**ПРОГРАММНЫЙ ПРОДУКТ**

**«ХОСТЕЛ»**

**ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ**

2022

**СОДЕРЖАНИЕ**

[**1. Описание функциональной деятельности процесса 3**](#_Toc119409777)

[**2. Требования к хранимым данным 7**](#_Toc119409778)

[**3. Связь Lazarus и Access 10**](#_Toc119409779)

[**4. Тестовые сценарии 12**](#_Toc119409780)

[**ПРИЛОЖЕНИЕ А Листинг тестов 17**](#_Toc119409781)

[**ПРИЛОЖЕНИЕ Б Выполнение тест-кейсов 22**](#_Toc119409782)

1. Описание функциональной деятельности процесса

На схеме отражены входные данные и полученные результаты.

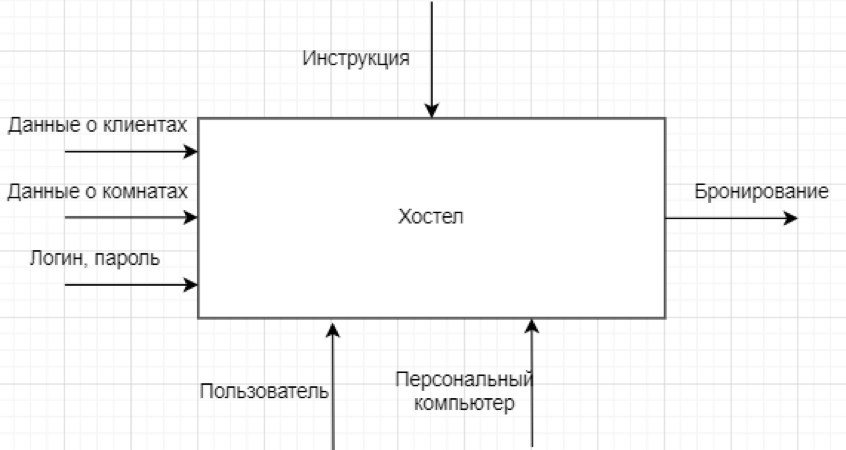


Рисунок 1 – Контекстная диаграмма

Входные данные приложения:

* Данные о клиентах - Фамилия, Имя, Отчество, почта, телефон клиента.
* Данные о комнатах - Название комнаты, количество мест в комнате.
* Логин и пароль администратора для использования базы данных из приложения.

Выходные данные:

* Бронирование клиента в базе данных.
* Создание комнаты в базе данных.
* Занесение клиента в базу данных.

Требования к запуску проекта:

* Наличие персонального компьютера;
* Пользователь.

Приложение Хостел выполняет функции:

* Занесения данных о новых клиентах в базу данных;
* Просмотр таблицы с информацией о клиентах хостела;
* Поиск клиентов по фамилии;
* Создание новых комнат хостела;
* Бронирование номера хостела.

Схема добавления новой записи.

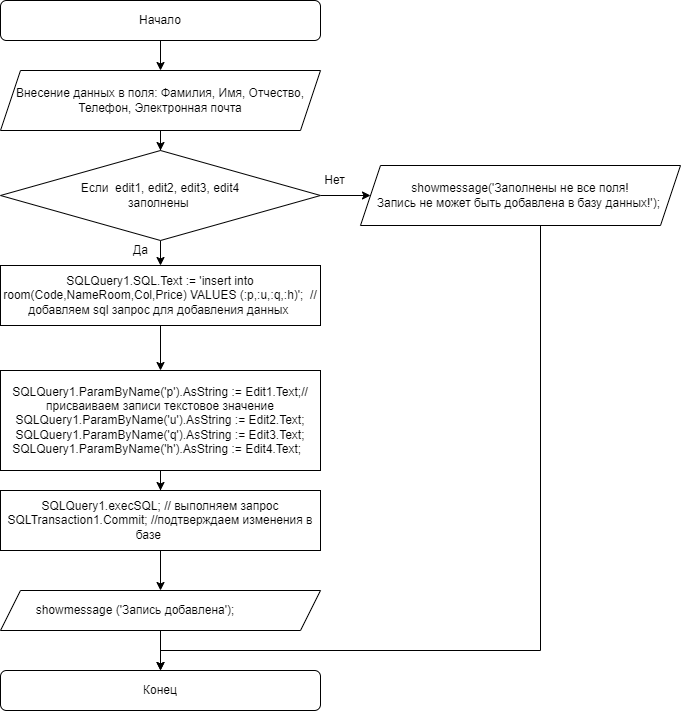


Рисунок 2 – Добавление записи

Схема редактирования записи.

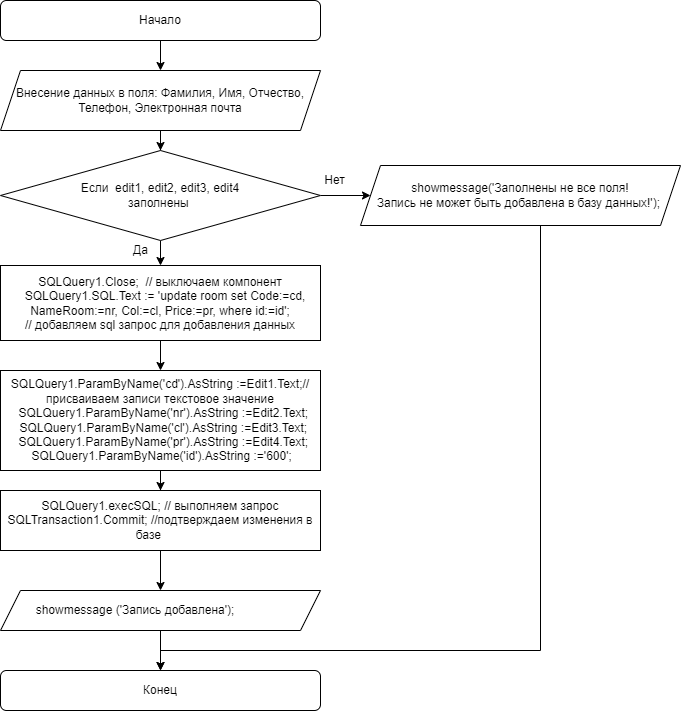


Рисунок 3 – Редактирование записи

Схема статистического запроса, определяющего количество зарегистрированных гостей.



Рисунок 4 – Количество гостей

1. Требования к хранимым данным

Для хранения данных о пользователях и комнатах используется база данных.

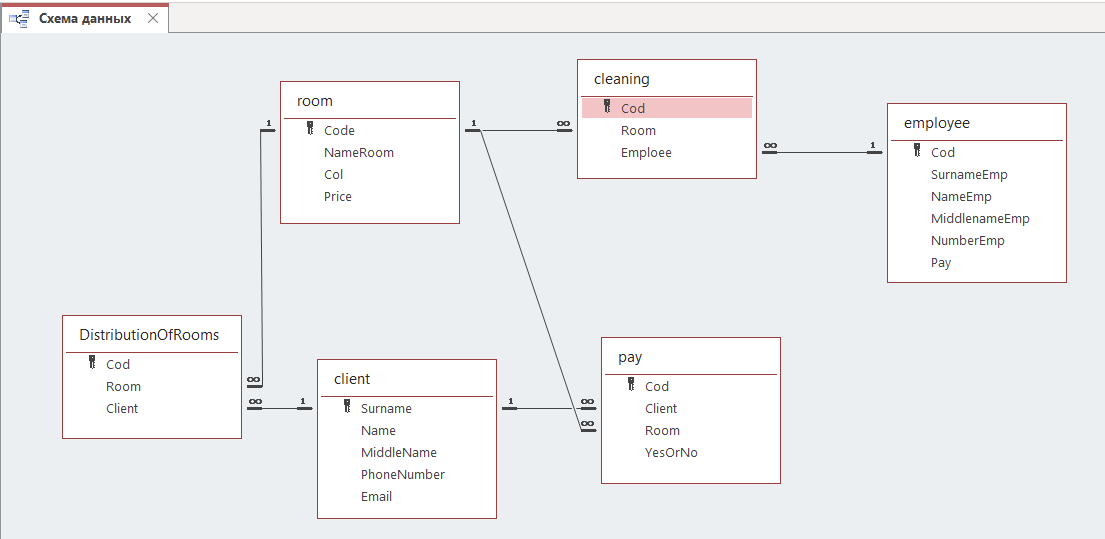


Рисунок 5 – Схема данных

Для хранения данных о клиентах используется таблица client.

Таблица 1

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Surname | Короткий текст | Фамилия клиента |
| Name | Короткий текст | Имя клиента |
| MiddleName | Короткий текст | Отчество клиента |
| PhoneNumber | Короткий текст | Номер телефона |
| Email | Короткий текст | Адрес электронной почты |

Таблица client связана с таблицей:

* Didtribution Of Rooms по полям Surname – client 1 ко многим.
* Pay по полям Surname – client 1 ко многим.

Для хранения данных о комнатах используется таблица room.

Таблица 2

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Code | Числовой | Номер комнаты |
| NameRoom | Короткий текст | Название комнаты |
| Col | Числовой | Количество мест в комнате |
| Price | Денежный | Цена места |

Таблица room связана с таблицей:

* Didtribution Of Rooms по полям Code – Room 1 ко многим.
* Cleaning по полям Code – Room 1 ко многим.
* Pay по полям Code – Room 1 ко многим.

Для хранения данных о сотрудниках организации используется таблица employee.

Таблица 3

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Code | Числовой | Номер сотрудника |
| SurnameEmp | Короткий текст | Фамилия сотрудника |
| NameEmp | Короткий текст | Имя сотрудника |
| MiddleNameEmp | Короткий текст | Отчество сотрудника |
| NumberEmp | Короткий текст | Номер телефона |
| Pay | Денежный | Заработная плата сотрудника |

Таблица employee связана с таблицей:

* Cleaning по полям Code – Employee 1 ко многим.

Отображение забронированных комнат происходит в таблице Didtribution Of Rooms.

Таблица 4

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Code | Числовой | Код бронирования |
| Room | Числовой | Номер комнаты |
| Client | Короткий текст | Фамилия клиента |
| StartDate | Дата и время | Дата заезда |
| EndDate | Дата и время | Дата выезда |
| People | Числовой | Количество человек |

Таблица Didtribution Of Rooms связана с таблицей:

* Client по полям client – Surname1 ко многим.
* Pay по полям client – Surname 1 ко многим.

Информация об оплаченных комнатах храница в таблице pay.

Таблица 5

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Cod | Числовой | Код оплаты |
| Client | Короткий текст | Фамилия клиента |
| Room | Числовой | Номер комнаты |
| YesOrNo | Логический | Оплачено Да/Нет |

Таблица pay связана с таблицей:

* Client по полям client – Surname 1 ко многим.
* Room по полям Room – Code 1 ко многим.

Информация о уборке комнат содержится в таблице cleaning.

Таблица 6

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Значение |
| Cod | Числовой | Код уборки |
| Room | Числовой | Номер комнаты |
| Code | Числовой | Номер сотрудника |

Таблица cleaning связана с таблицей:

* Room по полям Room – Code 1 ко многим.
* Employee по полям Employee – Code 1 ко многим.

1. Связь Lazarus и Access

Системный DSN

* Вводим в поиске меню Пуска слово ODBC, запускаем 64-bit версию.
* Переходим в меню Системный DSN(System DSN)
* Нажимаем кнопку Добавить

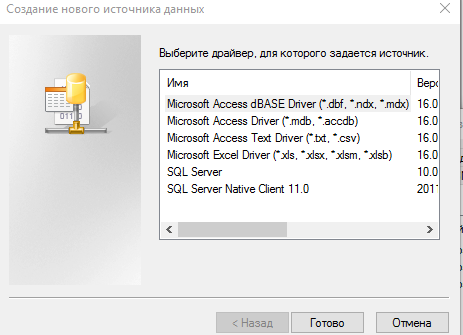


Рисунок 6 – Создание нового источника данных

Теперь у вас есть всё необходимое, нажимаем по Microsoft Access Driver(\*mdb, \*accdb).

Открылось такое окно – Рисунок 7.

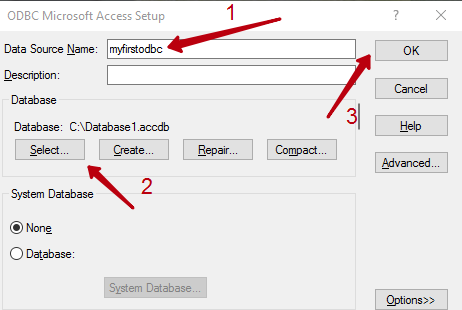


Рисунок 7 – Занесение своей базы данных

Заполняем Data Source Name, называя «hostel».

Далее выбираем нашу БД, нажав кнопку Select.

Нажимаем ОК и уже переходим в Lazarus.

В Lazarus на форму добавлены следующие компоненты:

Из вкладки Data Controls - TDBGrid.

Из вкладки Data Access - TDataSource

Из вкладки SQLDb - TODBCConection, SQLTransaction и SQLQuery

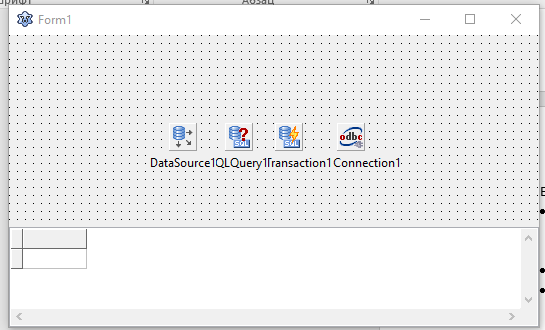


Рисунок 8 – Компоненты формы

1. Тестовые сценарии

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1.AddClient |
| Название тестирования | Сохранение данных клиента при регистрации. |
| Резюме испытания | Заполнение полей регистрации: Фамилия, Имя, Отчество, Телефон, E-mail. Занесение введенных данных в таблицу базы – client. |
| Шаги тестирования | Включение компонента связи с базой данных Access.  Присвоение полям текстовых значений.  Выполнение запроса с помощью компонента SQLQuery.  Подтверждение изменений в базе данных с помощью компонента SQLTransaction. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery.  Данные для ввода. |
| Ожидаемый результат | Внесение новых данных в таблицу client. |
| Фактический результат | Внесение новых данных в таблицу client. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1.AddRoom |
| Название тестирования | Сохранение данных о комнате. |
| Резюме испытания | Заполнение полей: Номер комнаты, Название комнаты, Количество мест в комнате, Цена бронирования одного места комнаты. Занесение введенных данных в таблицу базы – room. |
| Шаги тестирования | Включение компонента связи с базой данных Access.  Присвоение полям текстовых значений.  Выполнение запроса с помощью компонента SQLQuery.  Подтверждение изменений в базе данных с помощью компонента SQLTransaction. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery.  Данные для ввода. |
| Ожидаемый результат | Внесение новых данных в таблицу room. |
| Фактический результат | Внесение новых данных в таблицу room. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1.DeleteRoom |
| Название тестирования | Удаление комнаты. |
| Резюме испытания | Выбор комнаты из списка и ее удаление. |
| Шаги тестирования | Выделение поля из таблицы TBGrid. Нажатие кнопки удалить. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery. TBGrid. |
| Ожидаемый результат | Удаление информации о комнате в таблице room. |
| Фактический результат | Удаление информации о комнате в таблице room. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1.EditRoom |
| Название тестирования | Изменение данных о комнате. |
| Резюме испытания | Заполнение полей: Номер комнаты, Название комнаты, Количество мест в комнате, Цена бронирования одного места комнаты. Занесение измененных данных в таблицу базы – room. |
| Шаги тестирования | Включение компонента связи с базой данных Access.  Присвоение полям текстовых значений.  Выполнение запроса с помощью компонента SQLQuery.  Подтверждение изменений в базе данных с помощью компонента SQLTransaction. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery.  Данные для ввода. |
| Ожидаемый результат | Внесение измененных данных в таблицу room. |
| Фактический результат | Внесение измененных данных в таблицу room. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1. ViewStatistics |
| Название тестирования | Просмотр статистики. |
| Резюме испытания | Открытие формы, на которой выводится количество зарегистрированных клиентов хостела. |
| Шаги тестирования | Включение компонента связи с базой данных Access.  Выполнение запроса о количестве записей таблицы с помощью компонента SQLQuery.  Отображение результата в таблице TBGrid. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery. TBGrid. |
| Ожидаемый результат | Вывод количества записей таблицы client. |
| Фактический результат | Вывод количества записей таблицы client. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1. SearchClient |
| Название тестирования | Поиск клиента. |
| Резюме испытания | Заполнение поля Фамилия. Вывод клиентов, имеющих введенную фамилию. |
| Шаги тестирования | Ввод в поле поиска ‘Иванов’. Нажатие кнопки 'Поиск'.  Включение компонента связи с базой данных Access.  Выполнение запроса с помощью компонента SQLQuery.  Вывод данных в таблице TBGrid. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery. TBGrid,  Данные для ввода. |
| Ожидаемый результат | Вывод зарегистрированных клиентов с фамилией ‘Иванов’. |
| Фактический результат | Вывод зарегистрированных клиентов с фамилией ‘Иванов’. |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1. LoginVerificationClient |
| Название тестирования | Вывод информации о клиентах при авторизации пользователя. |
| Резюме испытания | Заполнение полей регистрации: Логин, Пароль. Вывод страницы «Клиенты». |
| Шаги тестирования | Ввод значений логина(«admin») и пароля(«1234»).  Открытие страницы «Клиенты». |
| Данные тестирования | Компоненты ввода Edit, кнопка Button. |
| Ожидаемый результат | Вывод страницы «Клиенты». |
| Фактический результат | Вывод страницы «Клиенты». |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1. LoginVerificationRoom |
| Название тестирования | Вывод информации о комнатах при авторизации пользователя. |
| Резюме испытания | Заполнение полей регистрации: Логин, Пароль. Вывод страницы «Комнаты». |
| Шаги тестирования | Ввод значений логина(«admin») и пароля(«1234»).  Открытие страницы «Комнаты». |
| Данные тестирования | Компоненты ввода Edit, кнопка Button. |
| Ожидаемый результат | Вывод страницы «Комнаты». |
| Фактический результат | Вывод страницы «Комнаты». |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| Идентификатор тест-кейса | TTestCase1. Booking |
| Название тестирования | Бронирование комнаты. |
| Резюме испытания | Внесение информации о бронировании комнаты и выполнение её резервирования. |
| Шаги тестирования | Заполнение данных о Заезде/Выезде, ввод количества резервируемых мест, ввод фамилии клиента, выбор комнаты. Отправка данных о регистрации брони в базу данных. |
| Данные тестирования | Компоненты связи Lazarus и Access: TDataSource, TODBCConection, SQLTransaction, SQLQuery. TBGrid,  Данные для ввода. |
| Ожидаемый результат | Вывод страницы «Подтверждения бронирования». |
| Фактический результат | Вывод страницы «Подтверждения бронирования». |
| Статус *(Выполнен/Провален)* | Выполнен. |
| Листинг процедуры | Приложение А |

**ПРИЛОЖЕНИЕ А  
 Листинг тестов**

unit TestCase1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, fpcunit, testutils, testregistry, Unit1, Unit2, Unit3, Unit5, Unit6, Unit7, Unit8;

type

TTestCase1= class(TTestCase)

protected

procedure SetUp; override;

procedure TearDown; override;

published

procedure AddClient;

procedure AddRoom;

procedure DeleteRoom;

procedure EditRoom;

procedure ViewStatistics;

procedure SearchClient;

procedure LoginVerificationClient;

procedure LoginVerificationRoom;

procedure OutputInformation;

procedure Booking;

procedure Transition;

end;

implementation

procedure TTestCase1.AddRoom;

begin

Form7.SQLQuery1.Close; // выключаем компонент

Form7.SQLQuery1.SQL.Text := 'insert into room(Code,NameRoom,Col,Price) VALUES (:p,:u,:q,:h)'; // добавляем sql запрос для добавления данных

Form7.SQLQuery1.ParamByName('p').AsString := '1';// присваиваем записи текстовое значение

Form7.SQLQuery1.ParamByName('u').AsString := 'Xtnsht[vtcnyfz';

Form7.SQLQuery1.ParamByName('q').AsString := '4';

Form7.SQLQuery1.ParamByName('h').AsString := '2000';

Form7.SQLQuery1.execSQL; // выполняем запрос

Form7.SQLTransaction1.Commit; //подтверждаем изменения в базе

end;

procedure TTestCase1.AddClient;

begin

Form2.SQLQuery1.Close; // выключаем компонент

Form2.SQLQuery1.SQL.Text := 'insert into client (Surname,Name,MiddleName,PhoneNumber,Email) values(:p,:u,:r,:t,:z);'; // добавляем sql запрос для добавления данных

Form2.SQLQuery1.ParamByName('p').AsString := 'Фамилия';// присваиваем записи текстовое значение

Form2.SQLQuery1.ParamByName('u').AsString := 'Имя';

Form2.SQLQuery1.ParamByName('r').AsString := 'Отчество';

Form2.SQLQuery1.ParamByName('t').AsString := '89124562134';

Form2.SQLQuery1.ParamByName('z').AsString := 'el\_pochta@mail.com';

Form2.SQLQuery1.ExecSQL; // выполняем запрос

Form2.SQLTransaction1.Commit; //подтверждаем изменения в базе

end;

procedure TTestCase1.DeleteRoom;

begin

Form7.SQLQuery1.Delete;

end;

procedure TTestCase1.EditRoom;

begin

Form7.SQLQuery1.Close; //выключаем компонент

Form7.SQLQuery1.SQL.Text := 'update room set Code:=cd, NameRoom:=nr, Col:=cl, Price:=pr, where id:=id';

//добавляем sql запрос для добавления данных

Form7.SQLQuery1.ParamByName('p').AsString := '1';// присваиваем записи текстовое значение

Form7.SQLQuery1.ParamByName('u').AsString := 'Xtnsht[vtcnyfz';

Form7.SQLQuery1.ParamByName('q').AsString := '4';

Form7.SQLQuery1.ParamByName('h').AsString := '2000';

Form7.SQLQuery1.ParamByName('id').AsString :='600';

Form7.SQLQuery1.execSQL; // выполняем запрос

Form7.SQLTransaction1.Commit; //подтверждаем изменения в базе

end;

procedure TTestCase1.ViewStatistics;

begin

Form8.Show;

Form6.Hide;

end;

procedure TTestCase1.SearchClient;

var s1 :string;

begin

s1:=’Иванов’;

Form6.SQLQuery1.Close;// закрываем датасет

Form6.SQLQuery1.SQL.Text := 'select \* from client where Surname = :s1'; // добавляем наш запрос

Form6.SQLQuery1.ParamByName('s1').AsString:=edit1.text;// присваиваем требуемый параметр

Form6.SQLQuery1.Open;// открываем запрос

end;

procedure TTestCase1.LoginVerificationClient;

var c,k: string;

begin

c:='admin';

k:='1234';

Form5.Edit1.Text=c;

Form5.Edit2.Text=k;

Form6.Show;

Form5.Hide;

end;

procedure TTestCase1.LoginVerificationRoom;

var c,k: string;

begin

c:='admin';

k:='1234';

Form5.Edit1.Text=c;

Form5.Edit2.Text=k;

Form7.Show;

Form5.Hide;

end;

procedure TTestCase1.OutputInformation;

begin

Form3.Label7.Caption := Form1.MaskEdit1.Text;

Form3.Label8.Caption := Form1.MaskEdit2.Text;

Form3.Label9.Caption := Form1.Edit1.Text;

end;

procedure TTestCase1.Booking;

begin

// проверить заполнены ли все поля

if (edit1.Text<>'') then

begin

Form3.SQLQuery1.Close; // выключаем компонент

Form3.SQLQuery1.SQL.Text := 'insert into DistributionOfRooms(Room,Client,StartDate,EndDate,People) VALUES (:p,:u,:r,:o,:t)'; // добавляем sql запрос для добавления данных

Form3.SQLQuery1.ParamByName('p').AsString := '1';// присваиваем записи текстовое значение

Form3.SQLQuery1.ParamByName('u').AsString := Edit1.Text;

Form3.SQLQuery1.ParamByName('r').AsString := Label7.Caption;

Form3.SQLQuery1.ParamByName('o').AsString := Label8.Caption;

Form3.SQLQuery1.ParamByName('t').AsString := Label9.Caption;

Form3.SQLQuery1.ExecSQL; // выполняем запрос

Form3.SQLTransaction1.Commit; //подтверждаем изменения в базе

Form4.Show;

Form3.Hide;

end

else

showmessage('Заполните поле "Фамилия"');

end;

procedure TTestCase1.Transition;

begin

Form1.Show;

Form4.Hide;

end;

procedure TTestCase1.SetUp;

begin

end;

procedure TTestCase1.TearDown;

begin

end;

initialization

RegisterTest(TTestCase1);

end.

**ПРИЛОЖЕНИЕ Б  
 Выполнение тест-кейсов**

