ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

Кафедра программной и системной инженерии

**Отчет по предмету «Программная инженерия»**

**Тема: Управляющая компания в сфере ЖКХ**

Выполнил:

студент группы ПИ 155-2

Кочнева Е.В.

Проверил:

ассистент

Красиков В.Е.

Оглавление

[Описание предметной области 3](#_Toc533368599)

[Разработка базы данных 5](#_Toc533368600)

[Разработка веб-службы 11](#_Toc533368601)

[Реализация клиентского приложения 13](#_Toc533368602)

[Заключение 14](#_Toc533368603)

[Список использованной литературы 15](#_Toc533368604)

# Описание предметной области

**Управляющая компания (УК) в сфере жилищно-коммунального хозяйства (ЖКХ) – это организация, созданная для осуществления управленческой деятельности многоквартирным домом, обеспечения его необходимыми ресурсами, поддерживающими надлежащее санитарное и техническое состояние объекта.**

**По большому счёту, УК ЖКХ являются посредниками между жильцами дома и компаниями, предоставляющими услуги населению. С технической стороны управляющие компании осуществляют накопление информации об объекте управления, о существующих проблемах технического характера, ведут учёт износа и амортизации имущества, планируют и организуют мероприятия по ремонту и восстановлению зданий и сооружений. Если рассматривать правовой аспект деятельности УК ЖКХ, то между ними и собственниками жилья складываются потребительские отношения, в которых такие компании выступают первыми подрядчиками со всеми вытекающими из гражданского законодательства обязанностями и юридическими последствиями.**

**Как и у любой организации в гражданских правоотношениях, у УК ЖКХ есть свои определённые обязанности перед собственниками жилья в доме, который находится на обслуживании. Например, такие компании обязаны проводить следующие работы:**

* **поддерживать техническое состояние дома в надлежащем виде, делая его безопасным для жильцов;**
* **проверять состояние вверенного имущества раз в полгода: крыша, чердак, подвал, системы водоотведения и канализации и т.д. Все обнаруженные проблемы должны документироваться и устраняться в разумные сроки;**
* **выполнять мелкий и текущий ремонт зданий;**
* **подготавливать дома и прилегающие территории к зиме (весне);**
* **устранять неполадки и аварии в доме;**
* **создавать благоприятные условия для предоставления гражданам коммунальных услуг соответствующего качества;**
* **контролировать соблюдение норм и правил пожарной безопасности.**

**Не стоит забывать и про организационные обязанности УК ЖКХ. К таким можно отнести регистрацию граждан в квартирах; проведение собраний жильцов для принятия решений касаемо дома, уведомления их о существующих проблемах и общей информации в сфере ЖКХ.**

# Разработка базы данных

База данных – это совокупность данных, организованная по определенным правилам, предусматривающая общие принципы описания, хранения, манипулирования данными, независимыми от прикладных программ.

На первом этапе проектирования базы данных необходимо определить цель создания базы данных, основные ее функции и информацию, которую она должна содержать. Воспользуемся методологией IDEF1x.

Методология моделирования IDEF1x, является расширением стандарта IDEF1 и предназначена для описания данных (информации). В ее основе лежит язык семантического моделирования, основанного на концепции "сущность — связь", позволяющей определять данные и связи между ними. Методология используется для создания информационной модели предметной области с помощью идентификации ее сущностей и связей между ними. Чаще всего такая методология используется для описания данных в целях последующей автоматизации их обработки с помощью систем управления базами данных. Связи в IDEF1X представляют собой ссылки, соединения и ассоциации между сущностями. Связи – это глаголы, которые показывают, как соотносятся сущности между собой. Диаграмма IDEF1x (см. рис. 1), разработанная для текущего проекта – управляющей компании в сфере ЖКХ, включает в себя в т.ч. следующие связи между сущностями:

* житель может создавать несколько заявок жильцов;
* должность содержит список требуемого оборудования;
* сотрудник может несколько раз уходить в отпуск;
* в отделе имеется несколько должностей;
* и т.д.

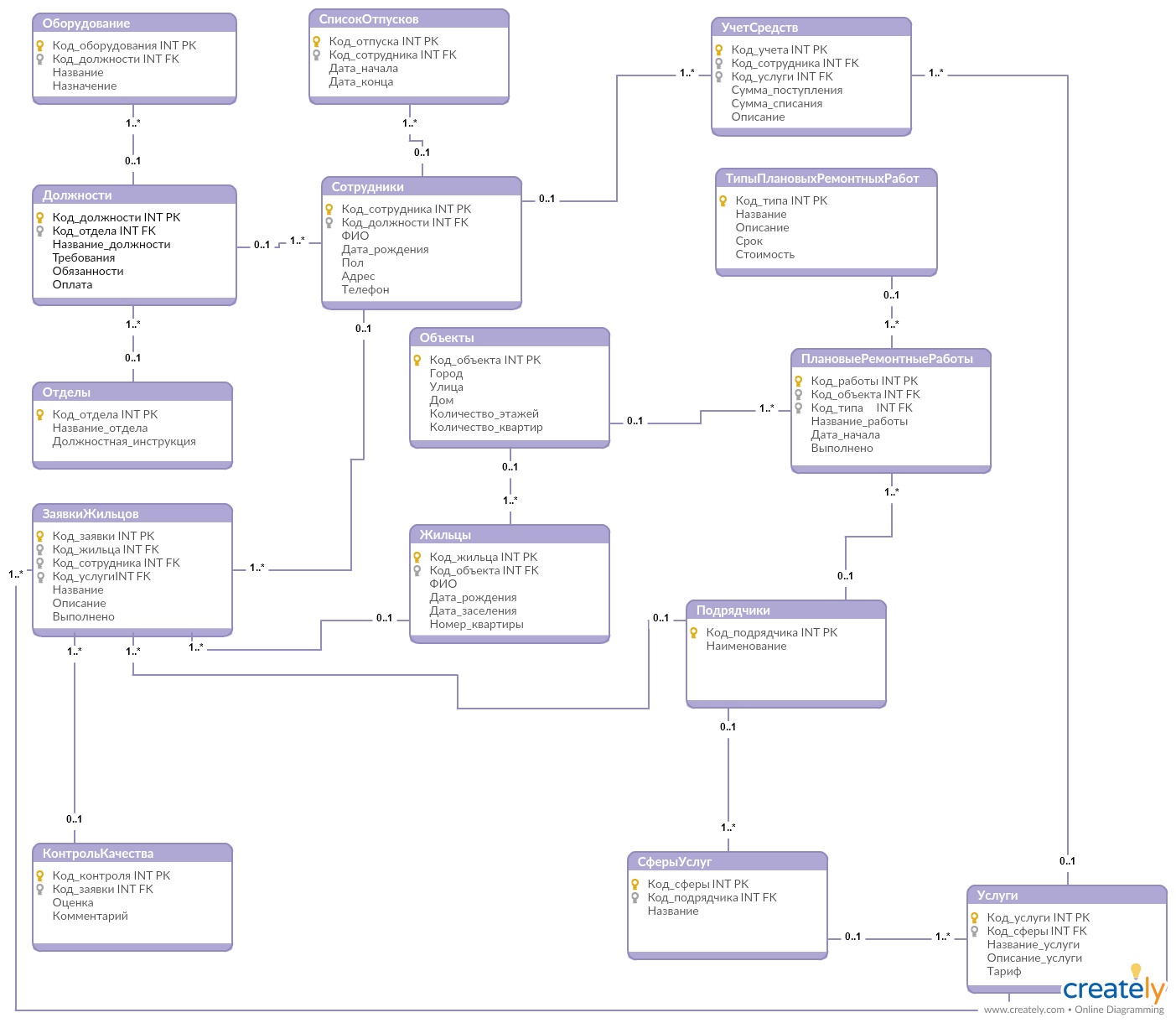


Рисунок 1. Диаграмма IDEF1x

После разработки диаграммы IDEF1x требуется создать базу данных с помощью специального программного обеспечения.

СУБД – система управления базами данных – программное обеспечение, предназначенное для управления БД и возможности получения пользователями необходимой информации из базы. В задачи СУБД входят:

* формирование и поддержание БД;
* обработка информации;
* прием запросов;
* предоставление информации пользователям;
* обеспечение целостности и реорганизации ценностей БД;
* организация совместной работы пользователей.

На сегодняшний день существует множество различных систем управления базами данных. Все они используют разные средства и функции, но преимущественно у всех СУБД в основе лежат одинаковые понятия.

Мной выбрана СУБД MS SQL Server, т.к. приложение разрабатывается на языке С# и продукт MS SQL наиболее оптимально подходит для данной задачи.

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

База данных (см. рис. 2), созданная в системе управления базами данных MS SQL, включает в себя 15 таблиц, описанных в диаграмме IDEF1x.

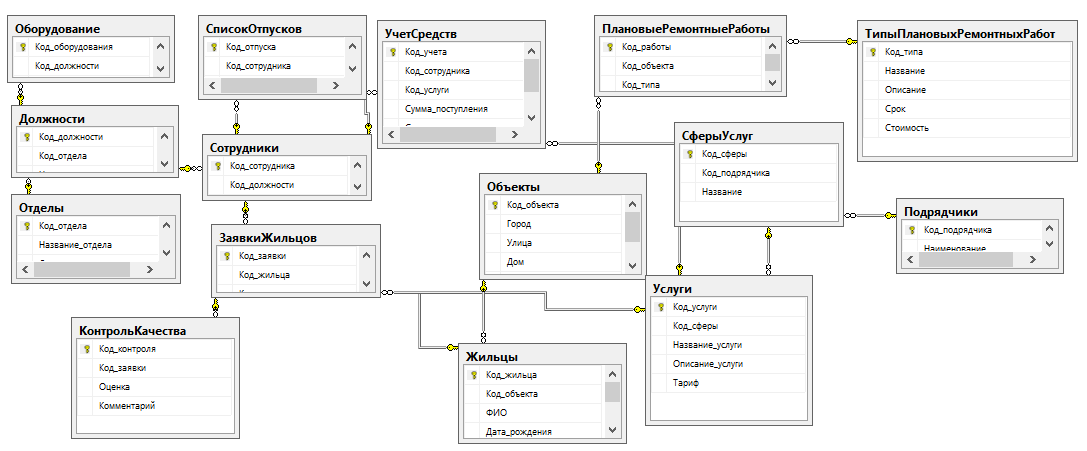


Рисунок 2. Диаграмма базы данных

В итоге, созданы и заполнены данными следующие таблицы:

* должности (см. рис. 3);
* жильцы (см. рис. 4);
* заявки жильцов (см. рис. 5);
* контроль качества (см. рис. 6);
* оборудование (см. рис. 7);
* объекты (см. рис. 8);
* отделы (см. рис. 9);
* плановые ремонтные работы (см. рис. 10);
* подрядчики (см. рис. 11);
* сотрудники (см. рис. 12);
* список отпусков (см. рис. 13);
* сферы услуг (см. рис. 14);
* типы плановых ремонтных работ (см. рис. 15);
* услуги (см. рис. 16);
* учет средств (см. рис. 17).

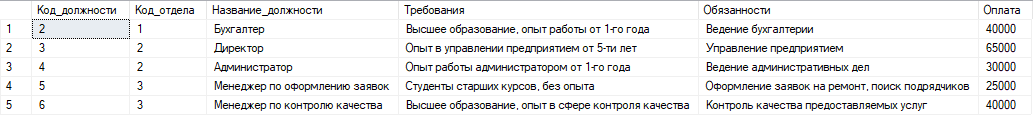


Рисунок 3. Таблица «Должности»



Рисунок 4. Таблица «Жильцы»

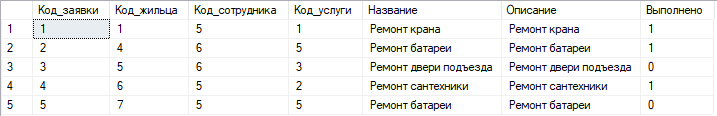


Рисунок 5. Таблица «Заявки жильцов»

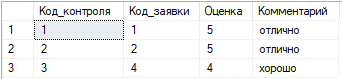


Рисунок 6. Таблица «Контроль качества»



Рисунок 7. Таблица «Оборудование»



Рисунок 8. Таблица «Объекты»

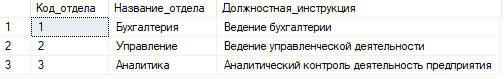


Рисунок 9. Таблица «Отделы»

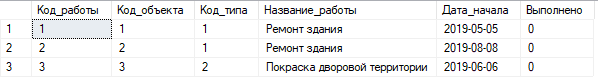


Рисунок 10. Таблица «Плановые ремонтные работы»

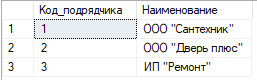


Рисунок 11. Таблица «Подрядчики»

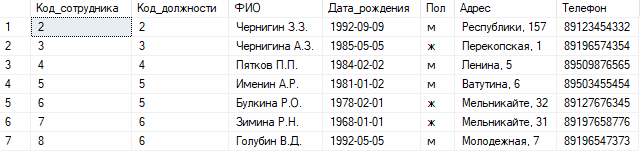


Рисунок 12. Таблица «Сотрудники»

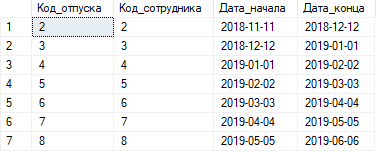


Рисунок 13. Таблица «Список отпусков»

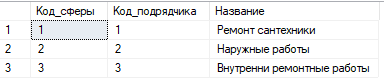


Рисунок 14. Таблица «Сферы услуг»

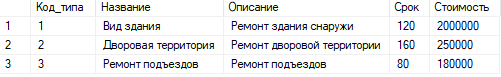


Рисунок 15. Таблица «Типы плановых ремонтных работ»

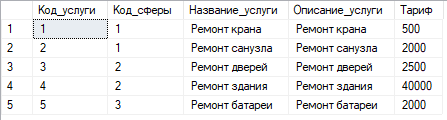


Рисунок 16. Таблица «Услуги»

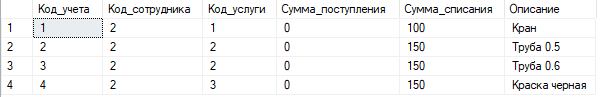


Рисунок 17. Таблица «Учет средств»

# Разработка веб-службы

Для отделения бизнес-логики и логики работы с БД от клиентского приложения используются веб-службы, осуществляющие взаимодействие по REST.

Веб-служба, веб-сервис (англ. web service) — идентифицируемая уникальным веб-адресом (URL-адресом) программная система со стандартизированными интерфейсами, а также HTML-документ сайта, отображаемый браузером пользователя. Веб-службы могут взаимодействовать друг с другом и со сторонними приложениями посредством сообщений, основанных на определённых протоколах и соглашениях.

REST (сокращение от англ. Representational State Transfer — «передача состояния представления») — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. REST определяет ряд архитектурных принципов проектирования Web-сервисов, ориентированных на системные ресурсы, включая способы обработки и передачи состояний ресурсов по HTTP разнообразными клиентскими приложениями, написанными на различных языках программирования. За последние несколько лет REST стала преобладающей моделью проектирования Web-сервисов.

Как происходит управление информацией сервиса – это целиком и полностью основывается на протоколе передачи данных. Наиболее распространенный протокол конечно же HTTP. Так вот, для HTTP действие над данными задается с помощью методов: GET (получить), PUT (добавить, заменить), POST (добавить, изменить, удалить), DELETE (удалить). Таким образом, действия CRUD (Create-Read-Updtae-Delete) могут выполняться как со всеми 4-мя методами, так и только с помощью GET и POST. Для каждой таблицы созданы контроллеры, в которых прописаны данные методы (приложение 1). С помощью строки http://localhost:3987/api/Должности, например, можно получить список всех имеющихся должностей (см. рис. 18).

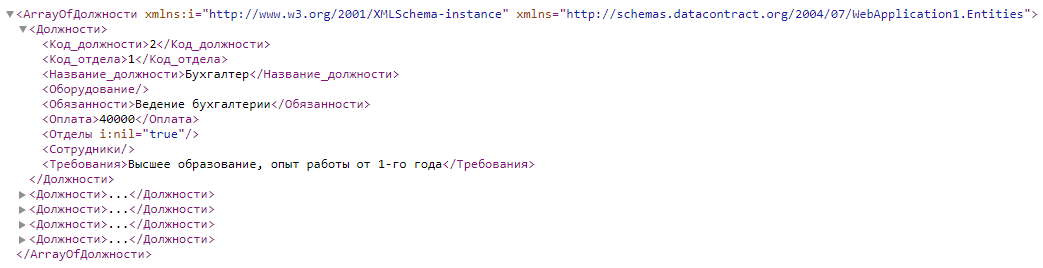


Рисунок 18. Список всех должностей

# Реализация клиентского приложения

Для удобного управления базой данных с помощью веб-форм создано клиентское приложение (см. рис. 19). Для каждой таблицы разработана отдельная веб-форма, где можно изменить, удалить или добавить данные, а также посмотреть список данных (см. рис. 20). Для каждой таблицы имеется возможность отсортировать данные по любому полю, например, по оплате (от минимальной до максимальной).

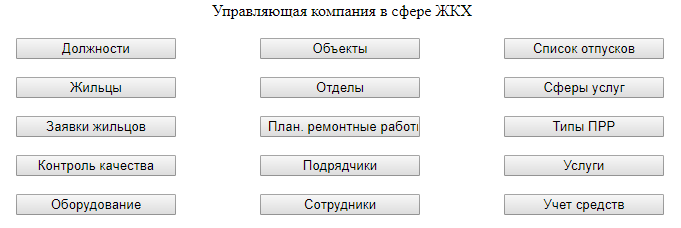


Рисунок 19. Главная страница приложения

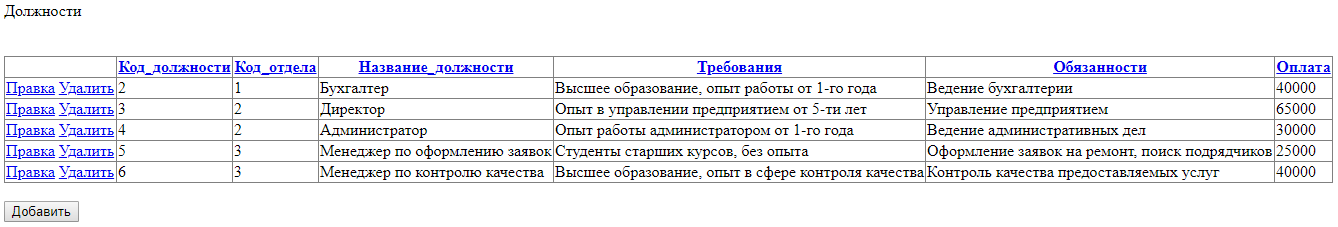


Рисунок 20. Страница управления должностями

# Заключение

В ходе разработки приложения «Управляющая компания в сфере ЖКХ» была создана диаграмма IDEF1x, по данной диаграмме – создана и заполнена база данных в СУБД MS SQL Server. Далее были подключены веб-службы, осуществляющие взаимодействие по REST и реализовано клиентское приложение, которое позволяет пользователю в удобном формате работать с базой данных.

# Список использованной литературы

1. «Управляющие компании ЖКХ» [Электронный ресурс], URL: http://ack1.ru/Library/upravlyaushchie\_kompanii\_ZHKH.html
2. «Основы методологии IDEF1x» [Электронный ресурс], URL: https://www.cfin.ru/vernikov/idef/idef1x.shtml
3. «Microsoft SQL Server» [Электронный ресурс], URL: https://ru.wikipedia.org/wiki/Microsoft\_SQL\_Server
4. «Веб-сервисы» [Электронный ресурс], URL: https://www.ibm.com/developerworks/ru/library/ws-restfu/index.html
5. «Архитектура REST» [Электронный ресурс], URL: https://habr.com/post/38730/

ПРИЛОЖЕНИЕ 1

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Description;

using WebApplication1.Entities;

namespace WebApplication1.Controllers

{

public class ДолжностиController : ApiController

{

private ZkhEntities db = new ZkhEntities();

public ДолжностиController()

{

db.Configuration.ProxyCreationEnabled = false;

}

internal void Ok(IHttpActionResult httpActionResult)

{

throw new NotImplementedException();

}

// GET: api/Должности

public IQueryable<Должности> GetДолжности()

{

return db.Должности;

}

// GET: api/Должности/5

[ResponseType(typeof(Должности))]

public IHttpActionResult GetДолжности(int id)

{

Должности должности = db.Должности.Find(id);

if (должности == null)

{

return NotFound();

}

return Ok(должности);

}

// PUT: api/Должности/5

[ResponseType(typeof(void))]

public IHttpActionResult PutДолжности(int id, Должности должности)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

if (id != должности.Код\_должности)

{

return BadRequest();

}

db.Entry(должности).State = EntityState.Modified;

try

{

db.SaveChanges();

}

catch (DbUpdateConcurrencyException)

{

if (!ДолжностиExists(id))

{

return NotFound();

}

else

{

throw;

}

}

return StatusCode(HttpStatusCode.NoContent);

}

// POST: api/Должности

[ResponseType(typeof(Должности))]

public IHttpActionResult PostДолжности(Должности должности)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

db.Должности.Add(должности);

db.SaveChanges();

return CreatedAtRoute("DefaultApi", new { id = должности.Код\_должности }, должности);

}

// DELETE: api/Должности/5

[ResponseType(typeof(Должности))]

public IHttpActionResult DeleteДолжности(int id)

{

Должности должности = db.Должности.Find(id);

if (должности == null)

{

return NotFound();

}

db.Должности.Remove(должности);

db.SaveChanges();

return Ok(должности);

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

private bool ДолжностиExists(int id)

{

return db.Должности.Count(e => e.Код\_должности == id) > 0;

}

}

}