Инструкция разработчика по программе MySA

Оглавление

1.Принцип работы программы.

1.1.При запуске программы происходит соединение с сервером баз данных Access (2 соединения с \\S1\v1003\NTAB_main.mdb) и SQL(IP - 173.33.7.30), используя данные из SE.ini. Устанавливается директорию для наблюдения (\\Work_3\arhiv\) и выводятся все данные из инфо таблицы (таблица logs) на вкладку «Параметры».

1.2.Каждый день в 8:00 программа обновляет таблицы users, dolgnost, shedules, tree, contr базы SQL, Access или обе, беря данные из последнего файла(imp*.mdb) занесенного в инфо таблицу (logs).

1.3.Если были внесены изменения в наблюдаемую папку (\\Work_3\arhiv\) , то через 15 мин проверяется о всех ли файлах imp*.mdb есть информация в инфо таблице(logs), и если ее нету – то данные заносятся в инфо таблицу(logs), а также в базу данных Access и SQL, в зависимости от получателя.

1.4.При выборе действия «Обработать» проверяется о всех ли файлах imp*.mdb есть информация в инфо таблице(logs), и если ее нету – то данные заносятся в инфо таблицу(logs), а также в базу данных Access и SQL, в зависимости от получателя

1.5.При выборе действия «Инф.польз.» обновляются таблицы users, dolgnost, shedules, tree, contr базы SQL, Access или обе, данные берутся из последнего файла(imp*.mdb) занесенного в инфо таблицу (logs).

1.6. Дополнительно каждые 100 секунд программа сворачивается в область уведомлений (Область уведомлений — элемент панели инструментов среды рабочего стола («панель задач» в Windows), используемый для нужд длительно запущенных, но при этом не постоянно используемых программ. Область уведомлений имеет неофициальное название «системный трей», расположена в правом нижнем углу экрана). При нажатии на иконке в трее, программа становится видимой. Принудительное сворачивание программы в трей происходит по нажатию на «Свернуть». Остановка выборки (пункт 1.3 и 1.4) из базы производится нажатием на «Остановить» 1 раз, двойным нажатием – производится выход из программы. Нажатием «Настроить» производится переход на форму настроек.

2.Алгоритм работы программы

Пункт 1.1. Запуск программы

Есть глобальные переменные описанные в файле Unit1.h для работы с SA.ini файлом.

AnsiString DIR_L, LOG, MyEvlog, SERVER;

Остальные переменные и компоненты описаны в Приложении1.

При запуске программы запускается процедура ReadP, в ней происходит считывание файла SA.ini в глобальные переменные:

```
TIniFile *in1 = new TIniFile(ExtractFilePath(Application->ExeName)+"\\SE.ini");
DIR_L = in1->ReadString("SA","path","");
LOG = in1->ReadString("SA","Logs","");
SERVER = in1->ReadString("SA","Server","");
ACC->DefaultDatabase = in1->ReadString("SA","Base","");
MyEvlog = in1->ReadString("SA","EvlogName","");
delete in1;
```

Происходит соединение с базой SQL, используя автоматически сформированную строку:

```
MySQL->ConnectionString = "Provider=MSDASQL.1;Persist Security Info=False;Extended Properties=\"DRIVER=SQL Server;SERVER="+SERVER+";UID=sa;PWD=sa;APP=Microsoft Office 2003;WSID=C0;DATABASE=main;Network=DBMSSOCN;AutoTranslate=No;QuotedId=No\"";
```

И 2 соединения с базой Access :

```
ACC->ConnectionString = "Provider=MSDASQL.1;Persist Security Info=False;Extended Properties=\"DBQ="+ACC->DefaultDatabase+";Driver={Driver do Microsoft Access (*.mdb)};" +
"DriverId=25;FIL=MS Access;MaxBufferSize=2048;MaxScanRows=8;PageTimeout=5;" +
"SafeTransactions=0;Threads=3;UID=admin;UserCommitSync=Yes;\"";
```

AccConnection->ConnectionString = ACC->ConnectionString;

В данных соединениях использовались следующие параметры:

Provider - Имя драйвера источника данных (SQLOLEDB.1 для MS SQL server)

Persist Security Info - необходимость использования шифрованого канала

Extended Properties - тип строка. Дополнительные параметры провайдера. Провайдеры могут иметь внутренние параметры, которые не представлены в свойствах, для установки этих параметров и существует это свойство.

DRIVER - имя драйвера для текущего подключения

SERVER - сервер базы данных, у нас SERVER=173.33.7.30

UID = "sa" 'логин пользователя SQL-сервера

PWD = "sa" 'пароль пользователя SQL-сервера

APP - Имя приложения, вызывающего функцию SQLDriverConnect (необязательно). Если это значение указано, оно хранится в столбце program_name в таблице master.dbo.sysprocesses и возвращается функциями sp_who и APP_NAME.

WSID -Идентификатор рабочей станции. Обычно это сетевое имя компьютера, на котором находится приложение (необязательно). Если это значение указано, хранится в системной таблице master.dbo.sysprocesses в столбце hostname и возвращается функциями sp_who и HOST_NAME

DATABASE - Имя базы данных SQL Server, используемой для соединения по умолчанию. Если ключевое слово Database не указано, используется база данных, определенная по умолчанию для имени входа.База данных по умолчанию из источника данных ODBC переопределяет базу данных по умолчанию, определенную для имени входа. База данных должна существовать, если не указан параметрAttachDBFileName. Если также указано ключевое слово AttachDBFileName, выполняется присоединение первичного файла, на который оно указывает, после чего присваивается имя базы данных, указанное ключевым словом Database.

Network -Допустимые значения: dbnmpntw (именованные каналы) и dbmssocn (TCP/IP).Одновременное указание значения для ключевого слова Network и префикса протокола в ключевом слове Server является ошибкой.

AutoTranlate - Если имеет значение «yes», строки символов ANSI, которыми обмениваются клиент и сервер, переводятся с использованием Юникода, чтобы минимизировать проблемы сопоставления символов

национальных алфавитов кодовых страниц клиента и сервера.Если имеет значение «no», перевод символов не выполняется.

QuotedId - Если имеет значение «yes», параметр QUOTED_IDENTIFIERS при соединении устанавливается в значение ON. SQL Server использует правила ISO в отношении использования кавычек в инструкциях SQL. Если параметр равен «no», то параметр QUOTED_IDENTIFIERS для соединения отключается (принимает значение OFF). В этом случае SQL Server следует правилам устаревших версий Transact-SQL относительно использования кавычек в инструкциях SQL. Дополнительные сведения см. в разделе Действие параметров ISO.

DBQ-путь к базе Base=\\S1\v1003\NTAB_main.mdb

Производится вывод сообщения о наличии или отсутствии данных и файлов

```
if(!FileExists(ACC->DefaultDatabase))
  Memo1->Lines->Add("Не найден путь к базе: "+ACC->DefaultDatabase);
else
  Memo1->Lines->Add("Файл базы на S1: " + ACC->DefaultDatabase);
  if(MyEvlog == "") Memo1->Lines->Add("Не найдена выходная таблица: " + MyEvlog);
  else Memo1->Lines->Add("Выходная таблица: " + MyEvlog);
  if(LOG == "") Memo1->Lines->Add("Не найдена таблица: " + LOG);
  else Memo1->Lines->Add("info - таблица: [" + LOG+"]");
```

Активируется компонент для наблюдения за папкой(\\Work_3\arhiv\)

```
FM->FolderName=DIR_L;
FM->Active=true;
```

Создается запрос для вывода информации на вкладку «Параметры». Для этого выбираются все данные из инфо таблицы logs и сортируются по полю files в обратном порядке (от z до a)

```
Logs_Query->SQL->Text = "select * from " + LOG + " order by files desc";
Logs_Query->Open();
```

На этом подготовительный этап заканчивается.

Пункт 1.2. Обновление информации о пользователях в 8:00

Есть компонент TRxClock *Cl (Описание в приложении 1), позволяющий отображать время. Он имеет следующие настройки в Object Inspector

AlarmEnabled = True;
AlarmHour = 8
AlarmMinute = 0
AlarmSecond = 0

Данные настройки позволяют в 8:00 утра запускать функцию ClAlarm(), которая в свою очередь запускает функцию Upd_info()

```
Memo1->Lines->Add("Сброс инф. польз. "+IntToStr(Upd_info())+" сек.");
```

Функция Upd_info проводит обновление информации в базах в зависимости от получателя. Это происходит следующим образом.

Происходит соединение с базой SQL, посылается запрос через переменную SysQuery, из этой базы извлекается имя последнего файла добавленного в лог файл(logs).

```
try {MySQL->Open();}
catch (Exception &EOleException)  {
  Memo1->Clear();
 Memo1->Lines->Add("Ошибка! Недоступен сервер MySQL: ["+SERVER+"]"); return -1;
  }
SysQuery->SQL->Text="select files from "+LOG+" order by files";
SysQuery->Open();
SysQuery->Last();
```

Получив имя последнего файла imp*.mdb, программа соединяется с ним как с базой данных Access.

```
AnsiString cfile=SysQuery->FieldByName("files")->AsString;
```

```
if(FileExists(DIR_L+"\\"+cfile))
 {
  AccConnection->DefaultDatabase = DIR_L+"\\"+cfile;
  AccConnection->Open();
 }
else {Memo1->Lines->Add("Ошибка!");return -1;}
```

И в зависимости от выбранного получателя, обновляет таблицы users, dolgnost, shedules, tree, contr баз данных SQL, Access или обеих используя функции update_info или update_infoA соответственно. То есть происходит вызов функции для каждой таблицы.

```
update_info("users",cfile);
update_info("dolgnost",cfile);
update_info("shedules",cfile);
update_info("tree",cfile);
update_info("contr",cfile);
```

Функция update_info использует следующий алгоритм (update_infoA работает аналогично, только для Access).

void update_info(AnsiString table_name,AnsiString file_copy); у нее есть 2 параметра имя таблицы и имя файла найденного в лог файле(последнего файла).

Первым делом очищается таблицу table_name в базе SQL через переменную Command.

```
Command->CommandText="delete from "+table_name;
Command->Execute();
```

Используя переменную ABase(описание в Приложении 1) соединенную с базой imp*.mdb, выбираются все поля таблицы table_name файла file_copy и сохраняются вместо удаленных в таблицу table_name базы SQL.

```
ABase->SQL->Text="select * from
\""+DIR_L+"\\"+file_copy+"\"."+table_name;
ABase->Open();
while(!ABase->Eof)
{
Cmd="insert into "+table_name+"(";
```

```
for(int i=0;i<ABase->FieldCount;i++)
{
if(i!=0)Cmd = Cmd+", "+ABase->Fields->Fields[i]->FieldName;
else Cmd = Cmd+ABase->Fields->Fields[i]->FieldName;
}
Cmd = Cmd+") values(";
for(int j=0;j<ABase->FieldCount;j++)
{
if(j>0)Cmd=Cmd+", \'"+ABase->FieldByName(ABase->Fields->Fields[j]-
>FieldName)->AsString+"\'";
else Cmd=Cmd+"\'"+ABase->FieldByName(ABase->Fields->Fields[j]-
>FieldName)->AsString+"\'";
}
Cmd=Cmd+")";
Command->CommandText=Cmd;
Command->Execute();
ABase->Next();
}
```

Таким образом происходит обновление 5 таблиц : users, dolgnost, shedules, tree, contr для базы SQL, Access или обеих в зависимости от получателя.

Пункт 1.3. Обработка данных с электронной проходной

Есть компонент TRxFolderMonitor *FM (описание в Приложении 1), который позволяет отслеживать изменения в папке указанной в FolderName. За какими именно изменениями следить указывается в свойстве Filter . У него установлены следующие свойства в Object Inspector:

FolderName = \\Work_3\EXPORT
Filter ->
fnFileName = true
fnDirName = true
fnSize = true
fnLastWrite = true

Папка FolderName изменяется на папку указанную в переменной DIR_L и берется из SA.ini(описано в Пункт 1.1).Остальные значения указывают, что отслеживаем :

fnFileName - любые изменения имен файлов в отслеживаемом каталоге и, возможно, его подкаталогах (переименование, создание или удаление файлов);

fnDirName - любые изменения имен директорий в отслеживаемом каталоге и, возможно, его подкаталогах (создание или удаление каталогов);

fnSize - любые изменения размера файлов в отслеживаемом каталоге и, возможно, его подкаталогах;

fnLastWrite - любые изменения времени последний записи (last write-time) в файл в отслеживаемом каталоге и, возможно, его подкаталогах.

То есть, как только в данный каталог будет добавлен новый файл или изменен старый, будет вызвана функция FMChange, которая запустит таймер

Timer1->Enabled=true;

Для Timer1 установлен интервал свойством Interval = 900000 мс, то есть через 900 секунд, после изменений в папке \\Work_3\\arhiv, произойдет вызов функции Timer1Timer(), произойдет отключение таймера и вызов функции обработки инфо таблицы.

Timer1->Enabled=false;
Find();

В *Find()* первым делом проверяется соединение с сервером SQL

try {MySQL->Open();}
catch (Exception &EOleException){Memo1->Lines->Add("Ошибка! Недоступен сервер MySQL: ["+SERVER+"]");return false;}
if(!MySQL->Connected)return false;

и если оно есть то, директория \\Work_3\arhiv\ делается основной и дальше производится работа с файлами imp*.mdb. Если такой файл присутствует в данной директории, то пока не найден последний файл с именем типа imp*.mdb, проверяется, занесена ли о нем информация в лог файл(logs). Если информации нету, то она заносится в инфо таблицу, а информация из данного файла, используя функцию f_list(), заносится в базу SQL, Access или обе(в зависимости от режима). Копия файла размещается в папку *c:\\arhiv*\ копьютера на котором запущена MySA

TSearchRec sr;

```
SetCurrentDir(DIR_L);
if(FindFirst("imp*.mdb",faAnyFile,sr)==0)
{
do
 {
Form1->SysQuery->SQL->Text="select files from "+LOG+" where
files='"+sr.Name+"'";
Form1->SysQuery->Open();
if(Form1->SysQuery->FieldByName("files")->AsString=="")
 {
 t_list=GetTickCount();
 Memo1->Lines->Add("Обработка: " + sr.Name);
 row_str=f_list(sr.Name, RG1->ItemIndex, RG2->ItemIndex);
    SysCommand->CommandText=
      "insert into "+LOG+"(files,rows,time_out,rw) values('"+
        sr.Name+"','"+row_str+"',"+FloatToStr((GetTickCount()-
t_list)/1000)+
          ","+IntToStr(prov)+")";
  SysCommand->Execute();
  AnsiString St1=DIR_L+"\\"+sr.Name,St2="c:\\arhiv\\"+sr.Name;
 CopyFile(St1.c_str(), St2.c_str(), true);
  Memo1->Lines->Add("Обработка файла: ~
"+FloatToStr((GetTickCount()-t_list)/1000)+" сек.");
    Application->ProcessMessages();
 }
}while(FindNext(sr)==0);
FindClose(sr);
```

Функция f_list(AnsiString in_file, int Regim, int Serv), получает 3 параметра in_file - имя реально найденного файла, Regim - режим работы (0- обновление, 1- по макс.дате, 2 - добавление), Serv - кто получатель(0 – SQL, 1 –Access, 2 - оба) и работает по следующему алгоритму.

Переменной, отвечающей за выход из цикла, присваивается значение, так что бы цикл продолжался

```
Res = false;
```

Если происходит работа не только с базой SQL, но и Access, то программа соединяется с ней. Переменная ACC связывается с базой _MAIN.MDB

```
if(Serv != 0)
if(FileExists(ACC->DefaultDatabase)!=0)
 {
  ACC->Open();
  if(!ACC->Connected)Memo1->Lines->Add("Ошибка открытия базы:
"+ACC->DefaultDatabase);
 }
```

Переменная AccConnection связывается с базой Access, посредством свойства ConnectionString описанного ранее, а именно с файлом переданным в качестве параметра в функцию (imp*.mdb). Проверяется наличие данного файла, произошло ли соединение. Выводятся соответствующие сообщения.

```
if(FileExists(DIR_L+"\\"+in_file)!=0)
 {
  AccConnection->DefaultDatabase = DIR_L+"\\"+in_file;
  try {AccConnection->Open();}
  catch (Exception &EOleException)
  {
   Memo1->Lines->Add("Ощибка! Ошибка открытия базы: "+
AccConnection->DefaultDatabase);
   return false;
  }
  if(!AccConnection->Connected)return -1;
 }
else {Memo1->Lines->Add("Ошибка!");return -1;}
```

Определяется максимальная и минимальная дата в файле 2-й базы(переданный параметр imp*.mdb)

```
MinMax->SQL->Text="select min(evdate) , max(evdate) from
\""+DIR_L+"\\"+in_file+"\".evlog";
MinMax->Open();
```

В зависимости от того какой режим работы : по макс. дате или нет, выбираются данные из таблицы evlog переданного файла imp*.mdb. Если по максимальной дате, то берутся только данные, дата которых выше максимальной даты в таблице evlog базы SQL.

```
if(Regim!=1)
ABase->SQL->Text="select * from \""+DIR_L+"\\"+in_file+"\".evlog
where evcode in(1,5,6) and user_id<>0 and userdirection in (0,1)";
if(Regim==1)
{
SysQuery->SQL->Text = "select max(evdate) as max from "+MyEvlog;
SysQuery->Open();
ABase->SQL->Text="select * from \""+DIR_L+"\\"+in_file+"\".evlog
where evcode in(1,5,6) and user_id<>0 and userdirection in (0,1) and evdate>"+
FormatDateTime("#mm/dd/yyyy hh.mm.ss#", SysQuery-
>FieldByName("max")->AsDateTime);
SysQuery->Close();
}
ABase->Open();
```

Выбираются данные из файла, проверяется параметр обработки.

Если параметр обработки – обновление, то из таблицы evlog (SQL, Access или из обеих баз данных, в зависимости от получателя) удаляются данные.

```
if(Regim==0)
   {
      Command->CommandText="delete from "+MyEvlog+" where
evdate>=\'"+
      FormatDateTime("yyyy-mm-dd hh:mm:ss", MinMax-
>FieldByName("expr1000")->AsDateTime)+"\' and evdate<=\'"+
      FormatDateTime("yyyy-mm-dd hh:mm:ss", MinMax-
>FieldByName("expr1001")->AsDateTime)+"\'";
      CommandA->CommandText="delete from "+MyEvlog+" where
evdate>="+
      FormatDateTime("#mm/dd/yyyy hh.mm.ss#", MinMax-
>FieldByName("expr1000")->AsDateTime)+" and evdate<="+
      FormatDateTime("#mm/dd/yyyy hh.mm.ss#", MinMax-
>FieldByName("expr1001")->AsDateTime);
```

```
    switch(Serv)
    {
    case 0:
       Command->Execute();
       break;
    case 1:
       CommandA->Execute();
       break;
    case 2:
       CommandA->Execute();
       Command->Execute();
       break;
    }
  }
```

Итак, проведены все предварительные действия, данные получены, таблицы очищены (при необходимости). Происходит добавление данных. Строится запрос на добавление, где вначале перечисляются, какие поля будут заполняться, а затем какие значения этим полям будут присваиваться. Поскольку известен изначальный формат полей таблицы evlog базы imp*.mdb, то будут производиться соответствующие преобразования значений в строку, т.к. запрос имеет тип AnsiString. Сформировав запрос для SQL и Access таблицы, программа выполняет запрос в зависимости того кто получатель SQL, Access или оба. Таким образом, заносятся данные из таблицы evlog файла imp*.mdb в базу SQL, MAIN.MDB или в обе

```
while(!ABase->Eof)
 {
  if(Res) {PB->Visible=false; return -2;}
  Cmd="insert into "+MyEvlog+"(";
  CmdA = "insert into "+MyEvlog+"(";
  for(int i=0;i<ABase->FieldCount;i++)
   {
    if(i!=0)
     {
      Cmd = Cmd+", "+ABase->Fields->Fields[i]->FieldName;
      CmdA += ", "+ABase->Fields->Fields[i]->FieldName;
     }
```

```
        else
         {
          Cmd = Cmd+ABase->Fields->Fields[i]->FieldName;
          CmdA += ABase->Fields->Fields[i]->FieldName;
         }
       }

      CmdA+= ") values(";
      Cmd = Cmd+") values(";
      for(int j=0;j<ABase->FieldCount;j++)
       {
        if(j>0)
         {
          if(j==1)
           {
            CmdA += ", "+FormatDateTime("#mm/dd/yyyy hh.mm.ss#",
ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsDateTime);
            Cmd += ", \'"+FormatDateTime("yyyy-mm-dd hh:mm:ss", ABase-
>FieldByName(ABase->Fields->Fields[j]->FieldName)->AsDateTime)+"\'";
           }
          else
           if(j<21)
            {
             CmdA += ", \'"+ABase->FieldByName(ABase->Fields-
>Fields[j]->FieldName)->AsString+"\'";
             Cmd += ", \'"+ABase->FieldByName(ABase->Fields-
>Fields[j]->FieldName)->AsString+"\'";
            }
          else
           {
            CmdA += ", "+FormatDateTime("#mm/dd/yyyy hh.mm.ss#",
ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsDateTime);
            Cmd += ", \'"+ABase->FieldByName(ABase->Fields->Fields[j]-
>FieldName)->AsDateTime+"\'";
           }
         }
        else
         {
```

```
                Cmd=Cmd+"\'"+ABase->FieldByName(ABase->Fields->Fields[j]-
>FieldName)->AsString+"\'";
                CmdA+="\'"+ABase->FieldByName(ABase->Fields->Fields[j]-
>FieldName)->AsString+"\'";
              }
            }
          Application->ProcessMessages();
          Cmd=Cmd+")";
          CmdA+=")";

          CommandA->CommandText=CmdA;
          Command->CommandText=Cmd;

          switch(Serv)
          {
          case 0:
            Command->Execute();
            break;
          case 1:
            CommandA->Execute();
            break;
          case 2:
            CommandA->Execute();
            Command->Execute();
            break;
          }
          prov++;
          PB->StepBy(1);
          ABase->Next();
          }
```

Если заполнение прошло успешно, возвращается количество записей

```
return ABase->RecordCount;
```

Таким образом, происходит добавление данных в инфо таблицу logs и базы SQL, Access или обе.

Пункт 1.4. Нажатие на «Обработать» Действует аналогично Пункту 1.3, только вызывается принудительно

Пункт 1.5. Нажатие на «Инф.польз.» Действует аналогично Пункту 1.2, только вызывается принудительно

Пункт 1.6. Дополнительные возможности

Есть таймер TTimer *Timer2(описание в Приложении 1). Для Timer2 установлен интервал свойством Interval = 100000 мс, то есть через 100 секунд произойдет вызов функции Label2Click(), действие аналогичное нажатию на «Свернуть». Форма сворачивается, остается видимым только отображение в трее.

Form1->Hide();

При двойном нажатии на иконку в трее вызывается функция RxT1DblClick и программа становится видимой

Form1->Visible=true;

При нажатии 1 раз на «Остановить» происходит изменение параметра, который является условием выхода из цикла при обработке баз данных, что приведет к остановке работы программы

Res = true;

При нажатии дважды, программа закончит обработку текущих событий и закроется

Application->Terminate();

При двойном нажатии на «Настроить» происходит переход на форму настроек

ServForm->ShowModal();

На данной форме можно вводить данные и сохранять их в файл SA.ini.

А именно в строку:

eServer – ip адрес SQL сервера

ePath – путь к каталогу экспорта(каталог за которым ведется наблюдение), в нем располагаются imp*.mdb файлы

ePathAcc - путь к фалу базы MAIN.MDB

eLogs – как называется информационная таблица базы(хранит какие файлы внесены в базу)

eEvlog**;** - как называется таблица для хранения поступающей информации (хранить всю информацию из файлов imp*.mdb)

При нажатии на кнопку «Сохранить и закрыть» происходит сохранение введенной информации, вызов функции для повторной подготовке к работе программы и закрытие формы

```
TIniFile *Oini=new TIniFile(ExtractFilePath(Application->ExeName)+"\\SE.ini");
Oini->WriteString("SA", "path", ePath->Text);
Oini->WriteString("SA", "Base", ePathAcc->Text);
Oini->WriteString("SA", "Logs", eLogs->Text);
Oini->WriteString("SA", "EvlogName", eEvlog->Text);
Oini->WriteString("SA", "Server", eServer->Text);
Oini->UpdateFile();
delete Oini;
Form1->ReadP();
Close();
```

При двойном щелчке на строке ввода ePath, происходит заполнение данных через диалог открытия файла в Windows. Взят путь до любого файла imp*.mdb без самого имени файла, то есть сохраняется каталог, где хранятся файлы imp*.mdb.

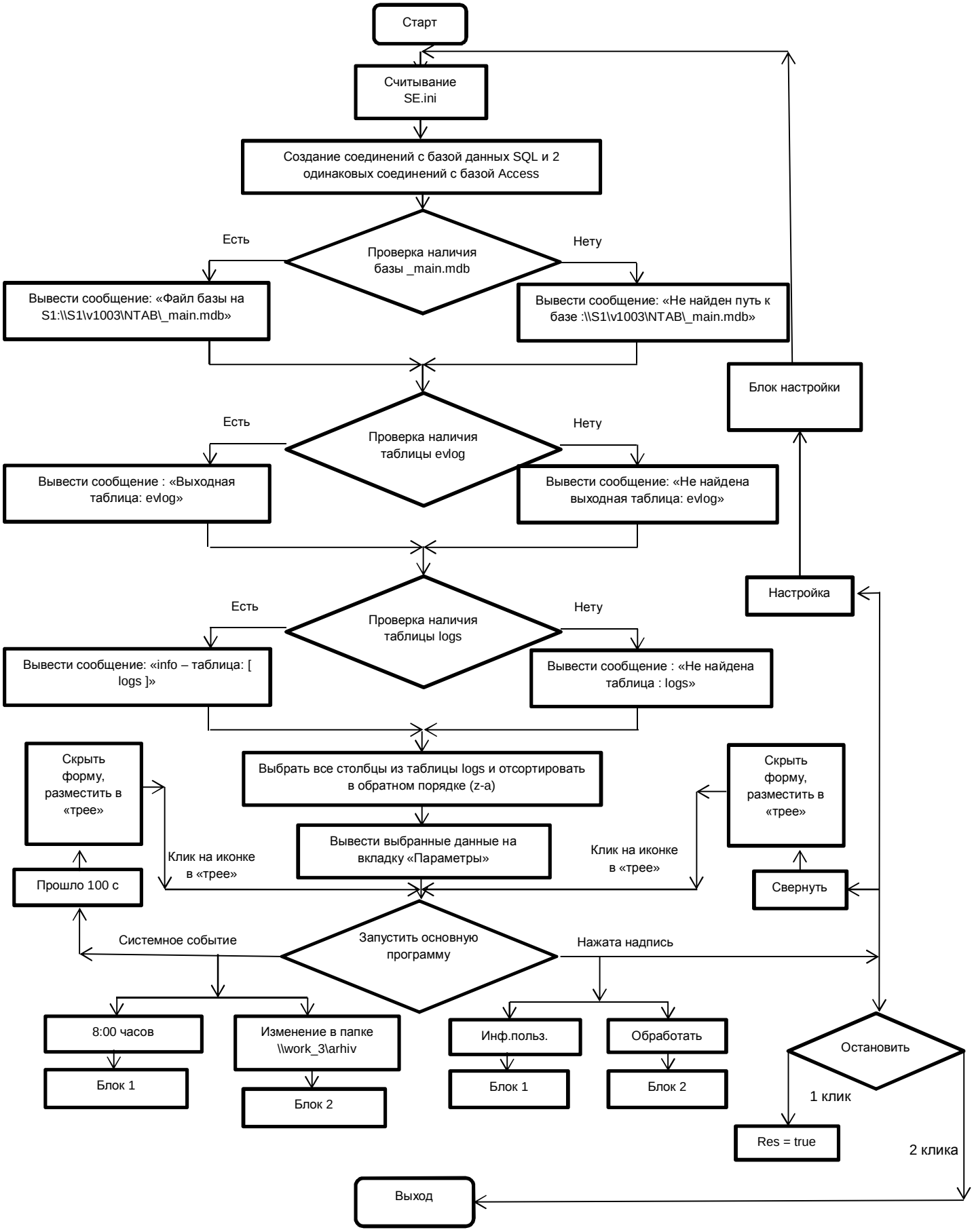
```
OpenDialog1->DefaultExt = ".mdb";
OpenDialog1->Filter = "Base(*.mdb)|*.mdb";
if(OpenDialog1->Execute())
 {
   ePath->Text = ExtractFilePath(OpenDialog1->FileName);
 }
```

При двойном щелчке на строке ввода ePathAcc, заполнение данных происходит через диалог открытия файла в Windows. Берется путь до файла _main.mdb

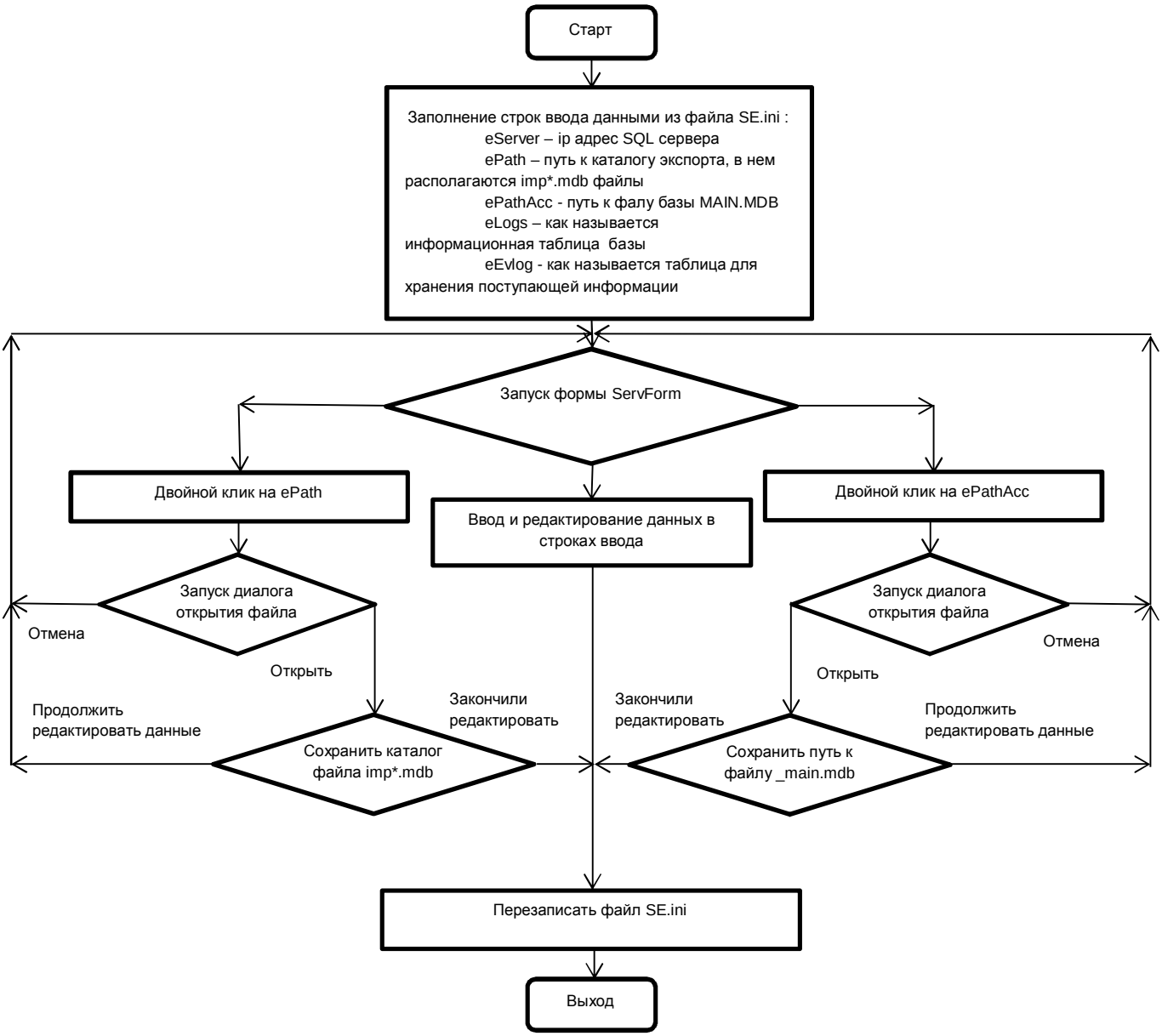
```
OpenDialog1->DefaultExt = ".mdb";
OpenDialog1->Filter = "Base(*.mdb)|*.mdb";
if(OpenDialog1->Execute())
 {
  ePathAcc->Text = OpenDialog1->FileName;
 }
```

3.Блок – схемы работы программы

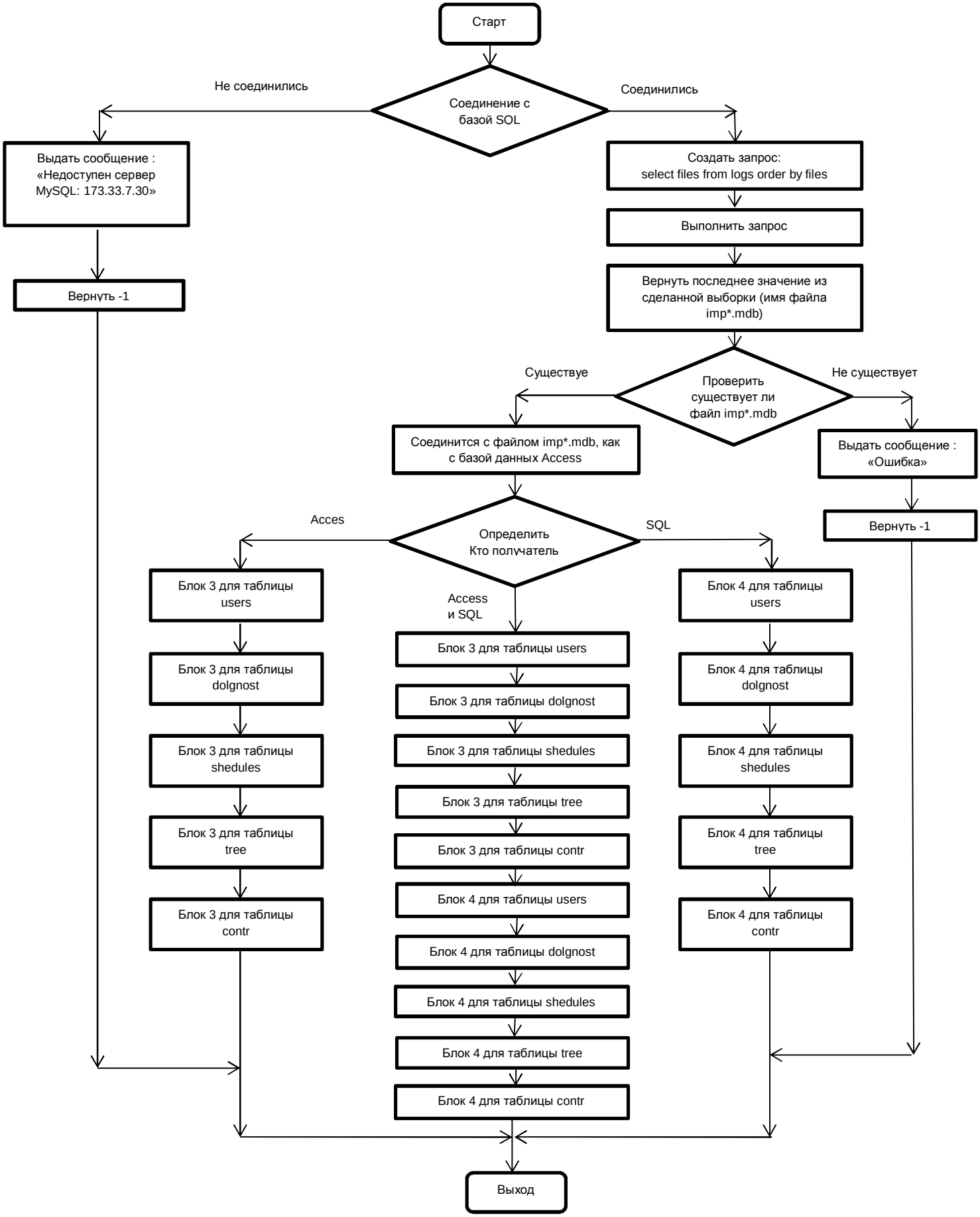
Блок –схема «Основная программа»



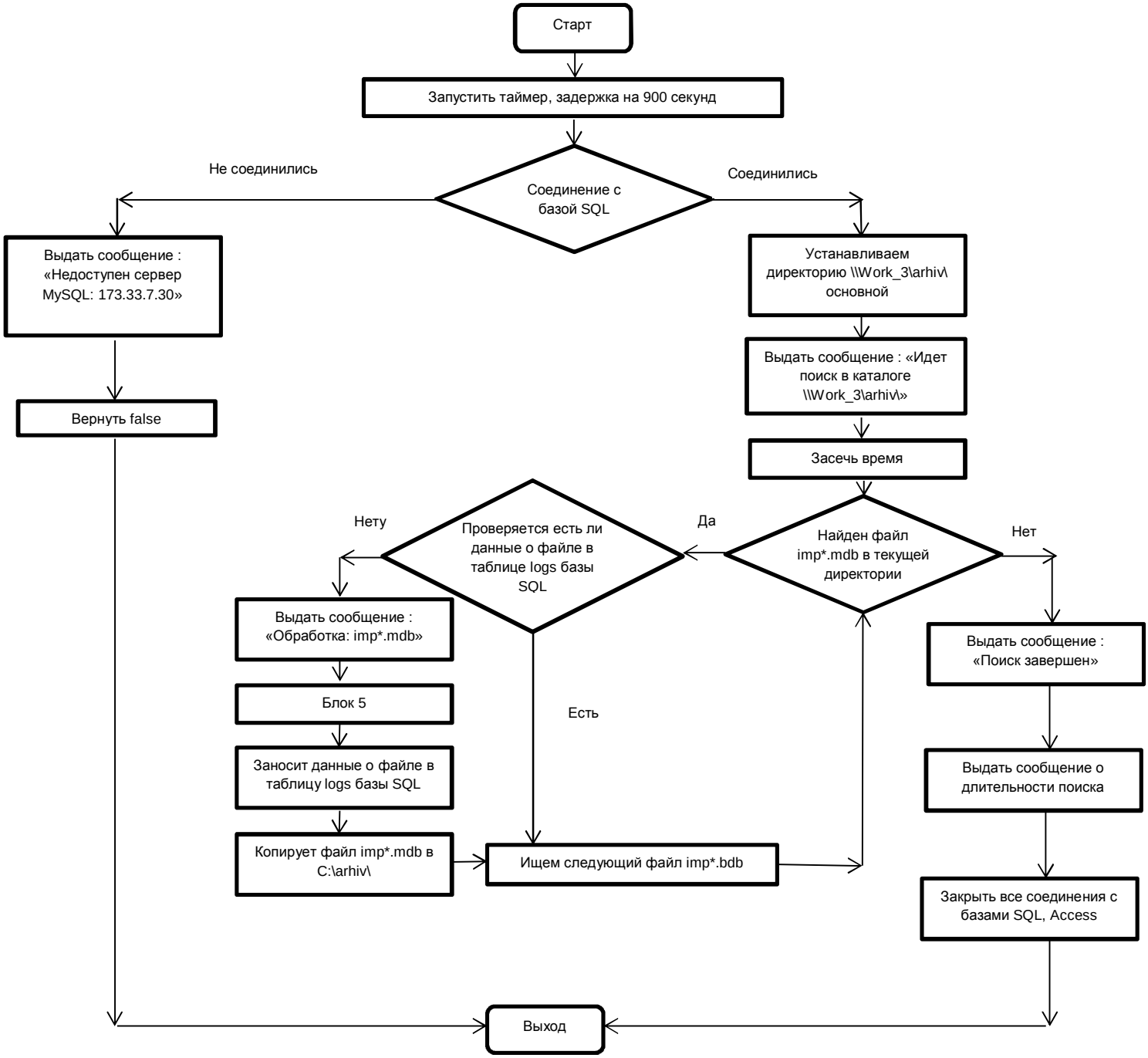
Блок – схема «Блок настройки»



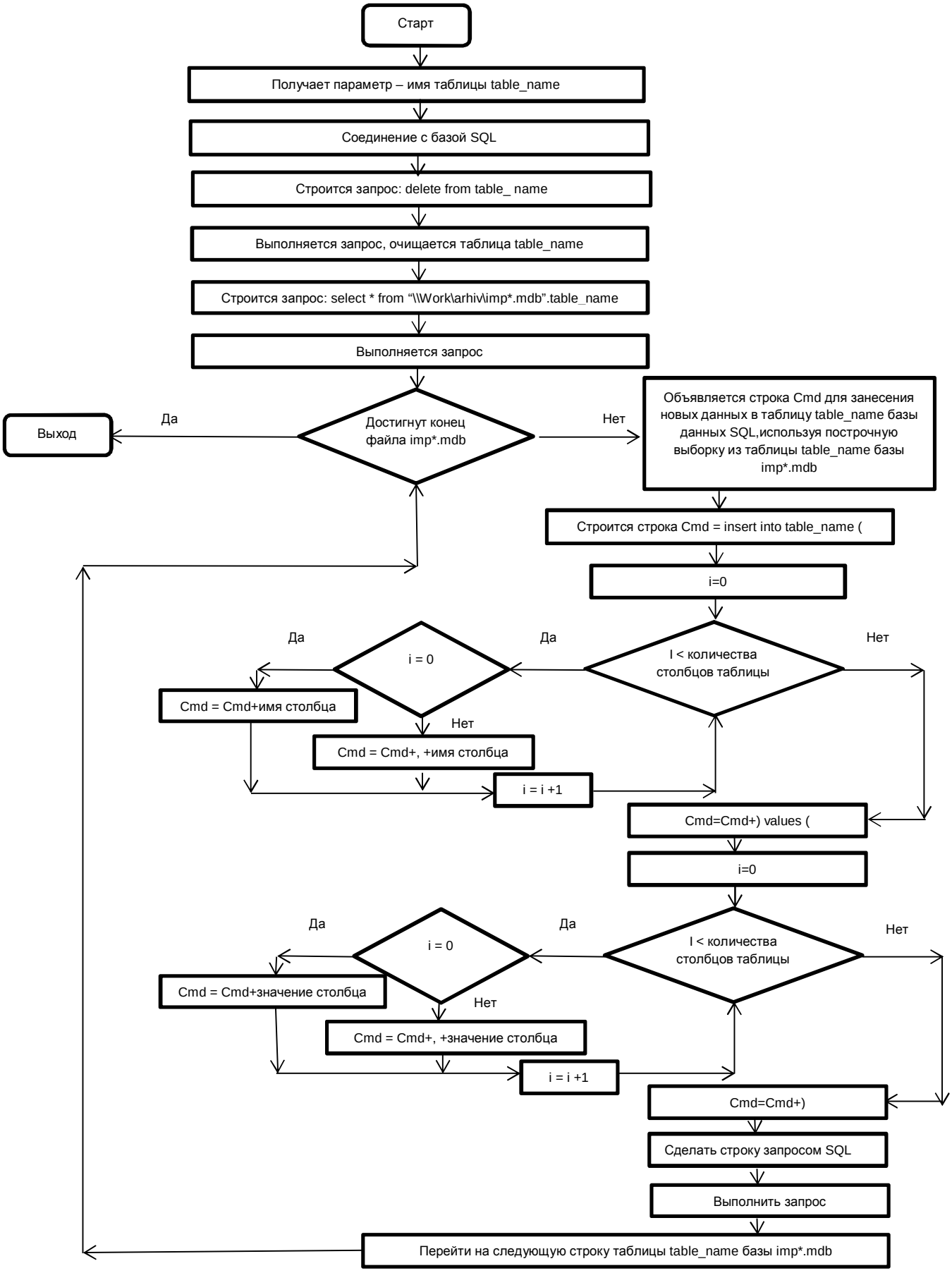
Блок – схема «Блок 1: Обновление информации о пользователях»



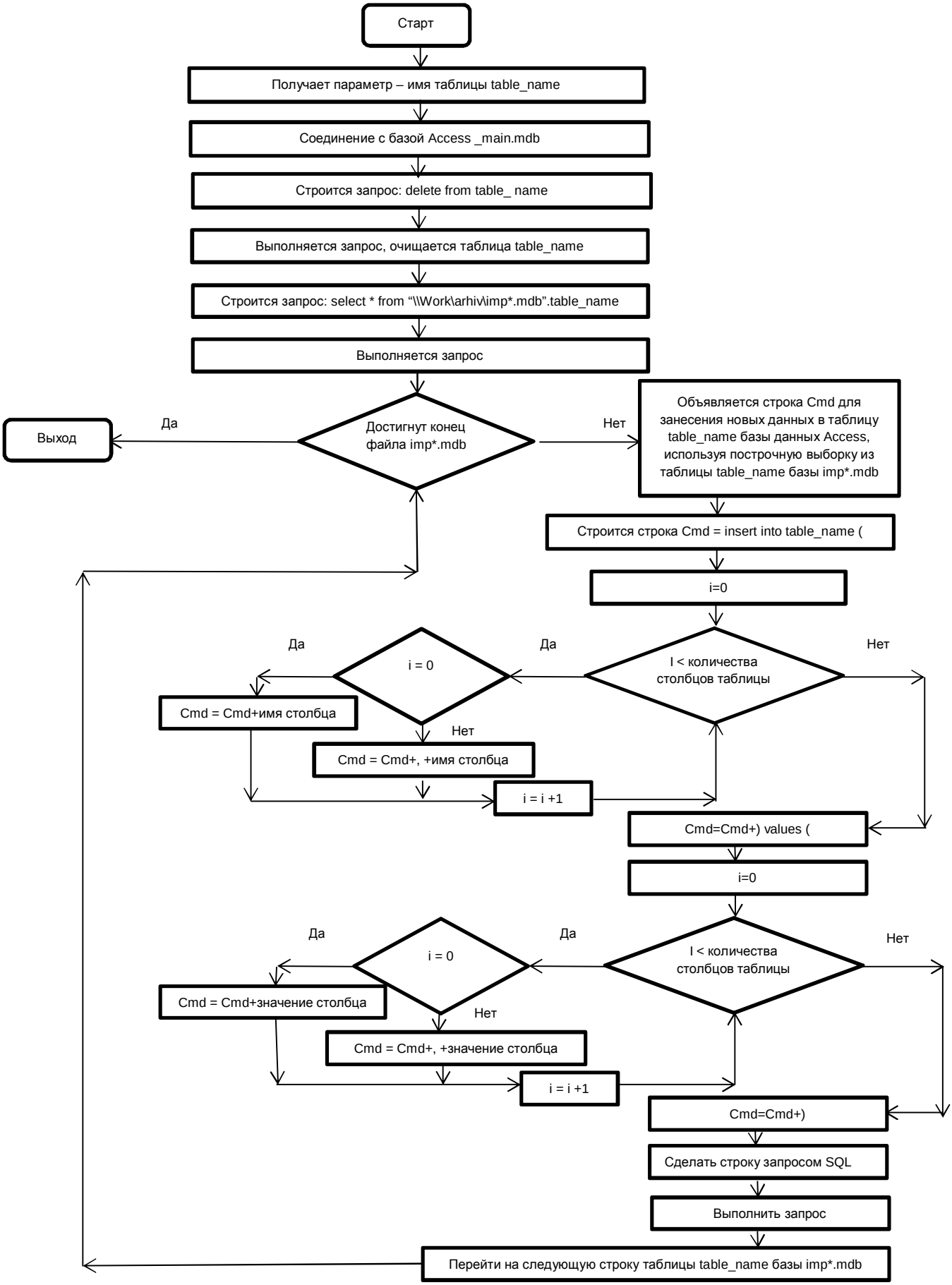
Блок – схема «Блок 2: Обновление информации в таблице logs »



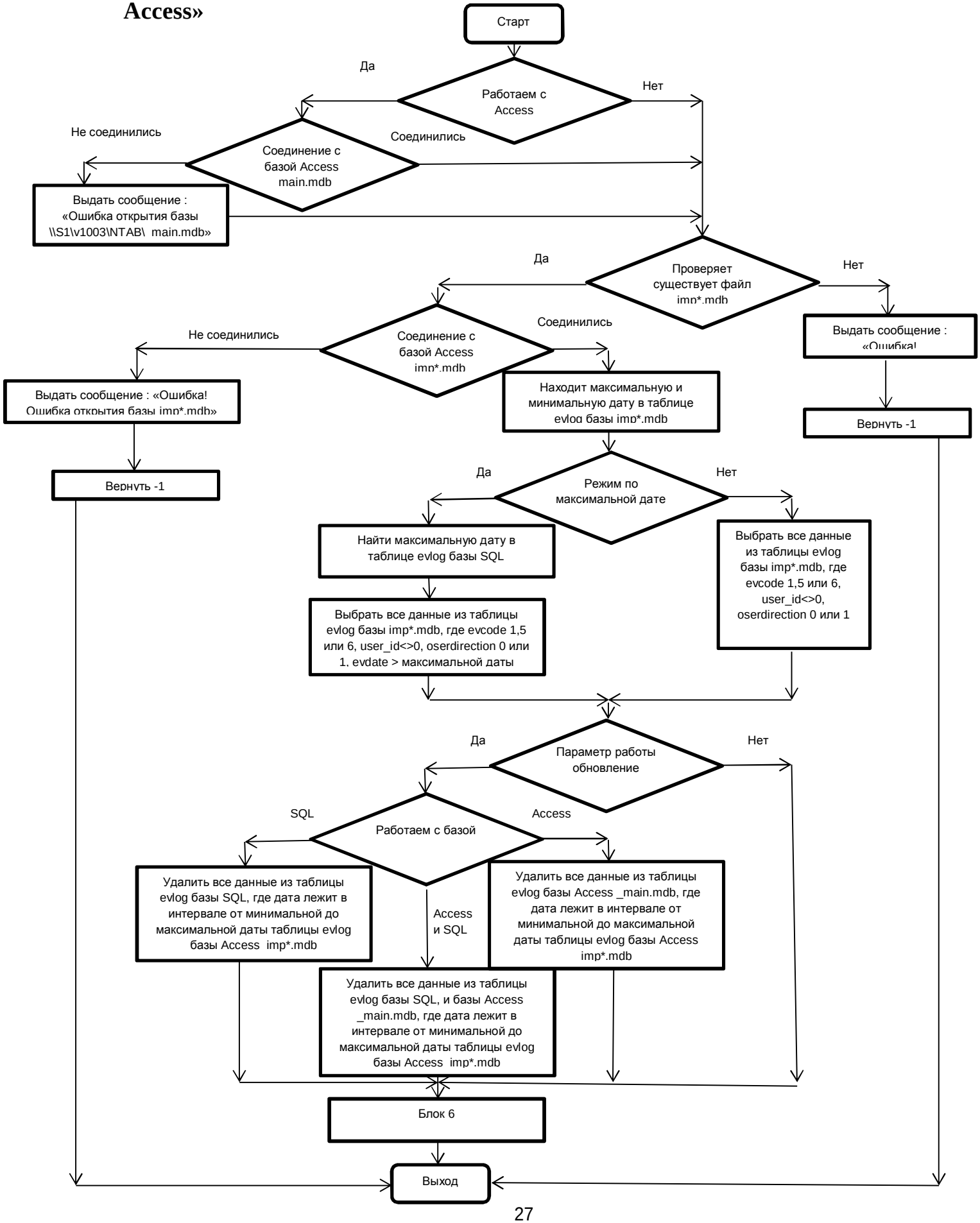
Блок – схема «Блок 3: Занесение новых данных в таблицы users, dolgnost, shedules , tree , contr базы данных SQL»



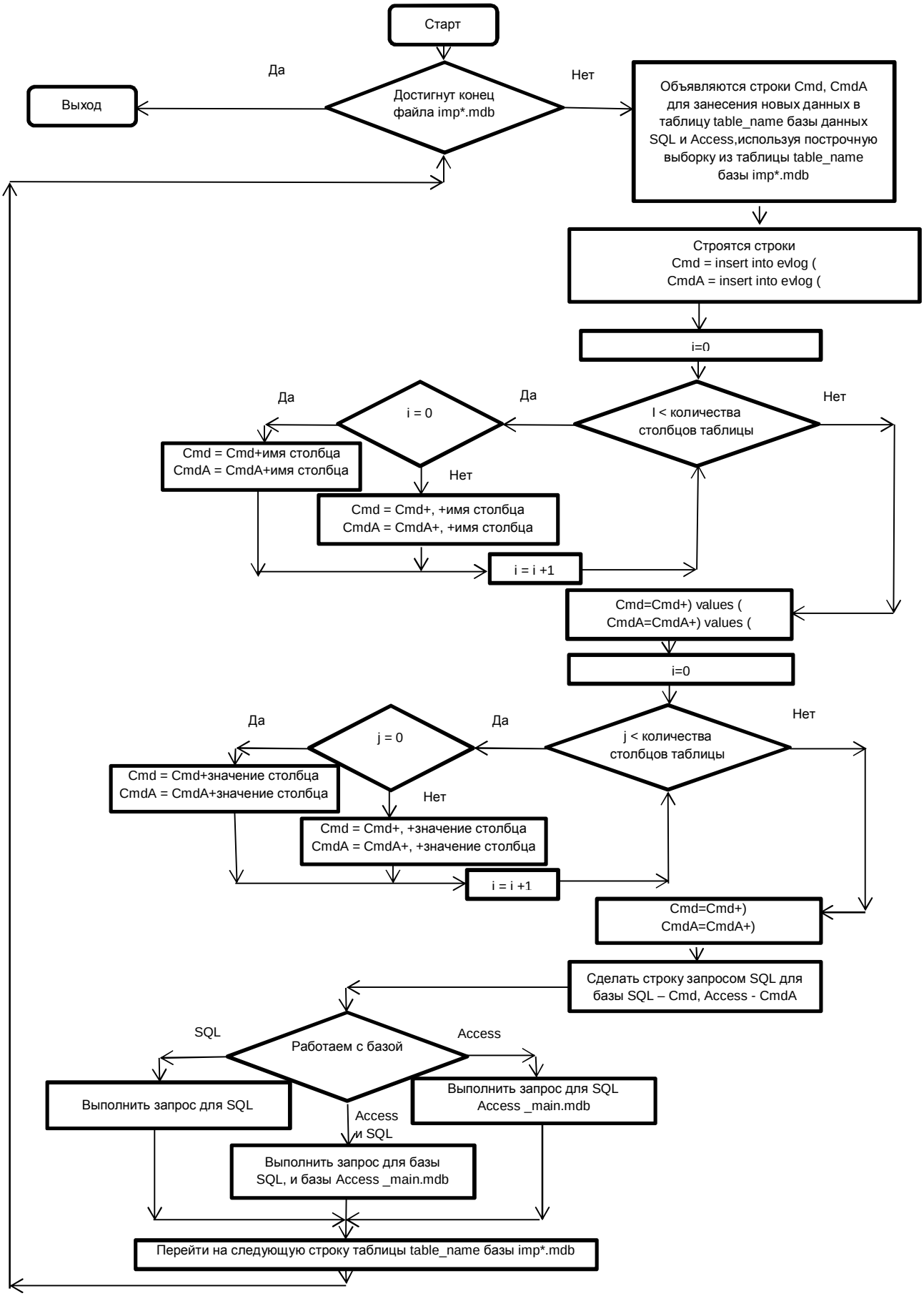
Блок – схема «Блок 4: Занесение новых данных в таблицы users, dolgnost, shedules , tree , contr базы данных Access»



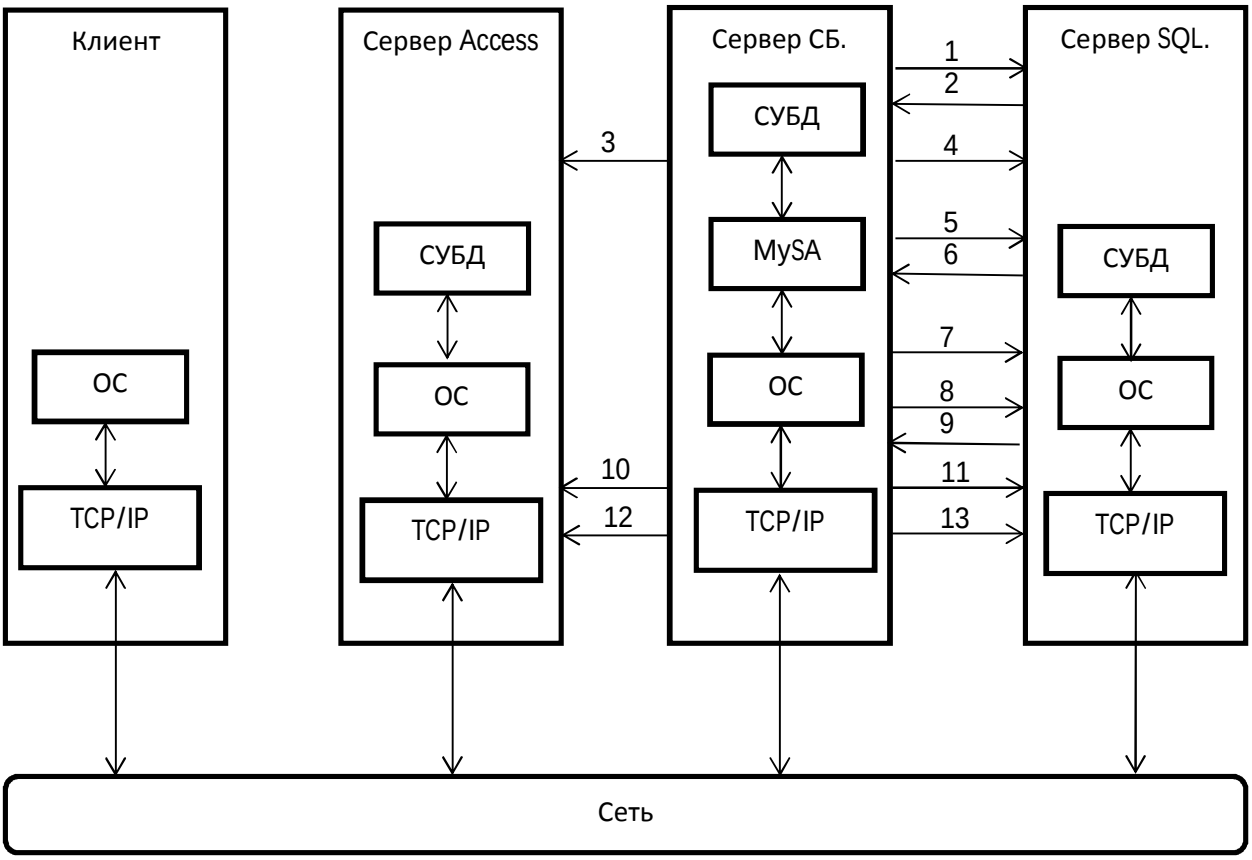
Блок – схема «Блок 5: Изменение таблицы evlog в базах данных SQL и Access»



Блок – схема «Блок 6: Создание запросов на добавление в таблицу evlog баз данных SQL и Access»



4.Сетевая модель работы программы



1. Запрос на извлечение списка файлов imp*.mdb из информационной таблицы logs и их сортировки по алфавиту в обратном порядке от z к a.
2. Ответ на запрос, возвращает имя последнего файла imp*.mdb(подробнее в «Пункте 1.2.» и блок схеме «Блок 1»)
3. Обновление таблиц users, dolgnost, shedules, tree, contr, предварительно удалив из них всю информацию. Будет вызван запрос или нет зависит от опции получателя.
4. Обновление таблиц users, dolgnost, shedules, tree, contr, предварительно удалив из них всю информацию. Будет вызван запрос или нет зависит от опции получателя.
5. Запрос на наличие информации о файле imp*.mdb в таблице logs.
6. Ответ на запрос, есть информация о файле или нет информации о файле. Если ответ нет, то формируются последующие запросы, если есть – ничего не происходит(подробнее в «Пункте 1.3» и блок схеме «Блок 2») .
7. Занести данные о файле imp*.mdb в таблицу logs, а именно значения колонок files,rows,time_out,rw.
8. Запрос на выбор максимальной даты из колонки evdate таблицы evlog

9. Ответ на запрос. Возвращает максимальную дату в колонке evdate из таблицы evlog (подробнее в «Пункте 1.3» и блок схеме «Блок 5»).

10. Удаление данных из таблицы evlog. Предварительно определяется максимальная и минимальная дата в таблице evlog файла imp*.mdb. Удаляются данные, где дата находится в интервале от минимальной до максимальной. Вызов зависит от получателя и опции обновления

11. Удаление данных из таблицы evlog. Предварительно определяется максимальная и минимальная дата в таблице evlog файла imp*.mdb. Удаляются данные, где дата находится в интервале от минимальной до максимальной. Вызов зависит от получателя и опции обновления

12. Заносим данные в таблицу evlog, данные берутся из таблицы evlog файла imp*.mdb. Предварительно определяется максимальная и минимальная дата в таблице evlog файла imp*.mdb. Если данные нужны с учетом максимальной даты, то предварительно выполняем запрос 8. Если не нужно учитывать дату, то пункты 8,9 можно пропустить. Запрос на добавление строится с учетом параметра дата или нет в зависимости от условия заполнения. Будет вызван запрос или нет зависит от опции получателя.

13. Заносим данные в таблицу evlog, данные берутся из таблицы evlog файла imp*.mdb. Предварительно определяется максимальная и минимальная дата в таблице evlog файла imp*.mdb. Если данные нужны с учетом максимальной даты, то предварительно выполняем запрос 8. Если не нужно учитывать дату, то пункты 8,9 можно пропустить. Запрос на добавление строится с учетом параметра дата или нет в зависимости от условия заполнения. Будет вызван запрос или нет зависит от опции получателя.

Приложение 1: Типы данных

Компоненты программы

Модуль Unit1;

TPageControl *PCtrl; Компонент TPageControl относится к группе многостраничных панелей

TTabSheet *TabSheet1; Страница компонента TPageControl – «Обзор»

TTabSheet *TabSheet2; Страница компонента TPageControl – «Параметры»

Во время проектирования работу со страницами проще всего производить, щелкнув на компоненте правой кнопкой мыши и выбрав из всплывшего меню соответствующий раздел: NewPage - создать новую страницу, NextPage или Previous Page - перейти к следующей или предыдущей странице. Каждая создаваемая страница является объектом типа TTabSheet. Это панель, на которой можно размещать любые управляющие компоненты, окна редактирования и т.п. Ее основные свойства:

Name Имя, по которому можно ссылаться на страницу

Caption Надпись, которая появляется на ярлычке закладки

PageIndex Индекс, по которому можно ссылаться на страницу

ImageIndex Индекс изображения, которое может появляться на ярлычке закладки

TPanel *Panel1; Панель на которой расположены все нижние компоненты TRadioGroup *RG1, TRadioGroup *RG2, TPanel *Panel3, TLabel *Label1, TLabel *Label2;

TPanel *Panel2; Верхняя панель «Настройка», на ней расположен TRxClock *Cl;

TPanel *Panel3; Панель расположенная на Panel1, на ней расположены TLabel *Label4, TLabel *Label3;

TLabel *Label1; Метка «Остановить», имеет 2 события Label1Click, Label1DblClick.

TLabel *Label2; Метка «Свернуть», имеет 1 событие Label2Click.

TLabel *Label3; Метка «Обработать», имеет 1 событие Label3Click.

TLabel *Label4; Метка «Инф.польз.», имеет 1 событие Label4Click.

TRxClock *Cl; Позволяет отображать время. У него есть настройки. При установке значения AlarmEnabled в True, при наступлении времени суток, заданного свойствами AlarmHour, AlarmMinute и AlarmSecond,

происходит событие OnAlarm, в обработчике которого вы можете произвести необходимые действия. У нас AlarmHour=8, и вызывает событие OnAlarm.

TRadioGroup *RG1; Применяется для формирования группы регулярно размещенных радиокнопок, из которых в любой момент времени может быть включена только одна. TRadioGroup - это панель, которая содержит радиокнопки, регулярно расположенные столбцами и строками. Из радиокнопок группы может быть включена только одна. При включении какой-то кнопки все остальные выключаются. Надпись в левом верхнем углу панели определяется свойством Caption. Надписи кнопок и их количество определяются свойством Items, имеющим тип TStrings. Во время проектирования задание свойства Items осуществляется вызываемым из Инспектора Объектов редактором списков строк. Сколько строчек вы запишете в нем, столько и будет кнопок. Определить, какую из кнопок выбрал пользователь, можно по свойству ItemIndex, которое показывает индекс выбранной кнопки (начинаются с 0). По умолчанию ItemIndex = -1, что означает отсутствие выбранной кнопки.
В данную группу входят Обновление, По макс. дате, Добавление

TRadioGroup *RG2; В данную группу входят SQL-сервер, Access Base , Все

TRichEdit *Memo1; Многострочное окно редактирования текстов в обогащенном формате .rtf, позволяющее производить выбор цвета, шрифта, поиск текста и т.д. Компонент TRichEdit представляет собой многофункциональное средство редактирования текстов, позволяющее работать с обогащенным форматом .rtf, т.е. выбирать различные атрибуты форматирования для разных фрагментов текста. В этом основное отличие TRichEdit от более простого компонента TMemo, в котором атрибуты форматирования одинаковы для всего текста.
TTimer *Timer1;Используется для запуска процедур, функций и событий в указанные интервалы времени. Запускается через 900 секунд при изменении в папке.
TTimer *Timer2;Запускается при простое 100 секунд и сворачивает программу в трей
TProgressBar *PB; Предназначен для отображения хода процессов, занимающих заметное время.

TDBNavigator *DBNavigator1; TDBNavigator - навигатор, позволяющий пользователю перемещаться по записям набора данных, редактировать данные и пересылать их в базу данных. Связан с DataSource2.

TDBGridEh *DBGridEh1; Компонент TDBGridEh позволяет отображать и редактировать записи наборов данных в виде таблицы Связан с DataSource2.

TDataSource *DataSource2; Компонент TDataSource представляет собой источник данных, который обеспечивает связь между набором данных и компонентами отображения и редактирования данных. Источник данных DataSet - > Logs_Query.

TDataSource *DataSource1; Источник данных DataSet - > ABase.

TADOQuery *Logs_Query; Компонент **ADOQuery** может использоваться в приложениях **ADO** вместо компонента **Query** приложений **BDE**, выполняющего аналогичные функции. Он применяется для выполнения произвольных запросов **SQL.** База данных задается свойством **ConnectionString**, или свойством **Connection**, подключающим компонент к **ADOConnection. Connection ->MySQL.**

TADOQuery *ABase; Соединение с базой **Connection** ->AccConnection

TADOQuery *SysQuery; Соединение с базой **Connection -> MySQL**

TADOQuery *MinMax; Соединение с базой **Connection** ->AccConnection

TADOConnection *MySQL; Компонент TADOConnection используется для соединения с различными источниками ADO и представляет собой версию объекта ADO Connection для Delphi.

Применение компонента TADOConnection дает разработчику ряд преимуществ:

- все компоненты доступа к данным ADO обращаются к хранилищу данных через одно соединение;
- возможность напрямую задать объект провайдера соединения;
- строка подключения хранится в одном месте, вместо того чтобы храниться в нескольких разных компонентах;
- выполнение транзакций;
- возможность выполнять команды ADO;
- расширенное управление соединением при помощи методов-обработчиков событий.

Компонент TADOConnection выполняет роль концентратора соединения с хранилищем данных. Для установления связи нужно с помощью свойства ConnectionString сформировать связные параметры и затем установить значение True в свойство Connected или вызвать метод Open. Для разрыва связи выполняется метод Close компонента или в его свойство Connected устанавливается значение False. До и после открытия и

закрытия соединения разработчик может использовать соответствующие стандартные методы-обработчики событий.

У нас класс MySQL используется для соединения с сервером SQL по указанному IP адрессу

TADOConnection *ACC; У нас используется для соединение с базой Access

TADOConnection *AccConnection; У нас используется для соединение с базой Access

TADOCommand *SysCommand; Компонент TADOCommand предназначен в основном для выполнение команд, не возвращающих результаты, таких как SQL-операторы языка определения данных DDL(Data Definition Language). К предложениям DDL относятся практически все запросы, которые не начинаются зарезервированным словом Select.

Текст исполняемой команды хранится в свойстве CommandText. За один прием компонент TADOCommand способен исполнить только одну команду.

При некоторых обстоятельствах компонент TADOCommand способен возвращать результаты. Для этого в него включены три реализации метода Execute, две из которых как раз и предназначены для создания набора записей. Использование возвращаемого набора данных возможно с помощью компонента-посредника TADODataSet:

```
ADODataSet1.RecordSet:=ADOCommand1.Execute;
```

Соединение с базой **Connection -> MySQL**

TADOCommand *Command; Соединение с базой **Connection -> MySQL**

TADOCommand *CommandA; Соединение с базой **Connection -> ACC**

TRxTrayIcon *RxT1; Компонента TRxTrayIcon предназначена для отображения иконки, заданной свойством Icon, в системной области (tray) панели задач (TaskBar) Windows95 или Windows NT 4.0 и старше. Иконка может быть статической (задается свойством Icon) либо анимированной (при задании значения свойству Icons и установке свойства Animated в значение True) - в этом случае частота смены изображения иконки задается свойством Interval.

Вы можете показывать и скрывать иконку, изменяя значение свойства Active. Свойство Hint задает текстовую строку, которая появляется над иконкой, когда вы останавливаете над ней мышь. Для тестирования компоненты в режиме дизайна Вы можете отобразить иконку в системной области панели задач, установив свойство ShowDesign в True.

Вы можете задать значение свойства PopupMenu - заданное вами меню (при значении его свойства AutoPopup = True) будет появляться при нажатии правой кнопки мыши на иконке в системной области. Если это меню имеет пункт "по-умолчанию" (со значением свойства Default = True, только в 32-

битной версии), то при двойном нажатии левой кнопки мыши будет вызван метод Click этого пункта меню. Вы можете также самостоятельно обработать события от мыши за счет использования обработчиков событий OnClick, OnDblClick и др., если не были вызваны обработчики по-умолчанию.

TRxFolderMonitor *FM; Компонента TRxFolderMonitor предназначена для отслеживания изменений в каталогах файловой системы, таких как создание, удаление, изменение файлов или каталогов, смена аттрибутов файлов и т.д. Когда происходит одно из изменений, определенных свойством Filter , вызывается событие OnChange.

Модуль SForm;

TBitBtn *BitBtn1; - кнопка с иконкой, «Сохранить и закрыть»
TEdit *ePath; - строка ввода к каталогу экспорта
TEdit *ePathAcc; - строка ввода пути к фалу базы
TLabel *Label1; - метка «Путь к каталогу экспорта:»
TLabel *Label2; - метка «Путь к файлу базы:»
TOpenDialog *OpenDialog1; - диалог открытия файлов Windows
TEdit *eLogs; - строка ввода списка обработки
TLabel *Label3; - метка «Список обр.:»
TEdit *eEvlog; - строка ввода получателя
TLabel *Label4; - метка «Получатель:»
TEdit *eServer; - строка ввода сервера MySQL
TLabel *Label5; - метка «Сервер MySQL:»

Приложение 2: Текст программы

Unit1.cpp

```
//---------------------------------------------------------------------------

#include <vcl.h>   //подключает заголовочный файл
//vcl.h - содержит описание визуальных компонентов Builder
#include <IniFiles.hpp>
#pragma hdrstop //завершает список файлов заголовков

#include "Unit1.h"      //подключает наш модуль
#include "SForm.h"      //подключает модуль другой формы
//---------------------------------------------------------------------------
#pragma package(smart_init) //говорит о том что модули включенные в пакет инициировать в порядке их зависимости
//подключение внешних библиотек(компонентов)
#pragma link "RXShell"
#pragma link "RXClock"
#pragma link "RxNotify"
#pragma link "RxGIF"
#pragma link "DBGridEh"
#pragma resource "*.dfm" //сообщает что для формы необходимо использовать файл
//*.dfm с именем данного файла(Unit1.dfm)
#define MYSQL_Serv 0x1    //определяет переменную
#define ACC_Serv 0x2     //определяет переменную
#pragma
//объявление переменных
TForm1 *Form1;        //форма
TDateTime min,max;     //переменные даты и времени
AnsiString debug_string="";    //строка
int prov;
bool Res = false;
//---------------------------------------------------------------------------

__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{

}
//---------------------------------------------------------------------------
/*Regim - 0 -режим работы обновление
        1 -режим работы по макс.дате
        2 -добавление
  Serv - получатель
       0 - SQL
```

```
      1 - Access
      2 - Все
           */
int TForm1::f_list(AnsiString in_file, int Regim, int Serv)
{
Application->ProcessMessages();  //заставляет обработать очередь сообщений приложения
немедленно
Res = false;
DateSeparator='/';
prov=0;
if(Serv != 0) //признак что база не только SQL
if(FileExists(ACC->DefaultDatabase)!=0) //проверяет существует ли файл _MAIN.MDB
//ACC - компонент TADOConnection связанный с базой Access
 {
   ACC->Open();  //открывает файл
   //выводит сообщение если файл не открыт
   if(!ACC->Connected)Memo1->Lines->Add("Ошибка открытия базы: "+ACC->DefaultDatabase);
 }

if(FileExists(DIR_L+"\\"+in_file)!=0) //проверяет существование файла
 {
   AccConnection->DefaultDatabase = DIR_L+"\\"+in_file;  //задает базу данных по умолчанию
   try {AccConnection->Open();} //открыть
   catch (Exception &EOleException)   //обработка исключения EOleException
   {
    Memo1->Lines->Add("Ощибка! Ошибка открытия базы: "+ AccConnection->DefaultDatabase);
//выводит строку
    return false; //выйти из функциив вернув 0
   }
   if(!AccConnection->Connected)return -1;  //если не соеденился вернуть -1
 }
else {Memo1->Lines->Add("Ошибка!");return -1;} //если нету файла вернуть -1

//MinMax -объект TADOQuery
//по умолчанию select min(evdate) , max(evdate) from
"\\work_3\export\imp001_050530_030003.mdb".evlog
//MinMax связан с базой AccConnection
//заполнить запрос: выбрать максимальную и минимальную дату(колонка evdate) из таблицы
evlog
//файла in_file переданного в функцию
MinMax->SQL->Text="select min(evdate) , max(evdate) from \""+DIR_L+"\\"+in_file+"\".evlog";
//открыть запрос(выполнить)
MinMax->Open();

if(Regim!=1)    //если параметр в вызове не равен 1    (не по максимальной дате)
//ABase -объект TADOQuery связанный с AccConnection   база imp*.mdb
```

```
//заполнить запрос : выбрать из таблицы evlog переданного файла  imp*.mdb все данные
//удовлетворяющие условию: параметр evcode равен 1,5 или 6, параметр user_id не равен 0
//userdirection равно 0 или 1.
ABase->SQL->Text="select * from \""+DIR_L+"\\"+in_file+"\".evlog where evcode in(1,5,6) and
user_id<>0 and userdirection in (0,1)";

if(Regim==1)    //если параметр в вызове равен 1   -режим по максимальной дате
{
//SysQuery -объект TADOQuery, связан с базой SQL
//заполнить запрос: выбрать максимальную дату как параметр max из таблицы evlog базы SQL
SysQuery->SQL->Text = "select max(evdate) as max from "+MyEvlog;
//выполнить
SysQuery->Open();
//ABase -объект TADOQuery связанный с AccConnection  база imp*.mdb
//заполнить запрос:выбрать из таблицы evlog переданного файла  imp*.mdb все данные
//удовлетворяющие условию параметр evcode равен 1,5 или 6, параметр user_id не равен 0
//userdirection равно 0 или 1, и evdate больше максимальной даты таблицы evloge базы SQL
ABase->SQL->Text="select * from \""+DIR_L+"\\"+in_file+"\".evlog where evcode in(1,5,6) and
user_id<>0 and userdirection in (0,1) and evdate>"+
FormatDateTime("#mm/dd/yyyy hh.mm.ss#", SysQuery->FieldByName("max")->AsDateTime);
//преобразовать данные полученные из запроса
//к текущему виду
SysQuery->Close();//закрыть запрос(закончить выполнение)
}
//выполнить запрос
ABase->Open();




if(Regim==0)    //если параметр в вызове равен 0   (обновление базы)
  {
  //Для SQL
  //Command - TADOCommand , SQL
  //заполнить запрос
  // удалить все из таблицы evlog базы SQL где evdate>min значения колонки
  // evdate из таблицы evlog файла imp*.mdb,и  evdate<max значения из тойже таблицы
    Command->CommandText="delete from "+MyEvlog+" where evdate>=\'"+
    FormatDateTime("yyyy-mm-dd hh:mm:ss", MinMax->FieldByName("expr1000")->AsDateTime)+"\'
and evdate<=\'"+
    FormatDateTime("yyyy-mm-dd hh:mm:ss", MinMax->FieldByName("expr1001")->AsDateTime)+"\'";
     //Для Access
  //CommandA - TADOCommand    ACC - _MAIN.MDB
  // удалить все из таблицы evlog базы _MAIN.MDB где evdate>min значения колонки
  // evdate из таблицы evlog файла imp*.mdb,и  evdate<max значения из тойже таблицы
    CommandA->CommandText="delete from "+MyEvlog+" where evdate>="+
```

```
    FormatDateTime("#mm/dd/yyyy hh.mm.ss#", MinMax->FieldByName("expr1000")->AsDateTime)+"
and evdate<="+
    FormatDateTime("#mm/dd/yyyy hh.mm.ss#", MinMax->FieldByName("expr1001")->AsDateTime);

    switch(Serv) //выбрать какой запрос выполнить
    {
    case 0:                    //SQL
      Command->Execute(); //выполнить 1-й запрос
      break;
    case 1:              //Access
      CommandA->Execute(); //выполнить 2-й запрос
      break;
    case 2:                //Все
      //выполнить оба запроса
      CommandA->Execute();
      Command->Execute();
      break;
    }
  }

AnsiString Cmd="", CmdA="";       //переменные
 //PB-TProgressBar
 //установить прогресс выполнения от 0 до количества записей
PB->Position = 0;  //на начале
PB->Max=ABase->RecordCount; //по количеству записей
PB->Visible=true; //сделать видимой
//ABase -объект TADOQuery связанный с AccConnection  база imp*.mdb
while(!ABase->Eof)   //пока не дошли до конца файла
 {
  if(Res) {PB->Visible=false; return -2;}
 //если Res стало истина выйти(признак нажатия "Остановить")
  Cmd="insert into "+MyEvlog+"(";      //присвоить строке
  CmdA = "insert into "+MyEvlog+"(";    //присвоить строке
  for(int i=0;i<ABase->FieldCount;i++)  //цикл до последнего столбца
   {
    if(i!=0)    //если не первый проход
     {
      Cmd = Cmd+", "+ABase->Fields->Fields[i]->FieldName; //дописываем в строку
      CmdA += ", "+ABase->Fields->Fields[i]->FieldName;   //дописываем в строку
     }
    else //если первый
     {
      Cmd = Cmd+ABase->Fields->Fields[i]->FieldName;
      CmdA += ABase->Fields->Fields[i]->FieldName;
     }
   }
```

```
  CmdA+= ") values(";
  Cmd = Cmd+") values(";
  for(int j=0;j<ABase->FieldCount;j++)
   {
    if(j>0) //если не первый проход
     {
      if(j==1) //дату дописать
       {
        CmdA += ", "+FormatDateTime("#mm/dd/yyyy hh.mm.ss#", ABase->FieldByName(ABase-
>Fields->Fields[j]->FieldName)->AsDateTime);
        Cmd += ", \'"+FormatDateTime("yyyy-mm-dd hh:mm:ss", ABase->FieldByName(ABase->Fields-
>Fields[j]->FieldName)->AsDateTime)+"\'";
       }
      else
       if(j<21)
        {
         CmdA += ", \'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
         Cmd += ", \'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
        }
      else
       {
        CmdA += ", "+FormatDateTime("#mm/dd/yyyy hh.mm.ss#", ABase->FieldByName(ABase-
>Fields->Fields[j]->FieldName)->AsDateTime);
        Cmd += ", \'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsDateTime+"\'";
       }
     }
    else //первый проход
     {
      Cmd=Cmd+"\'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
      CmdA+="\'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
     }
   }
  Application->ProcessMessages(); //выполнить всю очередь
  Cmd=Cmd+")"; //завершить строку
  CmdA+=")"; //завершить строку

  CommandA->CommandText=CmdA;  //заполнить текст коианд
  Command->CommandText=Cmd;    //заполнить текст коианд

  switch(Serv)    //выбрать по параметру и выполнить (зависит кто получатель)
  {
  case 0:                 //SQLServer
   Command->Execute();
   break;
  case 1:               //Access
```

```
   CommandA->Execute();
   break;
  case 2:               //все
   CommandA->Execute();
   Command->Execute();
   break;
  }
  prov++;       //   счетчик сколько записей добавили
  PB->StepBy(1);   //изменить позицию на 1
  ABase->Next();  //   переместить курсор на следующую запись
  }
PB->Visible=false;   //скрыть полосу выполнения
return ABase->RecordCount;    //вернуть количество строк
}
//---------------------------------------------------------------------------

bool TForm1::Find()
{
Memo1->Clear(); //очистить
try {MySQL->Open();}  //выполнить
//обработать исключение
catch (Exception &EOleException){Memo1->Lines->Add("Ошибка! Недоступен сервер MySQL:
["+SERVER+"]");return false;}
if(!MySQL->Connected)return false; //если не соединились - выйти
//FM - TRxFolderMonitor
FM->Active=false; //отключить
AnsiString row_str;  //строка
Memo1->Lines->Add("Идёт поиск в каталоге: "+DIR_L); //вывести сообщение
Application->ProcessMessages();   //обработать все события
DWORD start=GetTickCount(); //извлекает число миллисекунд, которые истекли с тех пор как
система была запущена
DWORD t_list; //переменная
TSearchRec sr;   //переменная для хранения данных о файле
SetCurrentDir(DIR_L);  //установить текущую дерикторию   читается из настроек
if(FindFirst("imp*.mdb",faAnyFile,sr)==0)   //поиск в каталоге   файла соответствующего описания
{
do
 {
 //заполнить запрос текущей формы
// Log  берется из ini файла, у нас Logs=logs
//выбрать имена файлов из таблицы LOG где имя файла совпадает с текущим
 Form1->SysQuery->SQL->Text="select files from "+LOG+" where files='"+sr.Name+"'";
 //выполнить
 Form1->SysQuery->Open();
 //если параметр не задан, тоесть результат запроса пуст, данных о файле нету
 if(Form1->SysQuery->FieldByName("files")->AsString=="")//если нет файла
```

```
 {
 t_list=GetTickCount();    //время
 Memo1->Lines->Add("Обработка: " + sr.Name); //вывести   имя файла
 row_str=f_list(sr.Name, RG1->ItemIndex, RG2->ItemIndex);//вызвать свою функцию исходя из того
какой режим выбран и получатель
 //заполнить команду записать в Лог
 //добавить в LOG значение колонок (files,rows,time_out,rw)
 //равные значениям (sr.Name,row_str,FloatToStr((GetTickCount()-t_list)/1000),+IntToStr(prov))
соответственно
    SysCommand->CommandText=
      "insert into "+LOG+"(files,rows,time_out,rw) values('"+
         sr.Name+"','"+row_str+"','"+FloatToStr((GetTickCount()-t_list)/1000)+
          "','"+IntToStr(prov)+")";
  //выполнить
  SysCommand->Execute();
  AnsiString St1=DIR_L+"\\"+sr.Name,St2="c:\\arhiv\\"+sr.Name; //обьявить 2 переменные
 CopyFile(St1.c_str(), St2.c_str(), true);//копирование файла в директорию
  Memo1->Lines->Add("Обработка файла: ~ "+FloatToStr((GetTickCount()-t_list)/1000)+" сек.");
//вывести сообщение
   Application->ProcessMessages();    //обработать события
 }
 }while(FindNext(sr)==0); //пока есть строки для обработки, тоесть есть еще файлы
удовлетворяющие условию
}
FindClose(sr);   //закрыть успешный поиск
Memo1->Lines->Add("Поиск завершен: "+DateToStr(Now())+" "+TimeToStr(Time()));    //выдает
время и дату
Memo1->Lines->Add("Время выполнения: "+FloatToStr((GetTickCount()-start)/1000)+" сек.");
//выдает время работы
FM->Active=true;   //включить
MySQL->Close();   //закрыть
ACC->Close();    //закрыть
AccConnection->Close();    //оключиться
return true;           //вернуть что завершено успехом
}
//---------------------------------------------------------------------------

void __fastcall TForm1::RxT1DblClick(TObject *Sender)
{
Form1->Visible=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Label1Click(TObject *Sender)
{
Res = true;
```

```
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Label2Click(TObject *Sender)
{
Form1->Hide();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Label3Click(TObject *Sender)
{
Panel3->Visible = false;
Find();
Panel3->Visible = true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::FormActivate(TObject *Sender)
{
ReadP();    //наша функция подготовки данных
}
//---------------------------------------------------------------------------

void __fastcall TForm1::RxT2DblClick(TObject *Sender)
{
Form1->Visible=true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::ClAlarm(TObject *Sender)
{
Memo1->Clear();      //очистка
Memo1->Lines->Add("Сброс инф. польз. "+IntToStr(Upd_info())+" сек."); //вывод строки
/*
IntToStr(Upd_info()) - преобразует целое в строку
Upd_info() - функция обновления информации
*/
}
//---------------------------------------------------------------------------

void __fastcall TForm1::FMChange(TObject *Sender)
{
Timer1->Enabled=true;    //включить таймер
}
//---------------------------------------------------------------------------
```

```
void TForm1::update_info(AnsiString table_name,AnsiString file_copy)
{
if(SysQuery->FieldByName("files")->AsString!="")
{
Command->CommandText="delete from "+table_name;
Command->Execute();

ABase->SQL->Text="select * from \""+DIR_L+"\\"+file_copy+"\"."+table_name;
ABase->Open();

PB->Max=ABase->RecordCount;   //полоса % выполнения
PB->Visible=true;

AnsiString Cmd="";
while(!ABase->Eof)         //пока не конец файла
{
Cmd="insert into "+table_name+"(";   //вставить в таблицу
for(int i=0;i<ABase->FieldCount;i++)   //количество столбцов
{
//для не первой строки
if(i!=0)Cmd = Cmd+", "+ABase->Fields->Fields[i]->FieldName;   //имя столбца
//для первой строки
else Cmd = Cmd+ABase->Fields->Fields[i]->FieldName; //имя столбца
}
Cmd = Cmd+") values(";
for(int j=0;j<ABase->FieldCount;j++)
{
if(j>0)Cmd=Cmd+", \'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
//значение столюца
else Cmd=Cmd+"\'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
//значение столбца
}
Cmd=Cmd+")";
Command->CommandText=Cmd; //запомнить всю строку
Command->Execute(); //выполнить
ABase->Next();//перейти к следующей строке
PB->StepBy(1);
}
PB->Visible=false;
Memo1->Lines->Add("Обновление таблицы \""+table_name+"\" выполнено: "+TimeToStr(Time()));
}
else Memo1->Lines->Add("Ошибка сброса инф. польз.");
Application->ProcessMessages();
}
//---------------------------------------------------------------------------
```

```
int TForm1::Upd_info()
{
try {MySQL->Open();}     //пробуем запустить обработку
catch (Exception &EOleException) //обработка исключения
 {
  Memo1->Clear(); //очистить
  Memo1->Lines->Add("Ошибка! Недоступен сервер MySQL: ["+SERVER+"]"); return -1; //вывести
ошибку
  }
FM->Active=false; //
SysQuery->SQL->Text="select files from "+LOG+" order by files";     //занести запрос
SysQuery->Open(); //выполнить запрос
SysQuery->Last(); //возвращает последнее значение
AnsiString cfile=SysQuery->FieldByName("files")->AsString;
if(FileExists(DIR_L+"\\"+cfile))    //проверить существует ли файл
 {
   AccConnection->DefaultDatabase = DIR_L+"\\"+cfile; //сделать базой по умолчанию
   AccConnection->Open(); //открыть
 }
else {Memo1->Lines->Add("Ошибка!");return -1;} //если нету выдать ошибку

DWORD stat=GetTickCount(); //проверить время
if(RG2->ItemIndex!=1)
{
   update_info("users",cfile);
   update_info("dolgnost",cfile);
   update_info("shedules",cfile);
   update_info("tree",cfile);
   update_info("contr",cfile);
}
if(RG2->ItemIndex!=0)
{
   update_infoA("users",cfile);
   update_infoA("dolgnost",cfile);
   update_infoA("shedules",cfile);
   update_infoA("tree",cfile);
   update_infoA("contr",cfile);
}
MySQL->Close();
AccConnection->Close();
FM->Active=true;
return ((GetTickCount()-stat)/1000);
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Label4Click(TObject *Sender)
```

```
{
Panel3->Visible = false;
int Ssec = Upd_info();
if(Ssec<0)Memo1->Lines->Add("Сброс инф. польз. ОШИБКА");
else Memo1->Lines->Add("Сброс инф. польз. "+IntToStr(Ssec)+" сек.");
Panel3->Visible = true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Timer1Timer(TObject *Sender)
{
Timer1->Enabled=false;   //отключить таймер
Find(); //вызвать функцию
}
//---------------------------------------------------------------------------

//аналогично void TForm1::update_info(AnsiString table_name,AnsiString file_copy)
void TForm1::update_infoA(AnsiString table_name,AnsiString file_copy)
{
if(FileExists(ACC->DefaultDatabase)!=0)
 {
   ACC->Open();
   if(!ACC->Connected)
    {
      Memo1->Lines->Add("Ошибка открытия базы: "+ACC->DefaultDatabase);
      return;
    }
 }
if(SysQuery->FieldByName("files")->AsString!="")
{
CommandA->CommandText="delete from "+table_name;
CommandA->Execute();

ABase->SQL->Text="select * from \""+DIR_L+"\\"+file_copy+"\"."+table_name;
ABase->Open();
PB->Max=ABase->RecordCount;
PB->Position = 0;
PB->Visible=true;

AnsiString Cmd="", CmdZag="";

CmdZag="insert into "+table_name+"(";
for(int i=0;i<ABase->FieldCount;i++)
 {
   if(i!=0)CmdZag+=", "+ABase->Fields->Fields[i]->FieldName;
   else CmdZag+=ABase->Fields->Fields[i]->FieldName;
```

```
 }
   CmdZag+=") values(";

while(!ABase->Eof)
{
Application->ProcessMessages();
Cmd="";
for(int j=0;j<ABase->FieldCount;j++)
 {
   if(j>0)Cmd=Cmd+", \'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
   else Cmd=Cmd+"\'"+ABase->FieldByName(ABase->Fields->Fields[j]->FieldName)->AsString+"\'";
 }
Cmd+=")";
Cmd = CmdZag + Cmd;
CommandA->CommandText=Cmd;
CommandA->Execute();
ABase->Next();
PB->StepBy(1);
}
PB->Visible=false;
Memo1->Lines->Add("Обновление таблицы \""+table_name+"\" выполнено: "+TimeToStr(Time()));
Application->ProcessMessages();
}
else Memo1->Lines->Add("Ошибка сброса инф. польз.");
ACC->Connected = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Panel2DblClick(TObject *Sender)
{
ServForm->ShowModal();
}
//---------------------------------------------------------------------------
void TForm1::ReadP()
{
MySQL->Connected = false;    //отключается от базы
//объявляется переменную типа TIniFile, инициируется переменная
//ExtractFilePath - извлекает путь до последнего файла
//Application->ExeName - возвращает путь с именем запущенного exe файла
TIniFile *in1 = new TIniFile(ExtractFilePath(Application->ExeName)+"\\SE.ini");
//считывает данные с ини файла
DIR_L = in1->ReadString("SA","path","");
LOG = in1->ReadString("SA","Logs","");
SERVER = in1->ReadString("SA","Server","");
ACC->DefaultDatabase = in1->ReadString("SA","Base","");
```

//инициирует строку соединения
//полная справка по адрессу http://msdn.microsoft.com/ru-ru/library/ms130822.aspx
//Provider - Имя драйвера источника данных (SQLOLEDB.1 для MS SQL server)
//Persist Security Info - необходимость использования шифрованого канала
//Extended Properties - тип строка. Дополнительные параметры провайдера. Провайдеры могут иметь внутренние параметры, которые не представлены в свойствах, для установки этих параметров и существует это свойство.
//DRIVER - имя драйвера для текущего подключения
//SERVER - сервер базы данных, у нас SERVER=173.33.7.30
//UID = "sa" 'логин пользователя SQL-сервера
//PWD = "sa" 'пароль пользователя SQL-сервера
/*APP - Имя приложения, вызывающего функцию SQLDriverConnect (необязательно). Если это значение указано, оно хранится в столбце program_name в таблице master.dbo.sysprocesses и возвращается функциями sp_who и APP_NAME.*/
/*WSID -Идентификатор рабочей станции. Обычно это сетевое имя компьютера, на котором находится приложение (необязательно). Если это значение указано, хранится в системной таблице master.dbo.sysprocesses в столбце hostname и возвращается функциями sp_who и HOST_NAME*/
/*DATABASE - Имя базы данных SQL Server, используемой для соединения по умолчанию. Если ключевое слово Database не указано, используется база данных, определенная по умолчанию для имени входа.База данных по умолчанию из источника данных ODBC переопределяет базу данных по умолчанию, определенную для имени входа. База данных должна существовать, если не указан параметрAttachDBFileName. Если также указано ключевое слово AttachDBFileName, выполняется присоединение первичного файла, на который оно указывает, после чего присваивается имя базы данных, указанное ключевым словом Database.*/
/*Network -Допустимые значения: dbnmpntw (именованные каналы) и dbmssocn (TCP/IP).Одновременное указание значения для ключевого слова Network и префикса протокола в ключевом слове Server является ошибкой.*/
/*AutoTranlate - Если имеет значение «yes», строки символов ANSI, которыми обмениваются клиент и сервер, переводятся с использованием Юникода, чтобы минимизировать проблемы сопоставления символов национальных алфавитов кодовых страниц клиента и сервера.Если имеет значение «no», перевод символов не выполняется.*/
/*QuotedId - Если имеет значение «yes», параметр QUOTED_IDENTIFIERS при соединении устанавливается в значение ON. SQL Server использует правила ISO в отношении использования кавычек в инструкциях SQL. Если параметр равен «no», то параметр QUOTED_IDENTIFIERS для соединения отключается (принимает значение OFF). В этом случае SQL Server следует правилам устаревших версий Transact-SQL относительно использования кавычек в инструкциях SQL. Дополнительные сведения см. в разделе Действие параметров ISO. */
//данная строка генерируется автоматически при помещении компонента ADOConnection и выборе ... в object Inspector -> ConnectionString

```
MySQL->ConnectionString = "Provider=MSDASQL.1;Persist Security Info=False;Extended
Properties=\"DRIVER=SQL Server;SERVER="+SERVER+
";UID=sa;PWD=sa;APP=Microsoft Office
2003;WSID=C0;DATABASE=main;Network=DBMSSOCN;AutoTranslate=No;QuotedId=No\"";
```

```
MyEvlog = in1->ReadString("SA","EvlogName","");
delete in1;  //освобождает переменную
FM->FolderName=DIR_L; //устанавливает директорию для наблюдения path=\\Work_3\arhiv\
FM->Active=true; //активирует наблюдение
Memo1->Lines->Clear(); //очистка окна

//DBQ-путь к базе Base=\\S1\v1003\NTAB\_main.mdb
//данная строка генерируется автоматически при помещении компонента ADOConnection и
выборе ... в object Inspector -> ConnectionString

ACC->ConnectionString = "Provider=MSDASQL.1;Persist Security Info=False;Extended
Properties=\"DBQ="+
ACC->DefaultDatabase+";Driver={Driver do Microsoft Access (*.mdb)};"+
"DriverId=25;FIL=MS Access;MaxBufferSize=2048;MaxScanRows=8;PageTimeout=5;"+
"SafeTransactions=0;Threads=3;UID=admin;UserCommitSync=Yes;\"";

AccConnection->ConnectionString = ACC->ConnectionString;
  Memo1->Lines->Add("Каталог экспорта: " + DIR_L); //вывести в окно редактирования
//проверяет наличие базы _main.mdb
if(!FileExists(ACC->DefaultDatabase))
  Memo1->Lines->Add("Не найден путь к базе: "+ACC->DefaultDatabase); //если нету
else
  Memo1->Lines->Add("Файл базы на S1: " + ACC->DefaultDatabase); //если есть

  //проверяет наличие строк в ини файле
  if(MyEvlog == "") Memo1->Lines->Add("Не найдена выходная таблица: " + MyEvlog);
//EvlogName=evlog
  else Memo1->Lines->Add("Выходная таблица: " + MyEvlog);
  if(LOG == "") Memo1->Lines->Add("Не найдена таблица: " + LOG); //Logs=logs
  else Memo1->Lines->Add("info - таблица: [" + LOG+"]");

//создать строку запроса,выбрать все столбцы из таблицы logs и отсортировать в обратном
порядке (z-a)
Logs_Query->SQL->Text = "select * from " + LOG + " order by files desc";
//запускает
Logs_Query->Open();
//данные выведутся на вкладку "параметры"
// Long_Query->Connection->MySQL (для соединения используем MySQL)
// DataSourse2->DataSet->Logs_Query
// DBGridEh1->DataSourse->DataSourse2

}
//---------------------------------------------------------------------------


void __fastcall TForm1::Label1DblClick(TObject *Sender)
```

```
{
Application->Terminate();
}
//---------------------------------------------------------------------------
```

SForm.cpp

```
//---------------------------------------------------------------------------

#include <vcl.h>
#include <IniFiles.hpp>   //юиюлиотека работыс ini файлами

#pragma hdrstop

#include "SForm.h"
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TServForm *ServForm;
//---------------------------------------------------------------------------
__fastcall TServForm::TServForm(TComponent* Owner)
  : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TServForm::BitBtn1Click(TObject *Sender)
{
/*
TIniFile -Тип объекта, отображающего файл .INI
Файлы .INI - это текстовые файлы, предназначенные в 16-разрядных Windows 3.x для хранения
информации о настройках различных приложений. Информация логически группируется в
разделы, каждый из которых начинается оператором заголовка, заключенным в квадратные
скобки. Например, [Desktop]. В строках, следующих за заголовком содержится информация,
относящаяся к данному разделу, в форме:
<ключ>=<значение>
<keyname>=<value>
Когда в приложении создается объект типа TIniFile, ему передается как FileName имя файла, с
которым он связан. Свойства и методы TIniFile позволяют читать из файла .INI, записывать в него
информацию, удалять целые разделы.*/
TIniFile *Oini=new TIniFile(ExtractFilePath(Application->ExeName)+"\\SE.ini");   //обькт класса TIniFile и
связываем его с файлом Se.ini
/*
ExtractFilePath( const FileName: string):  string Возвращает путь к каталогу (включая указание диска
и слэш перед именем файла), вырезанный из строки с полным именем файла FileName.
*/
Oini->WriteString("SA", "path", ePath->Text);     //путь к каталогу экспорта
/*
WriteString       Записывает значение Value в строку ключа Ident раздела Section. Если нужной
строки или раздела нет, они создаются.
```

```
*/
Oini->WriteString("SA", "Base", ePathAcc->Text); //путь к файлу базы
Oini->WriteString("SA", "Logs", eLogs->Text); //Список обр.
Oini->WriteString("SA", "EvlogName", eEvlog->Text);//получатель
Oini->WriteString("SA", "Server", eServer->Text);//Сервер
Oini->UpdateFile();
/*
UpdateFile       Очищает буфер и записывает файл на диск.
*/
delete Oini; //удалить обьект
Form1->ReadP();  //вызов функции из unit1.cpp для повторной подготовки работы программы
Close(); //закрыть форму
}
//---------------------------------------------------------------------------
void __fastcall TServForm::ePathDblClick(TObject *Sender)  //при двойном щелчке на
//строке ввода каталога базы экспорта произвести следующие действия
{
//установить для диалога открытия файла следующие параметры
//расшерение
OpenDialog1->DefaultExt = ".mdb";
//фильтер
OpenDialog1->Filter = "Base(*.mdb)|*.mdb";
//запустить диалог открытия файла
if(OpenDialog1->Execute())      //если работа диалога завершена успешно, то
 {
//Функция ExtractFilePath извлекает из FullFileName подстроку пути.
//Это часть полного имени файла включая конечный \ перед именем файла.
   ePath->Text = ExtractFilePath(OpenDialog1->FileName); //в строку вывести путь к файлу
 }
}
//---------------------------------------------------------------------------
void __fastcall TServForm::ePathAccDblClick(TObject *Sender)      //при двойном щелчке на
//строке ввода путь к файлу базы  произвести следующие действия
{
//установить для диалога открытия файла следующие параметры
//расшерение
OpenDialog1->DefaultExt = ".mdb";
//фильтер
OpenDialog1->Filter = "Base(*.mdb)|*.mdb";
//запустить диалог открытия файла
if(OpenDialog1->Execute())//если работа диалога завершена успешно, то
 {
   ePathAcc->Text = OpenDialog1->FileName;//в строку вывести путь к файлу
 }
}
//---------------------------------------------------------------------------
```

```
void __fastcall TServForm::FormShow(TObject *Sender)    //при отображении формы на экран
{
TIniFile *Rini = new TIniFile(".\\SE.ini"); //создает обьект и связывает его с файлом
//читаем данные из файла и сохраняем в строки
ePath->Text = Rini->ReadString("SA","path","");      //путь к каталогу экспорта
ePathAcc->Text = Rini->ReadString("SA","Base","");    //путь к файлу базы
eLogs->Text = Rini->ReadString("SA","Logs","");       //Список обр.
eEvlog->Text = Rini->ReadString("SA","EvlogName","");   //получатель
eServer->Text = Rini->ReadString("SA", "Server", "");   //Сервер
delete Rini;          //уничтожить обьект
}
//---------------------------------------------------------------------------
```

Приложение 3: Функции используемые в программе, краткое описание

Функции модуля Unit1

void __fastcall RxT1DblClick(TObject *Sender); - при двойном щелчке разворачивает программу из трея в обычный режим(вернее делает форму видимой, по щелчку на иконке).

void __fastcall Label1Click(TObject *Sender); - это нажатие на метку «Остановить», меняет глобальный признак, по которому происходит остановка цикла.

void __fastcall Label2Click(TObject *Sender); - это нажатие метки «Свернуть», сворачивает форму в трей(вернее просто прячет форму, но она остается видимой в трее)

void __fastcall Label3Click(TObject *Sender); - это нажатие на метку «Обработать», вызывает функцию Find

void __fastcall FormActivate(TObject *Sender); - вызывается при запуске программы, вызывает функцию ReadP.

void __fastcall ClAlarm(TObject *Sender); - вызывается по расписанию в 8:00:00 и вызывает Upd_info.

void __fastcall FMChange(TObject *Sender); - вызывается при изменении в отслеживаемой папке, включает таймер 1.

void __fastcall Label4Click(TObject *Sender); - вызывается при нажатии на метку «Инф.польз.», вызывает Upd_info.

void __fastcall Timer1Timer(TObject *Sender); - отключает Timer1 и вызывает функцию Find.

void __fastcall Panel2DblClick(TObject *Sender); - при двойном нажатии на панели «Настроить», активирует форму настроек ServForm.

void __fastcall Label1DblClick(TObject *Sender); - обрабатывает накопившиеся события и закрывает форму.

int f_list(AnsiString in_file, int Regim, int Serv); - получает 3 параметра in_file - имя реально найденного файла, Regim - режим работы (0-обновление, 1- по макс.дате, 2 - добавление), Serv - кто получатель(0 – SQL, 1 –Access, 2 - оба). Функция выбирает данные из таблицы evlog in_file и заносит данные в таблицу evlog базы SQL, Access или обеих, в зависимости от получателя. А в зависимости от режима меняется запрос на выборку из evlog in_file, и производим удаление из базы evlog SQL, Access или обеих. В случаи успеха возвращаем количество добавленных записей, при неудаче – код ошибки

bool Find(); - проверяет наличие файла imp*.mdb в инфо таблице logs, и если его нету в инфо таблице то заносим его в инфо таблицу, предварительно обработав функцией f_list(то есть занеся данные из файла в базу SQL, Access или обе в зависимости от режима). Возвращает true если все прошло успешно.

void update_info(AnsiString table_name,AnsiString file_copy); - 2 параметра имя таблицы и имя файла найденного в лог файле(последнего файла), очищаем таблицу table_name. Выбираем все данные из таблицы с именем table_name из файла file_copy. И заносим их в таблицу table_name SQL. Таким образом мы обновляем таблицу SQL.

void update_infoA(AnsiString table_name,AnsiString file_copy); - аналогично пред идущей только для Access

int Upd_info(); - функция обновления баз данных в зависимости от получателя

void ReadP(); - функция загружает исходные данные из ini файла и создает соединение с базами данных SQL и Access

Функции модуля SForm

void __fastcall BitBtn1Click(TObject *Sender); - вызывается при нажатии на кнопку «Сохранить и закрыть» Сохраняет данные в ini файл и закрывает форму ServForm.

void __fastcall ePathDblClick(TObject *Sender); - при двойном щелчке на строку ввода ePath открывает диалог открытия файла и делаем текс строки равный пути к файлу mdb(каталог).

void __fastcall ePathAccDblClick(TObject *Sender); - при двойном щелчке на строку ввода ePathAcc открывает диалог открытия файла и делаем текс строки равный пути к файлу mdb(полный путь к файлу).

void __fastcall FormShow(TObject *Sender); - выводит данные из ini файла в строки формы, при активации