

Рязанский станкостроительный колледж РГРТУ

МДК.01.01 Разработка программных модулей

Тема 6. Основы Entity Framework

Использование ListView при разработке БД

Рязань 2022

Оглавление

| | |
|---|----|
| Общие сведения | 3 |
| Элемент ListView | 4 |
| Реализация интерфейса для отображения БД | 5 |
| Сведения о структуре БД | 5 |
| Разработка макета интерфейса | 5 |
| Создания формы списка ListView..... | 6 |
| Использование форматов с применением StringFormat | 9 |
| Типовые операции с данными в таблице..... | 9 |
| Алгоритм добавления записи в БД..... | 10 |
| Алгоритм редактирования записи в БД | 12 |
| Алгоритм удаления записи в БД..... | 15 |
| Поиск информации..... | 16 |
| Фильтрация данных | 17 |
| Извлечение данных из элемента ListView | 17 |
| Практическая работа №22 | 18 |

Общие сведения

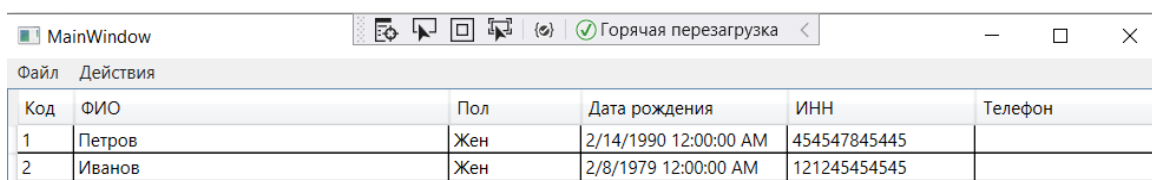
При разработке БД одной из важных задач является отображение информации из БД. Правильным подходом является продуманное отображение информации, т.е. главный критерий — это размещение информации в удобном виде для работы с ней пользователем этой программы и решение поставленных перед ним задач.

Ранее были рассмотрены такие способы отображения как *таблица и форма – бланк*, которые достаточны в большинстве случаев.

Таблица используется для отображения информации из БД в виде списка и может отображать информацию из одной таблицы или сгруппированную информацию из нескольких таблиц.

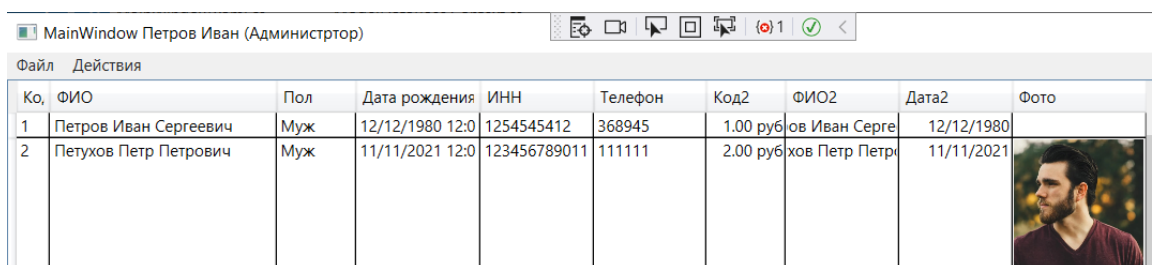
Форма – бланк используется для отображение подробной информации об одной записи из одной таблицы, или нескольких таблиц.

Но иногда возможности отображения информации в виде таблицы бывает недостаточны, особенно когда при выводе информации нужно использовать различные визуальные элементы, например картинки. В таких случаях вместо таблицы используется элемент *ListView*. Рассмотрим ниже примеры различного вывода с использованием таблиц и элемента *ListView*.



| Код | ФИО | Пол | Дата рождения | ИНН | Телефон |
|-----|--------|-----|-----------------------|--------------|---------|
| 1 | Петров | Жен | 2/14/1990 12:00:00 AM | 454547845445 | |
| 2 | Иванов | Жен | 2/8/1979 12:00:00 AM | 121245454545 | |

Рисунок 1 - Вывод информации в виде таблицы




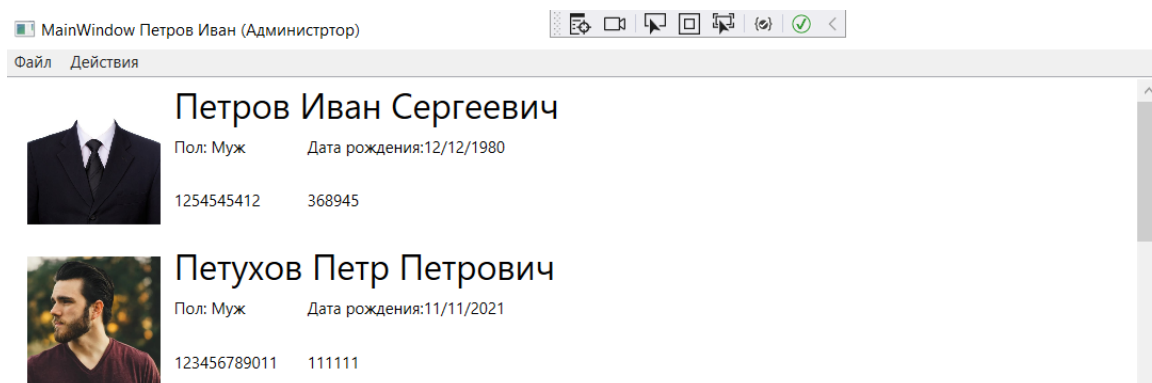
| Ко | ФИО | Пол | Дата рождения | ИНН | Телефон | Код2 | ФИО2 | Дата2 | Фото |
|----|-----------------------|-----|-----------------|--------------|---------|----------|---------------|------------|---|
| 1 | Петров Иван Сергеевич | Муж | 12/12/1980 12:0 | 1254545412 | 368945 | 1.00 руб | ов Иван Серге | 12/12/1980 | |
| 2 | Петухов Петр Петрович | Муж | 11/11/2021 12:0 | 123456789011 | 111111 | 2.00 руб | хов Петр Петр | 11/11/2021 |  |

Рисунок 2 - Вывод информации в виде таблицы с картинками





| Photo | Name | Gender | Date of Birth | INN |
|---|-----------------------|----------|---------------------------|---------------------|
|  | Петров Иван Сергеевич | Пол: Муж | Дата рождения: 12/12/1980 | 1254545412 368945 |
|  | Петухов Петр Петрович | Пол: Муж | Дата рождения: 11/11/2021 | 123456789011 111111 |

Рисунок 3 - Вывод информации в виде списка *ListView*

Элемент *ListView*

Элемент *ListView* позволяет отображать информацию в виде списка. Он унаследован от класса *ListBox*, поэтому может использоваться для отображения простого списка и многие приемы работы с ним аналогичны. Также в нем добавлены мощные средства для форматирования выводимой информации, которые мы и будем использовать.

Рассмотрим основные свойства элемент *ListView*.

Таблица 1. Основные свойства элемента *ListView*

| Свойство | Комментарий |
|--|---|
| <code>ItemCollection</code> <code>Items</code> | Элементы списка — коллекция строк |
| <code>ItemsPanelTemplate</code> <code>ItemsPanel</code> | Возвращает или задает шаблон, определяющий панель, управляющую размещением элементов. Значением по умолчанию является <i>StackPanel</i> , что означает, что элементы размещаются в виде списка. Если использовать <i>WrapPanel</i> , то элементы будут размещать в виде плитки. |
| <code>IEnumerable</code> <code>ItemsSource</code> | Получает или задает коллекцию, используемую для создания содержимого. |
| <code>object</code> <code>SelectedItem</code> | Возвращает или задает первый элемент в текущем выделении или возвращает значение <i>NULL</i> , если выделение является пустым. |
| <code>int</code> <code>SelectedIndex</code> | Получение или установка индекса первого элемента в текущем выделенном фрагменте или возврат значения минус один (-1), если выделенный фрагмент пуст. |
| <code>SelectionMode</code> | Возвращает или задает поведение выбора для списка <i>ListView</i> . <i>Single</i> - Пользователь может выбрать только один элемент. <i>Multiple</i> - Можно выбрать несколько элементов, не удерживая клавишу <i>Ctrl</i> , <i>Shift</i> или <i>Alt</i> . <i>Extended</i> - Пользователь может выбрать несколько последовательных элементов, удерживания нажатой клавишу <i>SHIFT</i> . |
| События | Комментарий |
| <code>SelectionChanged</code> | Возникает при изменении текущего выделения |

Реализация интерфейса для отображения БД

Сведения о структуре БД

Для демонстрации использования элемента *ListView*, используем базу данных абонент из лекции №9 Основы работы с БД. Эта база содержит информацию о некотором абоненте, а также его фотографию.

| | Имя столбца | Тип данных | Разрешить значения NULL |
|---|-------------|---------------|-------------------------------------|
| 🔑 | Id | int | <input type="checkbox"/> |
| | Fio | nvarchar(50) | <input type="checkbox"/> |
| | Gender | nvarchar(3) | <input type="checkbox"/> |
| | Age | datetime | <input checked="" type="checkbox"/> |
| | Inn | nvarchar(12) | <input checked="" type="checkbox"/> |
| | Phone | nvarchar(10) | <input checked="" type="checkbox"/> |
| | Photo | nvarchar(MAX) | <input checked="" type="checkbox"/> |

Рисунок 4 – Структура данных в таблице Абонент

```
public partial class Abonent
{
    Ссылка: 10
    public int Id { get; set; }
    Ссылка: 20
    public string Fio { get; set; }
    Ссылка: 10
    public string Gender { get; set; }
    Ссылка: 4
    public Nullable<System.DateTime> Age { get; set; }
    Ссылка: 3
    public string Inn { get; set; }
    Ссылка: 3
    public string Phone { get; set; }
    Ссылка: 7
    public string Photo { get; set; }
}
```

Рисунок 4 – Класс сущности Абонент

Разработка макета интерфейса

Изучив структуру БД приступаем к разработке макета интерфейса, т.е. способа размещения информации.

Для наших данных можно разработать строчный макет, где слева размещаем фотографию абонента, справа ФИО, и справа ниже в два столбика остальную информацию см. рисунок:


| | | |
|---|------------------------------|---------------------------|
|  | Петров Иван Семенович | |
| | Пол: Мужской | Дата рождения: 12.12.2022 |
| | ИНН: 1232145144 | Телефон: 910-900-45-33 |

Рисунок 5 – Строчный макет размещения информации

Также можно разработать плиточный макет, где размещение информации komponуем так, чтобы получилось квадратная плитка см. рисунок:

| | |
|--|--|
| Петров Иван Семенович | |
|  | |
| Пол: Мужской | |
| Дата рождения: 12.12.2022 | |
| ИНН: 1232145144 | |
| Телефон: 910-900-45-33 | |

Рисунок 6 – Плиточный макет размещения информации

Создания формы списка **ListView**

1. На форме размещаем элемент список (**ListView**), для плиточного макета отключите горизонтальную прокрутку.

```
<ListView x:Name="listView1" Grid.Row="1"
          ScrollViewer.HorizontalScrollBarVisibility="Disabled">
```

2. Объявляем глобальную переменную для получения доступа к контексту данных и получаем контекст данных. При загрузке формы с использованием контекста данных загружаем таблицу из БД. Привязываем или загружаем в **DataGrid** таблицу с данными с отслеживанием изменений контекста данных или без отслеживания изменений контекста данных.

```
//Получаем доступ к контексту данных
AbonentsEntities _db = AbonentsEntities.GetContext();
Ссылка: 1
private void mainWindow1_Loaded(object sender, RoutedEventArgs e)
{
    //Загружаем таблицу из БД
    _db.Abonents.Load();
    //Загружаем таблицу в listView с отслеживанием изменения контекста
    listView1.ItemsSource = _db.Abonents.Local.ToBindingList();
    //Загружаем таблицу в listView без отслеживания изменения контекста
    //listView1.ItemsSource = _db.Abonents.ToList();
}
```

3. В разметке **XAML** создаем шаблон отображения информации согласно макету. Т.к. макет по факту имеет табличный дизайн, используем контейнер **Grid** и согласно макету делаем в нем разметку.

```

<ListView.ItemTemplate>
    <DataTemplate>
        <Grid>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="120"></ColumnDefinition>
                <ColumnDefinition Width="100"></ColumnDefinition>
                <ColumnDefinition Width="*"></ColumnDefinition>
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
                <RowDefinition Height="20"></RowDefinition>
                <RowDefinition Height="20"></RowDefinition>
                <RowDefinition Height="*"></RowDefinition>
            </Grid.RowDefinitions>

```

4. Настраиваем основные выводимые поля, размещаем их в выбранных ячейках таблицы и привязываем помощью атрибута **Binding** к полям таблицы в базе данных.

```

<Image Width="100" Height="100" Grid.Row="0" Grid.Column="0" Grid.RowSpan="3"
    Stretch="UniformToFill" HorizontalAlignment="Center" Margin="10"
    Source="{Binding PhotoFull}">
    <!--<Image.Source>
        <Binding Path="PhotoFull">
            <Binding.TargetNullValue>
                <ImageSource>image/picture.jpg</ImageSource>
            </Binding.TargetNullValue>
        </Binding>
    </Image.Source>-->
</Image>
<TextBlock Text="{Binding Fio}" Grid.Row="0" Grid.Column="1" Grid.ColumnSpan="2"
    TextAlignment="Left" FontSize="26"></TextBlock>
<TextBlock Text="{Binding Gender, StringFormat={}}Пол: {0}" Grid.Row="1"
    Grid.Column="1" TextAlignment="Left"></TextBlock>
<TextBlock Text="{Binding Inn}" Grid.Row="2" Grid.Column="1" TextAlignment="Left"
<TextBlock Text="{Binding Age, StringFormat={}}Дата рождения: {0:d}"
    Grid.Row="1" Grid.Column="2" TextAlignment="Left"></TextBlock>
<TextBlock Text="{Binding Phone}" Grid.Row="2" Grid.Column="2" TextAlignment="Le-
Grid>

```

Обратите внимание, если фото в БД нет, на его месте будет пустое место, что не совсем красиво. Можно при отсутствии фото вставить типовое фото из заранее подготовленной картинке. Для этого добавляем картинку в проект *image/picture.jpg* и изменяем атрибут Source используя вложенное свойство TargetNullValue для привязки стандартной картинке, см. закомментированную зеленым часть кода выше.

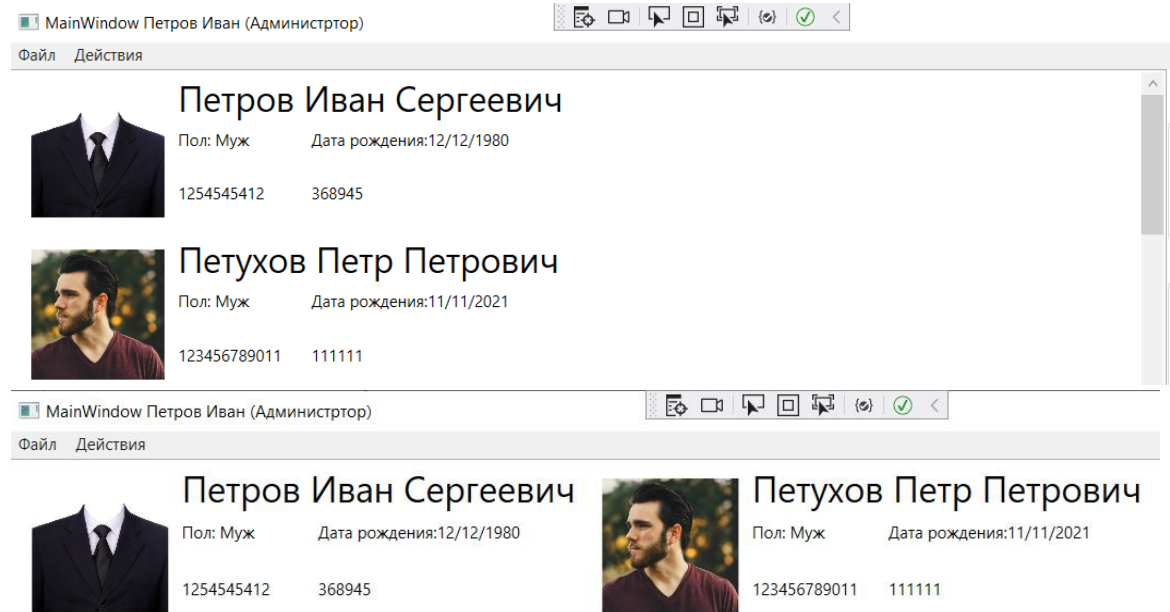
5. Если мы желаем настроить плиточный дизайн вывода, то настраиваем шаблон в свойстве **ItemPanel**, в данном случае в шаблоне указываем тип контейнера **WrapPanel**. В контейнер можно произвести типовые настройки выравнивания элементов.

```

<ListView x:Name="listView1" Grid.Row="1"
    ScrollViewer.HorizontalScrollBarVisibility="Disabled">
    <!--<ListView.ItemsPanel>
        <ItemsPanelTemplate>
            <WrapPanel></WrapPanel>
        </ItemsPanelTemplate>
    </ListView.ItemsPanel>-->

```

6. В итоге вы имеете вывод информации в виде списка или плитки



7. Если вы обратили внимание, то поле фото называется **Photo**, а в **ListView** указано для выбора поле **PhotoFull**.

Это связано с тем, что поле **Photo** содержит имя файла с картинкой, а для автоматической загрузки фото нужен полный путь файла с картинкой.

Для удобства файлы с картинками будем размещать в папке **image**, которая будет располагаться в папке с программой. Для формирования полного пути в класс сущности Абонент добавим поле **PhotoFull**, которое будет формировать полный путь к файлу.

```
public string Photo { get; set; }
```

Ссылка: 1

```
public string PhotoFull
{
    get
    {
        if (this.Photo == null)
        {
            return null;
        }
        else
        {
            string namePhoto = Directory.GetCurrentDirectory() + "\\image\\" + Photo;
            return namePhoto;
        }
    }
}
```


Использование форматов с применением StringFormat

Рассмотрим основные форматы, используемые при форматировании значений с использованием атрибута **StringFormat**.

| Описатель формата | Имя | Описание | Примеры |
|-------------------|----------------------------|---|---|
| "C" или "c" | Валюта | Результат: значение валюты. Поддерживается все числовые типы. | 123.456 ("C", en-US) -> \$123.46 -123.456 ("C3", en-US) -> (\$123.456) |
| "E" или "e" | Экспоненциальный (научный) | Результат: экспоненциальная нотация. Поддерживается все числовые типы. | 1052.0329112756 ("E", en-US) -> 1.052033E+003 -1052.0329112756 ("e2", en-US) -> -1.05e+003 |
| "F" или "f" | С фиксированной запятой | Результат: цифры целой и дробной частей с необязательным отрицательным знаком. Поддерживается все числовые типы. | 1234.567 ("F", en-US) -> 1234.57 1234.56 ("F4", en-US) -> -1234.5600 |
| "N" или "n" | Число | Результат: цифры целой и дробной частей, разделители групп и разделитель целой и дробной частей с необязательным отрицательным знаком. Поддерживается все числовые типы. | 1234.567 ("N", ru-RU) -> 1 234,57 -1234.56 ("N3", ru-RU) -> -1 234,560 |
| "d" | Короткий шаблон даты | | 2009-06-15T13:45:30 -> 6/15/2009 (en-US) |
| "D" | Полный шаблон даты. | | 2009-06-15T13:45:30 -> 15 июня 2009 г. (ru-RU) |
| "t" | Короткий шаблон времени. | | 2009-06-15T13:45:30 -> 1:45 PM (en-US) |
| "T" | Полный шаблон времени. | | 2009-06-15T13:45:30 -> 1:45:30 PM (en-US) |

Типовые операции с данными в таблице

Над данными в таблице обычно выполняются такие операции как добавления записи, изменение записи, удаление записи. Как правило, при добавлении или изменении записи открывается дополнительная форма, где добавляемая или изменяемая запись представлена в виде формы – бланка. Ввод или редактирование записи происходит на этой дополнительной форме. Так же часто при работе с текущей записью происходит

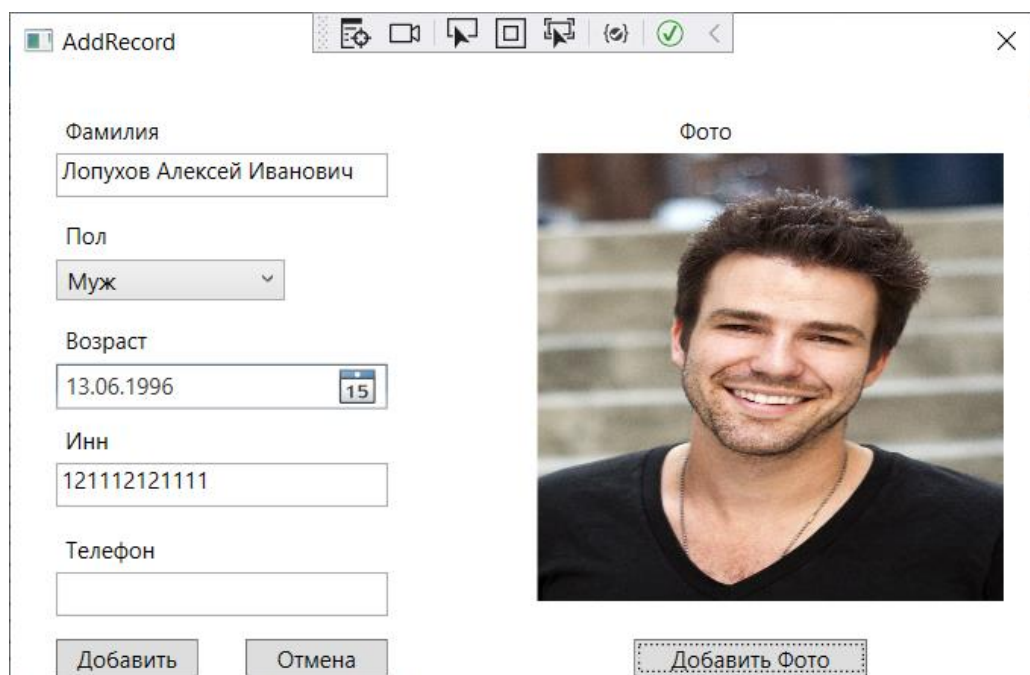
контроль данных по заданным критериям и в БД попадает только та информация, которая соответствует этим критериям.

В данном случае все действия происходят также, как и раньше. Обратим внимание только на то, что надо добавлять и изменять имя файла фото в БД. Соответственно при работе с фото выбранное фото будем копировать в папку image, при переборе файла, старый файл будем удалять. Изменения будут выделены красным цветом и отмечены комментарием *********

Алгоритм добавления записи в БД

1. На основную форму добавим кнопку *Добавить запись*. Кнопка *Добавить запись* будет открывать форму – бланк содержащие элементы для добавления новой записи.

2. Создаем форму *Добавить запись* см. пример. Эта форма будет содержать обычные *TextBox*, *ComboBox* для поля пол, *DatePicker* для поля дата рождения и *Image* для размещения фото.



3. На основной форме пишем код кнопки *Добавить запись*. Кнопка *Добавить запись* будет открывать форму *Добавить запись*.

```
private void btnAdd_Click(object sender, RoutedEventArgs e)
{
    //Открываем форму Добавить
    AddRecord winAddRecord = new AddRecord();
    winAddRecord.ShowDialog();
    listView1.Focus();
}
```

4. На форме *Добавить запись* объявляем глобальную переменную для получения доступа к контексту данных и получаем контекст данных. Создаем сущность таблицы. Создаем глобальную переменную – путь к файлу с фото

```
//Получаем доступ к контексту данных
AbonentsEntities _db = AbonentsEntities.GetContext();
//Создаем элемент таблицы
Abonent _abonent = new Abonent();
//*****
//Путь к файлу выбранного фото
OpenFileDialog open = new OpenFileDialog();
```

4. На форме *Добавить запись* пишем код кнопки *Добавить*. При добавлении обратите внимание на поле *Photo*, которое задается из глобальной переменной.

```
private void btnAdd_Click(object sender, RoutedEventArgs e)
{
    //Пример проверки каждого поля отдельно
    StringBuilder errors = new StringBuilder();
    if (tbFio.Text.Length == 0) errors.AppendLine("Введите фамилию");
    if (cbGender.Text != "Муж" && cbGender.Text != "Жен")
        errors.AppendLine("Введите пол Муж/Жен");
    if (tbInn.Text.Length != 12 ||
        double.TryParse(tbInn.Text, out double x) == false)
        errors.AppendLine("Неправильный ИНН");
    if (dpAge.Text.Length == 0) errors.AppendLine("Введите дату");
    /*
    // Проверка что абоненту более 18 лет
    DateTime dateNow = DateTime.Now; //Текущая дата
    DateTime dateAge = Convert.ToDateTime(dpAge.SelectedDate); //Дата рождения
    int yearNow = dateNow.Year; //Получение года
    int yearAge = dateAge.Year;
    if (yearNow - yearAge < 18)
    {
        errors.AppendLine("Введите правильно дату рождения");
    }
    */
    if (errors.Length > 0)
    {
        MessageBox.Show(errors.ToString());
        return;
    }
    //Заполняем этот элемент
    _abonent.Fio = tbFio.Text;
    _abonent.Gender = cbGender.Text;
    _abonent.Age = dpAge.SelectedDate;
    _abonent.Inn = tbInn.Text;
    _abonent.Phone = tbPhone.Text;
    //*****
    //Запоминаем имя фото если оно задано
    if (open.SafeFileName.Length != 0)
    {
        //Копируем выбранное фото в папку image
        string newNamePhoto = Directory.GetCurrentDirectory() +
            "\\image\\" + open.SafeFileName;
        File.Copy(open.FileName, newNamePhoto, true);
        _abonent.Photo = open.SafeFileName;
    }
}
```

```
try
{
    //Добавляем в БД
    _db.Abonents.Add(_abonent);
    //Сохраняем изменения
    _db.SaveChanges();
    //MessageBox.Show("Информация сохранена!");
    this.Close();
}
catch (Exception ex)
{
    _db.Abonents.Remove(_abonent);
    MessageBox.Show(ex.Message.ToString());
}
}
```

5. На форме *Добавить запись* пишем код кнопки *Добавить Фото*.

```
private void btnAddPhoto_Click(object sender, RoutedEventArgs e)
{
    open.Filter = "Все файлы | *.* | Файлы *.jpg|*.jpg";
    open.FilterIndex = 2;
    if (open.ShowDialog() == true)
    {
        BitmapImage photoImage = new BitmapImage(new Uri(open.FileName));
        imgPhoto.Source = photoImage;
    }
}
```

Алгоритм редактирования записи в БД

1. На основную форму добавим кнопку *Изменить запись*. Кнопка *Изменить запись* будет открывать форму – бланк содержащие элементы для изменения текущей записи.

2. Создаем форму *Изменить запись* см. предыдущий пример. Эта форма будет содержать обычные TextBox, ComboBox для поля пол, DateTimePicker для поля дата рождения и *Image* для размещения фото.

Так как форма *Изменить запись* имеет такой же вид как и форма *Добавить запись*, то можно в окне *Обозреватель решений* с помощью операций *Копировать* и *Вставить* сделать копию формы *Добавить запись*.

При создании копии:

- измените название формы в обозревателе решений;
- измените название класса в разметке XAML;
- измените название класса и конструктора в программном коде окна.

3. Создайте глобальный статический класс для передачи данных между формами. В этом классе опишем элемент Id для передачи кода текущей записи см. пример.

```
namespace PrimerBD
{
    Ссылка: 2
    public static class Data {
        public static int Id; //Код записи
    }
}
```

4. На основной форме пишем код кнопки **Изменить запись**. Кнопка **Изменить запись** будет открывать форму **Изменить запись**.

Чтобы изменять текущую запись запоминаем ее код.

Также т.к. при изменении записи контекст автоматически не обновляется, в таблице на основной форме изменения не отобразятся. Чтобы увидеть обновленную запись нужно вручную обновить таблицу.

Также таблица при нажатии кнопки **Изменить** теряет фокус и соответственно теряется выделение текущей записи, возвращаем фокус таблице см. код кнопки **Изменить запись**.

```
private void btnEdit_Click(object sender, RoutedEventArgs e)
{
    int indexRow = listView1.SelectedIndex;
    if (indexRow != -1)
    {
        //Получаем ключ текущей записи
        Abonent row = (Abonent)listView1.Items[indexRow];
        Data.Id = row.Id;
        //Открываем форму Редактировать
        EditRecord winEditRecord = new EditRecord();
        winEditRecord.ShowDialog();
        //Обновляем список
        listView1.Items.Refresh();
        listView1.Focus();
    }
}
```

5. На форме **Изменить запись** объявляем глобальную переменную для получения доступа к контексту данных и получаем контекст данных. Создаем сущность таблицы. Создаем глобальную переменную – путь к файлу с фото. Изменений нет!

```
//Получаем доступ к контексту данных
AbonentsEntities _db = AbonentsEntities.GetContext();
Abonent _abonent; //Элемент для работы с записью
//*****
//Путь к файлу выбранного фото
OpenFileDialog open = new OpenFileDialog();
```

6. На форме *Изменить запись* при загрузке формы отображаем содержимое текущей записи. **Обратите внимание на поле *Photo*.**

```
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    //Получаем запись по коду
    _abonent = _db.Abonents.Find(Data.Id);
    //Отображаем запись
    tbFio.Text = _abonent.Fio;
    cbGender.Text = _abonent.Gender;
    dpAge.SelectedDate = _abonent.Age;
    tbInn.Text = _abonent.Inn;
    tbPhone.Text = _abonent.Phone;
    //*****
    //Отображение фото
    if (_abonent.Photo != null)
    {
        BitmapImage photoImage = new BitmapImage(new Uri(_abonent.PhotoFull));
        imgPhoto.Source = photoImage;
    }
}
```

7. На форме *Изменить запись* пишем код кнопки *Изменить*. Учтем заданные ограничения полей.

```
private void btnEdit_Click(object sender, RoutedEventArgs e)
{
    //Пример проверки каждого поля отдельно
    StringBuilder errors = new StringBuilder();
    if (tbFio.Text.Length == 0) errors.AppendLine("Введите фамилию");
    if (cbGender.Text != "Муж" && cbGender.Text != "Жен")
        errors.AppendLine("Введите пол Муж/Жен");
    if (tbInn.Text.Length != 12 ||
        double.TryParse(tbInn.Text, out double x) == false)
        errors.AppendLine("Неправильный ИНН");
    if (dpAge.Text.Length == 0) errors.AppendLine("Введите дату");
    /*
    // Проверка что абоненту более 18 лет
    DateTime dateNow = DateTime.Now; //Текущая дата
    DateTime dateAge = Convert.ToDateTime(dpAge.SelectedDate); //Дата рождения
    int yearNow = dateNow.Year; //Получение года
    int yearAge = dateAge.Year;
    if (yearNow - yearAge < 18)
    {
        errors.AppendLine("Введите правильно дату рождения");
    }
    */
    if (errors.Length > 0)
    {
        MessageBox.Show(errors.ToString());
        return;
    }
    //Заполняем этот элемент
    _abonent.Fio = tbFio.Text;
    _abonent.Gender = cbGender.Text;
    _abonent.Age = dpAge.SelectedDate;
    _abonent.Inn = tbInn.Text;
    _abonent.Phone = tbPhone.Text;
}
```



```

//*****
//Запоминаем имя фото
if (open.SafeFileName.Length != 0)
{
    //Если это новое имя файла
    if (open.SafeFileName != _abonent.Photo)
    {
        //Копируем выбранное фото в папку image
        string newNamePhoto = Directory.GetCurrentDirectory() +
            "\\image\\" + open.SafeFileName;
        File.Copy(open.FileName, newNamePhoto, true);
        //Задаем новое имя фото
        _abonent.Photo = open.SafeFileName;
    }
}

try
{
    //Добавляем в БД
    //_db.Abonents.Add(_abonent);
    //Сохраняем изменения
    _db.SaveChanges();
    //MessageBox.Show("Информация сохранена!");
    this.Close();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message.ToString());
}
}

```

Алгоритм удаления записи в БД

1. На основную форму добавим кнопку *Удалить запись*. Кнопка *Удалить запись* будет удалять текущую запись с подтверждением.

2. На основной форме пишем код кнопки *Удалить запись*.

Вначале используем элемент *MessageBox* для вывода окна подтверждения удаления. При подтверждении удаления, удаляем текущую запись.

```

private void btnDelete_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult result;
    result = MessageBox.Show("Удалить запись?", "Удаление записи",
        MessageBoxButton.YesNo, MessageBoxImage.Warning);
    if (result == MessageBoxResult.Yes)
    {
        try
        {
            //Получаем текущую запись
            Abonent row = (Abonent)listView1.SelectedValue;
            //int indexRow = listView1.SelectedIndex;
            //Abonent row = (Abonent)listView1.Items[indexRow];
            //Abonent row = (Abonent)listView1.SelectedItems[0];

```

```

        //Удаляем запись
        _db.Abonents.Remove(row);
        _db.SaveChanges();
    }
    catch (ArgumentOutOfRangeException)
    {
        MessageBox.Show("Выберите запись");
    }
}
}

```

Поиск информации

Поиск информации в БД задача довольно непростая и может осуществляться разными способами, которые зависят от того какой конечный вариант нужно получить.

Рассмотрим один из простых вариантов поиска в элементе *ListView*, в которой в цикле проходим нужную запись в поле и ищем заданное значение. Далее выделяем строку с найденным значением, поиск осуществляем налету и реализуем так называемый нечеткий поиск, т.е. поиск по частичному совпадению см. пример интерфейса и кода программы:

Поиск по ФИО

```

private void tbFind_TextChanged(object sender, TextChangedEventArgs e)
{
    foreach (var item in listView1.Items)
    {
        if (((Abonent)item).Fio.Contains(tbFind.Text))
        {
            listView1.SelectedItem = item; //Выделяем элемент
            listView1.ScrollIntoView(item); //Скролим к нему окно
            break;
        }
    }

    //for (int i = 0; i < ListView.Items.Count; i++)
    //{
    //    //Получаем строку таблицы
    //    var row = (Abonent)ListView.Items[i];
    //    try
    //    {
    //        if (row.Fio.Contains(tbFind.Text))
    //        {
    //            object item = ListView.Items[i];
    //            ListView.SelectedItem = item; //Выделяем элемент
    //            ListView.ScrollIntoView(item); //Скролим к нему окно
    //            break;
    //        }
    //    }
    //    catch { }
    //}
}

```


Фильтрация данных

Фильтрация данных - это выбор данных по заданному критерию (условию).

Одним из способов установить фильтр использовать **LINQ** выражения, например используя метод **Where** сформировать новую коллекцию с отобранными значениями, фильтр будем осуществлять налету и реализуем так называемый нечеткий фильтр, т.е. фильтр по частичному совпадению см. пример интерфейса и кода программы:

-Фильтр по ФИО —————

```
private void tbFiltered_TextChanged(object sender, TextChangedEventArgs e)
{
    if (tbFiltered.Text.Length > 0)
    {
        //Загружаем в коллекцию таблицу
        var abonent = _db.Abonents.ToList();
        //Формируем новую таблицу по фильтру
        //var filtered = _abonent.Where(p => p.Fio == txtFiltered.Text);
        var filtered = abonent.Where(p => p.Fio.ToLower().
            Contains(tbFiltered.Text.ToLower()));
        //var filtered = abonent.Where(p => p.Fio.StartsWith(txtFiltered.Text));
        //Отображаем полученную таблицу
        listView1.ItemsSource = filtered;
    }
    else
    {
        listView1.ItemsSource = _db.Abonents.Local.ToBindingList();
    }
}
```

Извлечение данных из элемента ListView

```
//Получаем текущую запись
Abonent row = (Abonent)listView1.SelectedValue;
//int indexRow = listView1.SelectedIndex;
//Abonent row = (Abonent)listView1.Items[indexRow];
//Abonent row = (Abonent)listView1.SelectedItems[0];
```

Практическая работа №22

Использование **ListView**.

1. Создайте БД по варианту задания и заполните ее данными, при необходимости доработайте БД, добавив необходимые атрибуты.
2. Разработайте модуль авторизации.

При запуске приложения окно входа – первое, что видит пользователь. На ней пользователю предлагается ввести свой логин и пароль или есть возможность перейти на экран просмотра товаров в роли гостя.

Только после удачной авторизации пользователь получает доступ к остальным модулям системы:

- авторизованный клиент может просмотреть товары;
- менеджер может просматривать товары;
- администратор может добавлять/редактировать/удалять товары.

Реализуйте необходимые интерфейсы для всех пользователей системы. После входа в любую учетную запись должна быть реализована возможность выхода на главный экран – окно входа. При переходе в любую учетную запись в интерфейсе (правый верхний угол) должны отображаться ФИО пользователя.

После первой попытки неуспешной авторизации система выдает сообщение о неуспешной авторизации, а затем помимо ввода логина и пароля просит ввести captcha, состоящую из 4 символов (цифры и буквы латинского алфавита) и графического шума.

САРТСНА - должна содержать минимум 4 символа (буква или цифра), которые выведены не в одной линии. Символы должны быть либо перечеркнуты либо наложены друг на друга.

После попытки неудачной авторизации с вводом captcha, система блокирует возможность входа на 10 секунд.

3. Отобразите информацию с использованием **ListView** по варианту задания.
4. Пользователь должен иметь возможность искать информацию, используя поисковую строку, в том числе по нескольким атрибутам одновременно.
5. Пользователь должен иметь возможность отфильтровать данные.
6. Реализуйте переход на окно добавления информации и окно редактирования выбранной информации.

Варианты заданий

Вариант 1.

Постановка задачи:

Необходимо хранить информацию о существующих складах (номер склада, адрес, телефон, фамилия руководителя склада), товарах (код, название, группа товара, фирма-производитель), а также о наличии товаров на конкретных складах с указанием количества товаров.

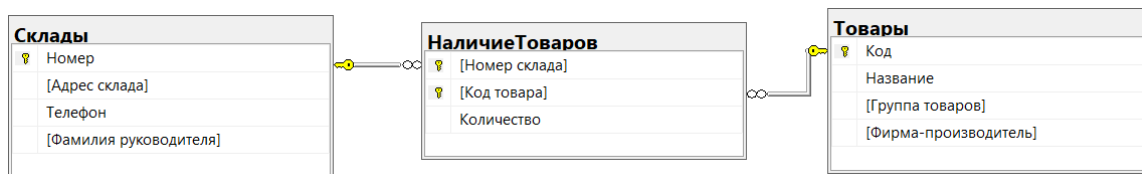


Схема базы данных

Таблица Склады

| Номер | Адрес склада | Телефон | Фамилия руководителя |
|-------|-------------------|---------|----------------------|
| 1 | Первомайская, 1 | 111111 | Иванов |
| 2 | Московское, 7 | 222222 | Сидоров |
| 3 | Касимовское, 3 | 111222 | Петров |
| 4 | Куйбышевское, 27 | 334455 | Ковалев |
| 5 | Шабулина, 12 | 121212 | Маматов |
| 6 | Яблочкова, 11 | 345678 | Маматов |
| 7 | Циолковского, 17 | 778877 | Сазонов |
| 8 | Павлова, 28 | 321321 | NULL |
| 9 | Новоселов, 60 | 223344 | Лоськов |
| 10 | Забайкальская, 14 | 445544 | Родин |

Таблица Товары

| Код | Название | Группа товаров | Фирма-производитель |
|-----|------------|----------------|---------------------|
| 1 | Колбаса | Продукты | Скопинский |
| 2 | Сыр | Продукты | Молкомбинат |
| 3 | Хлеб | Продукты | Хлебзавод 1 |
| 4 | Телевизор | Техника | Sony |
| 5 | Стол | Мебель | IKEA |
| 6 | Автокресло | NULL | NULL |
| 7 | Лопата | Хозтовары | Мехзавод |
| 8 | Мочалка | Хозтовары | Авангард |
| 9 | Мыло | Хозтовары | Свобода |
| 10 | Кастрюля | Хозтовары | Технопласт |

Таблица НаличиеТоваров

| Номер склада | Код товара | Количество |
|--------------|------------|------------|
| 1 | 3 | 5000 |
| 1 | 10 | 400 |
| 2 | 4 | 8000 |
| 2 | 7 | 1000 |
| 2 | 8 | 7000 |
| 3 | 3 | 500 |
| 3 | 8 | 9000 |
| 3 | 9 | 15000 |
| 4 | 1 | 2500 |
| 4 | 8 | 10000 |
| 5 | 2 | 1000 |
| 6 | 9 | 4500 |
| 6 | 3 | 2000 |
| 6 | 5 | 800 |
| 7 | 7 | 700 |
| 7 | 2 | 9000 |
| 8 | 1 | 3000 |
| 9 | 5 | 1000 |
| 9 | 8 | 5000 |
| 9 | 10 | 500 |

Интерфейс:

Необходимо отобразить информацию о товарах, также фото и количество товара на складе.

Вариант 2.

Постановка задачи:

Необходимо хранить информацию о фирмах-производителях продуктов (код фирмы, название фирмы, адрес, фамилия директора), продуктах (код, название, группа продуктов, вид упаковки), а также об объеме производства продуктов.

Схема базы данных

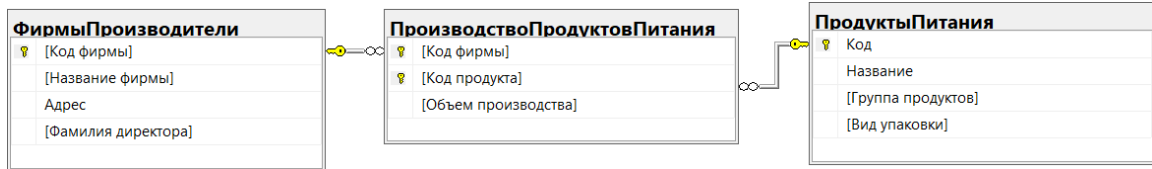


Таблица ФирмыПроизводители

| Код фирмы | Название фирмы | Адрес | Фамилия директора |
|-----------|------------------------------|---------------------------------------|-------------------|
| 1 | Шиловомясо | Шилово, ул. Рязанская, 118 | Титкин |
| 2 | Вим Биль Дан | Москва, Дмитровское шоссе, 108 | Пластинин |
| 3 | Рязаньхлеб | Рязань, ул. Военных автомобилистов, 3 | Поляков |
| 4 | Агропищекombинат Нива Рязани | Рязань, ул. Есенина, 9 | Горкин |
| 5 | Кортлав | Рязань, ул. 14 линия, 2 стр. 1 | Бренер |
| 6 | ОАО Лебедянский | Лебедянь, ул. Матросова, 7 | Кобзев |
| 7 | ОАО Макфа | Челябинская область, п. Рощино | Юревич |
| 8 | Красный Октябрь | Москва, Берсеневская наб., 6 | Даурской |
| 9 | ООО НЕП | С.-Петербург, ул. Тобольская, 3 | Зуев |
| 10 | ПКП Русь | Петушки, ул. Клязьменская, д. 2 | NULL |

Таблица ПродуктыПитания

| Код | Название | Группа продуктов | Вид упаковки |
|-----|--------------------|--------------------|-------------------|
| 1 | Сыр колбасный | Молочные | Целлофан |
| 2 | Молоко | Молочные | Картонная коробка |
| 3 | Хлеб бородинский | Хлебобулочные | Целлофан |
| 4 | Батон нарезной | Хлебобулочные | Целлофан |
| 5 | Батон окский | Хлебобулочные | NULL |
| 6 | Батон подмосковный | Хлебобулочные | Целлофан |
| 7 | Фруктовый сок | Питьевые | Картонная коробка |
| 8 | Макароны | Макаронные изделия | Целлофан |
| 9 | Капуста морская | Консервированные | Консервная банка |
| 10 | Шпроты рижские | Консервированные | Консервная банка |

Таблица ПроизводствоПродуктовПитания

| Код фирмы | Код продукта | Объем производства |
|-----------|--------------|--------------------|
| 1 | 2 | 100 |
| 2 | 2 | 250 |
| 2 | 7 | 300 |
| 3 | 3 | 500 |
| 3 | 4 | 750 |
| 3 | 6 | 180 |
| 3 | 5 | 260 |
| 4 | 1 | 1200 |
| 4 | 2 | 200 |
| 4 | 3 | 350 |
| 4 | 4 | 380 |
| 4 | 5 | 710 |
| 4 | 6 | 220 |
| 5 | 1 | 600 |
| 5 | 2 | 450 |
| 6 | 2 | 500 |
| 6 | 7 | 1400 |
| 9 | 3 | 1200 |
| 10 | 9 | 1600 |
| 10 | 10 | 120 |

Интерфейс:

Необходимо отобразить информацию о продуктах питания, также фото и объем производства.

Вариант 3.

Постановка задачи:

Необходимо хранить информацию о преподавателях (табельный номер, фамилия, должность, кафедра, стаж), дисциплинах (код, название, направление (гуманитарное, техническое и т.д.)), а также о распределении нагрузки по преподавателям с указанием номера группы студентов, семестра и количества часов.

Схема базы данных

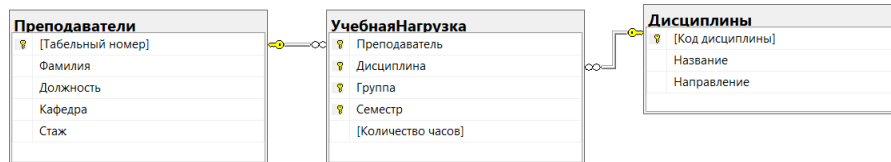


Таблица Преподаватели

| Табельный номер | Фамилия | Должность | Кафедра | Стаж |
|-----------------|----------|-----------|---------|------|
| 11 | Галкина | доцент | ЭВМ | 8 |
| 12 | Чичиков | доцент | ЭВМ | 19 |
| 13 | Иванов | доцент | ЭВМ | 35 |
| 14 | Орлов | NULL | ИФП | 12 |
| 15 | Туркин | доцент | САПР ВС | 21 |
| 16 | Чебышев | профессор | ИФП | 15 |
| 17 | Цаплина | NULL | ВМ | 9 |
| 18 | Павлушин | доцент | САПР ВС | 2 |
| 19 | Чечеткин | доцент | САПР ВС | 5 |
| 20 | Соловьев | ассистент | САПР ВС | NULL |

Таблица Дисциплины

| Код дисциплины | Название | Направление |
|----------------|-----------------------------|----------------|
| 1 | Информатика | NULL |
| 2 | ЭиЭ | техническое |
| 3 | ИиКГ | NULL |
| 4 | История | NULL |
| 5 | Философия | гуманитарное |
| 6 | Математический анализ | математическое |
| 7 | Технологии программирования | техническое |
| 8 | ПУ ЭВМ | техническое |
| 9 | Операционные системы | техническое |
| 10 | Базы данных | техническое |

Таблица УчебнаяНагрузка

| Преподаватель | Дисциплина | Группа | Семестр | Количество часов |
|---------------|------------|--------|---------|------------------|
| 11 | 7 | 641 | 5 | 32 |
| 11 | 10 | 641 | 5 | 68 |
| 12 | 9 | 640 | 5 | 68 |
| 13 | 1 | 840 | 1 | 68 |
| 13 | 1 | 841 | 1 | 68 |
| 14 | 4 | 840 | 1 | 68 |
| 14 | 4 | 841 | 1 | 68 |
| 15 | 3 | 643 | 5 | 68 |
| 16 | 5 | 740 | 4 | 68 |
| 16 | 5 | 748 | 4 | 68 |
| 17 | 6 | 840 | 1 | 51 |
| 17 | 6 | 840 | 2 | 51 |
| 17 | 6 | 841 | 1 | 51 |
| 17 | 6 | 841 | 2 | 51 |
| 18 | 2 | 640 | 5 | 51 |
| 18 | 2 | 641 | 5 | 51 |
| 18 | 2 | 648 | 5 | 51 |
| 19 | 3 | 640 | 5 | 68 |
| 19 | 3 | 641 | 5 | 68 |
| 19 | 8 | 640 | 5 | 32 |

Интерфейс:

Необходимо отобразить информацию о учебной нагрузке, также фото преподавателя.

Вариант 4.

Постановка задачи:

Необходимо хранить информацию о районах различных областей (код, название района, название области, фамилия главы администрации района), культурах (код, название, семейство), а также об урожайности культур.

Схема базы данных

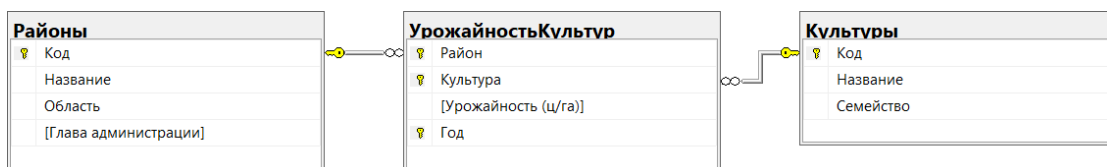


Таблица Районы

| Код | Название | Область | Глава администрации |
|-----|-----------------|--------------------|---------------------|
| 1 | Сараевский | Рязанская | Толмачев |
| 2 | Сасовский | Рязанская | Рыбин |
| 3 | Пронский | Рязанская | Казakov |
| 4 | Спасский | Нижегородская | Евдокимов |
| 5 | Приаргунский | Читинская | Пичуренко |
| 6 | Вельский | Архангельская | Колотилоv |
| 7 | Ейский | Краснодарский край | NULL |
| 8 | Красноармейский | Краснодарский край | Тимофеев |
| 9 | Михайловский | Волгоградская | Семисотов |
| 10 | Кузоватовский | Ульяновская | Вильчик |

Таблица Культуры

| Код | Название | Семейство |
|-----|-----------------|-------------|
| 101 | Пшеница | Злаки |
| 102 | Картофель | Пасленовые |
| 103 | Кукуруза | Злаки |
| 104 | Гречиха | Гречишные |
| 105 | Сахарная свекла | Маревые |
| 106 | Горох | Бобовые |
| 107 | Ячмень | Злаки |
| 108 | Рожь | Злаки |
| 109 | Рис | Злаки |
| 110 | Виноград | Виноградные |

Таблица УрожайностьКультур

| Район | Культура | Урожайность (ц/га) | Год |
|-------|----------|--------------------|------|
| 1 | 101 | 9 | 2018 |
| 1 | 107 | 16 | 2019 |
| 2 | 101 | 27 | 2018 |
| 2 | 102 | 141 | 2020 |
| 2 | 105 | 200 | 2019 |
| 2 | 108 | 17 | 2018 |
| 3 | 101 | 18 | 2019 |
| 3 | 102 | 130 | 2019 |
| 3 | 108 | 20 | 2018 |
| 4 | 101 | 15 | 2020 |
| 4 | 101 | 28 | 2018 |
| 6 | 102 | 173 | 2019 |
| 6 | 102 | 300 | 2018 |
| 7 | 103 | 61 | 2020 |
| 7 | 110 | 42 | 2020 |
| 8 | 109 | 40 | 2018 |
| 9 | 101 | 17 | 2018 |
| 9 | 103 | 67 | 2020 |
| 10 | 106 | 32 | 2019 |
| 10 | 108 | 30 | 2019 |

Интерфейс:

Необходимо отобразить информацию о урожайности культур, также область, семейство и фото культуры.

Вариант 5.

Постановка задачи:

Необходимо хранить информацию о национальностях (код, название, раса, язык), странах (номер страны, название, материк, столица, численность населения), а также об этническом составе каждой отдельной страны (в тыс.).

Схема базы данных

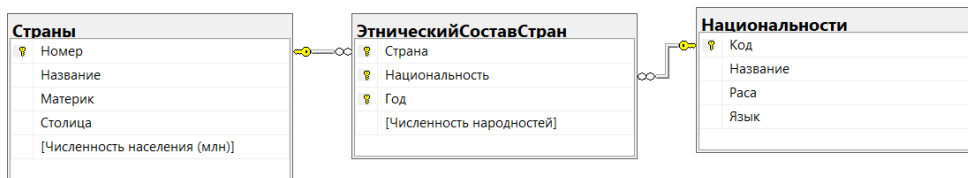


Таблица Национальности

| Код | Название | Раса | Язык |
|-----|----------|--------------|-------------|
| 1 | Русские | Европеоидная | Русский |
| 2 | Украинцы | Европеоидная | Украинский |
| 3 | Эстонцы | Европеоидная | Эстонский |
| 4 | Беларусы | Европеоидная | Белорусский |
| 5 | Хауса | Негроидная | Английский |
| 6 | Арабы | Европеоидная | Арабский |
| 7 | Немцы | Европеоидная | Немецкий |
| 8 | Монголы | Монголоидная | Монгольский |
| 9 | Китайцы | Монголоидная | Китайский |
| 10 | Евреи | Европеоидная | Иврит |

Таблица Страны

| Номер | Название | Материк | Столица | Численность населения (млн) |
|-------|------------|------------------|------------|-----------------------------|
| 1 | Россия | Евразия | Москва | 141 |
| 2 | Украина | Евразия | Киев | 43 |
| 3 | Эстония | Евразия | NULL | NULL |
| 4 | Белоруссия | Евразия | Минск | 9 |
| 5 | Нигерия | Африка | Лагос | 80 |
| 6 | Алжир | Африка | Алжир | 1 |
| 7 | Египет | Африка | Каир | 38 |
| 8 | Канада | Северная Америка | Оттава | 23 |
| 9 | Монголия | Евразия | Улан-Батор | 1,5 |
| 10 | Китай | Евразия | Пекин | 1000 |

Таблица ЭтническийСоставСтран

| Страна | Национальность | Год | Численность народностей |
|--------|----------------|------|-------------------------|
| 1 | 1 | 2019 | 94000 |
| 1 | 6 | 2020 | 0,9 |
| 1 | 7 | 2019 | 1000 |
| 1 | 9 | 2019 | 900 |
| 1 | 9 | 2020 | 2000 |
| 1 | 10 | 2019 | 20000 |
| 2 | 1 | 2020 | 20000 |
| 2 | 3 | 2019 | 2 |
| 3 | 1 | 2019 | 300 |
| 3 | 9 | 2020 | 0,3 |
| 4 | 2 | 2020 | 2000 |
| 5 | 5 | 2020 | 60000 |
| 5 | 6 | 2019 | 10000 |
| 6 | 8 | 2018 | 1 |
| 7 | 6 | 2020 | 35000 |
| 7 | 10 | 2019 | 2000 |
| 8 | 4 | 2020 | 1 |
| 8 | 9 | 2018 | 1000 |
| 9 | 9 | 2019 | 1 |
| 10 | 7 | 2019 | 1 |

Интерфейс:

Необходимо отобразить информацию о урожайности культур, также область, семейство и фото культуры.

Вариант 6.

Постановка задачи:

Необходимо хранить информацию об учащихсся (табельный номер, фамилия, год рождения, контактный телефон), предлагаемых языках (код, название, языковая группа, фамилия руководителя направления), а также о посещении учащимися занятий по конкретным языкам с указанием даты начала занятий, длительности и стоимости курсов.

Схема базы данных

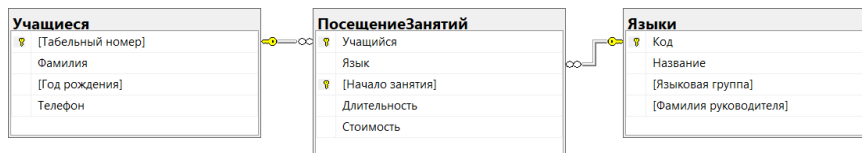


Таблица Языки

| Код | Название | Языковая группа | Фамилия руководителя |
|-----|-------------|-------------------|----------------------|
| 1 | Английский | Западногерманская | Рюмин |
| 2 | Польский | Славянская | Муравьев |
| 3 | Испанский | Романская | Балабанов |
| 4 | Русский | Славянская | Гасперт |
| 5 | Итальянский | Романская | Муравьев |
| 6 | Французский | Романская | Моле |
| 7 | Немецкий | Западногерманская | Исаев |
| 8 | Латинский | Италийская | Бродский |
| 9 | Шведский | Скандинавская | Бергман |
| 10 | Чешский | Славянская | Чудов |

Таблица Учащиеся

| Табельный номер | Фамилия | Год рождения | Телефон |
|-----------------|------------|--------------|---------|
| 1 | Иванов | 1980 | 112211 |
| 2 | Гришин | 1975 | 554637 |
| 3 | Морозов | 1985 | NULL |
| 4 | Троцкий | 1992 | 346712 |
| 5 | Медведев | 1983 | 543321 |
| 6 | Гусев | 1988 | 543622 |
| 7 | Кудрин | 1993 | 234214 |
| 8 | Разин | 1979 | 554325 |
| 9 | Пугачев | 1990 | 367162 |
| 10 | Болотников | 1987 | 123435 |

Таблица ПосещениеЗанятий

| Учащийся | Язык | Начало занятия | Длительность | Стоимость |
|----------|------|-------------------------|--------------|-----------|
| 1 | 7 | 2020-02-25 15:00:00.000 | 2 | 200,0000 |
| 1 | 7 | 2020-04-02 12:00:00.000 | 1 | 100,0000 |
| 2 | 2 | 2020-02-05 11:00:00.000 | 2 | 300,0000 |
| 2 | 2 | 2020-05-12 09:00:00.000 | 3 | 450,0000 |
| 5 | 6 | 2020-05-01 10:00:00.000 | 2 | 450,0000 |
| 4 | 9 | 2020-02-05 14:00:00.000 | 5 | 5000,0000 |
| 4 | 9 | 2020-05-30 12:00:00.000 | 5 | 5000,0000 |
| 5 | 8 | 2020-04-05 14:00:00.000 | 4 | 4000,0000 |
| 5 | 8 | 2020-05-06 09:00:00.000 | 2 | 2000,0000 |
| 6 | 10 | 2020-01-28 20:00:00.000 | 2 | NULL |
| 7 | 4 | 2020-02-20 15:00:00.000 | 3 | 250,0000 |
| 7 | 3 | 2020-05-10 13:00:00.000 | 4 | 2250,0000 |
| 7 | 7 | 2020-05-14 15:00:00.000 | 3 | 500,0000 |
| 7 | 4 | 2020-10-20 12:00:00.000 | 2 | NULL |
| 8 | 4 | 2020-02-02 17:00:00.000 | 1 | 100,0000 |
| 8 | 4 | 2020-08-05 14:00:00.000 | 3 | 2000,0000 |
| 9 | 5 | 2020-08-22 14:00:00.000 | 1 | 300,0000 |
| 10 | 4 | 2020-02-01 11:00:00.000 | 2 | 500,0000 |
| 10 | 5 | 2020-02-01 15:00:00.000 | 4 | 2000,0000 |
| 10 | 3 | 2020-07-03 18:00:00.000 | 3 | 2000,0000 |

Интерфейс:

Необходимо отобразить информацию о учащихсся, а также сумму, потраченную на занятия и фото.

Вариант 7.

Постановка задачи:

Необходимо хранить информацию о предоставляемых в салоне видах услуг (код, название, цена), клиентах (фамилия, адрес, контактный телефон, год рождения), а также о предварительной записи клиентов на услуги с указанием даты и времени посещения.

Схема базы данных

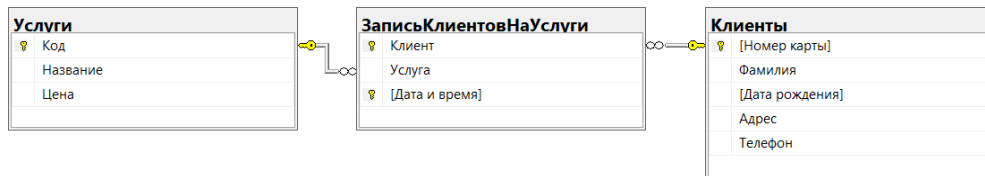


Таблица Услуги

| Код | Название | Цена |
|-----|------------------------|-----------|
| 1 | Стрижка | 150,0000 |
| 2 | Мелирование | 750,0000 |
| 3 | Покраска | 200,0000 |
| 4 | Мытье головы | 100,0000 |
| 5 | Сушка | 100,0000 |
| 6 | Лечение волос | 1000,0000 |
| 7 | Плетение косичек | 500,0000 |
| 8 | Торжественная прическа | 800,0000 |
| 9 | Укладка | 200,0000 |
| 10 | Бритье | 50,0000 |

Таблица Клиенты

| Номер карты | Фамилия | Дата рождения | Адрес | Телефон |
|-------------|----------|---------------|------------------|---------|
| 1 | Иванова | 1975-10-15 | Тимакова, 5 | 325531 |
| 2 | Петров | 1970-08-05 | Зубкова, 6 | 316940 |
| 3 | Сидорова | 1980-11-10 | Гоголя, 26 | 418122 |
| 4 | Гуляев | 1977-05-11 | Лермонтова, 13 | 775020 |
| 5 | Жукова | 1989-02-08 | NULL | 333911 |
| 6 | Васечкин | 1990-10-06 | Циолковского, 10 | NULL |
| 7 | Соколова | 1985-12-20 | Гагарина, 53 | 779750 |
| 8 | Орлов | 1988-06-22 | Дзержинского, 18 | 589551 |
| 9 | Бугреева | 1988-07-30 | Спортивная, 15 | NULL |
| 10 | Уткин | 1979-01-25 | Есенина, 20 | 321520 |

Таблица ЗаписьКлиентовНаУслуги

| Клиент | Услуга | Дата и время |
|--------|--------|-------------------------|
| 2 | 4 | 2020-03-01 12:10:00.000 |
| 2 | 5 | 2020-03-01 15:00:00.000 |
| 3 | 1 | 2020-03-02 10:20:00.000 |
| 3 | 9 | 2020-04-03 15:20:00.000 |
| 4 | 2 | 2020-04-03 17:00:00.000 |
| 5 | 1 | 2020-06-05 10:00:00.000 |
| 5 | 3 | 2020-06-05 12:00:00.000 |
| 5 | 4 | 2020-09-10 17:00:00.000 |
| 5 | 4 | 2020-10-10 12:00:00.000 |
| 6 | 1 | 2020-11-12 12:00:00.000 |
| 6 | 10 | 2020-11-15 10:00:00.000 |
| 7 | 6 | 2020-11-16 12:00:00.000 |
| 9 | 7 | 2020-12-19 10:00:00.000 |
| 8 | 4 | 2020-12-20 10:00:00.000 |
| 8 | 3 | 2020-12-20 17:00:00.000 |
| 8 | 5 | 2020-12-29 10:00:00.000 |
| 3 | 8 | 2020-12-31 10:00:00.000 |
| 5 | 8 | 2020-12-31 12:00:00.000 |
| 7 | 8 | 2020-12-31 15:00:00.000 |
| 9 | 8 | 2020-12-31 17:00:00.000 |

Интерфейс:

Необходимо отобразить информацию о клиентах, а также фото.

Вариант 8.

Постановка задачи:

Необходимо хранить информацию о существующем ателье (номер ателье, название, адрес, телефон), видах предоставляемых услуг (код, название, длительность выполнения), а также о стоимости таких услуг в конкретном ателье.

Схема базы данных

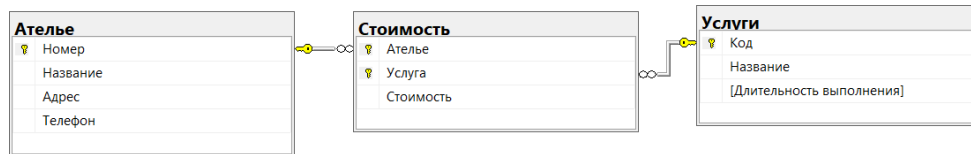


Таблица Ателье

| Номер | Название | Адрес | Телефон |
|-------|--------------|------------------|---------|
| 1 | Золотое Руно | Новая, 28/32 | 250085 |
| 2 | Золушка | Разина, 2А | 445791 |
| 3 | Имидж | Карла Маркса, 11 | 289622 |
| 4 | Канесса | Энгельса, 35 | 445044 |
| 5 | Карина | Дзержинского, 40 | NULL |
| 6 | Кутюрье | Халтурина, 1Б | 324088 |
| 7 | Макошь | Фрунзе, 15 | 456077 |
| 8 | Миранда-А | Крупской, 27 | 557890 |
| 9 | Мода | ЛенКом, 13 | 296057 |
| 10 | Образ | Урицкого, 25 | 411840 |

Таблица Услуги

| Код | Название | Длительность выполнения |
|-----|------------------|-------------------------|
| 1 | Укорачивание | 10 |
| 2 | Подгонка | 22 |
| 3 | Корректировка | 29 |
| 4 | Перекрой | 30 |
| 5 | Изменение фасона | 21 |
| 6 | Замена подкладки | 15 |
| 7 | Замена фурнитуры | 8 |
| 8 | Вышивка | NULL |
| 9 | Пошив | 35 |
| 10 | Штопка | 17 |

Таблица Стоимость

| Ателье | Услуга | Стоимость |
|--------|--------|-----------|
| 1 | 1 | 400,0000 |
| 1 | 2 | 850,0000 |
| 1 | 3 | 700,0000 |
| 1 | 4 | 1000,0000 |
| 2 | 1 | 600,0000 |
| 2 | 10 | 200,0000 |
| 4 | 6 | 500,0000 |
| 4 | 7 | 60,0000 |
| 5 | 9 | 3800,0000 |
| 5 | 10 | 200,0000 |
| 6 | 2 | 780,0000 |
| 7 | 4 | 670,0000 |
| 7 | 5 | 900,0000 |
| 7 | 6 | 540,0000 |
| 8 | 7 | 100,0000 |
| 8 | 9 | 4200,0000 |
| 9 | 5 | 1100,0000 |
| 9 | 9 | 7800,0000 |
| 10 | 1 | 500,0000 |
| 10 | 9 | 400,0000 |

Интерфейс:

Необходимо отобразить информацию о ателье, а также фото.

Вариант 9.

Постановка задачи:

Необходимо хранить информацию о редких растениях (название, семейство, раздел), странах (название, материк, столица), а также о произрастании растений в отдельных странах с указанием приблизительного количества корней за последние три года.

Схема базы данных

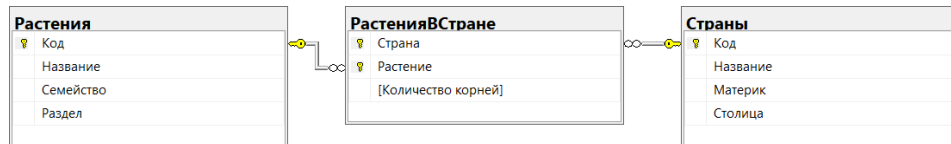


Таблица Растения

| Код | Название | Семейство | Раздел |
|-----|-------------------------|-----------------|-------------------|
| 1 | Мутинус собачий | Веселковые | Грибы |
| 2 | Рогатик пестиковый | Клавариевые | Грибы |
| 3 | Пиррозия язычная | Многоножковые | Папоротниковидные |
| 4 | Лепторумора Микеля | Аспидиевые | Папоротниковидные |
| 5 | Марсилея щетинистая | Марсилеевые | Папоротниковидные |
| 6 | Стереокаулон обнаженный | Стереокаулиевые | NULL |
| 7 | Асахиния Шоландера | Пармелиевые | Лишайники |
| 8 | Кладония вулканная | Кладониевые | Лишайники |
| 9 | Корникулярия степная | Уснеевые | Лишайники |
| 10 | Телосхистес Желтоватый | Телосхистовые | Лишайники |

Таблица Страны

| Код | Название | Материк | Столица |
|-----|-----------|------------------|---------|
| 1 | Россия | Евразия | Москва |
| 2 | Чехия | Евразия | Прага |
| 3 | Германия | Евразия | Берлин |
| 4 | Польша | Евразия | Варшава |
| 5 | Франция | Евразия | Париж |
| 6 | Австралия | Австралия | NULL |
| 7 | Латвия | Евразия | Рига |
| 8 | Канада | Северная Америка | Оттава |
| 9 | Мексика | Северная Америка | Мехико |
| 10 | Нигерия | Африка | Абуджа |

Таблица РастенияВСтране

| Страна | Растение | Количество корней |
|--------|----------|-------------------|
| 1 | 3 | 300 |
| 3 | 1 | 10 |
| 4 | 2 | 500 |
| 5 | 2 | 230 |
| 5 | 5 | 100 |
| 9 | 9 | 56 |
| 1 | 6 | 400 |
| 6 | 5 | 100 |
| 1 | 2 | 560 |
| 5 | 1 | 20 |
| 7 | 7 | 500 |
| 8 | 9 | 140 |
| 9 | 10 | 56 |
| 2 | 5 | 122 |
| 3 | 5 | 166 |
| 3 | 4 | 122 |
| 4 | 9 | 200 |
| 5 | 4 | 20 |
| 6 | 1 | 145 |
| 7 | 2 | 546 |

Интерфейс:

Необходимо отобразить информацию о растениях, а также фото.

Вариант 10.

Постановка задачи:

Необходимо хранить информацию о совхозах области (код, название совхоза, название района, фамилия председателя), видах скота (код, название вида, порода), а также о поголовье скота в различных совхозах за последние три года.

Схема базы данных

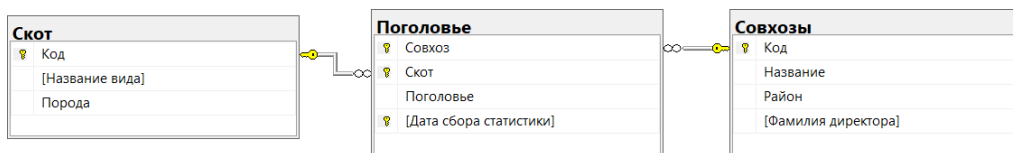


Таблица Совхозы

| Код | Название | Район | Фамилия директора |
|-----|----------|-------------|-------------------|
| 1 | Знамя | Касимовский | Орехов |
| 2 | Заря | Рыбновский | Слюков |
| 3 | Искра | Шацкий | Петров |
| 4 | Воля | Рязанский | Попов |
| 5 | Урожай | Рязанский | Баев |
| 6 | Авангард | Скопинский | Иванов |
| 7 | Победа | Пронский | Зотов |
| 8 | Успех | Спасский | Рынин |
| 9 | Сила | NULL | Пылев |
| 10 | Звезда | Сараевский | Зайцев |

Таблица Скот

| Код | Название вида | Порода |
|-----|---------------|---------------|
| 1 | Бык | Черный |
| 2 | Бык | Красный |
| 3 | Лошадь | Киргизская |
| 4 | Лошадь | Калмыцкая |
| 5 | Свинья | Белая |
| 6 | Овца | Грубошерстная |
| 7 | Курица | Минорка |
| 8 | Утка | Пекинская |
| 9 | Свинья | Эстонская |
| 10 | Овца | Тонкорунная |

Таблица Поголовье

| Совхоз | Скот | Поголовье | Дата сбора статистики |
|--------|------|-----------|-----------------------|
| 1 | 2 | 100 | 2020-01-01 |
| 1 | 5 | 110 | 2020-01-07 |
| 2 | 4 | 95 | 2020-01-01 |
| 2 | 6 | 50 | 2020-02-03 |
| 3 | 1 | 55 | 2020-03-01 |
| 3 | 6 | 220 | 2020-05-04 |
| 4 | 3 | 200 | 2020-01-12 |
| 4 | 8 | 80 | 2020-07-23 |
| 5 | 5 | 75 | 2020-04-14 |
| 5 | 10 | 45 | 2020-09-17 |
| 6 | 2 | 130 | 2020-01-01 |
| 6 | 4 | 111 | 2020-08-19 |
| 7 | 6 | 120 | 2020-01-01 |
| 7 | 7 | 150 | 2020-11-21 |
| 7 | 9 | 91 | 2020-01-31 |
| 8 | 7 | 80 | 2020-01-12 |
| 10 | 1 | 150 | 2020-09-07 |
| 10 | 4 | 90 | 2020-05-11 |
| 10 | 8 | 70 | 2020-06-18 |
| 10 | 10 | 60 | 2020-07-22 |

Интерфейс:

Необходимо отобразить информацию о поголовье, а также породу и фото скота.

Вариант 11.

Постановка задачи:

Необходимо хранить информацию о языках (код, название, языковая группа, вид знаковой системы (кириллица, латиница, иероглифы и т.п.)), странах (код страны, название, материк, столица, количество жителей), а также об этническом составе каждой отдельной страны за последние 3 года с указанием численности населения, говорящего на каждом из языков.

Схема базы данных

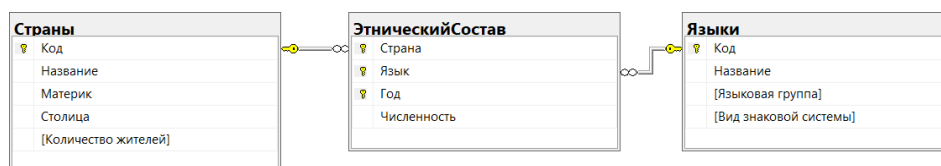


Таблица Страны

| Код | Название | Материк | Столица | Количество жителей |
|-----|----------|------------------|-----------|--------------------|
| 1 | Россия | Евразия | Москва | 143782000 |
| 2 | Франция | Евразия | Париж | 60424000 |
| 3 | США | Северная Америка | Вашингтон | 293027000 |
| 4 | Германия | Евразия | Берлин | 82424000 |
| 5 | Болгария | Евразия | София | 7640000 |
| 6 | Швеция | Евразия | Стокгольм | 9100000 |
| 7 | Япония | NULL | NULL | 127333000 |
| 8 | Китай | Евразия | Пекин | 1300000000 |
| 9 | Украина | Евразия | NULL | 47732000 |
| 10 | Канада | Северная Америка | Оттава | 32507000 |

Таблица Языки

| Код | Название | Языковая группа | Вид знаковой системы |
|-----|--------------|-----------------|----------------------|
| 1 | русский | славянская | кириллица |
| 2 | украинский | славянская | кириллица |
| 3 | белорусский | славянская | кириллица |
| 4 | французский | романская | латиница |
| 5 | американский | германская | латиница |
| 6 | немецкий | германская | NULL |
| 7 | болгарский | славянская | NULL |
| 8 | шведский | германская | латиница |
| 9 | японский | японская | иероглифы |
| 10 | китайский | сино-тибетская | иероглифы |

Таблица ЭтническийСостав

| Страна | Язык | Год | Численность |
|--------|------|------|-------------|
| 1 | 1 | 2020 | 143000000 |
| 1 | 2 | 2019 | 500000 |
| 1 | 2 | 2020 | 600000 |
| 1 | 10 | 2018 | 30000 |
| 1 | 10 | 2019 | 34000 |
| 1 | 10 | 2020 | 40000 |
| 2 | 4 | 2019 | 59000000 |
| 2 | 4 | 2020 | 60000000 |
| 3 | 5 | 2018 | 290000000 |
| 3 | 5 | 2020 | 292000000 |
| 4 | 1 | 2019 | 20000 |
| 4 | 6 | 2019 | 82000000 |
| 5 | 7 | 2020 | 7600000 |
| 6 | 8 | 2019 | 9000000 |
| 6 | 8 | 2020 | 9100000 |
| 7 | 9 | 2020 | 127000000 |
| 8 | 10 | 2018 | 1299000000 |
| 8 | 10 | 2020 | 1300000000 |
| 10 | 4 | 2018 | 30000000 |
| 10 | 4 | 2020 | 32000000 |

Интерфейс:

Необходимо отобразить информацию о странах, а также фото.

Вариант 12.

Постановка задачи:

Необходимо хранить информацию о существующих магазинах (номер магазина, название, адрес, телефон), комплектующих (модель, название вида, фирма-производитель, цена), а также о наличии комплектующих в конкретных магазинах с указанием их количества за последние три года.

Схема базы данных

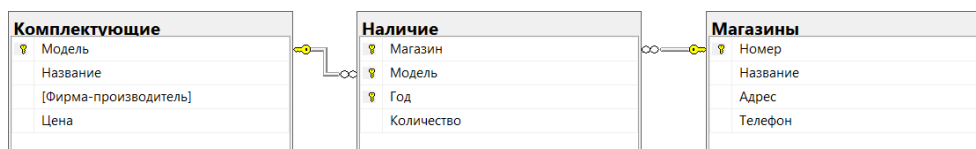


Таблица Комплектующие

| Модель | Название | Фирма-производитель | Цена |
|--------|---------------------|---------------------|------------|
| 22 | Очки для компьютера | MATSUDA | 700,0000 |
| 27 | Компьютерная мышь | SAMSUNG | 500,0000 |
| 113 | Компьютерная мышь | BENQ | 400,0000 |
| 222 | Клавиатура | NULL | 530,0000 |
| 276 | Привод DVD+RW | Asus | 1000,0000 |
| 336 | Ноутбук | BENQ | 36000,0000 |
| 377 | Монитор | Acer | 6000,0000 |
| 576 | Колонки | DAEWOO | 3500,0000 |
| 900 | Чистящие салфетки | NULL | 250,0000 |
| 3000 | Коврик для мыши | NULL | 100,0000 |
| 5100 | Ноутбук | HP | 27000,0000 |

Таблица Магазины

| Номер | Название | Адрес | Телефон |
|-------|--------------------|-----------------------|---------|
| 1 | Компьютерный мир | Советская, 36 | 336772 |
| 2 | Техносила | Грибоедова, 32 | 554411 |
| 3 | НИКС | Полетаева, 104 | 123456 |
| 4 | Эльдорадо | Солнечная, 2 | 5561789 |
| 5 | Техномир | Гагарина, 4 | NULL |
| 6 | Формоза | Циолковского, 75 | 757522 |
| 7 | Альт | Свободы, 99 | 276349 |
| 8 | От А до Я | Новоселов, 22 | NULL |
| 9 | НИТИ | Октябрьская, 83 | NULL |
| 10 | Все для компьютера | Интернациональная, 44 | 192837 |

Таблица Наличие

| Магазин | Модель | Год | Количество |
|---------|--------|------|------------|
| 1 | 336 | 2019 | 50 |
| 1 | 336 | 2020 | 70 |
| 2 | 576 | 2018 | 60 |
| 3 | 377 | 2019 | 20 |
| 3 | 576 | 2018 | 30 |
| 3 | 5100 | 2019 | 40 |
| 3 | 5100 | 2020 | 30 |
| 4 | 222 | 2018 | 63 |
| 5 | 3000 | 2018 | 300 |
| 5 | 3000 | 2019 | 350 |
| 5 | 3000 | 2020 | 250 |
| 6 | 113 | 2019 | 300 |
| 6 | 576 | 2019 | 90 |
| 8 | 377 | 2018 | 40 |
| 8 | 900 | 2018 | 300 |
| 9 | 22 | 2019 | 200 |
| 9 | 22 | 2020 | 250 |
| 10 | 222 | 2018 | 90 |
| 10 | 222 | 2019 | 80 |
| 10 | 222 | 2020 | 100 |

Интерфейс:

Необходимо отобразить информацию о комплектующих, а также общее количество и фото.

Вариант 13.

Постановка задачи:

Необходимо хранить информацию о существующих в городе фирмах (код, название, адрес, фамилия директора), постоянных клиентах (табельный номер, фамилия, адрес, телефон), а также о турах, заказанных клиентами с указанием страны, даты отъезда и стоимости тура.

Схема базы данных

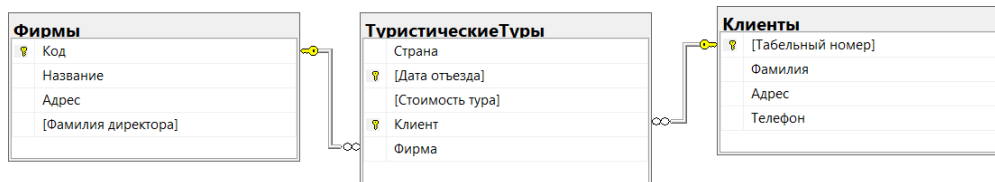


Таблица Фирмы

| Код | Название | Адрес | Фамилия директора |
|-----|-------------|-----------------------|-------------------|
| 1 | Алые паруса | Вокзальная, 11 | Шахов |
| 2 | Арго | Есенина, 72 | Маркова |
| 3 | БризТур | Ленина, 24 | NULL |
| 4 | Визази | Ленина, 10 | Хрипин |
| 5 | Гулливер | NULL | Горскина |
| 6 | Компас | Первомайский пр-т, 15 | Воронин |
| 7 | Лиоктур | Колхозный пр-т, 15 | Звягина |
| 8 | Колумб | Первомайский пр-т, 82 | Бурмистров |
| 9 | Магеллан | Вокзальная, 34 | Колткова |
| 10 | Спутник | Садовая, 22 | Осипова |

Таблица Клиенты

| Табельный номер | Фамилия | Адрес | Телефон |
|-----------------|------------|-----------------|---------|
| 1 | Бараненков | Сенная, 8 | 256397 |
| 2 | Кузнецова | Маяковского, 9 | 765220 |
| 3 | Макаренко | Соборная, 15 | 281099 |
| 4 | Гусев | Красноярская, 3 | 902190 |
| 5 | Крюкова | Гагарина, 77 | 678243 |
| 6 | Митасов | Почтовая, 58 | 152976 |
| 7 | Азарьева | Кольцова, 1 | 148934 |
| 8 | Данилова | Горького, 17 | 910543 |
| 9 | Воронцов | Введенская, 110 | 239844 |
| 10 | Афанасьева | Татарская, 15 | 234588 |

Таблица ТуристическиеТуры

| Страна | Дата отъезда | Стоимость тура | Клиент | Фирма |
|----------------|--------------|----------------|--------|-------|
| Абхазия | 2020-07-10 | 2000,0000 | 7 | 8 |
| Австрия | 2020-08-29 | 30000,0000 | 8 | 3 |
| ОАЭ | 2020-01-17 | 20000,0000 | 10 | 2 |
| Болгария | 2020-01-01 | 12000,0000 | 1 | 5 |
| Великобритания | 2020-11-07 | 18000,0000 | 4 | 9 |
| Германия | 2020-03-13 | 15000,0000 | 8 | 8 |
| Греция | 2020-11-19 | 25000,0000 | 1 | 1 |
| Египет | 2020-04-27 | 35000,0000 | 4 | 1 |
| Индонезия | 2020-10-15 | 39000,0000 | 10 | 4 |
| Испания | 2020-02-02 | 57000,0000 | 1 | 2 |
| Италия | 2020-09-13 | 55000,0000 | 6 | 8 |
| Кипр | 2020-02-14 | 40000,0000 | 9 | 5 |
| Россия | 2020-04-27 | 10000,0000 | 2 | 10 |
| Тунис | 2020-09-18 | 42000,0000 | 9 | 3 |
| Турция | 2020-12-22 | 30000,0000 | 2 | 9 |
| Финляндия | 2020-04-27 | 40000,0000 | 7 | 6 |
| Франция | 2020-05-30 | 27000,0000 | 5 | 6 |
| Хорватия | 2020-10-08 | 13000,0000 | 5 | 4 |
| Чехия | 2020-03-25 | 14000,0000 | 2 | 1 |
| Швейцария | 2020-06-06 | 50000,0000 | 6 | 1 |

Интерфейс:

Необходимо отобразить информацию о турах, а также адрес, телефон и фото клиента.

Вариант 14.

Постановка задачи:

Необходимо хранить информацию о существующих организациях города (код, название, адрес, телефон, количество сотрудников), о распространяемых изданиях (индекс, название, тип [газета, журнал и т.п.], количество страниц, цена), а также о подписке организаций на издания с указанием даты начала подписки, количества месяцев и скидки на стоимость подписки в процентах.

Схема базы данных

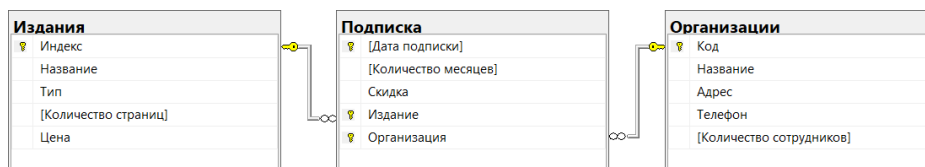


Таблица Организации

| Код | Название | Адрес | Телефон | Количество сотрудников |
|-----|------------------|------------------------|---------|------------------------|
| 1 | ООО МобПро | Декабристов, 12 | 745504 | 100 |
| 2 | ООО Опекс | Ленинградский пр-т, 62 | 771619 | 50 |
| 3 | ООО Росгидромет | Кулузовский пр-т, 26 | NULL | 30 |
| 4 | ООО Балттекстиль | Мичуринский пр-т, 3 | 925116 | 150 |
| 5 | ООО СтройДом | Ленина, 3 | 283455 | 40 |
| 6 | ЗАО Власко | Первомайский пр-т, 7 | 695836 | 15 |
| 7 | ООО Интертехно | Гагарина, 5 | NULL | 10 |
| 8 | ООО Авиценна | Почтовая, 9 | 678433 | 7 |
| 9 | ООО Артурпроект | Полевая, 2 | NULL | 8 |
| 10 | ОАО Биопрепарат | Первомайский пр-т, 14 | 678723 | 9 |

Таблица Издания

| Индекс | Название | Тип | Количество страниц | Цена |
|--------|---------------|--------|--------------------|----------|
| 1 | Панорама | газета | 30 | 12,0000 |
| 2 | Телесемь | газета | 40 | 15,0000 |
| 3 | I Love You | NULL | 60 | 30,0000 |
| 4 | Лиза | журнал | 70 | 35,0000 |
| 5 | Из рук в руки | газета | 50 | 7,0000 |
| 6 | Ярмарка | NULL | 100 | 10,0000 |
| 7 | Maxim | журнал | 150 | 90,0000 |
| 8 | Мир новостей | газета | 70 | 15,0000 |
| 9 | Сosmo | журнал | 100 | 100,0000 |
| 10 | Voi | журнал | 80 | 60,0000 |

Таблица Подписка

| Дата подписки | Количество месяцев | Скидка | Издание | Организация |
|---------------|--------------------|--------|---------|-------------|
| 2020-01-06 | 5 | 0,02 | 1 | 3 |
| 2020-01-06 | 3 | 0,01 | 9 | 5 |
| 2020-01-08 | 7 | 0,12 | 1 | 7 |
| 2020-02-21 | 3 | 0 | 2 | 5 |
| 2020-03-15 | 3 | 0,02 | 2 | 8 |
| 2020-03-29 | 7 | 0 | 3 | 6 |
| 2020-04-03 | 4 | 0,5 | 3 | 10 |
| 2020-04-12 | 1 | 0,01 | 4 | 1 |
| 2020-04-14 | 8 | 0,02 | 4 | 10 |
| 2020-05-10 | 12 | 0,13 | 4 | 6 |
| 2020-05-17 | 10 | 0,11 | 3 | 7 |
| 2020-06-12 | 1 | 0,01 | 6 | 2 |
| 2020-06-19 | 8 | 0 | 9 | 3 |
| 2020-07-24 | 6 | 0 | 7 | 3 |
| 2020-08-30 | 3 | 0 | 6 | 8 |
| 2020-09-07 | 2 | 0,02 | 6 | 1 |
| 2020-09-16 | 5 | 0,01 | 1 | 9 |
| 2020-10-03 | 3 | 0 | 7 | 2 |
| 2020-11-28 | 2 | 0 | 10 | 7 |
| 2020-12-01 | 6 | 0,01 | 10 | 6 |

Интерфейс:

Необходимо отобразить информацию о подписках, а также количество страниц, цена и фото издания.

Вариант 15.

Постановка задачи:

Необходимо хранить информацию о дисциплинах (код, название, количество часов), о преподавателях (табельный номер, ФИО, должность, возраст, телефон), а также о проведении занятий с указанием дня недели, времени, группы и аудитории.

Схема базы данных

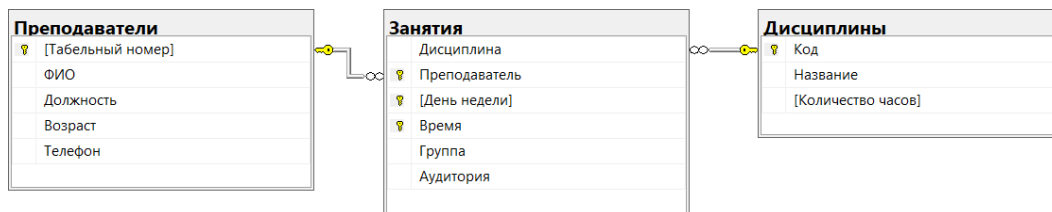


Таблица Дисциплины

| Код | Название | Количество часов |
|-----|--------------------------|------------------|
| 1 | Базы данных | 64 |
| 2 | Экономика | 32 |
| 3 | Математический анализ | 51 |
| 4 | Философия | NULL |
| 5 | Программирование | 64 |
| 6 | Физика | 32 |
| 7 | Математическая логика | 51 |
| 8 | Организация ЭВМ и систем | 64 |
| 9 | Информатика | 64 |
| 10 | Основы ИТ | 32 |

Таблица Преподаватели

| Табельный номер | ФИО | Должность | Возраст | Телефон |
|-----------------|------------------------------|-----------------------|---------|---------|
| 1 | Галкина Наталья Николаевна | Доцент | 26 | 345519 |
| 2 | Смирнов Максим Анатольевич | Доцент | 46 | 255563 |
| 3 | Майоров Валентин Васильевич | Профессор | 30 | 211577 |
| 4 | Щавелев Анатолий Анатольевич | Доцент | 41 | 554812 |
| 5 | Пушкин Александр Николаевич | Доцент | 50 | 985601 |
| 6 | Орлов Александр Павлович | Доцент | 51 | 475056 |
| 7 | Павлов Александр Викторович | Доцент | 31 | 448003 |
| 8 | Есенина Светлана Ивановна | Доцент | 32 | 389908 |
| 9 | Устинов Дмитрий Игоревич | Старший преподаватель | 30 | 542132 |
| 10 | Белкин Андрей Витальевич | Доцент | 40 | 219965 |

Таблица Занятия

| Дисциплина | Преподаватель | День недели | Время | Группа | Аудитория |
|------------|---------------|-------------|----------|--------|-----------|
| 1 | 1 | Вторник | 15:20:00 | 643 | 210 |
| 1 | 1 | Понедельник | 09:55:00 | 640 | 210 |
| 5 | 1 | Среда | 11:40:00 | 641 | 210 |
| 9 | 1 | Четверг | 13:35:00 | 648 | 210 |
| 2 | 2 | Понедельник | 09:55:00 | 740 | 268 |
| 2 | 2 | Пятница | 08:10:00 | 721 | 312 |
| 3 | 3 | Пятница | 11:40:00 | 741 | 466 |
| 3 | 3 | Четверг | 13:35:00 | 748 | 310 |
| 4 | 4 | Четверг | 13:35:00 | 745 | 311 |
| 5 | 5 | Вторник | 11:40:00 | 543 | 206 |
| 5 | 5 | Понедельник | 09:55:00 | 541 | 206 |
| 6 | 6 | Вторник | 11:40:00 | 746 | 358 |
| 7 | 7 | Пятница | 11:40:00 | 748 | 333 |
| 7 | 7 | Четверг | 09:55:00 | 740 | 333 |
| 5 | 8 | Пятница | 08:10:00 | 740 | 208 |
| 8 | 8 | Среда | 09:55:00 | 741 | 210 |
| 8 | 9 | Вторник | 11:40:00 | 840 | 208 |
| 9 | 9 | Среда | 09:55:00 | 841 | 208 |
| 8 | 9 | Вторник | 09:55:00 | 740 | 206 |
| 5 | 9 | Среда | 15:20:00 | 743 | 206 |

Интерфейс:

Необходимо отобразить информацию о занятиях, а также фото преподавателя.