Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**ИВАНОВСКИЙ ГОСУДАРСТВЕННЫЙ ХИМИКО-ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет техники, управления и цифровой инфраструктуры**

**Кафедра информационных технологий и цифровой экономики**

**КУРСОВОЙ ПРОЕКТ**

**ПО ДИСЦИПЛИНЕ**

**УПРАВЛЕНИЕ ДАННЫМИ**

**Тема:** Разработка базы данных MS SQL

и приложения средствами Visual Studio.

Вариант №6

**Выполнил студент: Мурашов Виктор Витальевич курс/гр. 3/147**

ФИО

**Проверил: Куваев Александр Евгеньевич**

ФИО

Иваново 2020

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**ИВАНОВСКИЙ ГОСУДАРСТВЕННЫЙ ХИМИКО-ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет** ТУиЦИ

**Кафедра** ИТиЦЭ

**Направление** 09.03.02 Информационные системы и технологии

**Профиль** Информационные системы и технологии

**Дисциплина** Управление данными

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

**Студенту** Мурашову Виктору Витальевичу курс/гр. **3/147**

(фамилия, имя, отчество - полностью)

**Тема:** Разработка базы данных MS SQL и приложения средствами Visual Studio. Вариант №6

Исходные данные: *законодательные и подзаконные нормативные акты; энциклопедическая и справочная литература; статистические и аналитические материалы; монографии; данные профессиональных периодических изданий; интернет-ресурсы; самостоятельно собранные первичные материалы.*

Содержание работы:

Введение: *цель и задачи работы; предмет и объект исследования; обоснование структуры работы.*

Теоретическая часть должна содержать: *системный анализ предметной области, инфологическую модель БД, список запросов, которые должны быть реализованы в БД, список требований к приложению*

В практической части необходимо: *сгенерировать БД, реализовать SQL-запросы, разработать программное приложение к БД.*

Заключение*: основные выводы по проекту, обобщение основных предложений и рекомендаций.*

Руководитель работы:  *Куваев А.Е.*

**Календарный план-график**

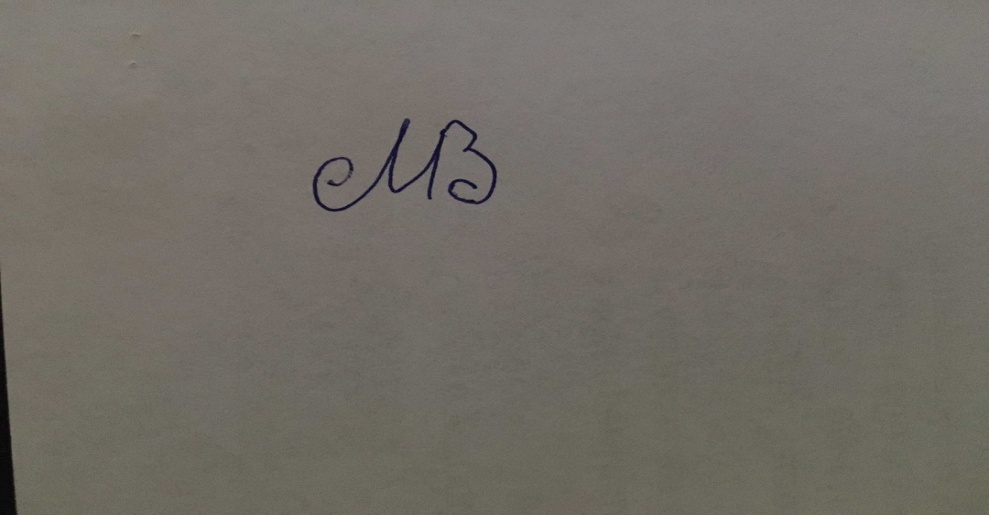
|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование этапов выполнения курсовой работы | Срок выполнения этапов | Примечание |
| 1 | Исследование предметной области и составление логической модели БД | 17.02.20 – 17.03.20 |  |
| 2 | Создание физической модели БД | 18.03.20 – 31.03.20 |  |
| 3 | Генерация БД средствами MS SQL и реализация SQL-запросов | 01.04.20 – 14.04.20 |  |
| 4 | Разработка приложения средствами Visual Studio | 15.04.20 – 12.05.20 |  |
| 5 | Оформление пояснительной записки | 13.05.20 – 17.05.20 |  |
| 6 | Защита курсового проекта | 18.05.20 - 23.05.20 |  |

Дата выдачи задания 17.02.2020

Дата представления законченной работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись /Ф.И.О.)

Студент  Мурашов В.В.\_\_\_\_\_\_

(Подпись /Ф.И.О.)

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc41258843)

[1 СИСТЕМНЫЙ АНАЛИЗ 6](#_Toc41258844)

[1.1Описание предметной области 6](#_Toc41258845)

[1.2 Логическая модель «Как есть» 7](#_Toc41258846)

[1.3 Логическая модель «Как должно быть» 8](#_Toc41258847)

[1.4 Список запросов, которые должны быть реализованы в БД 9](#_Toc41258848)

[2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 11](#_Toc41258849)

[3. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ 25](#_Toc41258850)

[3.1 Таблица «Виды работ» 27](#_Toc41258851)

[3.2 Таблица «Соискатели» 28](#_Toc41258852)

[3.3 Таблица «Работодатели» 28](#_Toc41258853)

[3.4 Таблица «Вакансии» 29](#_Toc41258854)

[3.5 Таблица «Сделки» 29](#_Toc41258855)

[3.6 Поиск записей 30](#_Toc41258856)

[3.7 Экспорт таблицы Excel 33](#_Toc41258857)

[3.8 Отчет по сделкам 34](#_Toc41258858)

[ЗАКЛЮЧЕНИЕ 36](#_Toc41258859)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 37](#_Toc41258860)

[ПРИЛОЖЕНИЕ А. СКРИПТ БАЗЫ ДАННЫХ 38](#_Toc41258861)

[ПРИЛОЖЕНИЕ Б. ПЕРЕЧЕНЬ ФАЙЛОВ 45](#_Toc41258862)

# ВВЕДЕНИЕ

Данная пояснительная записка описывает процесс создание базы данных, организации логики базы данных, создания приложения, работающего с этой базой данных.

В первый этап разработки входит ознакомление с исходным заданием, построение инфологических моделей «как есть» и «как должно быть». Также в этот этап входит составление модели данных в нотации IDEF1x, составления вопросов и требований того, что должна делать система.

Во второй этап разработки входит описание процесса разработки физической модели базы данных SQL, создание представлений, хранимых процедур, наполнения базы данных и выполнения запросов.

В третий этап разработки входит описание работы разработанного приложения для работы с базой данных, - работа с таблицами, добавление, удаление и редактирование записей таблиц, поиск записей в базе данных и отчетность.

ОСНОВНАЯ ЧАСТЬ

# 1 СИСТЕМНЫЙ АНАЛИЗ

ЗАДАНИЕ «БЮРО ПО ТРУДОУСТРОЙСТВУ»

## 1.1Описание предметной области

Вы работаете в бюро по трудоустройству. Вашей задачей является отслеживание финансовой стороны работы компании.

Деятельность бюро организована следующим образом: бюро готово искать работников для различных работодателей и вакансии для ищущих работу специалистов различного профиля. При обращении к вам клиента-работодателя его стандартные данные (название, вид деятельности, адрес, телефон) фиксируются в базе данных. При обращении к вам клиента-соискателя его стандартные данные (фамилия, имя, отчество, квалификация, профессия, иные данные) также фиксируются в базе данных. По каждому факту удовлетворения интересов обеих сторон составляется документ. В документе указываются соискатель, работодатель, должность и комиссионные (доход бюро).

Таблицы:

Работодатели (Код работодателя, Название, Вид деятельности, Адрес, Телефон).

Соискатели (Код соискателя, Фамилия, Имя, Отчество, Квалификация, Вид деятельности, Иные данные, Предполагаемый размер заработной платы).

Сделки (Код соискателя, Код работодателя, Должность, Комиссионные).

Развитие постановки задачи:

Оказалось, что база данных не совсем точно описывает работу бюро. В базе фиксируется только сделка, а информация по открытым вакансиям не хранится. Кроме того, для автоматического поиска вариантов необходимо вести справочник «Виды деятельности».

Внести в структуру таблиц изменения, учитывающие эти факты, и изменить существующие запросы. Добавить новые запросы.

В этапе системного анализа проводится осмысление поставленной задачи. Далее необходимо составить инфологические модели

## 1.2 Логическая модель «Как есть»

* Работодатель (Код работодателя, Название, Вид деятельности, Адрес, Телефон).
* Соискатель (Код соискателя, Фамилия, Имя, Отчество, Квалификация, Вид деятельности, Иные данные, Предполагаемый размер заработной платы).
* Сделка (Код соискателя, Код работодателя, Должность, Комиссионные). (Рисунок 1.2)

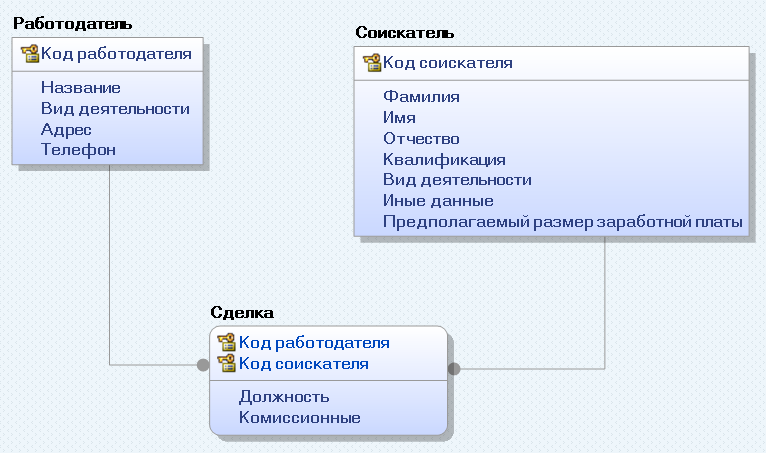


Рисунок 1.1 – Построение модели «как есть» в нотации IDEF1x

Исходя из анализа развития постановки задачи можно сделать вывод, что:

* Появляется новая сущность «Вакансия» и «Вид деятельности»;
* Сущность «Вакансия» уточняет все вакансии, нужные работодателям, туда входят атрибуты: Код вакансии (уникальный), Должность, Открытость, Зарплата;
* Сущность «Вид деятельности» хранит в себе Код деятельности (уникальный) и Наименование деятельности;
* Между сущностями «Работодатель» - «Вид деятельности» и «Вид деятельности» - «Соискатель» устанавливается неспецифичная связь (многие-ко-многим);
* Между сущностями «Работодатель» - «Вакансия», «Вакансия» - «Сделка», «Соискатель» - «Сделка» устанавливается идентифицирующая;

В таком случае, получим:

## 1.3 Логическая модель «Как должно быть»

* Работодатель (Код работодателя, Код деятельности, Название, Адрес, Телефон);
* Вакансия (Код вакансии, Код работодателя, Должность, Открытость, Зарплата);
* Соискатель (Код соискателя, Код деятельности, Фамилия, Имя, Отчество, Квалификация, Иные данные, Предполагаемый размер заработной платы)
* Вид деятельности (Код деятельности**,** Наименование деятельности)
* Сделка (Код вакансии, Код работодателя, Код сделки, Код соискателя, Комиссионные, Дата сделки)

Построение модели в нотации IDEF1x выполнено в программе Erwin Data Modeler 2018 R1 (Рисунок 1.2).

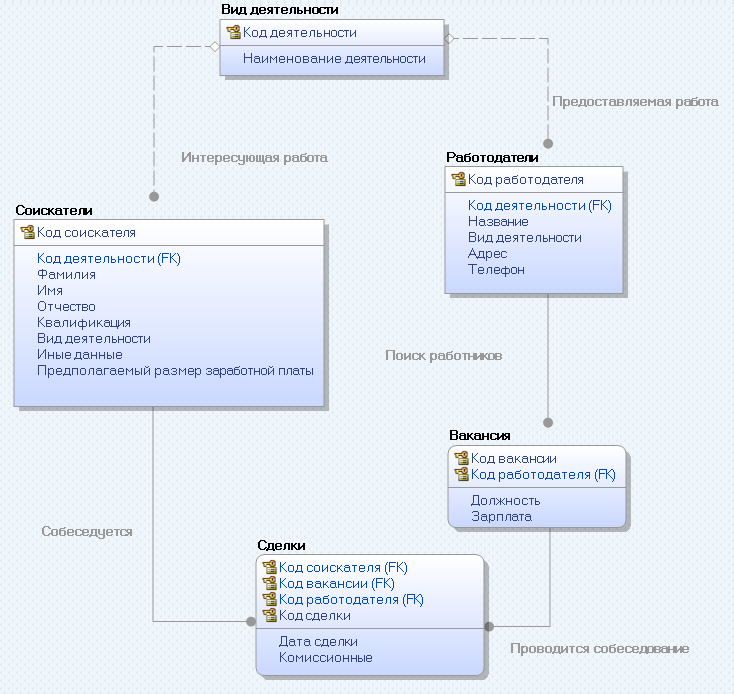


Рисунок 1.2 – Построение модели в нотации IDEF1x

## 1.4 Список запросов, которые должны быть реализованы в БД

1. Вывести список сделок с работодателем заключенных за последние два года?
2. Вывести должность с самой высокой зарплатой?
3. Вывести число соискателей, указывающих предполагаемый размер заработной платы свыше 50 тыс. руб.?
4. Вывести самую популярную должность, на которую устроено больше всего сотрудников?
5. Сколько программистов устроилось в компанию?
6. Сколько комиссионных было выплачено со сделок в этом году?
7. Кто из сотрудников владеет разговорным английским? (поле «Иные данные» = Разговорный английский)
8. У какого сотрудника самая высокая зарплата? Так же вывести его должность и квалификацию.
9. Какие компании нанимают программистов?
10. Какая должность была самые популярные позиции за последние пять лет?
11. Сколько в компании тестировщиков?
12. Какие соискатели имеют квалификацию «Аспирант»?
13. Сколько людей со средним специальным образованием устроилось в компанию?
14. Кто работает в компании аналитиком?
15. В какой компании работает Рогов Игорь и какая у него должность?

1.5 Набор требований, которым должно отвечать разрабатываемое приложение

* Обеспечить ведения списков всех соискателей;
* Обеспечить ведения списков работодателей;
* Обеспечить ведения списков вакансий;
* Обеспечить ведения списков типов работ;
* Обеспечить ведения списков сделок;
* Обеспечить возможность сортировки по годам, ФИО, зарплате, коду вакансии и коду сделки;
* Обеспечить возможность поиска записей в базе данных;
* Обеспечить выгрузку данных в таблицы Excel;
* Обеспечить возможность просмотра выплаты комиссионных по сделкам.

# 2 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Основная задача данного этапа заключается в преобразовании инфологической модели в физическую модель базы данных, преобразовании ее в схему базы данных для MS SQL Server 2008, используя Transact-SQL, проверки корректности сделанных преобразований и соответствию исходных требований, а также проектированию приложения к базам данных.

Физическая модель представлена на рисунке 2.1.

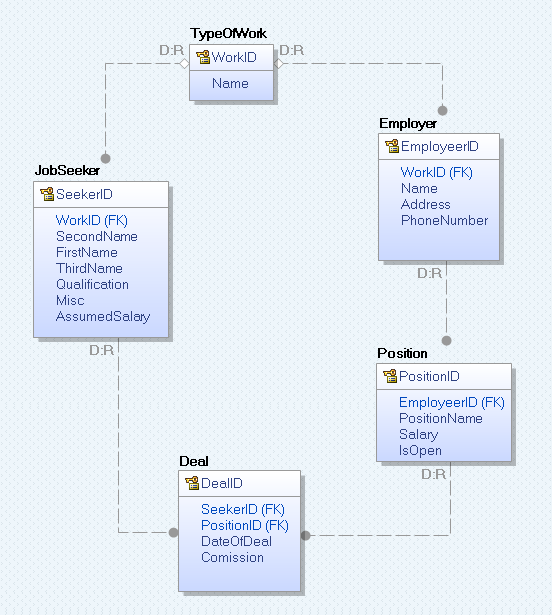


Рисунок 2.1 – Физическая модель

На рисунке 2.2 представлена физическая модель со всеми типами данных каждой таблицы.

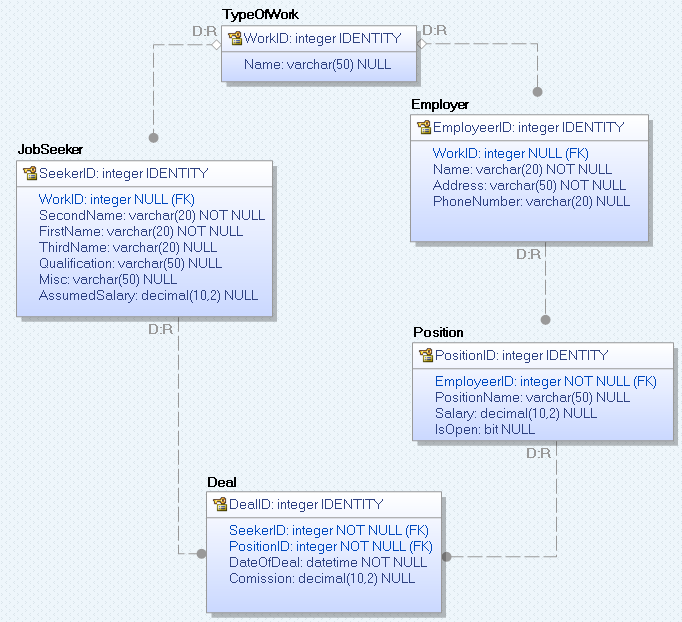


Рисунок 2.2 – Физическая модель со всеми типами данных

Представление базовой информации о соискателе показано на рисунке 2.3.

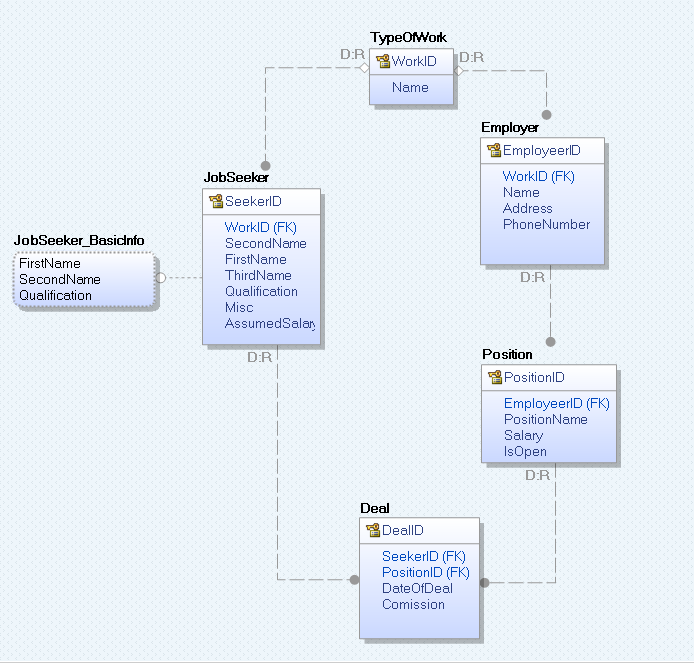


Рисунок 2.3 – Представление JobSeeker\_BasicInfo

После создания физической модели переносим ее в базу данных SQL с помощью Forward Engineering и заполним ее.

Далее представлены таблицы базы данных и запросы по заполнению таблиц.

Запрос SQL на добавление записи в таблицу TypeOfWork

INSERT INTO TypeOfWork (Name)

VALUES ('ИМЯ РАБОТЫ')

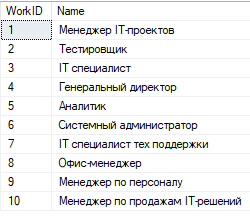


Рисунок 2.4 – Таблица Видов работ (TypeOfWork)

Запрос SQL на добавление записи в таблицу JobSeeker:

INSERT INTO JobSeeker (FirstName, SecondName, ThirdName, Qualification, AssumedSalary, Misc, WorkID)

VALUES ('ИМЯ', 'ФАМИЛИЯ', 'ОТЧЕСТВО', 'КВАЛИФИКАЦИЯ', 'ПРЕДПОЛАГАЕМАЯ ЗП', 'ПРОЧЕЕ', 'НОМЕР РАБОТЫ')

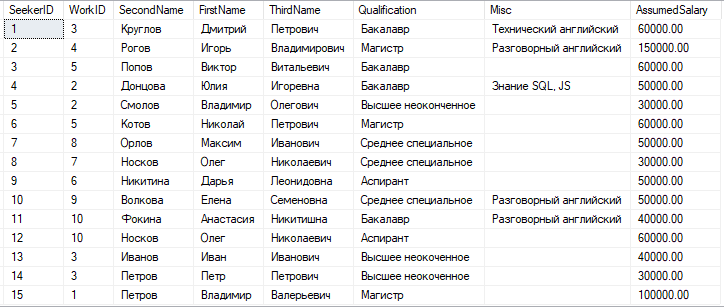


Рисунок 2.5 – Таблица соискателей (JobSeeker)

Запрос SQL на добавление записи в таблицу Employer

INSERT INTO Employer (Name,Address,PhoneNumber,WorkID)

values ( 'ИМЯ или НАЗВАНИЕ' , 'АДРЕС' , 'НОМЕР ТЕЛЕФОНА' , 'НОМЕР РАБОТЫ')

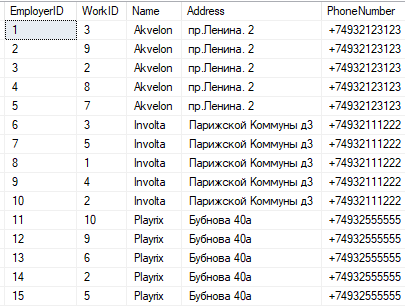


Рисунок 2.6 – Таблица работодателей (Employer)

Запрос SQL на добавление записи в таблицу Position

INSERT INTO Position (PositionName, IsOpen, Salary, EmployerID)

VALUES ('НАЗВАНИЕ ВАКАНСИИ, 'ОТКРЫТОСТЬ' ,'ЗАРПЛАТА' , 'НОМЕР РАБОТОДАТЕЛЯ')



Рисунок 2.7 – Таблица позиций в организациях (Position)

Запрос SQL на добавление записи в таблицу Deal

INSERT INTO Deal ( Commission, PositionID, SeekerID, DateOfDeal )

VALUES ('КОММИССИОННЫЕ', 'НОМЕР ВАКАНСИИ', 'НОМЕР СОИСКАТЕЛЯ', 'ДАТА СДЕЛКИ')

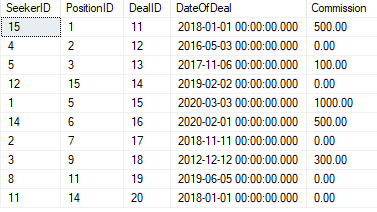


Рисунок 2.8 – Таблица сделок о приеме на работу (Deal)

Далее проверяем работу представлений (Рисунок 11-12).

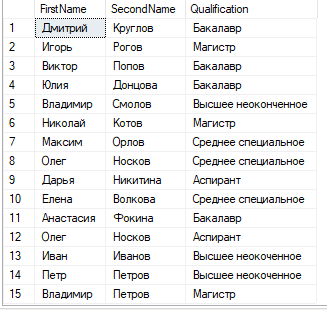


Рисунок 2.9 – Вывод представления JobSeeker\_BasicInfo

При работе с базой данных и представлениями, стало ясно, что JobSeeker\_BasicInfo показывает недостаточно информации о том, кем хотят работать соискатели. Поэтому с помощью конструктора представлений создаем представление JobSeeker\_NameOfWork\_Info, которое показывает информацию о соискателе, его квалификацию и вакансию на которую он бы хотел устроиться. (Рисунок 2.10)

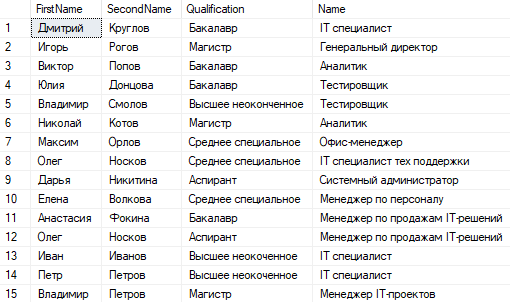


Рисунок 2.10 – Представление JobSeeker\_NameOfWork\_Info

Создадим ограничение для таблицы Deal. Нам нужно чтобы нельзя было добавить запись о приеме на работу позднее нынешней даты.

Создание ограничения:

ALTER TABLE Deal

ADD CONSTRAINT check\_date\_of\_deal CHECK (DateOfDeal < GETDATE() OR DateOfDeal is Null)

Проверка ограничения:

INSERT INTO Deal (DateOfDeal,Commission,PositionID,SeekerID)

VALUES ('01.01.2023',1,1,1)

Вывод:

Конфликт инструкции INSERT с ограничением CHECK "check\_date\_of\_deal". Конфликт произошел в базе данных "Vitya", таблица "dbo.Deal", column 'DateOfDeal'.

Выполнение данной инструкции было прервано.

Далее создаем хранимую процедуру CreateEmpPos, которая создает нового работодателя и вакансию от него, запрос на создание процедуры представлен ниже.

CREATE PROCEDURE CreateEmpPos

(

@EMPLOYER\_Name varchar(20),

@EMPLOYER\_Address varchar(50),

@EMPLOYER\_PhoneNumber varchar(20),

@WorkID int,

@PositionName varchar(50),

@POSITION\_IsOpen bit,

@POSITION\_Salary decimal(10,2)

)

AS

BEGIN

INSERT INTO Employer (Name,Address,PhoneNumber,WorkID)

VALUES (@EMPLOYER\_Name,@EMPLOYER\_Address,@EMPLOYER\_PhoneNumber,@WorkID)

INSERT INTO Position (PositionName,IsOpen,Salary,EmployerID)

VALUES (@PositionName,@POSITION\_IsOpen,@POSITION\_Salary,SCOPE\_IDENTITY())

END

RETURN

Запрос на инициализацию процедуры:

exec CreateEmpPos 'Google', 'г.Москва ул.Пушкина', '+12345678910', 2, 'Тестировщик', 1, 90000

После выполнения процедуры посмотрим добавленную запись в таблице (Рисунок 2.11).

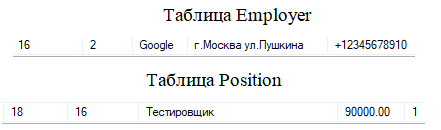


Рисунок 2.11 – Новый записи добавлены в таблицы Employer и Position

Далее создадим триггер SetDefaultPositionOpen для таблицы Position, который будет выставлять позиции значение 1(открытая), если при добавлении новой записи мы не указали значение в поле IsOpen.

Запрос на создание триггера:

CREATE TRIGGER SetDefaulPositionOpen ON Position

INSTEAD OF INSERT

AS

DECLARE

@PositionID AS int,

@EmployerID AS int,

@PositionName AS varchar(50),

@Salary AS decimal(10,2),

@IsOpen AS bit

SELECT @PositionID=PositionID, @EmployerID=EmployerID, @PositionName=PositionName, @Salary=Salary

FROM inserted

IF @IsOpen is NULL

BEGIN

INSERT INTO Position (IsOpen,EmployerID,PositionName,Salary)

VALUES(1,@EmployerID,@PositionName,@Salary)

END

IF @IsOpen is not Null

BEGIN

INSERT INTO Position (IsOpen,EmployerID,PositionName,Salary)

VALUES(@IsOpen, @EmployerID, @PositionName, @Salary)

END

Тестовый запрос для проверки триггера:

INSERT INTO Position (PositionName, Salary, EmployerID)

VALUES ('Тестировщик', 60000 , 10)

Проверяем таблицу Position



Рисунок 2.12 – Таблица Position с новой записью

После проверки триггера выполним запросы, которые были даны ранее (1 этап: Системный анализ):

1. Вывести список сделок с работодателем заключенных за последние два года?

SELECT COUNT(d.DealID) AS Date

FROM Deal d

WHERE d.DateOfDeal BETWEEN DATEADD(yy,-2,GETDATE()) AND GETDATE()



Рисунок 2.13 – Вывод запроса

1. Вывести должность с самой высокой зарплатой?

SELECT p.PositionName,p.Salary

FROM Position P

WHERE p.Salary=(SELECT MAX(Salary) FROM Position )



Рисунок 2.14 – Вывод запроса

1. Вывести число соискателей, указывающих предполагаемый размер заработной платы свыше 50 тыс. руб.?

SELECT COUNT(j.AssumedSalary) AS Number

FROM JobSeeker j

WHERE j.AssumedSalary>'50000'



Рисунок 2.15 – Вывод запроса

1. Вывести самую популярную должность, на которую устроено больше всего сотрудников?

SELECT b.PositionName

FROM (SELECT p.PositionName, COUNT(p.PositionName) AS postcount

FROM Position p

GROUP BY p.PositionName)AS b

WHERE b.postcount IN (SELECT MAX(b.postcount)

FROM (SELECT p.PositionName, COUNT(p.PositionName)AS postcount

FROM Position p

GROUP BY p.PositionName)AS b)



Рисунок 2.16 – Вывод запроса

1. Сколько программистов устроилось в компанию?

SELECT p.PositionID, j.SeekerID, j.FirstName, j.SecondName, j.Qualification

FROM JobSeeker j

JOIN Deal d ON d.SeekerID=j.SeekerID

JOIN Position p ON p.PositionID=d.PositionID

WHERE p.PositionName='IT специалист'



Рисунок 2.17 – Вывод запроса

1. Сколько комиссионных было выплачено со сделок в этом году?

SELECT SUM(d.Commission) AS Commision

FROM Deal d

WHERE YEAR(d.DateOfDeal)= YEAR(GETDATE())



Рисунок 2.18 – Вывод запроса

1. Кто из сотрудников владеет разговорным английским? (поле «Иные данные» = Разговорный английский)

SELECT j.FirstName, j.SecondName, j.Qualification, j.Misc

FROM JobSeeker j

WHERE j.Misc LIKE '%Разговорный английский%'

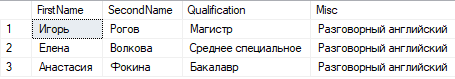


Рисунок 2.19 – Вывод запроса

1. У какого сотрудника самая высокая зарплата? Так же вывести его должность и квалификацию.

SELECT j.FirstName, j.SecondName, p.PositionName, j.Qualification, p.Salary

FROM Position P

JOIN Deal d ON d.PositionID=p.PositionID

JOIN JobSeeker j ON j.SeekerID=d.SeekerID

WHERE p.Salary =(SELECT MAX(Salary) FROM Position)



Рисунок 2.20 – Вывод запроса

1. Какие компании нанимают программистов?

SELECT e.Name, p.PositionName, p.IsOpen

FROM Employer e

JOIN Position p ON p.EmployerID=e.EmployerID

WHERE p.PositionName LIKE 'Тестировщик' AND p.IsOpen=1



Рисунок 2.21 – Вывод запроса

1. Какая должность была самые популярные позиции за последние пять лет?

SELECT b.PositionName FROM (SELECT p.PositionName, COUNT(p.PositionName)AS postcount FROM Position p

JOIN Deal d ON d.PositionID=p.PositionID

WHERE d.DateOfDeal BETWEEN DATEADD(yy,-5,GETDATE()) AND DATEADD(yy,0,GETDATE())

GROUP BY p.PositionName)AS b

WHERE b.postcount IN (SELECT MAX(b.postcount) FROM (SELECT p.PositionName, COUNT(p.PositionName)AS postcount FROM Position p

JOIN Deal d ON d.PositionID=p.PositionID

WHERE d.DateOfDeal BETWEEN DATEADD(yy,-5,GETDATE()) AND DATEADD(yy,0,GETDATE())

GROUP BY p.PositionName)AS b)



Рисунок 2.22 – Вывод запроса

1. Сколько в компании тестировщиков?

SELECT e.Name, COUNT(j.SeekerID) AS Count

FROM JobSeeker j

JOIN Deal d ON d.SeekerID=j.SeekerID

JOIN Position p ON p.PositionID=d.PositionID

JOIN Employer e ON e.EmployerID=p.EmployerID

WHERE p.PositionName='Тестировщик' AND p.IsOpen=0

GROUP BY e.Name



Рисунок 2.23 – Вывод запроса

1. Какие соискатели имеют квалификацию «Аспирант»?

SELECT \*

FROM JobSeeker j

WHERE j.Qualification='Аспирант'



Рисунок 2.13 – Вывод запроса

1. Сколько людей со средним специальным образованием устроилось в компанию?

Select COUNT(j.Qualification) AS Count FROM JobSeeker j

JOIN Deal d ON d.SeekerID=j.SeekerID

WHERE j.Qualification='Среднее специальное'



Рисунок 2.25 – Вывод запроса

1. Кто работает в компании аналитиком?

SELECT e.Name, j.FirstName, j.SecondName FROM JobSeeker j

JOIN Deal d ON d.SeekerID=j.SeekerID

JOIN Position p ON p.PositionID=d.PositionID

JOIN Employer e ON e.EmployerID=p.EmployerID

WHERE p.PositionName='Аналитик'



Рисунок 2.26 – Вывод запроса

1. В какой компании работает Рогов Игорь и какая у него должность?

SELECT j.FirstName, j.SecondName, e.Name, p.PositionName

FROM JobSeeker j

JOIN Deal d ON d.SeekerID=j.SeekerID

JOIN Position p ON p.PositionID=d.PositionID

JOIN Employer e ON e.EmployerID=p.EmployerID

WHERE j.SecondName='Рогов'



Рисунок 2.27 – Вывод запроса

# 3. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

В данном этапе было необходимо спроектировать приложение-ассистент для быстрой работы с базой данных. Проектирование приложения основывалось по следующим требованиям:

* Обеспечить ведения списков всех соискателей;
* Обеспечить ведения списков работодателей;
* Обеспечить ведения списков вакансий;
* Обеспечить ведения списков типов работ;
* Обеспечить ведения списков сделок;
* Обеспечить возможность сортировки по годам, ФИО, зарплате, коду вакансии и коду сделки;
* Обеспечить возможность поиска записей в базе данных;
* Обеспечить выгрузку данных в таблицы Excel;
* Обеспечить возможность просмотра выплаты комиссионных по сделкам.

Главное окно приложения представлено на рисунке 3.1.

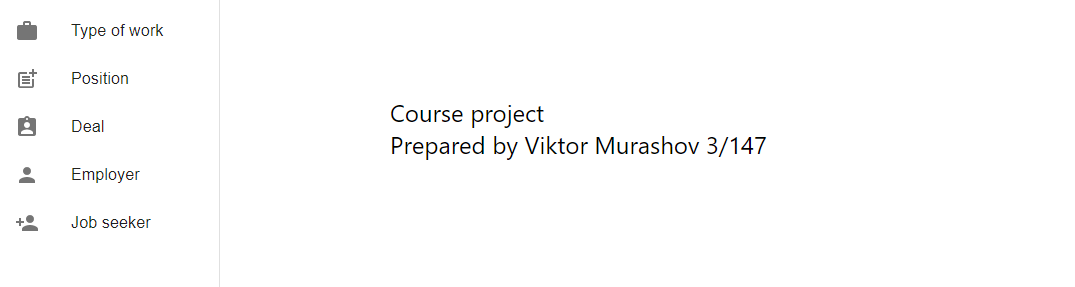


Рисунок 3.1 – Главное окно приложения

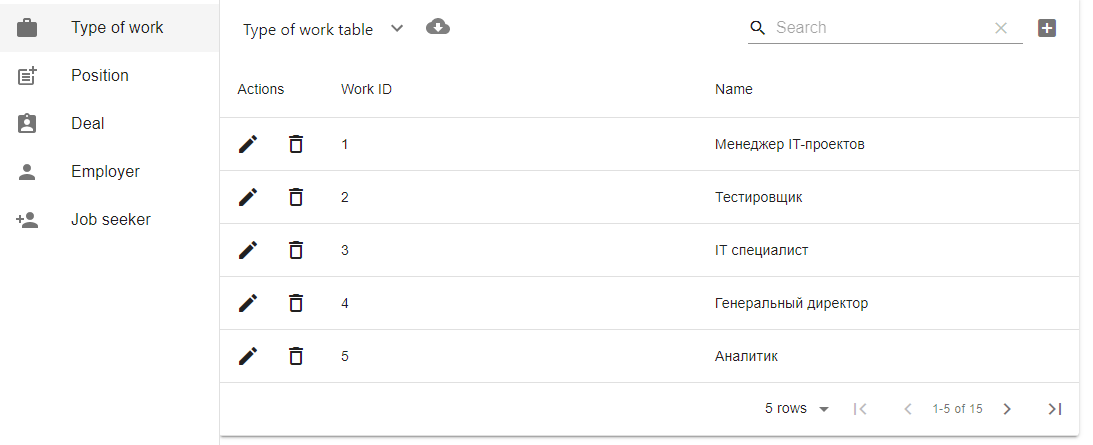
Нажимая кнопки в левой части приложения, можно переключаться между таблицами (Рисунки 3.2-3.6).

Рисунок 3.2 – Таблица «Виды работ»

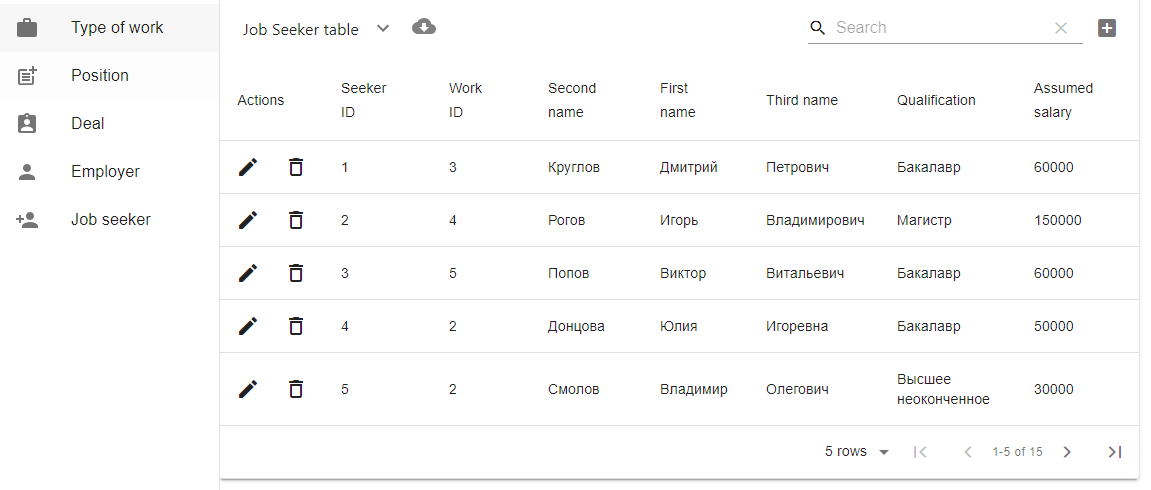


Рисунок 3.3 – Таблица «Соискатели»

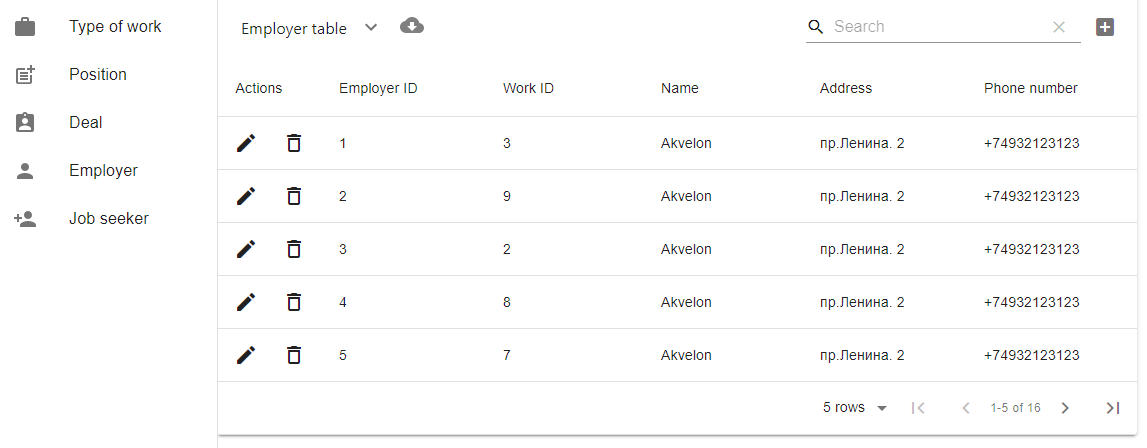


Рисунок 3.4 – Таблица «Работодатели»

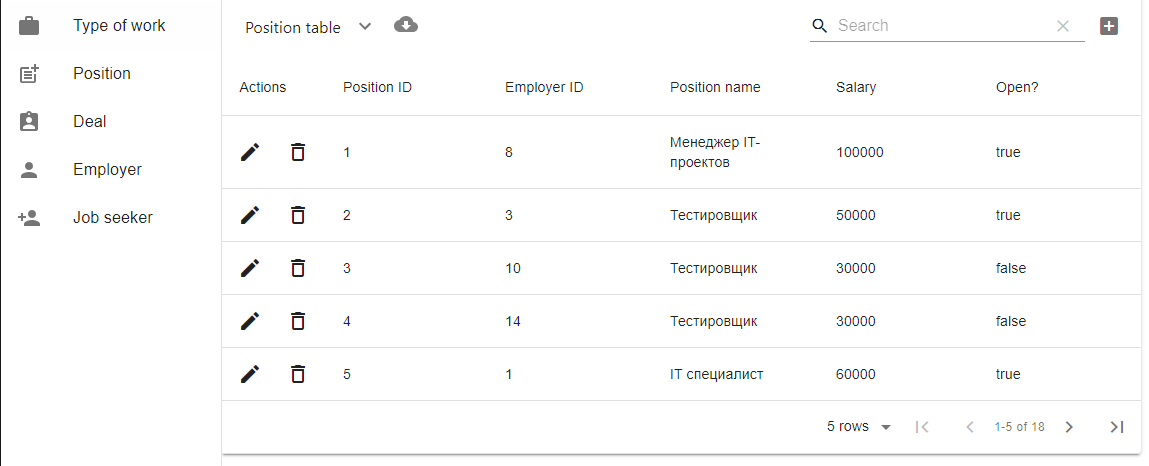


Рисунок 3.5 – Таблица «Вакансии»

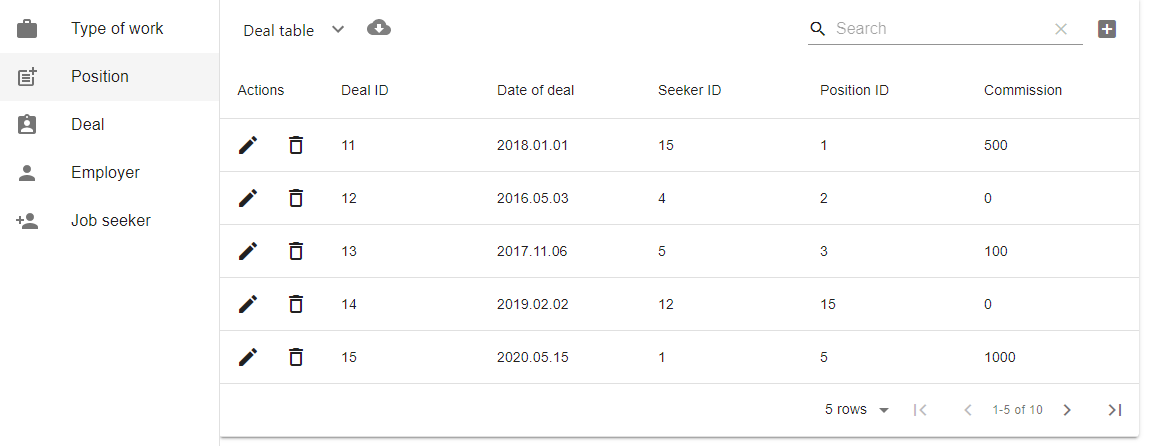


Рисунок 3.6 – Таблица «Сделки»

Далее рассмотрим подробнее вид окон и работу с разными таблицами:

## 3.1 Таблица «Виды работ»

Кнопка «+» в правом верхнем углу добавляет новую запись в таблицу (Рисунок 3.7).



Рисунок 3.7 – Добавление типа работы

Кнопка в виде «карандаша» загружает в форму «Типы работы» данные строки, чтобы ее изменить (Рисунок 3.8).

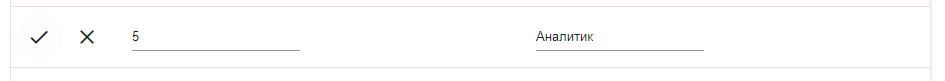


Рисунок 3.8 – Изменение типа работы

Кнопка в виде «Корзины» удаляет выбранную запись из таблицы (Рисунок 3.9).



Рисунок 3.9 – Удаление типа работы

## 3.2 Таблица «Соискатели»

Кнопка «+» в правом верхнем углу добавляет новую запись в таблицу (Рисунок 3.10).



Рисунок 3.10 – Добавление соискателя

Кнопка в виде «Карандаша» загружает в форму «Соискатели» данные строки, чтобы ее изменить (Рисунок 3.11).



Рисунок 3.11 – Изменение соискателя

Кнопка в виде «Корзины» удаляет выбранную запись из таблицы (Рисунок 3.12).



Рисунок 3.12 – Удаление соискателя

## 3.3 Таблица «Работодатели»

Кнопка «+» в правом верхнем углу добавляет новую запись в таблицу (Рисунок 3.13).



Рисунок 3.13 – Добавление работодателя

Кнопка в виде «Карандаша» загружает в форму «Работодатели» данные строки, чтобы ее изменить (Рисунок 3.14).



Рисунок 3.14 – Изменение работодателя

Кнопка в виде «Корзины» удаляет выбранную запись из таблицы (Рисунок 3.15).



Рисунок 3.15 – Удаление работодателя

## 3.4 Таблица «Вакансии»

Кнопка «+» в правом верхнем углу добавляет новую запись в таблицу (Рисунок 3.16).



Рисунок 3.16 – Добавление вакансии

Кнопка в виде «Карандаша» загружает в форму «Вакансии» данные строки, чтобы ее изменить (Рисунок 3.17).



Рисунок 3.17 – Изменение вакансии

Кнопка в виде «Корзины» удаляет выбранную запись из таблицы (Рисунок 3.18).



Рисунок 3.18 – Удаление вакансии

## 3.5 Таблица «Сделки»

Кнопка «+» в правом верхнем углу добавляет новую запись в таблицу (Рисунок 3.19).



Рисунок 3.19 – Добавление сделки

Кнопка в виде «Карандаша» загружает в форму «Сделки» данные строки, чтобы ее изменить (Рисунок 3.20).



Рисунок 3.20 – Изменение сделки

Кнопка в виде «Корзины» удаляет выбранную запись из таблицы (Рисунок 3.21).



Рисунок 3.21 – Удаление сделки

## 3.6 Поиск записей

Для поиска записей в базе данных, необходимо выбрать нужную таблицу, затем при заполнении строки поиска записи, будут отображаться результаты поиска. Строка поиска представлено на рисунке 3.22.



Рисунок 3.22 – Окно поиска

Для поиска записей в базе данных, также можно использовать фильтрацию, для этого необходимо сначала выбрать нужную таблицу, затем нажать на кнопку в виде “стрелки вниз”. Окна фильтров для разных таблиц представлено на рисунках 3.23-3.27.

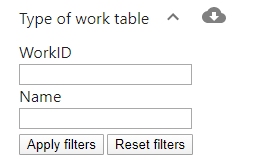


Рисунок 3.23 – Фильтры для типов работ

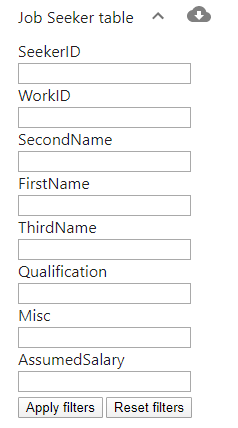


Рисунок 3.24 – Фильтры для соискателей

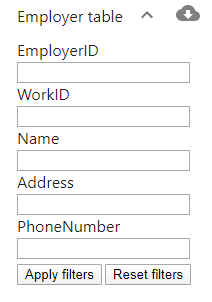


Рисунок 3.25 – Фильтры для работодателей

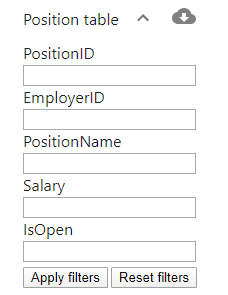


Рисунок 3.26 – Окно фильтров для вакансий

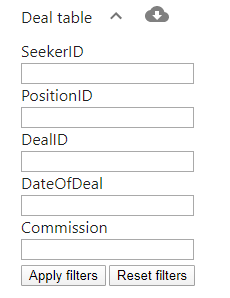


Рисунок 3.27 – Окно фильтров для сделок.

Рассмотрим пример использования фильтра. Найдем всех, у кого зарплата выше 50000у.е. и квалификация Магистр (Рисунок 3.28 и Рисунок 3.29).

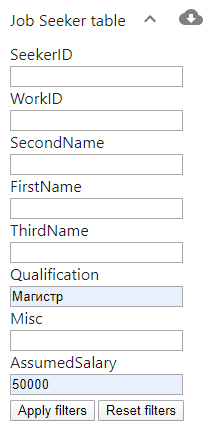


Рисунок 3.28 – Поиск магистров с зарплатой выше 50000



Рисунок 3.29 – Результат фильтрации

## 3.7 Экспорт таблицы Excel

Для выгрузки таблицы Excel необходимо сначала выбрать таблицу для сохранения, затем нажать на кнопку в виде ‘облака’, после этого сразу загрузиться, и откроется файл excel с нужной таблицей (Рисунок 3.30).

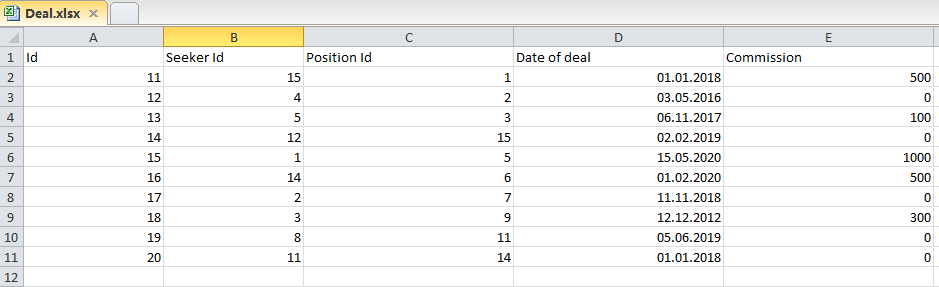


Рисунок 3.30 – Результат экспорта в Microsoft Excel

## 3.8 Отчет по сделкам

Для просмотра выплаты комиссионных по сделкам, необходимо перейти в таблицу сделок. График отчета представлен на рисунке 3.31.

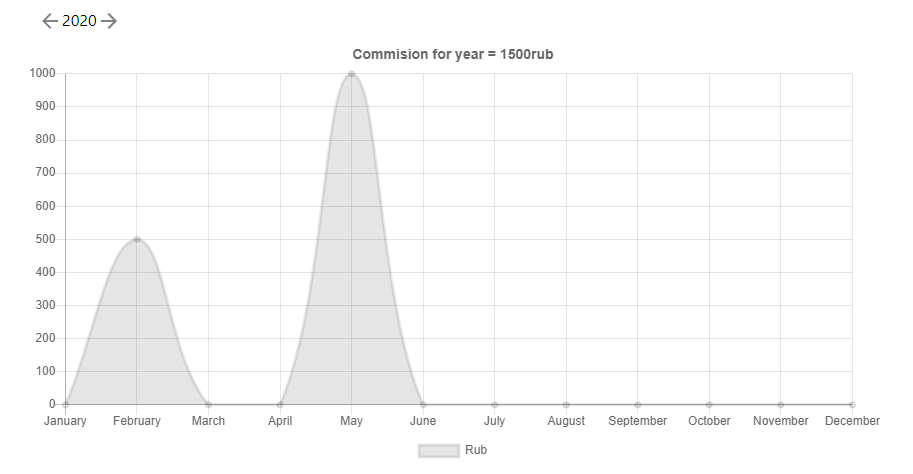


Рисунок 3.31 –График отчета по сделкам за 2020 год

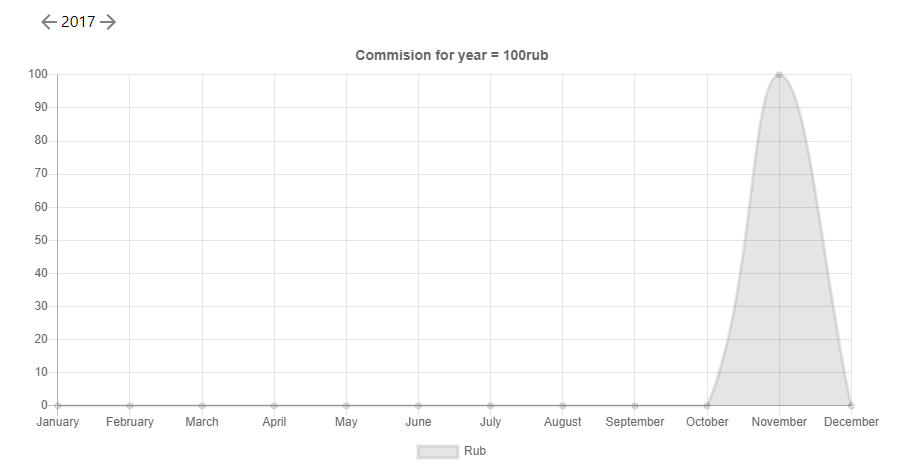
Также для изменения года можно использовать “стрелки” влево и вправо. Пример графика за 2017 год представлен на рисунке 3.32.

Рисунок 3.32 – График отчета по сделкам за 2017 год

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта по заданию «Бюро по трудоустройству» были выполнены различные задачи по анализу поставленной задачи, создания физической модели.

На основе физической модели была создана реляционная база данных, в которой дополнительно были созданы триггеры, ограничения, хранимые процедуры для корректной работы базы данных.

Помимо этого, было спроектировано приложение-ассистент для работы с данной базой данных. В этом приложении были реализованы функции просмотра, добавления, изменения и удаления записей базы данных, поиска записей, фильтрация для записей, экспорт таблиц и просмотр комиссионных в сделках.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Требования к оформлению квалификационных работ: метод. указания для студентов по направлению 230200 «Информационные системы» / Сост.: А.П.Власов, Н.А. Марчук: Иван. гос. хим.-технол. ун-т. – Иваново, 2010, 35 с.
2. Проектирование баз данных: лабораторный практикум / Э.Г. Галиаскаров, А.Ю. Крылов; Иван. гос. хим.-технол. ун-т.- Иваново, 2012.- 96 с.
3. Разработка приложений баз данных: лабораторный практикум / Э.Г. Галиаскаров и др.; Иван. гос. хим.-технол. ун-т. – Иваново, 2015. – 112 с.
4. Сайт обмена знаниями и решения проблем StackOverflow [Электронный ресурс] URL <https://stackoverflow.com/> (дата обращения 15.04.2020)
5. Упражнения по SQL [Электронный ресурс] URL <http://sql-ex.ru/> (дата обращения 15.04.2020)
6. Татьяна Карпова, курс «Базы данных: модели, разработка, реализация» Сайт «НОУ ИНТУИТ» [Электронный ресурс] URL <https://www.intuit.ru/studies/courses/1001/297/info> (дата обращения 01.04.2020)
7. Сайт с документацией по react, React [Электронный ресурс] URL <https://ru.reactjs.org/> (дата обращения 15.04.20)

# ПРИЛОЖЕНИЕ А. СКРИПТ БАЗЫ ДАННЫХ

CREATE TABLE [Deal]

(

[SeekerID] integer NOT NULL ,

[PositionID] integer NOT NULL ,

[DealID] integer NOT NULL IDENTITY ( 1,1 ) ,

[DateOfDeal] datetime NOT NULL ,

[Commission] decimal(10,2) NULL

)

ALTER TABLE [Deal]

ADD CONSTRAINT [XPKDeal] PRIMARY KEY CLUSTERED ([DealID] ASC)

CREATE TABLE [Employer]

(

[EmployerID] integer NOT NULL IDENTITY ( 1,1 ) ,

[WorkID] integer NULL ,

[Name] varchar(20) NOT NULL ,

[Address] varchar(50) NOT NULL ,

[PhoneNumber] varchar(20) NULL

)

ALTER TABLE [Employer]

ADD CONSTRAINT [XPKEmployer] PRIMARY KEY CLUSTERED ([EmployerID] ASC)

CREATE TABLE [JobSeeker]

(

[SeekerID] integer NOT NULL IDENTITY ( 1,1 ) ,

[WorkID] integer NULL ,

[SecondName] varchar(20) NOT NULL ,

[FirstName] varchar(20) NOT NULL ,

[ThirdName] varchar(20) NULL ,

[Qualification] varchar(50) NULL ,

[Misc] varchar(50) NULL ,

[AssumedSalary] decimal(10,2) NULL

)

ALTER TABLE [JobSeeker]

ADD CONSTRAINT [XPKJobSeeker] PRIMARY KEY CLUSTERED ([SeekerID] ASC)

CREATE TABLE [Position]

(

[PositionID] integer NOT NULL IDENTITY ( 1,1 ) ,

[EmployerID] integer NOT NULL ,

[PositionName] varchar(50) NULL ,

[Salary] decimal(10,2) NULL ,

[IsOpen] bit NULL

)

ALTER TABLE [Position]

ADD CONSTRAINT [XPKPosition] PRIMARY KEY CLUSTERED ([PositionID] ASC)

CREATE TABLE [TypeOfWork]

(

[WorkID] integer NOT NULL IDENTITY ( 1,1 ) ,

[Name] varchar(50) NULL

)

ALTER TABLE [TypeOfWork]

ADD CONSTRAINT [XPKTypeOfWork] PRIMARY KEY CLUSTERED ([WorkID] ASC)

CREATE VIEW [JobSeeker\_BasicInfo]([FirstName],[SecondName],[Qualification])

AS

SELECT [FirstName],[SecondName],[Qualification]

FROM [JobSeeker]

ALTER TABLE [Deal]

ADD CONSTRAINT [R\_2] FOREIGN KEY ([SeekerID]) REFERENCES [JobSeeker]([SeekerID])

ALTER TABLE [Deal]

ADD CONSTRAINT [R\_6] FOREIGN KEY ([PositionID]) REFERENCES [Position]([PositionID])

ALTER TABLE [Employer]

ADD CONSTRAINT [R\_3] FOREIGN KEY ([WorkID]) REFERENCES [TypeOfWork]([WorkID])

ALTER TABLE [JobSeeker]

ADD CONSTRAINT [R\_4] FOREIGN KEY ([WorkID]) REFERENCES [TypeOfWork]([WorkID])

ALTER TABLE [Position]

ADD CONSTRAINT [R\_5] FOREIGN KEY ([EmployerID]) REFERENCES [Employer]([EmployerID])

CREATE TRIGGER tD\_Employer ON Employer FOR DELETE AS

/\* erwin Builtin Trigger \*/

/\* DELETE trigger on Employer \*/

BEGIN

DECLARE @errno int,

@severity int,

@state int,

@errmsg varchar(255)

/\* erwin Builtin Trigger \*/

/\* Employer Поиск работников Position on parent delete restrict \*/

/\* ERWIN\_RELATION:CHECKSUM="0001141d", PARENT\_OWNER="", PARENT\_TABLE="Employer"

CHILD\_OWNER="", CHILD\_TABLE="Position"

P2C\_VERB\_PHRASE="Поиск работников", C2P\_VERB\_PHRASE="",

FK\_CONSTRAINT="R\_5", FK\_COLUMNS="EmployerID" \*/

IF EXISTS (

SELECT \* FROM deleted,Position

WHERE

/\* %JoinFKPK(Position,deleted," = "," AND") \*/

Position.EmployerID = deleted.EmployerID

)

BEGIN

SELECT @errno = 30001,

@errmsg = 'Cannot delete Employer because Position exists.'

GOTO error

END

CREATE TRIGGER tD\_JobSeeker ON JobSeeker FOR DELETE AS

/\* erwin Builtin Trigger \*/

/\* DELETE trigger on JobSeeker \*/

BEGIN

DECLARE @errno int,

@severity int,

@state int,

@errmsg varchar(255)

/\* erwin Builtin Trigger \*/

/\* JobSeeker Договариваются Deal on parent delete restrict \*/

/\* ERWIN\_RELATION:CHECKSUM="000103ab", PARENT\_OWNER="", PARENT\_TABLE="JobSeeker"

CHILD\_OWNER="", CHILD\_TABLE="Deal"

P2C\_VERB\_PHRASE="Договариваются", C2P\_VERB\_PHRASE="",

FK\_CONSTRAINT="R\_2", FK\_COLUMNS="SeekerID" \*/

IF EXISTS (

SELECT \* FROM deleted,Deal

WHERE

/\* %JoinFKPK(Deal,deleted," = "," AND") \*/

Deal.SeekerID = deleted.SeekerID

)

BEGIN

SELECT @errno = 30001,

@errmsg = 'Cannot delete JobSeeker because Deal exists.'

GOTO error

END

CREATE TRIGGER tD\_Position ON Position FOR DELETE AS

/\* erwin Builtin Trigger \*/

/\* DELETE trigger on Position \*/

BEGIN

DECLARE @errno int,

@severity int,

@state int,

@errmsg varchar(255)

/\* erwin Builtin Trigger \*/

/\* Position Проходят собеседование Deal on parent delete restrict \*/

/\* ERWIN\_RELATION:CHECKSUM="00010cb8", PARENT\_OWNER="", PARENT\_TABLE="Position"

CHILD\_OWNER="", CHILD\_TABLE="Deal"

P2C\_VERB\_PHRASE="Проходят собеседование", C2P\_VERB\_PHRASE="",

FK\_CONSTRAINT="R\_6", FK\_COLUMNS="PositionID" \*/

IF EXISTS (

SELECT \* FROM deleted,Deal

WHERE

/\* %JoinFKPK(Deal,deleted," = "," AND") \*/

Deal.PositionID = deleted.PositionID

)

BEGIN

SELECT @errno = 30001,

@errmsg = 'Cannot delete Position because Deal exists.'

GOTO error

END

CREATE TRIGGER tD\_TypeOfWork ON TypeOfWork FOR DELETE AS

/\* erwin Builtin Trigger \*/

/\* DELETE trigger on TypeOfWork \*/

BEGIN

DECLARE @errno int,

@severity int,

@state int,

@errmsg varchar(255)

/\* erwin Builtin Trigger \*/

/\* TypeOfWork Интересующая работа JobSeeker on parent delete restrict \*/

/\* ERWIN\_RELATION:CHECKSUM="00021280", PARENT\_OWNER="", PARENT\_TABLE="TypeOfWork"

CHILD\_OWNER="", CHILD\_TABLE="JobSeeker"

P2C\_VERB\_PHRASE="Интересующая работа", C2P\_VERB\_PHRASE="",

FK\_CONSTRAINT="R\_4", FK\_COLUMNS="WorkID" \*/

IF EXISTS (

SELECT \* FROM deleted,JobSeeker

WHERE

/\* %JoinFKPK(JobSeeker,deleted," = "," AND") \*/

JobSeeker.WorkID = deleted.WorkID

)

BEGIN

SELECT @errno = 30001,

@errmsg = 'Cannot delete TypeOfWork because JobSeeker exists.'

GOTO error

END

/\* erwin Builtin Trigger \*/

/\* TypeOfWork Предоставление работы Employer on parent delete restrict \*/

/\* ERWIN\_RELATION:CHECKSUM="00000000", PARENT\_OWNER="", PARENT\_TABLE="TypeOfWork"

CHILD\_OWNER="", CHILD\_TABLE="Employer"

P2C\_VERB\_PHRASE="Предоставление работы", C2P\_VERB\_PHRASE="",

FK\_CONSTRAINT="R\_3", FK\_COLUMNS="WorkID" \*/

IF EXISTS (

SELECT \* FROM deleted,Employer

WHERE

/\* %JoinFKPK(Employer,deleted," = "," AND") \*/

Employer.WorkID = deleted.WorkID

)

BEGIN

SELECT @errno = 30001,

@errmsg = 'Cannot delete TypeOfWork because Employer exists.'

GOTO error

END

# ПРИЛОЖЕНИЕ Б. ПЕРЕЧЕНЬ ФАЙЛОВ

1. Пояснительная записка к курсовому проекту в папке doc Пояснительная записка.docx
2. Инфологическая модель в папке erwin logicalModel.erwin
3. Физическая модель базы данных в папке erwin physicalModel.erwin
4. Файлы базы данных в папке db Recruting.mdf
5. Папки с исходным кодом приложения Server и Client

<https://github.com/viktormurashov/project_X.git>/tree/dev