NULL CHECK (Price >= 0) -- цена должна быть положительной
);
GO

-- Таблица сотрудников
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    Position NVARCHAR(100) DEFAULT 'Неизвестный', -- значение по умолчанию
    BirthDate DATE CHECK (BirthDate <= GETDATE()) -- дата рождения не может быть в будущем
);
GO

-- Таблица оказанных услуг (основные услуги клиентам)
CREATE TABLE Services (
    ServiceID INT PRIMARY KEY IDENTITY(1,1),  -- уникальный идентификатор записи оказанной услуги
    AdditionalServiceID INT NOT NULL FOREIGN KEY REFERENCES AdditionalServices(AdditionalServiceID) ON DELETE CASCADE ON UPDATE CASCADE,  --
    ClientID INT NOT NULL FOREIGN KEY REFERENCES Clients(ClientID) ON DELETE CASCADE ON UPDATE CASCADE,  -- клиент, которому оказана услуга
    EmployeeID INT NOT NULL FOREIGN KEY REFERENCES Employees(EmployeeID) ON DELETE CASCADE ON UPDATE CASCADE,  -- сотрудник, оказавший услугу
    Date DATE NOT NULL,  -- дата оказания услуги
    Cost DECIMAL(10, 2) NOT NULL CHECK (Cost >= 0)  -- стоимость услуги
);
GO
```

Рисунок 4 – Созданные таблицы 3

На рисунках 5, 6, 7 представлен код на языке *Transact SQL* тестового набора данных для таблицы Клиенты на стороне отношения один.

```sql
-- Создание временных таблиц для имен, фамилий, отчеств и профессий
CREATE TABLE #MaleNames (Name NVARCHAR(50));
CREATE TABLE #FemaleNames (Name NVARCHAR(50));
CREATE TABLE #LastNames (LastName NVARCHAR(50));
CREATE TABLE #FemaleLastNames (FemaleLastName NVARCHAR(50));
CREATE TABLE #MalePatronymics (Patronymic NVARCHAR(50));
CREATE TABLE #FemalePatronymics (Patronymic NVARCHAR(50));
CREATE TABLE #Professions (Profession NVARCHAR(100));

-- Заполнение временных таблиц
INSERT INTO #MaleNames (Name) VALUES
('Александр'), ('Максим'), ('Дмитрий'), ('Иван'), ('Анатолий'),
('Евгений'), ('Павел'), ('Артем'), ('Сергей'), ('Роман');

INSERT INTO #FemaleNames (Name) VALUES
('Анна'), ('Екатерина'), ('Мария'), ('Ольга'), ('Светлана'),
('Наталья'), ('Дарья'), ('Елена'), ('Татьяна'), ('Юлия');

INSERT INTO #LastNames (LastName) VALUES
('Иванов'), ('Петров'), ('Сидоров'), ('Кузнецов'), ('Семёнов'),
('Фёдоров'), ('Морозов'), ('Попов'), ('Васильев'), ('Зайцев');

INSERT INTO #FemaleLastNames (FemaleLastName) VALUES
('Иванова'), ('Петрова'), ('Сидорова'), ('Кузнецова'), ('Семёнова'),
('Фёдорова'), ('Морозова'), ('Попова'), ('Васильева'), ('Зайцева');

INSERT INTO #MalePatronymics (Patronymic) VALUES
('Александрович'), ('Максимович'), ('Дмитриевич'), ('Иванович'),
('Анатольевич'), ('Евгеньевич'), ('Павлович'), ('Артёмович'),
('Сергеевич'), ('Романович');

INSERT INTO #FemalePatronymics (Patronymic) VALUES
('Александровна'), ('Максимовна'), ('Дмитриевна'), ('Ивановна'),
('Анатольевна'), ('Евгеньевна'), ('Павловна'), ('Артёмовна'),
('Сергеевна'), ('Романовна');
```

Рисунок 5 – Заполененная таблица Клиентов 1

```
INSERT INTO #Professions (Profession) VALUES
('Программист'), ('Врач'), ('Учитель'), ('Инженер'), ('Маркетолог'),
('Дизайнер'), ('Адвокат'), ('Экономист'), ('Менеджер'), ('Фармацевт'),
('Архитектор'), ('Журналист'), ('Полицейский'), ('Психолог'), ('Слесарь'),
('Электрик'), ('Повар'), ('Шеф-повар'), ('Водитель'), ('Флорист');

-- Заполнение таблицы Клиенты с разнообразными именами, фамилиями и профессиями
DECLARE @i INT = 1;
DECLARE @MaleName NVARCHAR(50);
DECLARE @FemaleName NVARCHAR(50);
DECLARE @LastName NVARCHAR(50);
DECLARE @FemaleLastName NVARCHAR(50);
DECLARE @MalePatronymic NVARCHAR(50);
DECLARE @FemalePatronymic NVARCHAR(50);
DECLARE @Profession NVARCHAR(100);

WHILE @i <= 500
BEGIN
    -- Случайная профессия
    SELECT TOP 1 @Profession = Profession FROM #Professions ORDER BY NEWID();

    IF @i % 2 = 0
    BEGIN
        -- Женский клиент
        SELECT TOP 1 @FemaleName = Name FROM #FemaleNames ORDER BY NEWID();
        SELECT TOP 1 @FemaleLastName = FemaleLastName FROM #FemaleLastNames ORDER BY NEWID();
        SELECT TOP 1 @FemalePatronymic = Patronymic FROM #FemalePatronymics ORDER BY NEWID();

        INSERT INTO Clients (FirstName, LastName, MiddleName, Gender, BirthDate, ZodiacSignID, NationalityID, Profession)
        VALUES (
            @FemaleName,
            @FemaleLastName,
            @FemalePatronymic,
            'Женский',
            DATEADD(YEAR, -FLOOR(RAND() * 50) - 18, GETDATE()),  -- Возраст от 18 до 67 лет
            FLOOR(RAND() * 12) + 1,  -- Случайный знак зодиака
```

Рисунок 6 – Заполененная таблица Клиентов 2

```
            FLOOR(RAND() * 10) + 1,  -- Случайная национальность
            @Profession  -- Случайная профессия
        );
    END
    ELSE
    BEGIN
        -- Мужской клиент
        SELECT TOP 1 @MaleName = Name FROM #MaleNames ORDER BY NEWID();
        SELECT TOP 1 @LastName = LastName FROM #LastNames ORDER BY NEWID();ы
        SELECT TOP 1 @MalePatronymic = Patronymic FROM #MalePatronymics ORDER BY NEWID();

        INSERT INTO Clients (FirstName, LastName, MiddleName, Gender, BirthDate, ZodiacSignID, NationalityID, Profession)
        VALUES (
            @MaleName,
            @LastName,
            @MalePatronymic,
            'Мужской',
            DATEADD(YEAR, -FLOOR(RAND() * 50) - 18, GETDATE()),  -- Возраст от 18 до 67 лет
            FLOOR(RAND() * 12) + 1,  -- Случайный знак зодиака
            FLOOR(RAND() * 10) + 1,  -- Случайная национальность
            @Profession  -- Случайная профессия
        );
    END;

    SET @i = @i + 1;
END;

-- Удаление временных таблиц
DROP TABLE #Professions;
DROP TABLE #MaleNames;
DROP TABLE #FemaleNames;
DROP TABLE #LastNames;
DROP TABLE #FemaleLastNames;
DROP TABLE #MalePatronymics;
DROP TABLE #FemalePatronymics;
```

Рисунок 7 – Заполененная таблица Клиентов 3

На рисунках 8, 9, 10 представлен код на языке *Transact SQL* тестового набора данных для таблицы Сотрудники на стороне отношения один.

```
-- Создание временных таблиц для имен, фамилий, отчеств и позиций
CREATE TABLE #MaleNames (Name NVARCHAR(50));
CREATE TABLE #FemaleNames (Name NVARCHAR(50));
CREATE TABLE #LastNames (LastName NVARCHAR(50));
CREATE TABLE #FemaleLastNames (FemaleLastName NVARCHAR(50));
CREATE TABLE #MalePatronymics (Patronymic NVARCHAR(50));
CREATE TABLE #FemalePatronymics (Patronymic NVARCHAR(50));
CREATE TABLE #Positions (Position NVARCHAR(100));

-- Заполнение временных таблиц
INSERT INTO #MaleNames (Name) VALUES
('Александр'), ('Максим'), ('Дмитрий'), ('Иван'), ('Анатолий'),
('Евгений'), ('Павел'), ('Артем'), ('Сергей'), ('Роман');

INSERT INTO #FemaleNames (Name) VALUES
('Анна'), ('Екатерина'), ('Мария'), ('Ольга'), ('Светлана'),
('Наталья'), ('Дарья'), ('Елена'), ('Татьяна'), ('Юлия');

INSERT INTO #LastNames (LastName) VALUES
('Иванов'), ('Петров'), ('Сидоров'), ('Кузнецов'), ('Семёнов'),
('Фёдоров'), ('Морозов'), ('Попов'), ('Васильев'), ('Зайцев');

INSERT INTO #FemaleLastNames (FemaleLastName) VALUES
('Иванова'), ('Петрова'), ('Сидорова'), ('Кузнецова'), ('Семёнова'),
('Фёдорова'), ('Морозова'), ('Попова'), ('Васильева'), ('Зайцева');

INSERT INTO #MalePatronymics (Patronymic) VALUES
('Александрович'), ('Максимович'), ('Дмитриевич'), ('Иванович'),
('Анатольевич'), ('Евгеньевич'), ('Павлович'), ('Артёмович'),
('Сергеевич'), ('Романович');

INSERT INTO #FemalePatronymics (Patronymic) VALUES
('Александровна'), ('Максимовна'), ('Дмитриевна'), ('Ивановна'),
('Анатольевна'), ('Евгеньевна'), ('Павловна'), ('Артёмовна'),
('Сергеевна'), ('Романовна');
```

Рисунок 8 – Заполененная таблица Сотрудники 1

```sql
-- Добавление позиций сотрудников
INSERT INTO #Positions (Position) VALUES
('Менеджер по работе с клиентами'),
('Консультант'),
('HR специалист'),
('Маркетолог'),
('Аналитик'),
('IT специалист'),
('Бухгалтер'),
('Юрист'),
('Офис-менеджер'),
('Специалист по рекламе');

-- Заполнение таблицы Employees с разнообразными именами, фамилиями и позициями
DECLARE @i INT = 1;
DECLARE @MaleName NVARCHAR(50);
DECLARE @FemaleName NVARCHAR(50);
DECLARE @LastName NVARCHAR(50);
DECLARE @FemaleLastName NVARCHAR(50);
DECLARE @MalePatronymic NVARCHAR(50);
DECLARE @FemalePatronymic NVARCHAR(50);
DECLARE @Position NVARCHAR(100);
WHILE @i <= 500  -- 500 сотрудников
BEGIN
    -- Случайный выбор позиции
    SELECT TOP 1 @Position = Position FROM #Positions ORDER BY NEWID();

    IF @i % 2 = 0
    BEGIN
        -- Женский сотрудник
        SELECT TOP 1 @FemaleName = Name FROM #FemaleNames ORDER BY NEWID();
        SELECT TOP 1 @FemaleLastName = FemaleLastName FROM #FemaleLastNames ORDER BY NEWID();
        SELECT TOP 1 @FemalePatronymic = Patronymic FROM #FemalePatronymics ORDER BY NEWID();

        INSERT INTO Employees (FirstName, LastName, MiddleName, BirthDate, Position)
        VALUES (
```

Рисунок 9 – Заполененная таблица Сотрудники 2

```
        VALUES (
            @FemaleName,
            @FemaleLastName,
            @FemalePatronymic,
            DATEADD(YEAR, -FLOOR(RAND() * 40) - 18, GETDATE()),  -- Возраст от 18 до 58 лет
            @Position  -- Случайная должность
        );
    END
    ELSE
    BEGIN
        -- Мужской сотрудник
        SELECT TOP 1 @MaleName = Name FROM #MaleNames ORDER BY NEWID();
        SELECT TOP 1 @LastName = LastName FROM #LastNames ORDER BY NEWID();
        SELECT TOP 1 @MalePatronymic = Patronymic FROM #MalePatronymics ORDER BY NEWID();

        INSERT INTO Employees (FirstName, LastName, MiddleName, BirthDate, Position)
        VALUES (
            @MaleName,
            @LastName,
            @MalePatronymic,
            DATEADD(YEAR, -FLOOR(RAND() * 40) - 18, GETDATE()),  -- Возраст от 18 до 58 лет
            @Position  -- Случайная должность
        );
    END;
    SET @i = @i + 1;
END;

-- Удаление временных таблиц
DROP TABLE #MaleNames;
DROP TABLE #FemaleNames;
DROP TABLE #LastNames;
DROP TABLE #FemaleLastNames;
DROP TABLE #MalePatronymics;
DROP TABLE #FemalePatronymics;
```

Рисунок 10 – Заполененная таблица Сотрудники 3

На рисунке 11 представлен код на языке *Transact SQL* тестового набора данных для таблицы Услуги на стороне отношения многие.

```
-- Заполнение таблицы Services
DECLARE @i INT = 1;
DECLARE @ClientID INT;
DECLARE @EmployeeID INT;
DECLARE @AdditionalServiceID INT;
DECLARE @Date DATE;
DECLARE @Cost DECIMAL(10, 2);

WHILE @i <= 20000  -- Допустим, у нас 20 000 записей об оказанных услугах
BEGIN
    -- Случайный выбор существующего клиента, сотрудника и дополнительной услуги
    SET @ClientID = FLOOR(RAND() * 500) + 1;  -- Существующие клиенты (500 записей)
    SET @EmployeeID = FLOOR(RAND() * 100) + 1;  -- Существующие сотрудники (100 записей)
    SET @AdditionalServiceID = FLOOR(RAND() * 10) + 1;  -- Существующие дополнительные услуги (10 записей)

    -- Случайная дата оказания услуги за последние 2 года
    SET @Date = DATEADD(DAY, -FLOOR(RAND() * 730), GETDATE());

    -- Получение стоимости услуги из таблицы AdditionalServices
    SELECT TOP 1 @Cost = Price FROM AdditionalServices WHERE AdditionalServiceID = @AdditionalServiceID;

    -- Вставка в таблицу Services
    INSERT INTO Services (AdditionalServiceID, ClientID, EmployeeID, Date, Cost)
    VALUES (
        @AdditionalServiceID,  -- Случайная услуга
        @ClientID,  -- Случайный клиент
        @EmployeeID,  -- Случайный сотрудник
        @Date,  -- Случайная дата оказания услуги
        @Cost  -- Стоимость услуги
    );

    SET @i = @i + 1;
END;
```

Рисунок 11 – Заполененная таблица Услуги

На рисунках 12, 13, 14 представлен код на языке *Transact SQL* созданных трёх представлений.

```
CREATE VIEW ClientFullInfo AS
SELECT
    c.ClientID,
    c.FirstName,
    c.LastName,
    c.MiddleName,
    c.Gender,
    c.BirthDate,
    z.Name AS ZodiacSign,
    n.Name AS Nationality,
    c.Profession,
    p.Age,
    p.Height,
    p.Weight,
    p.ChildrenCount,
    p.MaritalStatus,
    p.BadHabits,
    p.Hobbies,
    con.Address,
    con.Phone,
    con.PassportData,
    ph.Photo
FROM
    Clients c
    LEFT JOIN PhysicalAttributes p ON c.ClientID = p.ClientID
    LEFT JOIN Contacts con ON c.ClientID = con.ClientID
    LEFT JOIN ZodiacSigns z ON c.ZodiacSignID = z.ZodiacSignID
    LEFT JOIN Nationalities n ON c.NationalityID = n.NationalityID
    LEFT JOIN Photos ph ON c.ClientID = ph.ClientID;
```

Рисунок 12 – Представление 1

```
CREATE VIEW EmployeeServices AS
SELECT
    e.EmployeeID,
    e.FirstName AS EmployeeFirstName,
    e.LastName AS EmployeeLastName,
    e.Position,
    s.ServiceID,
    s.ClientID,
    c.FirstName AS ClientFirstName,
    c.LastName AS ClientLastName,
    s.Date AS ServiceDate,
    s.Cost,
    a.Name AS ServiceName
FROM
    Employees e
    INNER JOIN Services s ON e.EmployeeID = s.EmployeeID
    INNER JOIN Clients c ON s.ClientID = c.ClientID
    INNER JOIN AdditionalServices a ON s.AdditionalServiceID = a.AdditionalServiceID;
```

Рисунок 13 – Представление 2

```sql
CREATE VIEW ClientServices AS
SELECT
    c.ClientID,
    c.FirstName AS ClientFirstName,
    c.LastName AS ClientLastName,
    s.ServiceID,
    a.Name AS ServiceName,
    s.Date AS ServiceDate,
    s.Cost,
    e.FirstName AS EmployeeFirstName,
    e.LastName AS EmployeeLastName
FROM
    Clients c
    INNER JOIN Services s ON c.ClientID = s.ClientID
    INNER JOIN AdditionalServices a ON s.AdditionalServiceID = a.AdditionalServiceID
    INNER JOIN Employees e ON s.EmployeeID = e.EmployeeID;
```

Рисунок 14 –Представление 3

На рисунке 15 представлено использование трёх созданных представлений средствами *Transact SQL*.

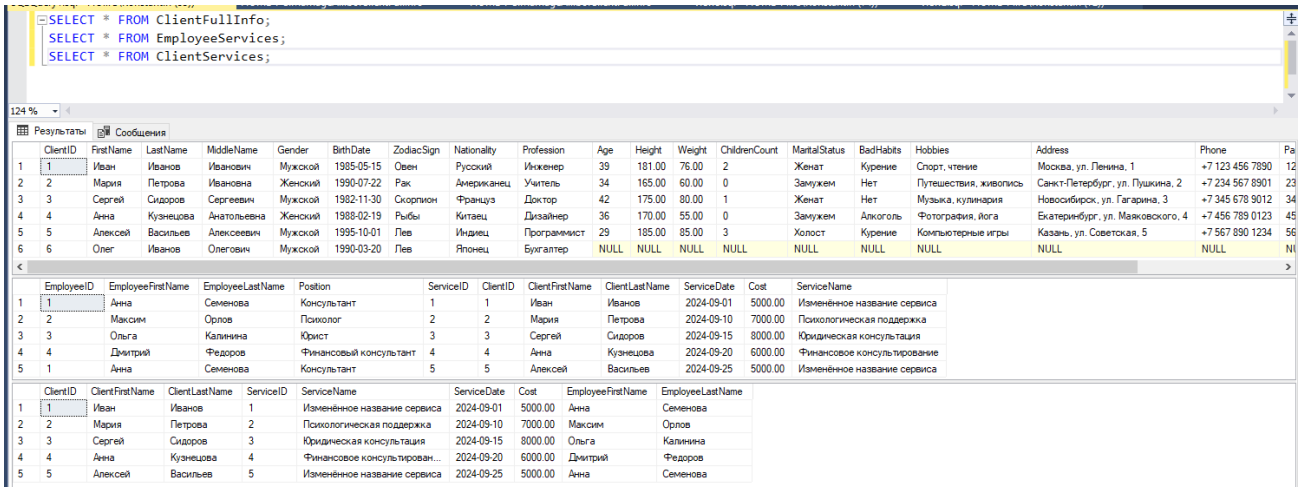

Рисунок 15 – Использование представлений

На рисунках 16, 17, 18 представлен код на языке *Transact SQL* созданных трёх хранимых процедур.

```sql
CREATE PROCEDURE InsertClient
    @FirstName NVARCHAR(50),
    @LastName NVARCHAR(50),
    @MiddleName NVARCHAR(50) = NULL,
    @Gender NVARCHAR(10),
    @BirthDate DATE,
    @ZodiacSignID INT = NULL,
    @NationalityID INT = NULL,
    @Profession NVARCHAR(100) = 'Безработный'
AS
BEGIN
    INSERT INTO Clients (FirstName, LastName, MiddleName, Gender, BirthDate, ZodiacSignID, NationalityID, Profession)
    VALUES (@FirstName, @LastName, @MiddleName, @Gender, @BirthDate, @ZodiacSignID, @NationalityID, @Profession);
END;
```

Рисунок 16 – Хранимая процедура 1

```
CREATE PROCEDURE UpdatePhysicalAttributes
    @ClientID INT,
    @Age INT,
    @Height DECIMAL(5,2),
    @Weight DECIMAL(5,2),
    @ChildrenCount INT,
    @MaritalStatus NVARCHAR(50),
    @BadHabits NVARCHAR(MAX) = NULL,
    @Hobbies NVARCHAR(MAX) = NULL
AS
BEGIN
    UPDATE PhysicalAttributes
    SET
        Age = @Age,
        Height = @Height,
        Weight = @Weight,
        ChildrenCount = @ChildrenCount,
        MaritalStatus = @MaritalStatus,
        BadHabits = @BadHabits,
        Hobbies = @Hobbies
    WHERE ClientID = @ClientID;
END;
```

Рисунок 17 – Хранимая процедура 2

```
CREATE PROCEDURE UpsertAdditionalService
    @AdditionalServiceID INT = NULL,
    @Name NVARCHAR(100),
    @Description NVARCHAR(MAX) = 'Нет описания',
    @Price DECIMAL(10, 2)
AS
BEGIN
    IF @AdditionalServiceID IS NULL
    BEGIN
        -- Вставка нового сервиса
        INSERT INTO AdditionalServices (Name, Description, Price)
        VALUES (@Name, @Description, @Price);
    END
    ELSE
    BEGIN
        -- Обновление существующего сервиса
        UPDATE AdditionalServices
        SET
            Name = @Name,
            Description = @Description,
            Price = @Price
        WHERE AdditionalServiceID = @AdditionalServiceID;
    END
END;
```

Рисунок 18 – Хранимая процедура 3

На рисунке 19 представлен вызов трёх хранимых процедур с параметрами для вставки и обновления данных в таблицах базы данных.

```sql
-- Вставка нового клиента
EXEC InsertClient
    @FirstName = 'Олег',
    @LastName = 'Иванов',
    @MiddleName = 'Олегович',
    @Gender = 'Мужской',
    @BirthDate = '1990-03-20',
    @ZodiacSignID = 5, -- Лев
    @NationalityID = 6, -- Итальянец
    @Profession = 'Бухгалтер';

    -- Обновление физических характеристик клиента
EXEC UpdatePhysicalAttributes
    @ClientID = 1,
    @Age = 39,
    @Height = 181.00,
    @Weight = 76.00,
    @ChildrenCount = 2,
    @MaritalStatus = 'Женат',
    @BadHabits = 'Курение',
    @Hobbies = 'Спорт, чтение';

    -- Вставка нового сервиса
EXEC UpsertAdditionalService
    @Name = 'Астрологическая консультация',
    @Description = 'Консультация по вопросам астрологии',
    @Price = 4500.00;

-- Обновление существующего сервиса
EXEC UpsertAdditionalService
    @AdditionalServiceID = 1, -- Сервис с ID = 1
    @Name = 'Изменённое название сервиса',
    @Description = 'Обновлённое описание сервиса',
    @Price = 5500.00;
```

Рисунок 19 – Вызов хранимых процедур

На рисунке 20 представлена строка подключения с удалённым сервером.

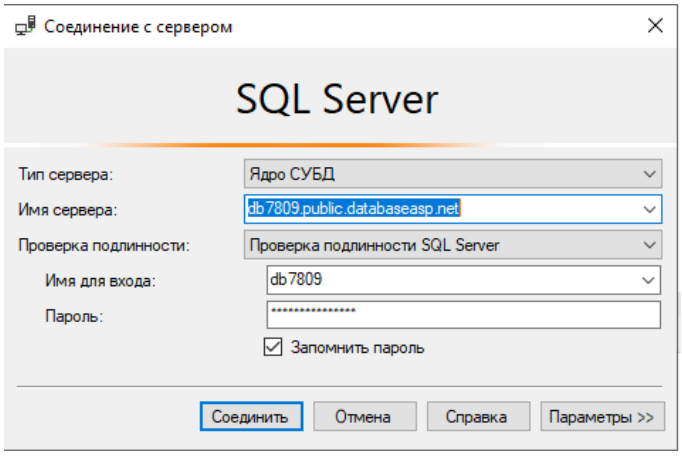

Рисунок 20 – Строка подключения

На рисунке 21 представлена созданная база данных на удалённом сервере.

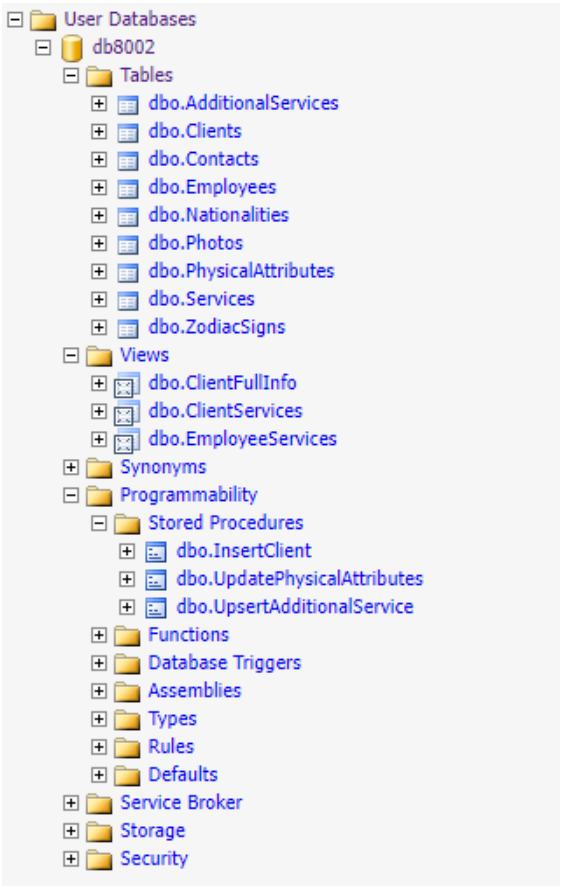

Рисунок 21 – Созданная БД на удалённом сервере

Выводы: была разработана серверная часть клиент-серверной информационной системы, основанной на базе данных в области брачных агентств с использованием средств СУБД *MS SQL Server*.