УВАЖАЕМЫЕ СТУДЕНТЫ! Выполните практические задания.

Результаты работы: файл-отчет предоставить преподавателю на e-mail: xvsviv@rambler.ru в трехдневный срок.

При возникновении вопросов по приведенному материалу обращаться по следующим номерам телефонов:072-138-93-11.

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Учебная практика

Работа с макросами в СУБД MS ACCESS

Цель работы: Ознакомиться с использованием макросов Access

Задание 1. Используется база данных Техникум.

1) Ознакомьтесь с макросами Access (см. текст Средства автоматизации Access после заданий к практике)

2) Создать форму, которая загружается автоматически при запуске Access и запрашивает статус посетителя. Если статус – руководитель группы, то следующая форма Студент открывается в режиме изменения данных, если преподаватель – то только для чтения, если другое – то выдается сообщение «статус неверен» и форма вообще не открывается

3) Создать форму Ведомость, которая при просмотре позволяет использовать фильтр по шаблону названия дисциплины, а также - по наименованию специальности

4) Предусмотреть возможность одновременного применения фильтра по названию дисциплины и наименованию специальности

5) Кнопка Применить выполняет фильтрацию по заданным условиям, а кнопка Отменить – отменяет фильтр, выводит все записи и очищает поля фильтра.

Пример формы для базы данных Торговля (см. рисунок 1)

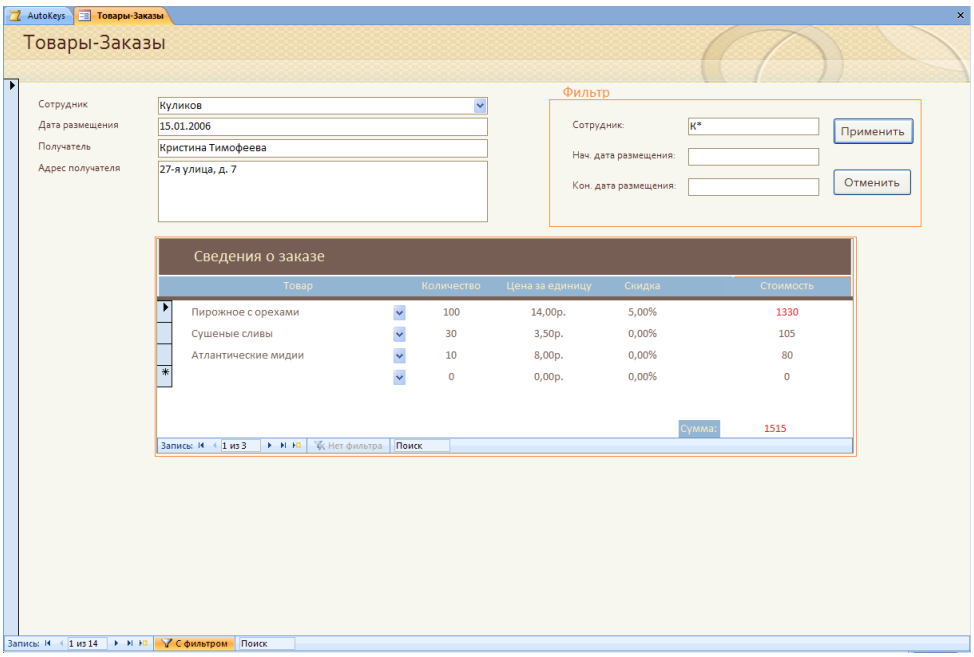


Рисунок 1

Средства автоматизации в Access

Для автоматизации работы в Access служат *макросы* (небольшие программы на языке макрокоманд системы Access) и *модули* (процедуры на языке Visual Basic for Application, VBA). С их помощью можно существенно расширить функциональные возможности создаваемого вами приложения и настроить его на нужды конкретных пользователей.

МАКРОСЫ

С помощью макросов можно выполнить практически все действия над объектами Access из тех, которые выполняются «в ручную». Основное назначение макросов — это создание удобного интерфейса приложения: чтобы формы и отчеты открывались при нажатии кнопок в форме или на панели инструментов или же привычным выбором команды меню; чтобы при открытии приложения пользователь видел на экране не окно База данных (Database), наполненное множеством таблиц, запросов, форм и отчетов, а некую понятную форму, с помощью которой можно было бы сразу производить желаемые действия и т. д.

Однако использование макросов имеет и некоторые недостатки, о которых здесь уместно сказать.

– Возможности макрокоманд ограничены по сравнению с возможностями языка VBA, поэтому в ряде случаев без программирования на VBA не обойтись, хотя сначала нужно быть уверенным, что эти дополнительные возможности действительно нужны. Язык VBA предоставляет более широкие возможности для работы с данными, позволяет использовать механизм программирования объектов для связи с другими приложениями, вызывать функции из библиотек динамической загрузки (DLL) Windows и создавать собственные специализированные функции.

– Макросы можно использовать практически везде, где применяются процедуры VBA, однако процедуры VBA, как правило, выполняются быстрее.

– Макросы являются объектами, существующими отдельно от форм и отчетов, в которых они используются, поэтому, когда этих объектов становится очень много, их поддержка достаточно трудоемка. Процедуры обработки событий VBA являются неотъемлемой частью форм и отчетов, и в этом есть свои преимущества. Например, при переносе форм и отчетов из одной базы данных в другую с ними автоматически переносятся связанные процедуры.

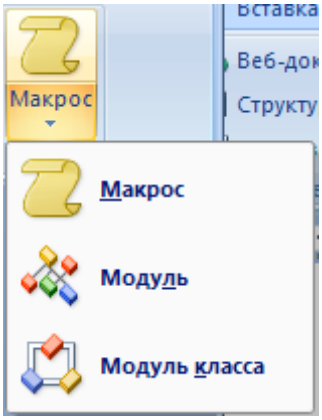
Тем не менее использование макросов оправдано тем, что их легко создавать, и для этого не нужно изучать синтаксис языка программирования.

Макрос в Access представляет собой структуру, состоящую из одной или нескольких макрокоманд, которые выполняются либо последовательно, либо в порядке, заданном определенными условиями. Каждая макрокоманда имеет определенное имя и, возможно, один или несколько аргументов, которые задаются пользователем. Например, при использовании макрокоманды ОткрытьФорму (OpenForm) в качестве аргументов необходимо задать, по крайней мере, имя открываемой формы и режим вывода ее на экран.

В приложении 1 содержится список макрокоманд, сгруппированных по категориям, подробности см. в справке на Microsoft Access.

Конструктор макросов

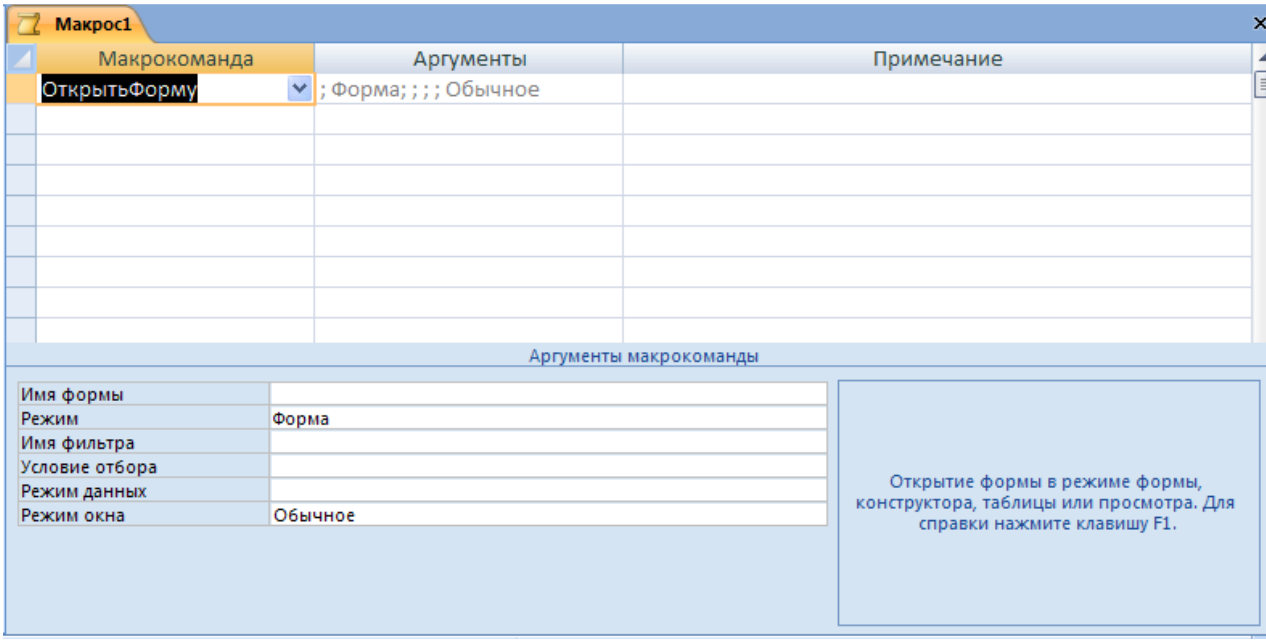
Находится на вкладке Создать\Другие\Макрос



Окно Конструктора макросов построено аналогично окну Конструктора таблиц, т. е. разделено по горизонтали на две части: панель описаний и панель аргументов.

Верхняя часть окна Конструктора — панель описаний — состоит из нескольких столбцов. По умолчанию на этой панели выводится два столбца: "Макрокоманда" (Action) и "Примечание" (Comments). Панель описаний позволяет определить последовательность макрокоманд, из которых состоит макрос.

Строка в столбце "Макрокоманда" (Action) представляет собой поле со списком, в котором можно выбрать нужную макрокоманду.



Строка в столбце "Примечание" (Comments) — это обычное текстовое поле, в которое можно ввести комментарий, описывающий выполняемое действие.

Когда поле "Макрокоманда" (Action) заполнено, в нижней части окна Конструктора макросов появляется панель аргументов, предназначенная для ввода значений аргументов соответствующей макрокоманды. Список полей на этой панели зависит от выбранной макрокоманды и может отсутствовать, если макрокоманда не имеет аргументов. Таким образом, при создании макросов не нужно запоминать список аргументов для каждой макрокоманды.

На рис. проиллюстрирован выбор макрокоманды ОткрытьФорму. На панели аргументов имеются два поля, которые позволяют задать имя фильтра и условие отбора записей. Для того чтобы задать условие отбора записей, можно воспользоваться Построителем выражений. Кроме того, можно сделать записи в этой форме недоступными для изменений, для чего в поле **Режим данных** (Data Mode) необходимо ввести значение **Только чтение** (Read Only).

Для ввода аргументов макрокоманды чаще всего требуется выбирать значения из списков или вводить выражения. Для ввода выражений можно воспользоваться Построителем выражений, кнопка которого находится справа от поля аргумента. Еще одна кнопка Построителя выражений находится на панели инструментов. Как и в других случаях, перед выражением нужно ставить знак равенства (=). Исключения составляют аргумент **Выражение** (Expression) макрокоманды ЗадатьЗначение (Setvalue) и аргумент **Число повторений** (Repeat Count) макрокоманды-ЗапускМакроса (RunMacro). Если ввести знак равенства перед выражением, задающим значение этих аргументов, то оно будет вычисляться дважды, что может привести к нежелательным результатам.

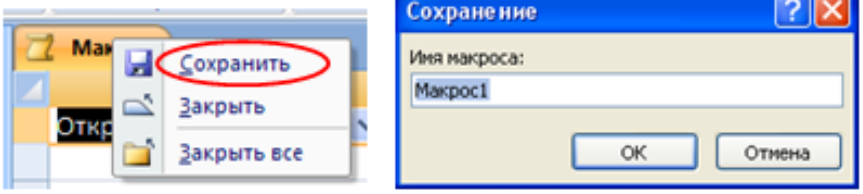
Если в качестве аргумента макрокоманды необходимо задать имя объекта базы данных, то его можно либо ввести с помощью клавиатуры, либо выбрать из раскрывающегося списка, либо указать название объекта, перетащив его из окна базы данных.

Если база данных большая, отбор записей в форме может занять некоторое время, поэтому рекомендуется вывести на экран курсор в форме песочных часов, который будет показывать, что идет обработка данных. Для

того чтобы сделать это, воспользуемся макрокомандой ПесочныеЧасы (Hourglass), в поле "Включить" (Hourglass On) панели аргументов установить значение Да (Yes). Чтобы вернуть первоначальный вид указателю мыши, в конце созданного макроса нужно ввести такую же макрокоманду ПесочныеЧасы (Hourglass), но со значением Нет (No) аргумента **Включить** (Hourglass On).

Сохранить Макрос

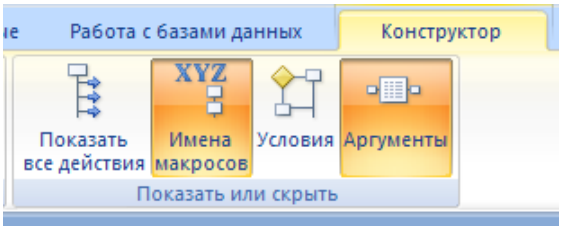
Чтобы сохранить макрос щелкаем по ярлычку вкладки с макросом и в контекстном меню выбираем Сохранить, затем следует задать имя макроса

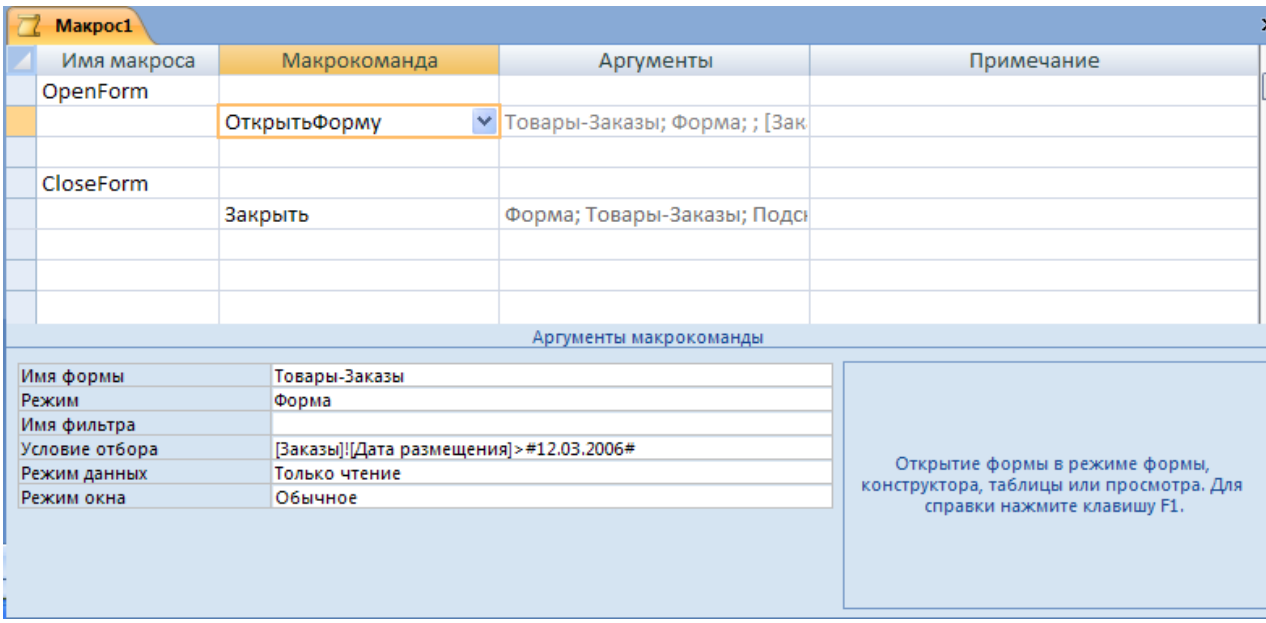


Создание групп макросов

При разработке приложения с использованием макросов количество макросов может оказаться очень большим. В одном объекте **Макрос** (Macro) можно объединить несколько макросов. Например, рекомендуется все макросы, связанные с событиями в форме или отчете, объединить в отдельный объект, соответствующий данной форме или отчету.Каждый макрос группы должен иметь свое имя, а имя объекта **Макрос** (Macro) будет являться именем группы макросов. Чтобы создать группу макросов:

1. Откройте Конструктор макросов
2. Нажмите кнопку **Имена макросов** (Macro Name) на Вкладке Конструктор\Показать или Скрыть\Имена макросов. На панели описаний в окне Конструктора появится еще один столбец — "Имя макроса" (Macro Name)





3. В этот столбец введите имя первого макроса. Остальные поля данной строки лучше оставить пустыми — это облегчит перемещение и копирование макрокоманд.

4. Начиная со следующей строки, введите все макрокоманды макроса и соответствующие аргументы для каждой макрокоманды.

5. Пропустите одну строку.

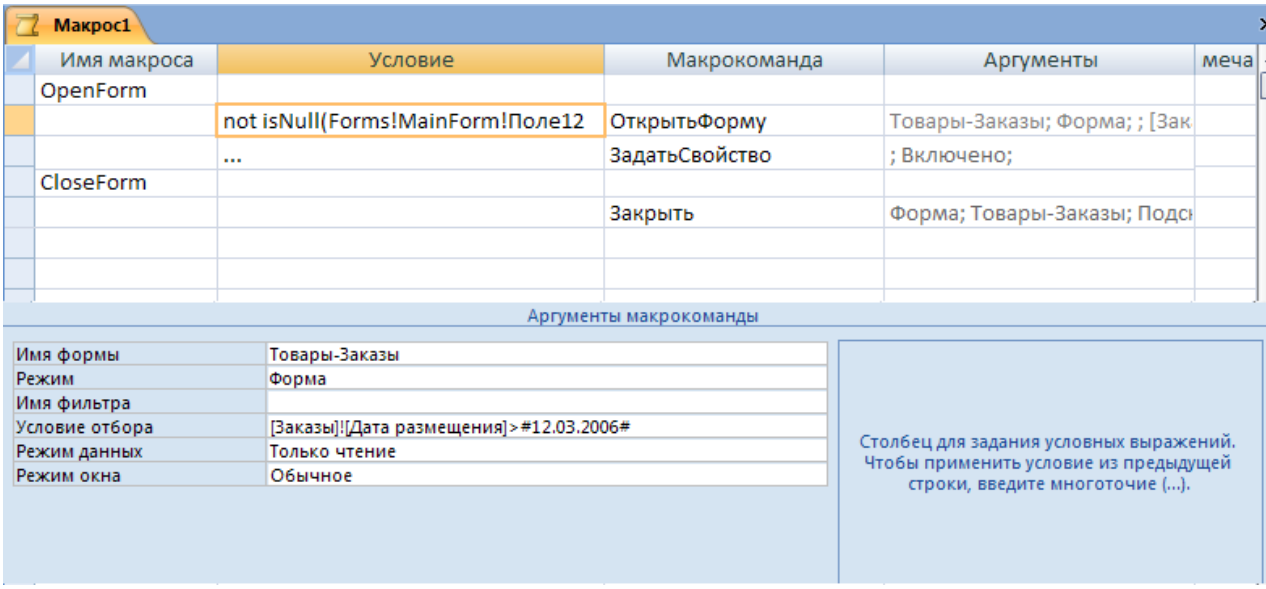
6. Повторите шаги 3—5 для каждого макроса.

Для запуска одного из макросов группы используется полное имя для ссылки на макрос. *Полное имя макроса* формируется таким образом: имяГруппы.имяМакроса.

Применение условий в макросах

Порядок выполнения макрокоманд в макросе может быть изменен. Для этого вводятся условия выполнения или пропуска макрокоманд. Эти условия задаются в виде выражений в специальном столбце, который появляется в окне Конструктора макросов, если нажата кнопка **Условия** 

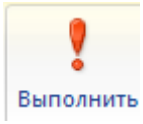
(на вкладке Конструктор\Показать или скрыть\Условия)



В окне виден столбец "Условие" (Condition). Если условие, заданное в этом столбце, истинно, выполняется макрокоманда, находящаяся в этой строке. Если условие ложно, соответствующая макрокоманда пропускается и выполняется следующая.

Если требуется при истинности условия выполнить сразу несколько макрокоманд, то для всех макрокоманд, кроме первой, в столбце "Условие" (Condition) ставится многоточие (...). Условие, которое должно проверяться, пишется в строке первой макрокоманды. Тогда, если оно истинно, выполняется весь набор макрокоманд от этого условия и до следующей макрокоманды с заданным условием, до следующего макроса или до конца макроса. Если условие ложно, пропускаются все команды, помеченные многоточием, включая макрокоманду с заданным условием. Далее выполняется макрокоманда, следующая за пропущенными, в которой содержится новое условие или поле "Условие" (Condition) не заполнено. Таким образом, в отличие от "большинства языков программирования, в макросах нет альтернативного ветвления. Для того чтобы создать макрос с двумя альтернативными ветвями, нужно сначала ввести условие и определить макрокоманды, исполняемые при выполнении этого условия. Сразу после них необходимо указать обратное условие и определить макрокоманды, исполняемые в альтернативной ветви.

Запуск макроса из окна Конструктора макросов

Этот способ применяется для тестирования только что созданного или исправленного макроса. Если макрос в окне Конструктора макросов один, то для его запуска нужно просто нажать кнопку **Выполнить** (Run) 

на панели инструментов или выбрать команду **Запуск, По шагам** (Run, Single Step) (последний способ позволяет отладить макрос). Однако, если макросов несколько, то с помощью этой кнопки или команды можно запустить только первый макрос в группе. Поэтому сначала рекомендуется создать и отладить все макросы отдельно, затем объединить их в группы.

Запуск макроса из области переходов

Дважды щелкните имя нужного макроса в области переходов

Замечание

Если в условиях или аргументах макрокоманд есть ссылки на объекты Access — формы, отчеты и пр., — они должны быть открыты перед запуском макроса, в противном случае будут выдаваться сообщения об ошибках.

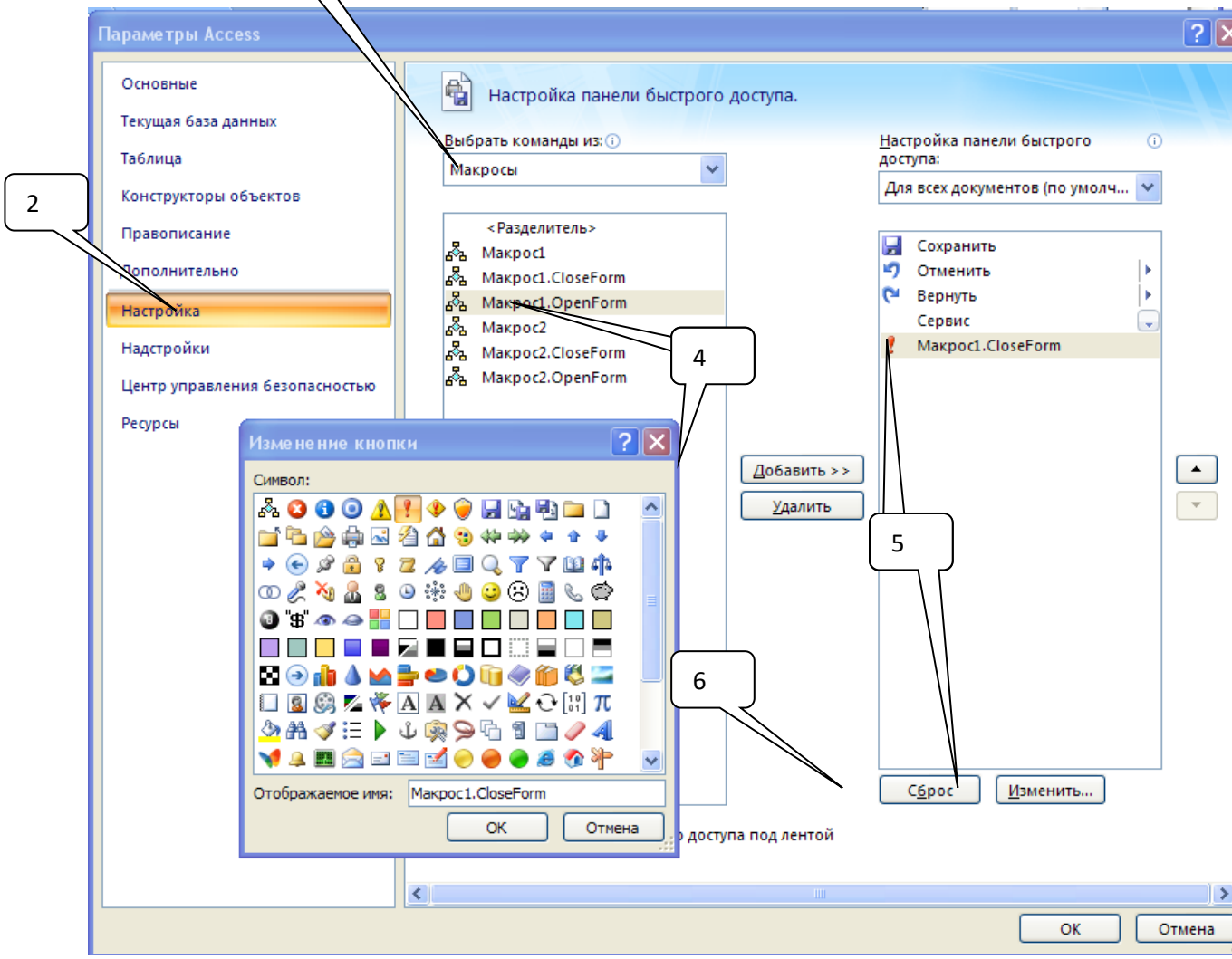
Запуск макроса с помощью кнопки на панели инструментов

Прямые способы запуска макросов являются простыми, но не самыми быстрыми. Существуют более удобные и быстрые способы. Наиболее удобный с нашей точки зрения способ запуска макроса — это создание специальной кнопки на панели инструментов. операция добавления специальной кнопки на стандартную панель инструментов чрезвычайно проста:

1. Щелкаете кнопку MS Office  , затем кнопку Параметры Access
2. В появившемся окне выбираете Настройка
3. На странице Настройка панели быстрого доступа устанавливаете поле Выбрать команды из: Макросы
4. В окне выбираете имя нужного макроса, щелкаете Добавить

5. Осталось настроить вид кнопки быстрого доступа. Для этого выделяем этот макрос в правом окне и щелкаем кнопку Изменить. Появится окно с доступными иконками.

6. Чтобы удалить макрос с панели быстрого доступа, опять выбираем его в правом окне и ш г кнопку Сброс



Запуск макроса с помощью комбинации клавиш

Для запуска макроса можно назначить комбинацию клавиш. Для этого необходимо создать специальную группу макросов — "AutoKeys". Эта группа макросов должна для каждой назначенной вами комбинации клавиш содержать макрокоманду запуска соответствующего макроса. Пример группы макросов "AutoKeys приведен на рис. Имя макроса— это запись комбинации клавиш <Ctrl>+<P>. Сам макрос состоит из одной макрокоманды ЗапускМакроса (RunMacro), которая запускает макрос "Макрос2.OpenForm"

Макрос "AutoKeys" просматривается каждый раз, когда пользователь вводит специальные комбинации клавиш, например <Ctrl>+<P>. Если введенная комбинация клавиш найдена в "AutoKeys", то запускается соответствующий макрос.

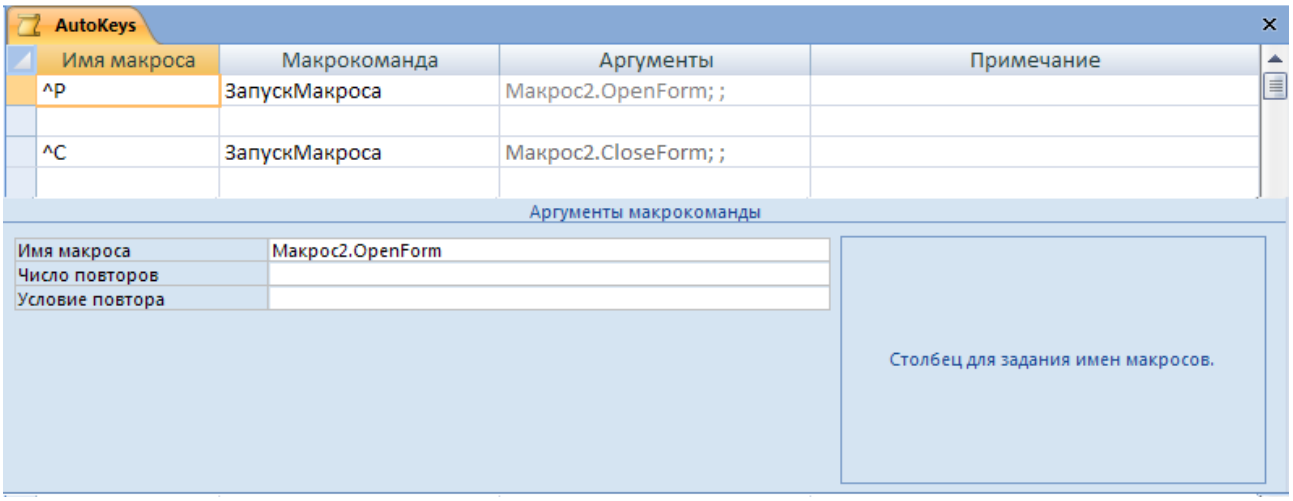


Рис. Пример макроса "AutoKeys"

Этот способ запуска макроса, безусловно, является самым быстрым, однако он имеет существенный недостаток: свободных комбинаций клавиш очень немного. И хотя комбинации, определенные в "AutoKeys", имеют более высокий приоритет, чем стандартные (например, <Ctrl>+<C> — копировать), заменять стандартное назначение комбинаций клавиш не рекомендуется. Разрешенные комбинации клавиш приведены в табл. 11.2. Комбинации с клавишей <Alt> в данном случае не применяются, т. к. используются для запуска команд меню и нажатия кнопок.

Обозначение	Комбинации клавиш
^A или ^4	<Ctrl> + буква или <Ctrl> + цифра
{F1}	Любая функциональная клавиша
^{F1}	<Ctrl> + любая функциональная клавиша
+ {F1}	<Shift> + любая функциональная клавиша
{Insert}	<Ins>
A {Insert}	<Ctrl> + <Ins>
+{Insert}	<Shift> + <Ins>
{Delete} или {Del}	<Del>
^{ Delete} или ^{De1}	<Ctrl> + <Del>
+{ Delete} или +{Del}	<Shift> + <Del>

Таблица 11.2. Разрешенные комбинации клавиш

Запуск макроса при открытии базы данных

При открытии базы данных Microsoft Access могут выполняться некоторые действия. Чаще всего это открытие специальной формы, так называемой Главной кнопочной формы, вывод специальных меню или панелей инструментов, скрытие стандартных меню и т. д. Для определения этих действий используется диалоговое окно **Параметры запуска** (Startup). Однако иногда требуется при запуске приложения выполнить более сложный набор действий, чем тот, что позволяет задать это диалоговое окно. Например, может потребоваться заранее открыть несколько форм (не показывая их на экране), чтобы потом их отображение не занимало много времени, проверить некоторые условия или запросить ввод некоторых данных. Все это можно сделать с помощью специального макроса, который называется "AutoExec". При открытии базы данных Access проверяет наличие этого макроса и, если он существует, выполняет его. При создании макроса "AutoExec" следует помнить, что Access сначала выполнит действия, определенные в окне **Параметры запуска** (Startup), а затем — макрос "AutoExec", поэтому в них не должно быть противоречивых действий.

Запуск макроса из другого макроса

Иногда требуется вызвать макрос из другого макроса. Это можно сделать с помощью макрокоманды ЗапускМакроса (RunMacro). Мы уже встречались с этой макрокомандой при описании макроса "AutoKeys". Здесь нужно отметить, что эта макрокоманда имеет три аргумента: кроме имени макроса, задается число повторов выполнения макрокоманды и условие повтора. Таким образом, эта макрокоманда позволяет организовывать циклы. Аргумент **Число повторов** (Repeat Count) задает количество вызовов макроса. Аргумент **Условие повтора** (Repeat Expression) является выражением, которое может принимать значение Истина или Ложь. Перед выполнением макрокоманды ЗапускМакроса (RunMacro) проверяется значение этого выражения. Если оно Истина, то макрос выполняется, если Ложь, то макрос не

выполняется и управление передается следующей макрокоманде. Если эти два аргумента не заданы, макрос выполняется только один раз. Если заданы оба эти аргумента, цикл вызова прекращается, когда макрос выполнился заданное число раз либо когда заданное условие окажется невыполненным и получит значение Ложь.

Замечание

Имя макроса в аргументе макрокоманды ЗапускМакроса (RunMacro) должно быть полным, т. е. должно иметь вид имяГруппы. имяМакроса, даже если вызываемый макрос находится в той же группе, что и вызывающий. Если условие повтора задано таким образом, что оно всегда истинно, то цикл окажется бесконечным. Можно прервать его с помощью комбинации клавиш <Ctrl>+<Break>. Если это не поможет, придется прервать работу Access, нажав комбинацию клавиш <Ctrl>+<Alt>+<Del>.

Назначение макроса событию

Наиболее часто макросы используются в приложении Access для обработки событий.

Событие — это любое действие, распознаваемое объектом, и можно определить реакцию объекта на событие.

События происходят в результате действий пользователя, выполнения инструкций VBA или генерируются системой. Примером событий является вывод на экран формы, отчета, ввод данных в текстовое поле, нажатие кнопки мыши или клавиши. Каждому из этих событий можно назначить макрос или процедуру VBA, которые будут автоматически выполняться в ответ на произошедшее событие. Практически все программирование в Access сводится к написанию макросов или процедур, обрабатывающих события, т. е. программируется реакция объектов на события. Существует большое количество различного рода событий, на которые реагируют объекты, причем часто возникает не одно, а целая последовательность событий. Поэтому необходимо обладать определенным умением, чтобы решить, какому событию следует назначить созданный вами макрос или процедуру VBA. Все тонкости

этого выбора см. в приложении 2. Здесь мы опишем только способ назначения макроса событию и приведем примеры обработки событий с помощью макросов.

Начнем с самого простого. Откроем форму "Клиенты" (Customers) в базе данных "Борей". В этой форме отображается информация о клиенте. Допустим, мы хотели бы видеть не только эту информацию, но и данные о купленных клиентом продуктах. Было бы хорошо создать кнопку **Заказы клиента,** при нажатии которой появлялась бы форма "Заказы" (Orders) с заказами только того клиента, который в данный момент выбран в форме "Клиенты" (Customers). Для того чтобы получить желаемое, создадим макрос, который будет выполняться, когда произойдет событие **Нажатие кнопки** (On Click) в форме "Клиенты" (Customers).

Чтобы создать макрос, определяющий описанную реакцию приложения на событие **Нажатие кнопки** (OnClick):

1. Откройте форму "Клиенты" (Customers) в режиме Конструктора.
2. Создайте кнопку в области заголовка формы. При этом кнопка мастера на панели элементов должна быть отжата, т. к. в противном случае вам будет предложено создать процедуру обработки событий, а не макрос.
3. Откройте окно свойств только что созданной кнопки, если оно еще не открыто, и раскройте вкладку **События** (Event).
4. Обратите внимание, сколько разных событий связано только с командной кнопкой. Помимо обычного нажатия, которое мы сейчас и будем использовать, в набор событий кнопки входят получение и потеря фокуса, двойной щелчок кнопкой мыши, простое перемещение указателя мыши над кнопкой и пр. Такое многообразие событий дает разработчику большие возможности по созданию удобного интерфейса пользователя. Теперь найдите в списке событие **Нажатие кнопки** (On Click) и установите курсор в соответствующую ячейку. Это поле со списком, и в данный момент оно пустое. Если открыть этот список, то первым его элементом будет **[Процедура**

обработки событий] ([Event procedure]), а далее идет перечень всех макросов, существующих в приложении (рис. 11.10).

5. Поскольку нужного нам макроса в списке нет, давайте его создадим. Для этого нажмите кнопку Построителя, находящуюся справа от поля. Появится окно **Построитель** (Choose Builder), в котором предлагается выбрать один из трех Построителей: **Выражения** (Expression Builder), **Макросы** (Macro Builder) и **Программы** (Code Builder) (рис. 11.11).

6. Выберите **Макросы** (Macro Builder) и нажмите кнопку **ОК.** Откроется"окно макросов и диалоговое окно, в которое нужно ввести имя создаваемого макроса. Введите имя Заказы клиента.

7. В макрос нужно добавить единственную макрокоманду ОткрытьФорму (OpenForm). Значения аргументов этой макрокоманды приведены в табл. 11.3.

Аргумент	**Значение**
Имя формы (Form Name)	Заказы (Orders)
Режим (View)	Форма (Form)
Условие отбора (Condition)	[КодКлиента] = [Forms] ! [Клиенты] ! [КодКлиента]
Режим данных (Data Mode)	Только чтение (Read Only)
Режим окна (Window Mode)	Обычное (Normal)

Таблица 11.3. Значения аргументов макрокоманды ОткрытьФорму (OpenForm)

Аргумент **Режим** (View) определяет режим, в котором форма должна быть открыта. Он может принимать значения: **Форма** (Form), **Конструктор** (Design), **Просмотр** (Print Preview), **Таблица** (Datasheet), **Сводная таблица** (PivotTable) и **Сводная диаграмма** (Pivot Chart). Аргумент **Условие отбора** (Condition) определяет условие для отбора записей, отображаемых в форме. Условие представляет собой выражение. В данном случае это выражение содержит ссылку на элемент управления **КодКлиента** (CustomerlD) в форме "Клиенты" (Customers), что позволяет выбрать все записи из таблицы "Заказы" (Orders), в которых код клиента равняется значению, заданному в поле "КодКлиента" (CustomerID) формы "Клиенты" (Customers). Аргумент **Режим**

данных (Data Mode) определяет способ работы с данными и может принимать одно из значений: **Добавление** (Add), **Изменение** (Edit) или **Только чтение** (Read Only). И наконец, аргумент **Режим окна** (Window Mode) определяет тип окна: **Обычное** (Normal), **Невидимое** (Hidden), **Значок** (Icon) и **Окно диалога** (Dialog).

8. Закройте окно макроса, сохранив изменения. В окне свойств кнопки в поле **Нажатие кнопки** (On Click) появится имя макроса "Заказы клиента".

9. Раскройте вкладку Макет (Format) и введите в поле **Подпись** (Caption) название кнопки: Заказы клиента. То же самое имя рекомендуется ввести в поле **Имя** (Name) на вкладке **Другие** (Other).

Теперь остается перейти в режим Формы и проверить, как макрос отрабатывает событие **Нажатие кнопки** (On Click). Если вы нигде не ошиблись, то должны увидеть на экране картинку, похожую на ту, что изображена на рис. 11.12.

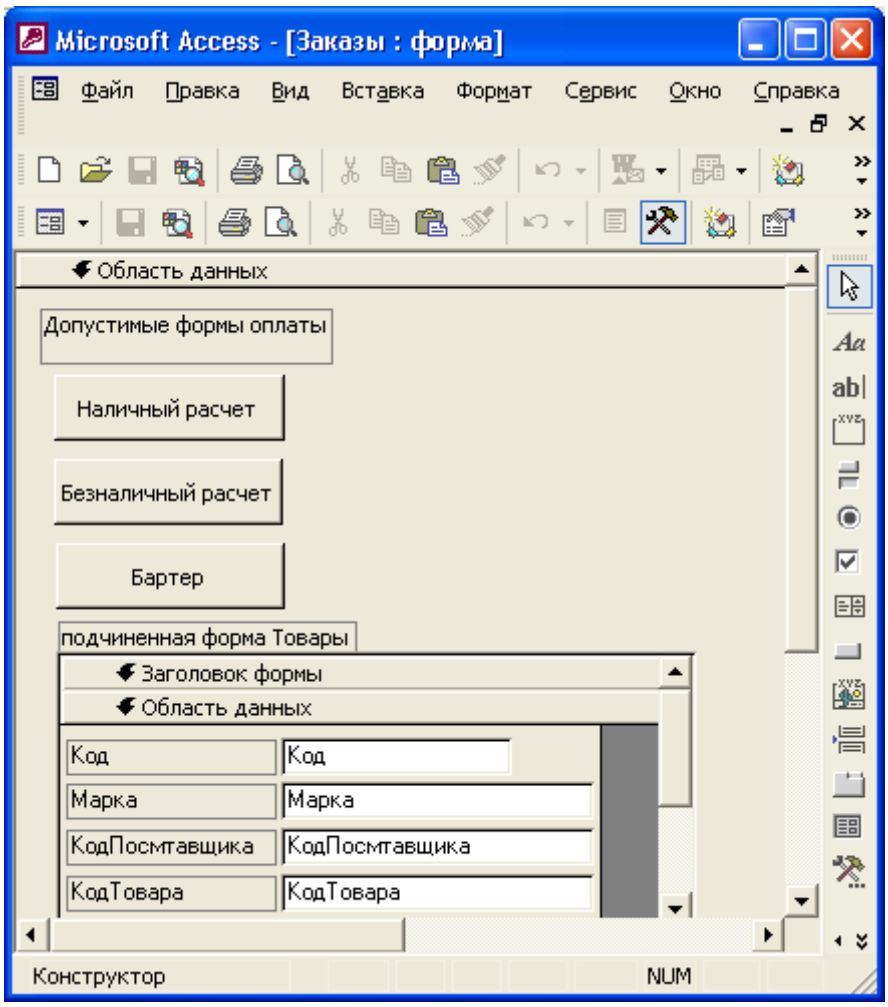


Рис. 11.12. Форма "Заказы"

Однако мы выполнили еще не все необходимые действия. Если перейти к следующей записи в форме "Клиенты" (Customers), данные в форме "Заказы" (Orders) уже не будут правильно отражать ситуацию — они не меняются. Нужно сделать так, чтобы эти данные изменялись синхронно с переходом к другим записям в форме "Клиенты" (Customers) либо чтобы это окно просто закрывалось. Рассмотрим реализацию второго варианта. Форма "Заказы" (Orders) открывается, когда мы нажимаем кнопку **Заказы клиента,** и становится активной. Требуется сделать так, чтобы она закрывалась, когда активной становится форма "Клиенты" (Customers). Для этого нужно выбрать соответствующее событие и задать процедуру или макрос его обработки. Если вы откроете окно **Свойства** (Properties) формы "Клиенты" (Customers) и раскроете вкладку **События** (Event), то среди множества событий увидите событие **Включение** (On Activate). Это как раз то событие, которое нам нужно.

Выберем это событие, но теперь мы не будем создавать новую группу макросов, а добавим новый макрос в группу макросов "Клиенты" (Customers), которая уже содержит макросы для формы "Клиенты" (Customers). Выберите эту группу макросов в раскрывающемся списке и нажмите кнопку Построителя. Откроется окно, в котором отобразятся два макроса. Добавим к ним еще один макрос. Назовем его "Закрытие" и введем макрокоманду Закрыть (Close) с соответствующими параметрами (табл. 11.4). Но перед закрытием формы необходимо проверить, открыта ли она. Для этого воспользуемся функцией isLoaded, которая возвращает значение Истина, если форма открыта, и Ложь — в противном случае. В качестве аргумента в функцию isLoaded нужно передать имя формы. Итак, в столбец **Условие** (Condition), напротив макрокоманды Закрыть (Close), введите IsLoaded("Заказы") (или IsLoaded ("Orders")).

Аргумент	**Значение**
Тип объекта (Object Type)	Форма (Forms)
Имя объекта (Object Name)	Заказы (Orders)
Сохранение (Save)	Нет (No)

Таблица 11.4. Значения аргументов макрокоманды Закрыть (Close)

Аргумент **Сохранение** (Save) позволяет определить, будет ли Access при закрытии формы отображать диалоговое окно для подтверждения сохранения измененных данных. Поскольку мы открывали форму "Заказы" (Orders) в режиме только для чтения, изменение данных запрещено, поэтому данному аргументу необходимо присвоить значение **Нет** (No). Готовый макрос показан на рис. 11.13.

Закройте окно макроса, сохранив изменения. Измените имя макроса, назначенного событию **Включение** (On Activate). Для этого выберите в списке или введите имя Клиенты.Закрыть (Customers.Close) и нажмите клавишу <Enter>. Теперь сохраните форму, перейдите в режим Формы и проверьте, как работают созданные вами макросы.

Вызов макроса из процедуры VBA

Как мы уже говорили, процедуры VBA и макросы в Access являются объектами близкими и часто взаимозаменяемыми, т. е. вместо макроса можно использовать процедуру VBA и наоборот. Существуют способы запуска макроса из процедуры VBA, запуска процедуры VBA из макроса, кроме того, Access позволяет преобразовать макрос в процедуру VBA. Для запуска процедуры VBA из макроса существует специальная макрокоманда ЗапускПрограммы(RunCode). Эта макрокоманда имеет один аргумент — имя вызываемой процедуры, хотя вызываться может только функция, а не подпрограмма.

Для запуска макроса из процедуры VBA применяется специальный метод ЗапускМакроса (RunMacro) объекта DoCmd, например:

DoCmd.RunMacro "Макрос2.OpenForm".

Объект DoCmd используется в процедуре VBA для выполнения макрокоманд Access. При этом английское имя нужной макрокоманды должно быть указано в качестве Метода объекта DoCmd, например строка процедуры

DoCmd.OpenForm "Клиенты"

позволяет открыть форму "Клиенты". Здесь "Клиенты"— аргумент макрокоманды. Аргументы перечисляются в предложении DoCmd через запятую. Таким способом может быть выполнено большинство макрокоманд.

Применение макросов

Макросы могут применяться для решения самых разнообразных задач. В первом разделе данной главы мы привели таблицу макрокоманд, сгруппированных по категориям. Теперь рассмотрим применение макрокоманд некоторых категорий.

Работа с данными в формах и отчетах

В эту категорию входит набор макрокоманд, обеспечивающих отбор данных, перемещение по данным и обновление данных в формах. Прежде чем перейти к примерам использования макросов, покажем, как применять ссылки

на формы, отчеты и элементы управления в аргументах и условиях макрокоманд, поскольку такие ссылки используются очень часто.

Ссылки на формы, отчеты и их свойства

Чтобы задать ссылку на форму или отчет, нужно сначала определить, в какое семейство (collection) входит объект, на который создается ссылка. Все открытые формы входят в семейство Формы (Forms), а открытые отчеты — в семейство Отчеты (Reports). Полная ссылка на форму или отчет должна состоять из двух частей: имяСемейства!имяОбъекта. Причем если имя объекта содержит пробелы или специальные символы, то его нужно заключить в квадратные скобки. Если пробелы в имени не используются, скобки можно не ставить. Таким образом, ссылка на форму будет выглядеть так: Forms![Заказы клиента] ИЛИ Forms!Клиенты Для отчета ссылки выглядят аналогично:

Reports! [Отчет о продажах] ИЛИ Reports!Прайс-лист Ссылка на свойство формы или отчета состоит из трех частей: имяСемейства!имяОбъекта.имяСвойства Например:

Forms!Клиенты.Visible или Reports![Продажи за период].MenuBar Свойство **Вывод на экран** (Visible) определяет, будет ли форма видна на экрану или спрятана, а свойство **Меню** (MenuBar) позволяет связать с отчетом или формой специальное меню.

Ссылки на элементы управления форм, отчетов и их свойства

Чтобы создать ссылку на элемент управления или его свойство, необходимо указать его имя. Если имя содержит пробелы, оно заключается в квадратные скобки. Ссылка на элемент управления в форме или отчете состоит из трех частей:

имяСемейства!имяОбъекта!имяЭлемента

Например, ссылка на элемент управления **Номер заказа** в форме "Заказы клиента" выглядит так:

Forms![Заказы клиента]![Номер заказа]

Ссылка на элемент управления **Сумма** в отчете "Продажи за период" выглядит так:

Reports![Продажи за период]![Сумма]

Ссылка на свойство элемента управления состоит из четырех частей:

имяСемейства!имяОбъекта!имяЭлемента.имяСвойства

Например:

Forms![Заказы клиента]![Номер заказа].Enabled

Свойство **Доступ** (Enabled) позволяет запретить или разрешить доступ к элементу управления.

Объект может иметь свойство, используемое по умолчанию. Это свойство применяется в том случае, когда имя свойства в ссылке явно не указано. Например, у элементов управления по умолчанию используется свойство **Значение** (Value), поэтому ссылка Forms! Товары! Цена позволяет получить доступ к значению, отображенному в текстовом поле "Цена".

Ссылки на подчиненные формы и отчеты

На подчиненную форму или отчет можно ссылаться так же, как и на любой другой элемент управления, т. к. подчиненная форма и подчиненный отчет являются одним из типов элементов управления. Например:

Forms!Заказы!ПодформаТовары

Здесь ПодформаТовары — это имя элемента управления в форме "Заказы", который представляет собой подчиненную форму.

А вот ссылка на элемент управления в подчиненной форме или отчете имеет особую структуру: после имени элемента управления, который является подчиненной формой, нужно сначала указать специальное свойство: Form — для форм или Report — для отчетов, а затем имя элемента управления, на который выполняется ссылка:

Фильтрация записей в формах, отчетах, таблицах

Перейдем к примерам применения макрокоманд Access. Для отбора записей в формах, отчетах, таблицах используется макрокоманда Приме нить Фильтр (ApplyFilter). Задать фильтр можно двумя способами: либо указав имя заранее созданного фильтра в аргументе **Имя фильтра** (Filter Name), либо непосредственно задав условие выборки в аргументе **Условие отбора** (Where

Condition). Если фильтр нужно применить сразу при открытии формы, тогда с событием **Открытие** (On Open) формы необходимо связать макрос, содержащий макрокоманду ПрименитьФильтр (ApplyFilter). Если нужно менять набор отображаемых записей в открытой форме динамически, поступают следующим образом:

- создают в этой форме набор полей, в которых можно задать условия отбора;
- создают кнопку **Применить фильтр,** с которой связывают макрос, содержащий макрокоманду ПрименитьФильтр (ApplyFilter). В качестве значения аргумента **Условие отбора** (Where Condition) для этой макрокоманды указывают выражение, содержащее ссылки на эти поля.

Пример такого решения приведен на рис. 11.15. На этом рисунке представлена форма "Товары" (Products), позволяющая просматривать товары с выборкой по различным критериям.

Эта форма построена на основе таблицы "Товары" (Products) с помощью Мастера автоматической генерации ленточных форм. Затем в режиме Конструктора форм к ней добавлены поля для определения критериев выбора и кнопки для применения и отмены фильтра. Соответствующие макросы, связанные с событием **Нажатие, кнопки** (On Click), приведены на рис. 11.16. Для отмены фильтра используется макрокоманда ПоказатьВсеЗаписи(ShowAllRecords).

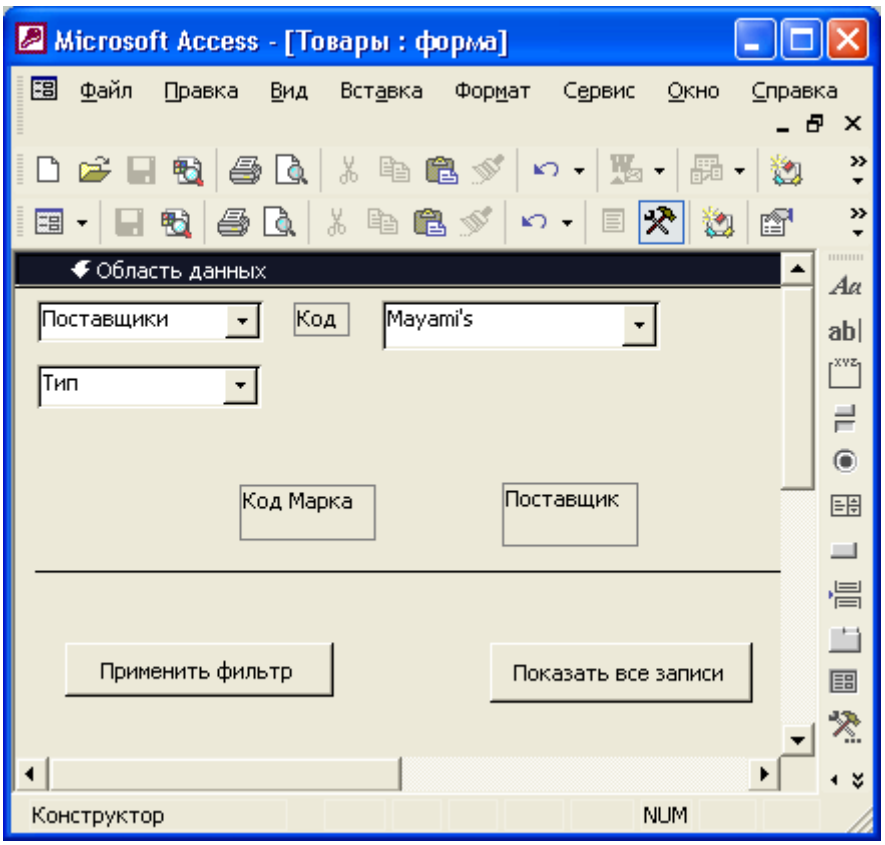


Рис. 11.15. Форма для отбора записей по заданным критериям

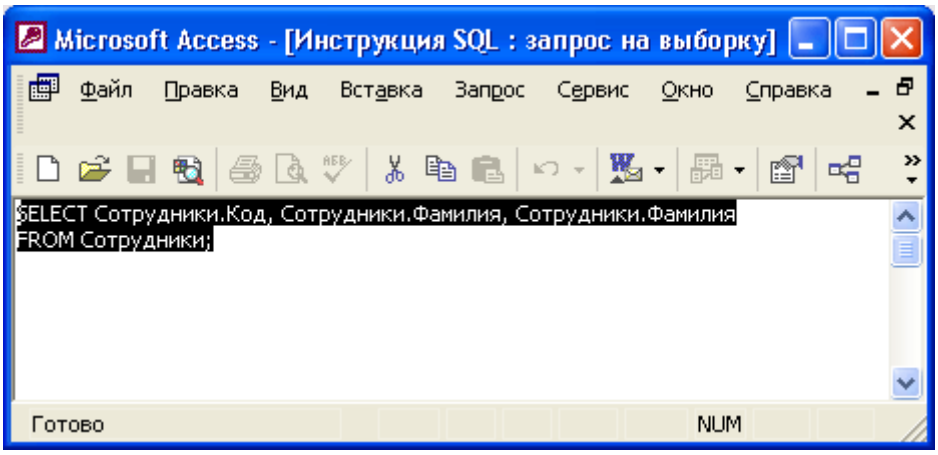


Рис. 11.16. Макросы для установки и отмены фильтра в форме "Товары"

На рис. 11.16 представлено два макроса: УстановитьФильтр и Отменить Фильтр. В диалоговом окне **Область ввода** (Zoom) отображено значение аргумента **Условие отбора** (Where Condition) макрокоманды ПрименитьФильтр (ApplyFilter). Обратите внимание, условие получилось довольно сложным, а длина поля **Условие отбора** (Where Condition) ограничена 255 символами. Поэтому, если бы мы включили в условия отбора еще одно поле, например "Поставки прекращены", нам пришлось бы отдельно

создать специальный фильтр и задать его имя в аргументе **Имя фильтра** (Filter Name).

В макросе "ОтменитьФильтр", кроме макрокоманды ПоказатьВсеЗаписи(ShowAllRecords), нужны еще две макрокоманды, чтобы очистить поля " ВыборПоставщика" и "ВыборТипа", т. е. присвоить им значение **Пусто** (Null). Это делается с помощью макрокоманды ЗадатьЗначение(SetValue).

Перемещение по данным

Эта группа макрокоманд связана с перемещением по записям и элементам управления. В качестве примера применения макрокоманд этой группы рассмотрим ситуацию, когда пользователь работает с формой "Клиенты" и хочет найти заказ текущего клиента, номер которого ему известен. Давайте вернемся к примеру, приведенному в *разд. "Назначение макроса событию"*. Тогда мы создали.в форме "Клиенты" кнопку **Заказы клиента** для отображения заказов выбранного клиента. Попробуем усовершенствовать эти формы. Если номер заказа известен пользователю, он может ввести его в текстовое поле **Поиск заказа,** которое можно добавить в форму "Клиенты" (Customers). Тогда в открываемой форме "Заказы" (Orders) следует сразу показать запись, содержащую соответствующий счет. Добавим поле "Поиск заказа" в заголовок формы и сделаем так, чтобы при открытии формы "Заказы" (Orders) проверялось значение этого поля и, если оно не пустое, производился бы поиск счета с указанным номером и выполнялся переход на соответствующую запись. В противном случае текущей становилась бы первая запись из отфильтрованного набора записей. На рис. 11.17 показана форма "Клиенты" (Customers) с новым полем и макрос "Поиск заказа", который связан с событием **Загрузка** (On Load) формы "Заказы" (Orders).

Поскольку добавленный макрос обработки события ссылается на элемент управления в форме "Клиенты" (Customers), он должен выполняться только в том случае, если эта форма открыта, и, кроме того, поиск должен

производиться только тогда, когда поле "Поиск заказа" не пусто. Исходя из этого формируются условия макроса. Прежде чем производить поиск записи по образцу в одном из полей, необходимо активизировать это поле, для чего используется макрокоманда КЭлементуУправления(GoToControl) . С ее помощью устанавливается фокус на поле "КодЗаказа" (Orderld). Но это поле в форме "Заказы" (Orders) недоступно, поэтому сначала нужно изменить значение свойства **Доступ** (Enabled) этого поля. В противном случае выполнение макрокоманды КЭлементуУправления (GoToControl) приведет к возникновению ошибки.

Макрокоманда КЭлементуУправления (GoToControl) используется, как правило", если требуется изменить стандартный порядок перехода между полями в форме. Обычно переход между полями выполняется с помощью клавиши <Tab>, однако иногда требуется пропустить ряд полей и переместить фокус к определенному элементу управления. Это может зависеть от каких-либо условий, например от значения поля. В этом случае на событие **После обновления** (After Update) для данного прля назначается макрос, который перемещает фокус на нужный элемент управления в форме. В качестве аргумента для макрокоманды необходимо указать короткое имя элемента управления. В случае излишне длинного имени переход к элементу управления не произойдет и отобразится сообщение об ошибке.

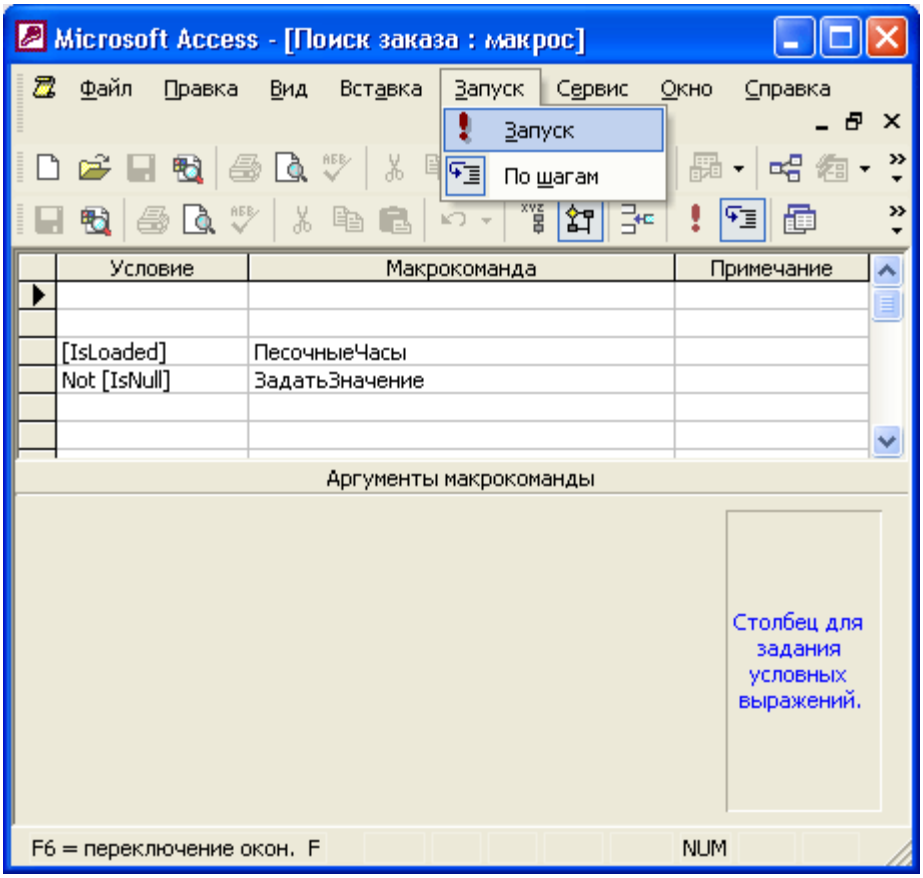


Рис. 11.17. Пример макроса для перемещения по данным

Обновление данных в формах и элементах управления

Последняя группа макрокоманд в рассматриваемой категории связана с обновлением данных в активных формах, таблицах и запросах. Если несколько человек в сети одновременно изменяют данные, то формы и таблицы у конкретного пользователя могут не отражать актуальных данных. Для того чтобы отображаемые данные соответствовали текущему состоянию базы данных, их необходимо обновить с помощью команды **Записи, Обновить** (Record, Refresh) *(см. разд.Работа с записями гл. 2).*

Похожая ситуация возникает даже в однопользовательском режиме, если в форме используется поле со списком, источником данных для которого является таблица или запрос. Если в исходную таблицу были добавлены записи, то в поле со списком они автоматически не появятся — нужно повторно выполнить запрос. Кроме полей со списком, к элементам управления, требующим обновления отображаемых данных, относятся также списки и элементы управления подчиненной формы, объекты OLE и

вычисляемые элементы управления, содержащие статистические функции по подмножеству записей, такие как DLookUp () или DSum ().

Для того чтобы выполнить обновление записей в формах, таблицах или элементах управления, используются макрокоманды Обновление(Requery), ПоказатьВсеЗаписи(ShowAllRecords) И ОбновитьОбъект(RepaintObject).

Макрокоманда Обновление (Requery) обновляет данные в объекте базы данных путем повторного просмотра источника данных. Макрокоманда имеет один аргумент, содержащий имя объекта, который следует обновить. Если обновляется активный объект, например форма, то поле аргумента следует оставить пустым. При этом макрокоманда будет повторно выполнять запрос, указанный в свойстве **Источник данных** (RecordSource) этой формы.

Рассмотрим пример использования макроса для обновления данных. В форме "Клиенты" (Customers) есть поле со списком "Страна" (Country). Источником данных для этого поля является запрос, который выбирает значения из поля "Страна" (Country) таблицы "Клиенты" (Customers):

SELECT DISTINCT Клиенты.Страна FROM Клиенты;

Если при вводе клиента в таблицу добавляется новое название страны, то в списке эта страна не появится, поскольку запрос будет выполнен повторно только при следующем открытии формы. Чтобы провести обновление списка стран раньше, следует назначить событию **После обновления** (After Update) формы макрос Клиенты.Обновление списка стран, который состоит из одной макрокоманды Обновление (Requery) со значением аргумента "Страна" (Country) (рис. 11.18).

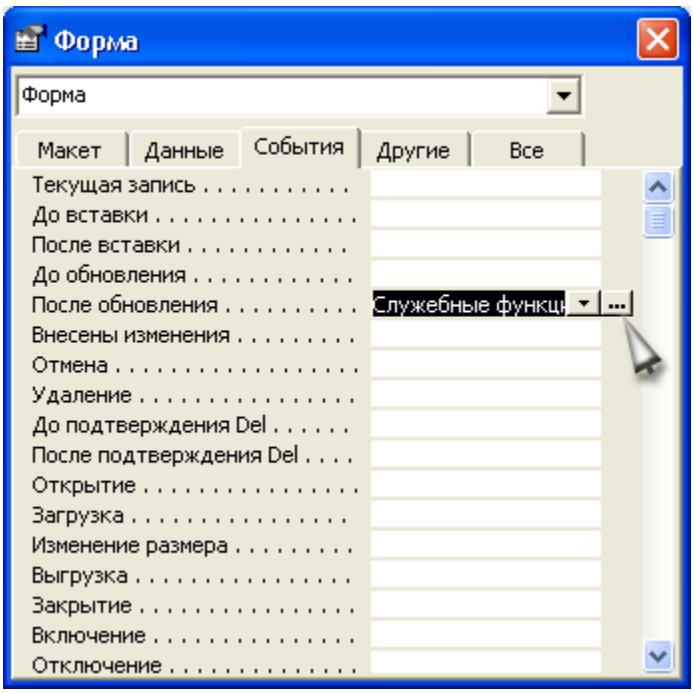


Рис. 11.18. Назначение макроса событию формы **После обновления**

Замечание

Макрокоманда Обновление (Requery) обновляет только один объект, поэтому если в форме существуют элементы управления, требующие обновления отображаемых данных, и в макросе, назначенном этой форме, используется макрокоманда Обновление (Requery) с пустым значением аргумента, то обновляться будут только записи в форме. Для каждого такого элемента управления нужно использовать отдельную макрокоманду. Если эта макрокоманда в качестве аргумента содержит имя элемента управления, то при ее выполнении обновляется только данный элемент. Записи в самой форме не обновляются.

С помощью макрокоманды Обновление (Requery) можно обновлять данные и в неактивной форме, точнее, не в той форме, которой назначен макрос. Однако в этом случае сначала необходимо выполнить макрокоманду ВыделитьОбъект(SelectObject), которая перенесет фокус на нужную форму, чтобы ее активизировать (потом фокус можно вернуть обратно).

Макрокоманда ПоказатьВсеЗаписи (ShowAllRecords), как уже отмечалось, отменяет действие фильтра и повторно просматривает источник записей. Ее часто используют для обновления данных в подчиненной форме.

Макрокоманда ОбновитьОбъект (RepaintObject) применяется только к объекту базы данных (к таблице, запросу, форме, отчету, странице, макросу и модулю) и не применяется к элементу управления. Она выполняет немедленное обновление указанного открытого объекта (если имя объекта не задано, обновляется активный объект), хотя при этом не производится повторное выполнение запроса к источнику данных. Обновление объекта не влияет на. отображение новых и удаленных записей, как это происходит при выполнении макрокоманды Обновление (Requery). Обычно макрокоманду ОбновитьОбъект (RepaintObject) применяют для отображения результатов изменения данных с помощью макрокоманд ЗадатьЗначение (SetValue), а также для повторного вычисления значений выражений в вычисляемых элементах управления.

Работа с объектами

Это наиболее обширная категория макрокоманд, часть из которых уже была описана в предыдущих разделах.

Рассмотрим сначала, как используется макрокоманда ЗадатьЗначение (SetValue), которая позволяет устанавливать значения свойств элементов управления в формах и отчетах. Мы уже использовали эту макрокоманду в предыдущих примерах.

Установка свойств элементов управления позволяет динамически, в зависимости от условий, делать эти элементы недоступными или невидимыми. Вот несколько примеров.

- Можно создать одну форму, в которой в зависимости от определенных условий, видимыми будут разные поля. Это позволяет использовать одну форму в нескольких случаях и не создавать для каждой ситуации дополнительную форму. Когда форм в приложении очень много, такая возможность становится очень полезной. Чтобы сделать элемент управления невидимым, следует задать значение **Ложь** (False) для свойства **Вывод на экран** (Visible) этого элемента. Чтобы показать элемент управления на экране, задайте для этого свойства значение **Истина** (True).

- В зависимости от текущего состояния работы с данными можно изменять доступность кнопок, предназначенных для выполнения определенных действий, или других элементов управления в форме. Чтобы сделать элемент управления недоступным, задайте значение **Ложь** (False) его свойству Доступ (Enabled). Чтобы элемент управления стал доступным, задайте значение **Истина** (True) для этого свойства.
- Можно запретить пользователю изменять данные в форме. Для этого свойству **Доступ** (Enabled) соответствующих полей следует присвоить значение **Ложь** (False), а свойству **Блокировка записей** (Locked) — значение **Истина** (True). Если необходимо запретить изменение во всех полях, установите значение **Ложь** (False) для следующих свойств формы: **Разрешить изменения** (Allow Edits), **Разрешить добавление** (Allow Additions), **Разрешить удаление** (Allow Delitions). Эти свойства можно изменять динамически, т. е. в процессе работы пользователя с формой. Например, можно разрешить или запретить редактирование данных после проверки прав пользователя.
- С помощью макрокоманды ЗадатьЗначение (SetValue) можно динамически изменять значение свойства **Источник записей** (RecordSource) для формы, что позволяет управлять содержанием отображаемых данных.

Существуют макрокоманды для открытия "и закрытия объектов Access: ОткрытьФорму(OpenForm), ОткрытьЗапрос(OpenQuery), ОткрытьОтчет(OpenReport) и т. д. Для открытия объекта каждого типа применяется отдельная макрокоманда, а для закрытия объекта используется общая для объектов всех типов макрокоманда Закрыть (Close). Тип объекта, к которому следует применить эту макрокоманду, указывается в качестве одного из ее аргументов. В предыдущих примерах уже использовались макрокоманды ОткрытьФорму(OpenForm) и Закрыть(Close) для активного объекта.

Макрокоманда ОткрытьЗапрос (OpenQuery) позволяет выполнить любой тип запроса, в том числе запрос на изменение данных. Если в качестве

аргумента этой макрокоманды задается имя запроса на выборку или перекрестного запроса, то результатом выполнения макрокоманды будет вывод на экран выбранных записей. Если же аргумент — имя запроса на изменение данных, то макрокоманда выполнит запрос, изменяя соответствующим образом данные в таблицах.

При выполнении запроса, изменяющего данные, на экране будут отображаться предупреждающие сообщения. Чтобы отключить вывод этих сообщений, используйте макрокоманду УстановитьСообщения (SetWarnings) со значением аргумента **Нет** (No). Только не забудьте после выполнения запроса снова включить вывод системных сообщений, применив ту же макрокоманду, но с аргументом Да (Yes). Иначе в Access не будут отображаться никакие системные сообщения, что может привести к выполнению нежелательных действий в приложении.

Макрокоманда ОткрытьПредставление (OpenView) аналогична макрокоманде ОткрытьЗапрос (OpenQuery), только применяется она в проектах Access 2000 и предназначена для работы с данными, хранящимися на сервере.

Макрокоманда ОткрытьСохраненнуюПроцедуру (OpenStoreProcedure) позволяет выполнить или открыть в режиме редактирования хранимую процедуру сервера.

В данном разделе описаны далеко не все макрокоманды. Для получения полной информации используйте справочную систему Access. Чтобы быстро получить справку:

1. Выберите команду **Справка, Справка по Micrisift** Access (Help, Microsoft Access Help).

2. Раскройте вкладку **Мастер ответов** (Answer Wizard) и введите в поле **Выберите действие** (What would you like to do?) имя макрокоманды. Нажмите кнопку **Найти** (Search).

Макрокоманды Access

Вид отображения		
Скрыть/показать; свернуть/развернуть	Восстановить	Вернуть развернутое или свернутое окно в исходное состояние
	ВыводНаЭкран	Скрыть или показать результаты текущей работы Макроса
	Развернуть	Позволяет видеть в активном окне максимально возможную часть объекта
	Свернуть	Уменьшаетя размер активного окна вплоть до маленькой строки заголовка в нижней части окна Access
	СдвигРазмер	Перемещение или изменение размеров активного окна
	Закрыть	Закрыть окно
Перемещение фокуса	ВыделитьОбъект	Выделить указанный объект базы данных.
	КЭлементуУправления	Перемещение фокуса на элемент управления в текущей записи открытой формы или запроса в режиме таблицы
Вид меню	ДобавитьМеню	Для форм, отчетов, БД в целом: Создать Пользовательские меню в группе Команды на вкладке Надстройки, Пользовательские контекстные меню, Глобальное контекстное меню
	ЗадатьКомандуМеню	Задать состояние пунктов меню (включены или отключены, выбраны или нет) в пользовательских и глобальных меню на вкладке Надстройки.
	ПанельИнструментов	Отобразить или скрыть группу команд на вкладке Надстройки.
Вид области переходов	ЗадатьОтображаемыеКатегории	Указать, какие из категорий будут отображаться в разделе Переход в категорию в области переходов
	ЗафиксироватьОбластьПереходов	Не позволяет допустить удаление объектов БД, отображенных в области переходов
	ПерейтиК	Можно контролировать отображение объектов базы данных в области переходов. Например, можно задать категории объектов базы данных или отфильтровать объекты так, чтобы отображались только определенные.
Формат вывода	ВывестиВФормате	Вывести данные из указанного объекта Access (таблицы, формы, отчета, модуля или страницы доступа к данным) в различных выходных форматах.

Работа с данными в формах и отчетах		
Отбор данных	ПрименитьФильтр	Применить фильтр, запрос или предложение WHERE инструкции SQL к таблице, форме или отчету
Перемещение по данным	НаЗапись	Делает указанную запись текущей в открытой форме, таблице или результирующем наборе запроса
	НайтиЗапись	Найти первый экземпляр данных, удовлетворяющих заданным условиям
	НаСтраницу	Перемещение фокуса в активной форме на первый элемент указанной страницы
	ПоискЗаписи	Найти заданную запись в таблице, запросе, форме или отчете.
	СледующаяЗапись	Нахождение следующей записи, удовлетворяющей условию, определенному предыдущей макрокомандой НайтиЗапись
Обновление данных или экрана	ОбновитьОбъект	Завершить все отложенные обновления экрана для указанного объекта базы данных или активного объекта базы данных,
	Обновление	Обновить данные указанного элемента управления активного объекта путем повторного запроса к источнику данных этого элемента управления
	ПоказатьВсеЗаписи	Удалить все фильтры, которые применялись к активной таблице, результирующему набору записей запроса илиы форме
Изменить значение, свойство	ЗадатьВременнуюПеременную	Создать временную переменную и присвоить ей значение. Впоследствии переменная может использоваться в качестве условия или как аргумент последующих макрокоманд, а также в других макросах, процедурах обработки событий, формах или отчетах.
	ЗадатьЗначение	Задать значение для поля, элемента управления или свойства в форме, форме в режиме таблицы или в отчете
	ЗадатьСвойство	Задать свойства элемента управления формы или отчета.
	УдалитьВременнуюПеременную	Удалить одиночную временную переменную, созданную с помощью команды ЗадатьВремПеременную
	УдалитьВсеВременные Переменные	Удалить все временные переменные, созданные с помощью команды ЗадатьВремПеременную.

Выполнение		
Выполнение команды	ВыполнитьКоманду	Выполнить встроенную команду Microsoft Office Access 2007.
Выполнение макроса, процедуры или запроса	Выполнить Макрос	Выполнить макрос.
	ЗапускМакроса	Запустить другой макрос из данного макроса
	ЗапускПрограммы	Вызвать процедуру типа Function Visual Basic для приложений (VBA).
	ЗапускЗапросаSQL	Выполнить в запрос на изменение с использованием соответствующей инструкции SQL, а также управляющий запрос.
	ОткрытьЗапрос	Открыть запрос на выборку или перекрестный запрос в режиме таблицы, режиме конструктора или режиме предварительного просмотра.
	ОткрытьМодуль	Открыть указанный модуль Visual Basic для приложений (VBA) на заданной процедуре.
	ОткрытьОтчет	Открыть отчет в режиме конструктора или режиме предварительного просмотра, а также вывести отчет на печать. Можно ограничить записи, которые будут печататься в отчете
	ОткрытьПредставление	Открыть представление в режиме таблицы, в режиме конструктора или в режиме предварительного просмотра.
	ОткрытьСохраненнуюПроцедуру	Открыть хранимую процедуру в режиме таблицы, режиме конструктора или в режиме предварительного просмотра. Вызов макрокоманды в режиме таблицы ведет к выполнению указанной хранимой процедуры
	ОткрытьСтраницуДоступаКДанным	Открыть страницы доступа к данным в режиме страницы или в режиме конструктора.
	ОткрытьСхему	Открыть схемы базы данных в режиме конструктора.
	ОткрытьТаблицу	Открыть таблицу в режиме таблицы, режиме конструктора или в режиме предварительного просмотра.
	ОткрытьФорму	Открыть формы в режиме формы, в режиме конструктора, в режиме предварительного просмотра или в режиме таблицы. Она позволяет выбирать режим ввода данных и режим окна для формы, а также ограничивать количество записей, отображаемых в форме.

	ОткрытьФункцию	Открыть определяемую пользователем функцию в режиме таблицы, в режиме конструктора встроенной функции, в режиме текстового редактора SQL или в режиме предварительного просмотра.
	ЗапускПриложения	Запустить приложение Microsoft Windows или MS-DOS
Прерывание выполнения	ОстановитьВсеМакросы	Остановить все работающие в данный момент макросы
	ОстановитьМакрос	Остановить работающий в данный момент макрос.
	ОтменитьСобытие	Отмена события, в результате которого запущен макрос, содержащий данную макрокоманду
Выход из Microsoft Access	Выход	Выйти из приложения Microsoft Office Access 2007.
Импорт/экспорт		
Передача объектов Microsoft Access в другие приложения	ВыполнитьСохраненныйИмпортЭкспорт	Выполнить операцию импорта или экспорта по сохраненной спецификации, созданной с помощью мастера импорта или мастера экспорта.
	ОтправитьОбъект	Включить Microsoft Office Access 2007 объект в режиме таблицы, форму, отчет, модуль или страницу доступа к данным в сообщение электронной почты
	ПереносБазыДанныхSQL	Перенести базу данных Microsoft SQL Server версии 7.0 или более поздней в другую такую базу данных.
	КопироватьОбъект	Копирует указанный объект БД в другую БД Access или в ту же БД под новым именем
Преобразование формата данных	ПреобразоватьБазуДанных	Импортировать или экспортировать данные между текущей базой данных Access (MDB или ACCDB) или проектом Access (ADP) и другой базой данных.Или связать таблицу из другой базы данных с текущей базой данных Access.
	ПреобразоватьСписокSharePoint	Импортировать и связать данные с узла Службы Microsoft Windows SharePoint Services 3.0.
	ПреобразоватьТекст	Импортировать или экспортировать текст между текущей базой данных Access 2007 (MDB или ACCDB) или проектом Access (ADP) и текстовым файлом. Или связать данные в текстовом файле с текущей базой данных Access.
	ПреобразоватьЭлектронуюТаблицу	Импортировать или экспортировать данные между текущей базой данных Access (MDB или ACCDB) или проектом Access (ADP) и файлом электронной

		таблицы. Или связать данные электронной таблицы Excel с текущей базой данных Access
Работа с объектами, сообщениями, макросами		
Копирование, переименование и сохранение объекта	КопироватьФайлБазыДанных	Создание копии подключенной к проекту Access текущей базы данных Microsoft SQL Server 7.0 или более поздней версии.
	Переименовать	Переименовать указанный объект базы данных.
	УдалитьОбъект	Удалить указанный объект БД
	Сохранить	Сохранить указанный или активный объект
Работа с сообщениями	Сообщение	Отображение окна сообщения, содержащего предупреждение или сведения
	УстановитьСообщения	Включение и отключение системных сообщений
Работа с макросами	ПриОшибке	Можно указать действие, выполняемое при возникновении ошибки в макросе.
	УстранитьОшибкуМакроса	Очистка данных об ошибке, хранящихся в объекте MacroError
	Шаг	Приостановить исполнение макроса и открыть диалоговое окна Пошаговое исполнение макроса.
Закрыть базу	ЗакрытьБазуДанных	Закрыть текущую БДСоздание копии подключенной к Access текущей БД MSSQL Server
Прочие	КомандыКлавиатуры	Применяется для передачи нажатий клавиш непосредственно в Microsoft Office Access 2007 или в активное приложение Windows.
	ПесочныеЧасы	Представление курсора мыши в виде песочных часов на время работы макроса
	Печать	Вывода на печать активного объекта открытой базы данных: печать таблиц, отчетов, форм, страниц доступа к данным и модулей.
	Сигнал	Звуковой сигнал через динамик компьютера