КУРСОВОЕ ПРОЕКТИРОВАНИЕ ПО БАЗАМ ДАННЫХ

ТЕМА: "Телефонная компания"

Одесса 2010

**Аннотация**

**Объектом исследования** является развитие навыков проектирования информационных систем и закрепление знаний, полученных за время изучения дисциплины "Базы данных".

**Цель работы** – разработка и создание базы данных для предметной области Аварийная служба.

**Метод исследования** – изучение видов деятельности в служба рекламы, составление и отладка программы, для работы с базой данных (программа написана на языке С++).

**Содержание**

Введение

1. Анализ предметной области

1.1 Рабочие цели системы

1.2 Этапы проектирования БД

1.3 Концептуальная модель

2. Преобразование модели ER в реляционную модель

3. Запросы к базе данных системы

4. Описание групп пользователей ИС

5. Текст программы

Заключение

Список литературы

**Введение**

**Цель работы**: является развитие навыков проектирования информационных систем и закрепление знаний, полученных за время изучения дисциплины "Базы данных".

**Задание:** предметная область "Реклама в коммерческом издании". Возможные виды деятельности: учет клиентов с возможными льготами; учет совершенных ими заказов по различным направлениям; расчет стоимости заказа с учетом типа, даты продолжительности; оплата клиентами услуг.

**1. Анализ предметной области**

**1.1 Рабочие цели системы**

Согласно заданию, программный продукт должен фиксировать такие данные как:

- клиентов с возможными льготами;

- заказ клиентов по различным типам;

- стоимость заказа с учетом типа, даты и продолжительности;

- оплата клиентом услуг.

В связи с этим были сформированы следующие таблицы в базе данных:

Клиенты;

Услуги;

Звонки;

Льготы.

А также таблицы, связывающие данные таблицы.

Так же ПП должен выполнять следующие функции:

Вывод списка услуг клиента;

Вывод списка заказов клиента;

Вывод списка клиентов по первым буквам;

Вывод списка заказов за данный промежуток;

Вывод списка заказов в данный промежуток;

Вывод количества клиентов с данным типом услуги;

Вывод количества неоплаченых счетов;

Вывод количества клиентов по услугам;

Вывод количества клиентов по льготам;

Вывод клиента с наибольшим долгом;

Для каждой льготы определить клиента с наименьшим сроком льготы;

Для каждой услуги определить а клиента с наименьшим сроком услуги;

Вывод клиентов, не имеющих льгот;

Вывод клиентов, не имеющих услуг;

Вывод состояний счетов;

Вывод состояний сроков услуг;

Обновить наличие долга;

Обновить отсутствие долга.

**1.2 Этапы проектирования БД**

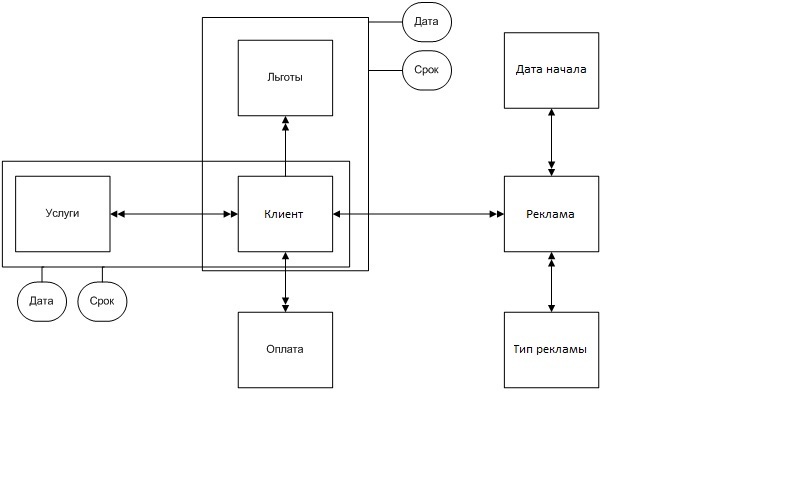
После анализа предметной области были сформированы ранее перечисленные таблицы.

В таблице "Клиент" был выбран первичный ключ "Идентификатор клиента", т.к. данного ключа достаточно для уникального определения каждого клиента, поскольку данные характеризующие клиента не могут повториться дважды.

В таблице "Услуги" и "Льготы" были также выбраны первичные ключи "Идентификатор", т.к. данных полей достаточно для уникального определения данных в таблицах.

При выборе данных таблиц и их связующих таблиц, поля были выбраны с учетом нормализации отношений, т.е. таблицы не содержат избыточное количество повторяющихся данных и не могут привести к различным родам аномалий.

**1.3 Концептуальная модель**



**2. Преобразование модели ER в реляционную модель**

|  |  |  |  |
| --- | --- | --- | --- |
| Клиент | | | |
| Поле | Описание | Тип Данных | Комментарий |
| id\_klient | Идентификационный код клиента | Счетчик | Необходим для уникального определения клиента |
| surname | Фамилия клиента | Текстовый | Данные о фамилии клиента |
| ab\_name | Имя клиента | Текстовый | Данные об имени клиента |
| Num | Номер телефона | Числовой | Данные о номера телефона клиента |
| bill | Состояние счета клиента | Денежный | Текущее состояние счета клиента |
| More Info | Информация о наличие или отсутствии долга | Текстовый | Дополнительные данные о состоянии счета |

|  |  |  |  |
| --- | --- | --- | --- |
| Реклама | | | |
| Поле | Описание | Тип Данных | Комментарий |
| id\_call | Идентификационный код рекламы | Счетчик | Необходим для уникального определения рекламы |
| id\_klient | Идентификационный код клиента | Числовой | Необходим для определения клиента |
| Data | Дата рекламы | Дата/время | Определение даты рекламы |
| time\_beg | Время начала звонка | Дата/время | Определение времени начала рекламы |
| time\_and | Время окончания звонка | Дата/время | Определение времени окончания рекламы |
| id\_type | Код типа рекламы | Числовой | Определение типа рекламы |

|  |  |  |  |
| --- | --- | --- | --- |
| Льгота | | | |
| Поле | Описание | Тип Данных | Комментарий |
| Id\_lgot | Идентификационный код льготы | Счетчик | Необходим для уникального определения льготы |
| Opisanie | Данные о льготе | Текстовый | Описание льготы |
| Stoimost | Процент от стоимости звонка | Числовой | Процент от стоимости рекламы |

|  |  |  |  |
| --- | --- | --- | --- |
| Услуга | | | |
| Поле | Описание | Тип Данных | Комментарий |
| Id\_service | Идентификационный код услуги | Счетчик | Необходим для уникального определения услуги |
| Opisanie | Данные об услуге | Текстовый | Описание услуги |
| Stoimost | Процент от стоимости звонка | Числовой | Процент от стоимости звонка |

|  |  |  |  |
| --- | --- | --- | --- |
| Направление | | | |
| Поле | Описание | Тип Данных | Комментарий |
| Id\_type | Идентификационный код типа | Счетчик | Необходим для уникального определения типа |
| Opisanie | Данные о типе | Текстовый | Описание типа |
| Stoimost | Процент от стоимости рекламы | Числовой | Процент от стоимости рекламы |

|  |  |  |  |
| --- | --- | --- | --- |
| Оплата | | | |
| Поле | Описание | Тип Данных | Комментарий |
| id\_oplata | Идентификационный код оплаты | Счетчик | Необходим для уникального определения оплаты |
| Data | Дата оплаты | Дата/время | Данные о дате внесения оплаты |
| Summ | Внесенная сумма | Денежный | Данные о внесенной сумме |
| id\_ klient | Идентификационный код клиента | Числовой | Определение клиента, выполнившего оплату |

|  |  |  |  |
| --- | --- | --- | --- |
| Клиент-льгота | | | |
| Поле | Описание | Тип Данных | Комментарий |
| id\_lg\_ab | Идентификационный код клиент-льготы | Счетчик | Необходим для уникального определения клиент-льготы |
| Data | Дата присвоения льготы | Дата/время | Данные о дате присвоения льготы |
| Srok | Дата завершения льготы | Дата/время | Данные о дате завершения льготы |
| id\_lgot | Идентификационный код льготы | Числовой | Определение присвоенной льготы |
| id\_klient | Идентификационный код клиента | Числовой | Определение клиента |

|  |  |  |  |
| --- | --- | --- | --- |
| Клиент-услуга | | | |
| Поле | Описание | Тип Данных | Комментарий |
| id\_ys\_ab | Идентификационный код клиент-услугы | Счетчик | Необходим для уникального определения клиент-услугы |
| Data | Дата присвоения услуги | Дата/время | Данные о дате присвоения услуги |
| Srok | Дата завершения услуги | Дата/время | Данные о дате завершения услуги |
| id\_lgot | Идентификационный код услуги | Числовой | Определение присвоенной услуги |
| id\_klient | Идентификационный код клиента | Числовой | Определение клиента |

**3. Запросы к базе данных системы**

Вывод списка услуг клиента

SELECT A.id\_service, A.opisanie, C.surname FROM services AS A, ys\_ab AS B, Klient AS C WHERE A.id\_service=B.id\_service and B.id\_klient=C.id\_klient and C.surname=(Фамилия\_Пользователя)

Вывод списка звонков клиента

SELECT A.id\_call, A.data, A.time\_beg, A.time\_and, C.opisanie, B.surname FROM Call AS A, Klient AS B, Napravlenie AS C WHERE A.id\_klient=B.id\_klient And A.id\_napravlenie=C.id\_napravlenie And B.surname=(Фамилия\_Пользователя)

Вывод списка клиентов по первым буквам

SELECT \* FROM Klient WHERE surname like Первые\_буквы%

Вывод списка клиентов по первым цифрам

SELECT \* FROM Klient WHERE num like Первые\_цифры%

Вывод списка звонков за данный промежуток

SELECT \* FROM call WHERE data Between Начальная\_Дата and Конечная\_Дата

Вывод списка услуг в данный промежуток

SELECT A.Data, A.Srok, B.Surname FROM ys\_ab AS A, Klient AS B WHERE A.id\_klient=B.id\_klient and A.Data between Начальная\_Дата and Конечная\_Дата

Вывод количества клиентов с данной услугой

SELECT Count(\*) FROM services WHERE services.id\_service=Идентификато\_услуги

Вывод количества отрицательных счетов

SELECT count(\*) FROM Klient WHERE bill<0;

Вывод количества клиентов по услугам

SELECT A.id\_service, B.opisanie, Count(\*) FROM ys\_ab AS A, services AS B WHERE A.id\_service=B.id\_service GROUP BY A.id\_service, B.opisanie;

Вывод количества клиентов по льготам

SELECT A.id\_lgot, B.opisanie, Count(\*)FROM lg\_ab AS A, lgot AS B WHERE A.id\_lgot=B.id\_lgot GROUP BY A.id\_lgot, B.opisanie;

Вывод клиента с наибольшим долгом

SELECT \* FROM klient AS A WHERE bill<= ALL (select bill from klient );

Вывод клиентов, имеющих услуги

SELECT \* FROM klient AS A WHERE A.id\_klient =ANY (select B.id\_klient from ys\_ab AS B);

Для каждой льготы определить клиента с наименьшим сроком льготы

SELECT A.opisanie, B.surname, B.ab\_name FROM lgot AS A, klient AS B, lg\_ab AS C WHERE A.id\_lgot=C.id\_lgot AND B.id\_klient=c.id\_klient AND C.srok<=All (select Srok from lg\_ab where lg\_ab.id\_lgot=A.id\_lgot);

Для каждой услуги определить клиента с наименьшим сроком услуги

SELECT A.opisanie, B.surname, B.ab\_name FROM services AS A, klient AS B, ys\_ab AS C WHERE A.id\_service=C.id\_service AND B.id\_klient=c.id\_klient AND C.srok<=All (select Srok from ys\_ab where ys\_ab.id\_service=A.id\_service);

Вывод клиентов, не имеющих льгот

SELECT A.surname FROM Klient AS A WHERE A.id\_klient not IN (select id\_klient from lg\_ab);

Вывод клиентов, не имеющих услуг

SELECT A.surname FROM Klient AS A WHERE A.id\_klient not IN (select id\_klient from ys\_ab);

Вывод состояний счетов

SELECT id\_klient,surname,ab\_name,'norm' from Klient where bill>0 UNION SELECT id\_klient,surname,ab\_name,'minus' from Klient where bill<0;

Вывод состояний сроков услуг

SELECT \*,Srok-Now() FROM ys\_ab where (Srok-Now())>0 union SELECT \*,Srok-Now() FROM ys\_ab where (Srok-Now())<0;

Обновить наличие долга

update klient set [More info]='imeet dolg' where id\_klient in (select id\_klient from klient where bill<0);

Обновить отсутствие долга

update klient set [More info]='ne imeet dolg' where id\_klient in (select id\_klient from klient where bill>0);

4. **Описание групп пользователей ИС**

В данной области были выделены две группы пользователей.

Первая группа – менеджер-оператор:

- возможность просмотра таблиц "Клиенты", "Рекламы", "Льготы-Клиенты", "Услуги-Клиенты", "Оплаты".

- возможность выполнения всех перечисленных в анализе требований действий.

- возможность внесения оплаты.

Вторая группа – администратор:

- возможность просмотра таблиц "Клиенты", "Услуги", "Льготы-Клиенты", "Услуги-Клиенты", "Оплаты".

- возможность выполнения всех перечисленных в анализе требований действий.

- возможность внесения оплаты.

- возможность просмотра журнала событий, произошедших в системе.

- возможность изменения всех таблиц (кроме журнала событий):

-редактирование записей.

-добавление записей.

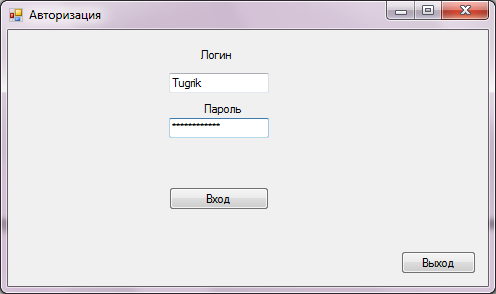
- удаление записей.

- возможность создания нового пользователя.

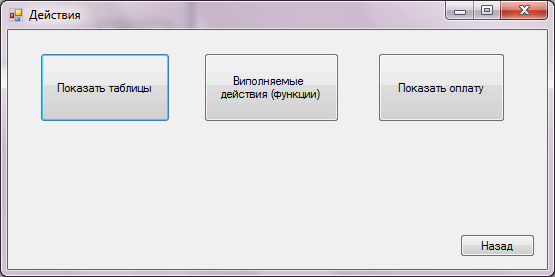
*Интерфейс ИС*

Программный продукт содержит одно окно, которое в зависимости от выполняемых действий преобразуется под выполняемое действие

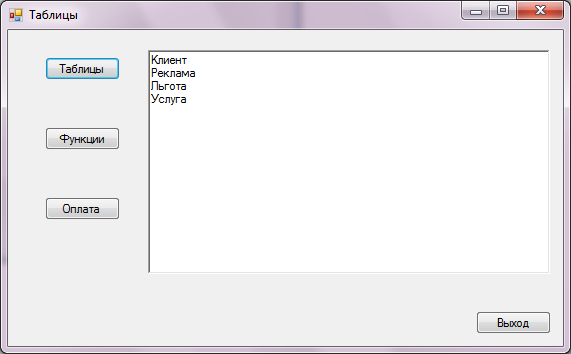
1. Первоначальное окно программы:



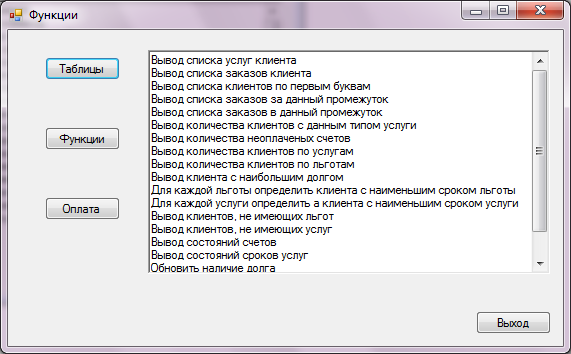
1. Окно программы, которое видит менеджер-оператор после ввода своего логина и пароля:



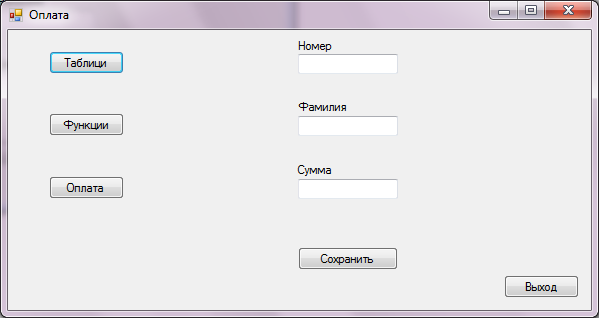
1. Окно программы, которое видит менеджер-оператор после нажатия кнопки "Таблицы":



Окно программы, которое видит менеджер-оператор после нажатия кнопки "Функции":

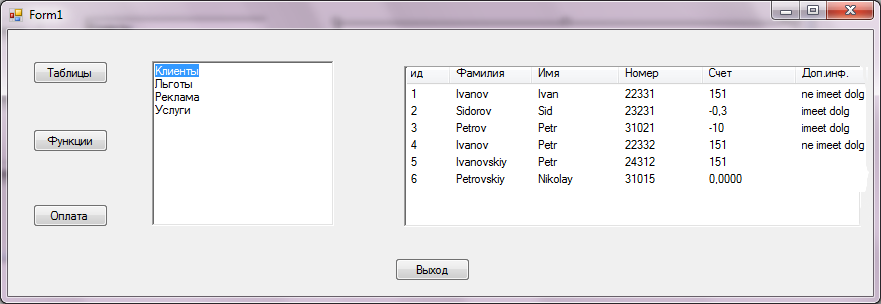


1. Окно программы, которое видит менеджер-оператор после нажатия кнопки "Оплата":

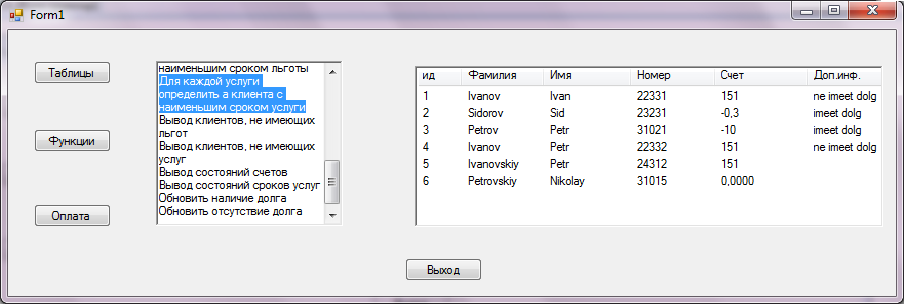


При этом, пользователь вводит номер в поле ввода номера и автоматически вписывается фамилия в поле фамилии, которою в последствии пользователь должен согласовать с клиентом.

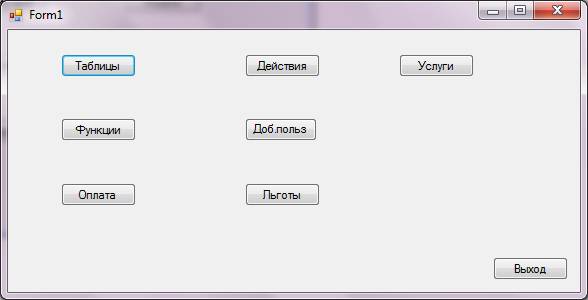
1. Окно программы, которое видит менеджер-оператор после выбора пункта в меню "Таблицы":



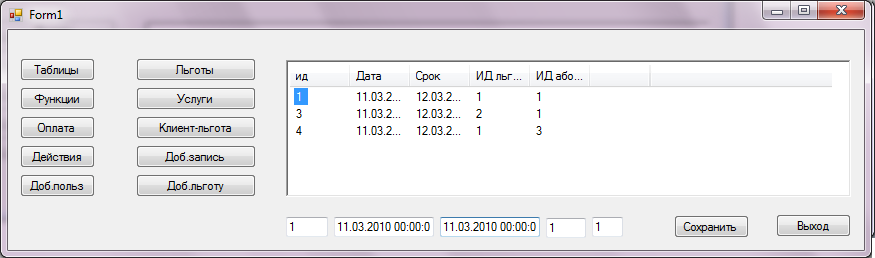
1. Окно программы, которое видит менеджер-оператор после выбора пункта в меню "Функции":



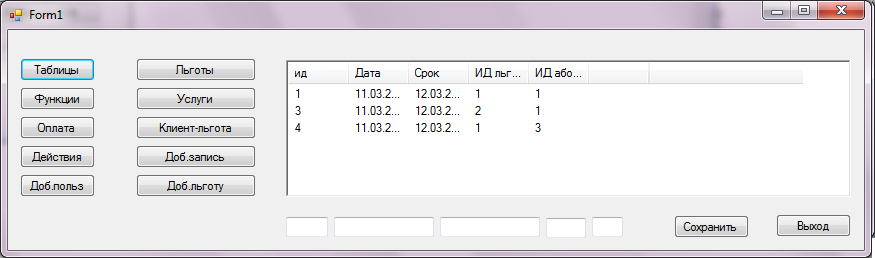
1. Окно программы, которое видит администратор после ввода своего логина и пароля:



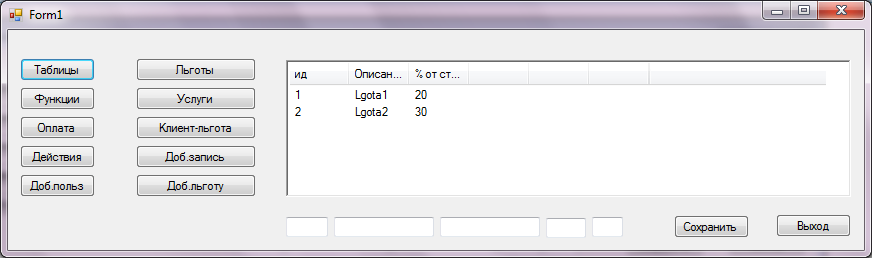
1. Окно программы, которое видит администратор после нажатия кнопки "Изменить Клиент-Льгота":



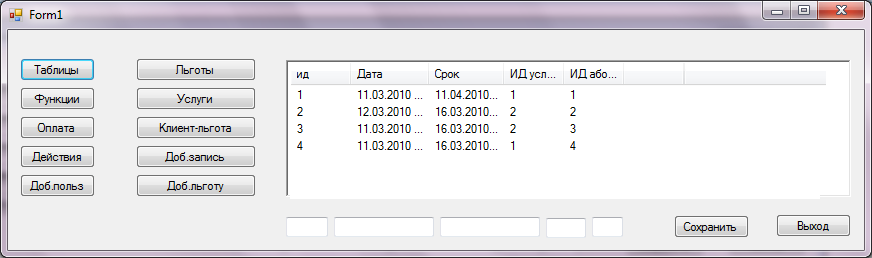
1. Окно программы, которое видит администратор после нажатия кнопки "Добавить запись":



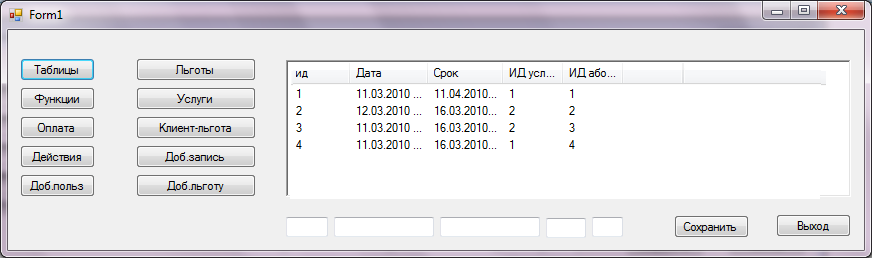
1. Окно программы, которое видит администратор после нажатия кнопки "Добавить льготу":



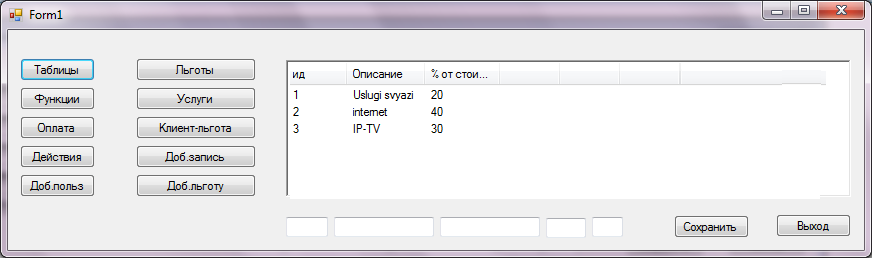
1. Окно программы, которое видит администратор после нажатия кнопки "Изменить Клиент-Услуга":



1. Окно программы, которое видит администратор после нажатия кнопки "Добавить запись":



1. Окно программы, которое видит администратор после нажатия кнопки "Добавить услугу":



*Сообщения программы, причины, их вызывающие, и реакция пользователя на сообщения*

* Сообщение о неверном имени пользователя или пароля.

При возникновении данной ошибки пользователю выводится сообщение о возникновении данной ситуации, очищается поле пароля и система ждет повторного ввода.

* Сообщение о неверном внесении оплаты.

При возникновении данной ошибки пользователю выводится сообщение о возникновении данной ситуации, очищаются все поля ввода. Требуется ввести верные данные.

* Сообщение о неверном внесении нового пользователя.

При возникновении данной ошибки пользователю выводится сообщение о возникновении данной ситуации, очищаются все поля ввода. Требуется ввести верные данные.

**5. Тексты программы**

#pragma once

int zap=0;//переменная для определения текущего запроса

int id=0;//переменная для определения текущего пользователя в системе

int level=0;//переменная для определения выдаваемых функций пользователю

int op\_new=-1;//переменная для определения действий кнопки "добавить"

int ChoiseToLV2=0;//переменная, определяющая выводимую и вводимую информацию в функциях.

//участок кода, реализующий один из запросов на вывод таблиц в программе..в основном в остальных запросах изменяется лишь значение переменной q, в которую и вписывается сам запрос.

private: System::Void listBox1\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e) {

this->listView1->Visible=true;

this->ClientSize = System::Drawing::Size(960, 580);

this->listView1->Items->Clear();

cn->Open();

int choise=this->listBox1->SelectedIndex;

if(choise==0)

{

String ^q="Select \* from Klient;";

System::Data::OleDb::OleDbCommand^cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

System::Data::OleDb::OleDbDataReader ^rdr=cmd->ExecuteReader();

int k=0;

while(rdr->Read())

{

this->columnHeader1->Text=L"ид";

this->listView1->Items->Add(Convert::ToString(rdr[0]));

this->columnHeader2->Text=L"Фамилия";

this->listView1->Items[k]->SubItems->Add(Convert::ToString(rdr[1]));

this->columnHeader3->Text=L"Имя";

this->listView1->Items[k]->SubItems->Add(Convert::ToString(rdr[2]));

this->columnHeader4->Text=L"Номер";

this->listView1->Items[k]->SubItems->Add(Convert::ToString(rdr[3]));

this->columnHeader5->Text=L"Счет";

this->listView1->Items[k]->SubItems->Add(Convert::ToString(rdr[4]));

this->columnHeader6->Text=L"Доп.инф.";

this->listView1->Items[k]->SubItems->Add(Convert::ToString(rdr[5]));

this->columnHeader7->Text=L"";

k++;

}

rdr->Close();

cn->Close();}}

//действия при нажатии на кнопку "Внести" в меню "Оплата".

private: System::Void button11\_Click(System::Object^ sender, System::EventArgs^ e) {

String ^q;

System::Data::OleDb::OleDbCommand ^cmd;

System::Data::OleDb::OleDbDataReader ^rdr;

cn->Open();

if(op\_new==1)

{

double summ\_ins=Convert::ToDouble(this->textBox8->Text);

double summ;

int chosen\_id;

q="Select \* from klient";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

while(rdr->Read())

{

if(Convert::ToString(rdr[1])==this->comboBox1->Text && Convert::ToInt32(rdr[3])==Convert::ToInt32(this->comboBox2->Text))

{

chosen\_id=Convert::ToInt32(rdr[0]);

summ=summ\_ins+Convert::ToDouble(rdr[4]);

q="update klient set bill ='"+Convert::ToString(summ)+"' where id\_klient="+chosen\_id+";";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

q="INSERT INTO Oplata ( Data, summ, id\_klient )values (Date(),'"+Convert::ToString(summ\_ins)+"','"+chosen\_id+"'); ";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

}

}

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Внесение суммы "+Convert::ToString(summ\_ins)+" на счет клиента с ID "+chosen\_id+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

}

//действия при нажатии на кнопку "Изменить"

private: System::Void button13\_Click(System::Object^ sender, System::EventArgs^ e) {

cn->Open();

if(ChoiseToLV2==1)

{

String ^q="update lg\_ab SET Data='"+this->textBox10->Text+"',Srok='"+this->textBox11->Text+"',id\_lgot="+Convert::ToInt32(this->textBox12->Text)+",id\_klient="+Convert::ToInt32(this->textBox13->Text)+" where id\_lg\_ab="+Convert::ToInt32(this->textBox9->Text);

System::Data::OleDb::OleDbCommand ^cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

System::Data::OleDb::OleDbDataReader ^rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Изменение записи в таблице клиент-льгота с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==4)

{

String ^q="update ys\_ab SET Data='"+this->textBox10->Text+"',Srok='"+this->textBox11->Text+"',id\_service="+Convert::ToInt32(this->textBox12->Text)+",id\_klient="+Convert::ToInt32(this->textBox13->Text)+" where id\_ys\_ab="+Convert::ToInt32(this->textBox9->Text);

System::Data::OleDb::OleDbCommand ^cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

System::Data::OleDb::OleDbDataReader ^rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Изменение записи в таблице клиент-услуга с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

cn->Close();

}

//действия при нажатии на кнопку "Удалить"

private: System::Void button14\_Click(System::Object^ sender, System::EventArgs^ e) {

cn->Open();

System::Data::OleDb::OleDbCommand ^cmd;

System::Data::OleDb::OleDbDataReader ^rdr;

if(ChoiseToLV2==1)

{

String ^q="delete from lg\_ab where id\_lg\_ab="+Convert::ToInt32(this->textBox9->Text);

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Удаление записи в таблице клиент-льгота с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==2)

{

String ^q="delete from lgot where id\_lgot="+Convert::ToInt32(this->textBox9->Text);

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Удаление записи в таблице льгота с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==3)

{

String ^q="delete from Klient where id\_klient="+Convert::ToInt32(this->textBox9->Text);

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Удаление записи в таблице Клиент с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==4)

{

String ^q="delete from ys\_ab where id\_ys\_ab="+Convert::ToInt32(this->textBox9->Text);

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Удаление записи в таблице клиент-услуга с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==5)

{

String ^q="delete from services where id\_service="+Convert::ToInt32(this->textBox9->Text);

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Удаление записи в таблице услуга с ID "+Convert::ToInt32(this->textBox9->Text)+"');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

rdr->Close();

cn->Close();

}

//Действия при нажатии на кнопку "Добавить"

private: System::Void button16\_Click(System::Object^ sender, System::EventArgs^ e) {

cn->Open();

System::Data::OleDb::OleDbCommand ^cmd;

System::Data::OleDb::OleDbDataReader ^rdr;

if(ChoiseToLV2==1)

{

String ^q="insert into lg\_ab (Data,Srok,id\_lgot,id\_klient) VALUES('"+this->textBox10->Text+"','"+this->textBox11->Text+"',"+Convert::ToInt32(this->textBox12->Text)+","+Convert::ToInt32(this->textBox13->Text)+")";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Добавление записи в таблицу клиент-льгота');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==2)

{

String ^q="insert into Lgot (Opisanie,Stoimost)VALUES('"+this->textBox10->Text+"',"+Convert::ToInt32(this->textBox11->Text)+")";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Добавление записи в таблицу Льготы');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==3)

{

String ^q="insert into Klient (Surname,ab\_name,Num,bill)VALUES('"+this->textBox10->Text+"','"+this->textBox11->Text+"',"+Convert::ToInt32(this->textBox12->Text)+","+Convert::ToDouble(this->textBox13->Text)+")";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Добавление нового клиента');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==4)

{

String ^q="insert into ys\_ab (Data,Srok,id\_service,id\_klient) VALUES('"+this->textBox10->Text+"','"+this->textBox11->Text+"',"+Convert::ToInt32(this->textBox12->Text)+","+Convert::ToInt32(this->textBox13->Text)+")";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Добавление записи в таблицу клиент-услуга');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

if(ChoiseToLV2==5)

{

String ^q="insert into services (Opisanie,Stoimost)VALUES('"+this->textBox10->Text+"',"+Convert::ToInt32(this->textBox11->Text)+")";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

q="insert into changes(id\_user,data,Opisanie) values("+id+",Now(),'Добавление записи в таблицу услуги');";

cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

rdr=cmd->ExecuteReader();

rdr->Close();

}

rdr->Close();

cn->Close();

}

//Действия, происходящие при внесении данных в поле ввода номера

private: System::Void comboBox2\_TextUpdate(System::Object^ sender, System::EventArgs^ e) {

cn->Open();

String ^q="SELECT surname FROM Klient WHERE num like '"+this->comboBox2->Text+"%';";

System::Data::OleDb::OleDbCommand^cmd=gcnew System::Data::OleDb::OleDbCommand(q,cn);

System::Data::OleDb::OleDbDataReader ^rdr=cmd->ExecuteReader();

while(rdr->Read())

{

this->comboBox1->Text=Convert::ToString(rdr[0]);

}

rdr->Close();

cn->Close();

}

**Заключение**

Данная курсовая работа заключала в себе анализ и создание базы данных для аварийной службы.

Целью курсовой работы является развитие навыков проектирования информационных систем и закрепление знаний, полученных за время изучения дисциплины "Базы данных".

Цели и задачи были полностью выполнены. Это означает, что материал по дисциплине "База данных" был полностью усвоен и принят за основу моему дальнейшему развитию.

**Список литературы**

1. Дейт, К. Дж. Введение в системы Баз Данных, 7-е издание.: Пер. с англ. Ю.Г. Гордиенко; Под ред. А.В. Слепцова / К. Дж. Дейт - М.: Издательский дом "Вильямс", 2001. – 1072с.
2. Роб, П., Коронел, К. Системы баз данных: проектирование, реализация и управление.- 5-е изд., перераб. и доп./ Пер. с англ. А. Никифоров; Под ред. Е. Кондукова / П. Роб, К. Коронел – СПб.: БХВ – Петербург, 2004. – 1040 с.
3. Вендров, A. M. Проектирование программного обеспечения информационых систем: Учебник / A. M. Вендров. - М.: "Финансы и статистика", 2000. - 352 с.
4. Калянов, Г. Н. CASE. Структурный системный анализ / Г.Н. Калянов. - М.: Лори, 1996. - 437 с.