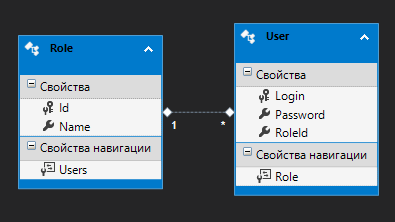
1. **Чтобы работало разделение по ролям, у нас обязательно должно выполняться одно условие – наличие ролей в БД! Для нормализации данных выносим роли в отдельную таблицу, связанную с пользователем.**



**Авторизация по ролям**

**Авторизация по ролям**

**Авторизация по ролям**

1. **Пишем следующий код в App.xaml.cs**

public static Entities.User CurrentUser = null;

С помощью данного кода мы создаем пустую переменную CurrentUser (Выбранный пользователь), которая будет хранить в себе информацию об авторизованном пользователе.

1. **При авторизации присваиваем переменной значение**

private void BtnLogin\_Click(object sender, RoutedEventArgs e) //Обработка кнопки «Войти» в авторизации

{

var currentUser = App.Context.Users.FirstOrDefault(p => p.Login == TBoxLogin.Text && p.Password == PBoxPassword.Password); //Сравниваем значение текстовых блоков со значениями в таблице User

if (currentUser != null) //Такой пользователь найден в таблице User

{

App.CurrentUser = currentUser; **//ПРИСВАИВАЕМ ЗНАЧЕНИЕ ПЕРЕМЕННОЙ**

NavigationService.Navigate(new ServicesPage());

}

else //Если пользователя нет

{

MessageBox.Show("Пользователь с такими данными не найден.", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

1. **При переходе на формы обозначаем, какие кнопки доступны и для какой роли**

public ServicesPage()

{

InitializeComponent();

if (App.CurrentUser.RoleId == 1) //Если роль 1 (например Менеджер)

{ //То кнопки редактирования и просмотра видны

BtnAddService.Visibility = Visibility.Visible;

BtnLookWrite.Visibility = Visibility.Visible;

}

else //Для всех других ролей

{ //Данные кнопки не видны

BtnAddService.Visibility = Visibility.Collapsed;

BtnLookWrite.Visibility = Visibility.Collapsed;

}

}

**Авторизация по ролям**

**Авторизация по ролям**



**Ура, Вы закончили еще одну тему!**

Возвращайтесь в приложение и переходите к следующему разделу.