Министерство труда и социальной защиты
Республики Беларусь

Государственное учреждение образования
«Республиканский институт повышения квалификации
и переподготовки работников
Министерства труда и социальной защиты
Республики Беларусь»

И. В. ВИНОГРАДОВА, С. С. ЩУПАК

РАБОТА В MICROSOFT ACCESS 2010

Пособие для слушателей переподготовки

Минск
«Колорград»
2017

УДК 004.92(075.9)
ББК 32.973.26-018.2я75
В49

Рекомендовано к изданию Советом института Государственного учреждения образования «Республиканский институт повышения квалификации и переподготовки работников Министерства труда и социальной защиты Республики Беларусь», протокол от 14.09.2017, №4.

Рецензенты:
заведующий кафедрой информационных систем и технологий Белорусского государственного технологического университета, кандидат технических наук, доцент *В. В. Смелов;*
начальник отдела информационного и технического обеспечения образовательного процесса факультета транспортных коммуникаций Белорусского национального технического университета, *Е. В. Баяревич*

Виноградова, И. В.

В49 Работа в Microsoft Access 2010 : пособие для слушателей переподготовки / И. В. Виноградова, С. С. Щупак. – Минск : Колорград, 2017. – 108 с.

ISBN 978-985-7189-85-4.

В пособии рассматриваются технологии проектирования баз данных, нормализации баз, выборки данных при помощи запросов, построения форм и отчетов. Рекомендовано использовать при проведении занятий в группах повышения квалификации и переподготовки по тематике «Организация и проектирование баз данных», «Средства визуального программирования приложений» специальности «Программное обеспечение информационных систем»

УДК 004.92(075.9)
ББК 32.81я75

ISBN 978-985-7189-85-4

Содержание

Введение **5**
Основные понятия баз данных **7**
Реляционные базы данных 9
Таблицы базы данных 10
Ключи и индексы 10
Разработка базы данных **11**
Постановка задачи 12
Нормализация данных 14
Первая нормальная форма 16
Вторая нормальная форма 19
Третья нормальная форма 20
Связи между таблицами 21
Создание проекта MS Access **22**
Создание новой базы данных **23**
Создание таблиц 24
Создание первичных ключей и индексов 28
Создание обычного индекса по полю таблицы 28
Создание простого первичного ключа 29
Создание составного первичного ключа 29
Контроль правильности ввода данных **30**
Добавление условия на значение поля 30
Добавление условия на значение записи 30
Создание связей между таблицами **32**
Создание связи «один-ко-многим» 32
Создание связи «один-к-одному» 35
Устранение проблем, возникающих при создании ключей 36
Устранение связи «многие-ко-многим» 36
Создание форм и отчетов MS Access 2010 ..... **37**
Автоматическое создание формы на основе таблицы 37
Применение мастера для создания формы 39
Создание простой формы в режиме конструктора 40
Создание сложной формы 46
Создание подчиненной формы 48

Добавление кнопки в форму для вызова другой формы.. 52
Удаление записи, отображаемой в форме..... 54
Поиск в Microsoft Access 2010..... 55
Проверка орфографии 57
Создание стартовой формы..... 57
Создание простого отчета..... 58
Создание отчета в режиме конструктора..... 60
Создание запроса 62
Добавление элементов в отчет..... 64
Включение в отчет даты, времени
и номеров страниц 67
Добавление кнопки в форму для запуска отчета..... 69
Вывод отчета MS Access на печать 70
Изменение отчета..... 71
Дополнительные возможности Microsoft Access 74
Сжатие базы данных..... 74
Преобразование базы данных в формат
MS Access 2007/2010 75
Анализ быстродействия базы данных 76
Сохранение базы данных в виде accde-файла..... 77
Анализ данных в Microsoft Excel 79
Повышение быстродействия Microsoft Access..... 80
Разделение данных и приложения 81
Просмотр и изменение свойств документа
MS Access 2010 84
Импортирование объекта в свою базу данных 85
Основные сведения о Visual Basic for Applications..... 86
Среда Visual Basic for Applications 86
Переменные, типы данных и константы 88
Стандартные функции и выражения VBA..... 94
Массивы..... 94
Операторы Visual Basic for Applications..... 96
Процедуры и функции..... 101
**Преобразование базы данных MS Access 2010
в базу MS SQL Server 2008..... 103**
Литература 105

Введение

С выходом в свет Microsoft Office Access 2010 в арсенал разработчиков к комбинации мощных современных технологий и развитых средств создания прикладных программ нового поколения добавились новые возможности.

Для изучения возможностей нового программного продукта лучше всего подходит конкретное, полностью законченное приложение. Именно на примере такого приложения построено данное пособие.

Материалы пособия позволят быстро освоить базовые возможности Microsoft Access 2010. Вначале описывается создание довольно несложного приложения. Далее – модификация приложения, придание ему основных черт профессиональной разработки. В нем появится меню, лента пользователя, многостраничные формы, объекты, контролирующие права доступа к расчетам, которые выполняет программный комплекс, а также отчеты, обеспечивающие передачу данных в другие продукты Microsoft Office 2010, развитый механизм поиска данных и другие необходимые процедуры, делающие работу конечного пользователя более комфортной.

Далее рассматривается процесс перевода нашего приложения на платформу «клиент-сервер». Исходя из соотношения цены и качества, Microsoft SQL Server является самой удобной серверной СУБД (система управления базами данных). Информационные системы уровня предприятия, построенные с использованием Microsoft SQL Server, выгодно отличаются невысокой суммарной стоимостью, а богатые возможности этой

СУБД являются одним из самых важных критериев при выборе продукта, который будет использоваться на предприятии при построении баз данных.

В пособии предложен самый легкий способ создания базы данных Microsoft SQL Server – конвертация созданной и успешно работающей базы Microsoft Access в Microsoft SQL Server средствами Access.

Основные понятия баз данных

Информационная система – это система обработки информации. Все деловые приложения хранят значительные объемы данных, являющиеся ядром информационной системы. Такая система в первую очередь призвана облегчить труд человека, а для этого она должна как можно лучше соответствовать сложной модели реального мира.

В составе информационной системы всегда можно выделить две составляющие:

- компьютерную инфраструктуру предприятия или организации, которая представляет собой совокупность сетевой, телекоммуникационной и организационной инфраструктур;
- функциональные подсистемы, обеспечивающие решение задач предприятия или организации и достижение их целей.

Первая составляющая носит название корпоративной сети.

Вторая составляющая информационной системы целиком относится к прикладной области и полностью зависит от задач, решаемых предприятием или организацией. Требования к функциональным подсистемам, как правило, противоречивы, поскольку выдвигаются специалистами разных прикладных областей.

Взаимосвязи между составляющими информационной системы в действительности очень сложны. На первый взгляд они кажутся независимыми. В самом деле построение компьютерной сети и протоколы обмена данными абсолютно не зависят от методов и программ, применяемых для автоматизации работы

предприятия. При более детальном рассмотрении легко выясняется, что функциональные подсистемы не могут существовать независимо от корпоративной сети. Невозможно эксплуатировать распределенную информационную систему без соответствующей ей сетевой инфраструктуры.

Нельзя строить корпоративную сеть, не принимая во внимание прикладную функциональность информационной системы.

Типовая информационная система в самых общих чертах включает в себя следующие составляющие:

- серверы, рабочие станции с их периферийными устройствами;
- коммуникационное оборудование (маршрутизаторы, шлюзы, коммутаторы, мосты и т. д.);
- системное программное обеспечение (операционные системы, серверы баз данных и т. д.);
- базы данных;
- системы управления базами данных;
- прикладное программное обеспечение, разработанное прикладными программистами (приложение);
- обслуживающий персонал.

Банк данных – это та же информационная система, в которой представлены функции централизованного накопления информации, хранящейся в нескольких базах данных.

База данных (БД) – совместно используемый набор логически связанных данных.

Система управления базами данных (СУБД) – программное обеспечение, с помощью которого пользователи могут создавать, модифицировать базу данных и осуществлять к ней контролируемый доступ.

Прикладное программное обеспечение (приложение) – комплекс программ, обеспечивающих автоматизацию обработки информации.

Приложения могут создаваться как в среде системы управления базами данных, так и вне среды системы управления базами.

Администратор сети – лицо, отвечающее за правильное функционирование аппаратно-программных средств сети, ее реконфигурацию и восстановление системного программного обеспечения после сбоев и отказов оборудования. В его обязанности входят также мероприятия по разграничению доступа к ресурсам сети и профилактические мероприятия.

Администратор базы данных – лицо, отвечающее за проектирование базы данных, защиту ее от несанкционированного доступа. В его обязанности также входит контроль за сохранностью, достоверностью хранимой в базе данных информации и обеспечение ее непротиворечивости и сохранности.

Разработчик программного комплекса (прикладной программист) – лицо, отвечающее за разработку приложения, созданного как в среде системы управления базами данных (например, MS Access), так и внешнего (например, MS Visual Studio).

Обслуживающий персонал – группа лиц, для работы которых и создана информационная система (работники предприятия или учреждения).

Реляционные базы данных

На сегодняшний день наиболее используемыми базами данных являются реляционные базы данных.

Такое широкое распространение связано с применением в них реляционной модели данных, обеспечивающей простоту, гибкость структуры, удобство реализации, а самое главное – наличие теоретического описания. Реляционная модель данных пришла на смену сетевой модели, век которой быстро закончился именно из-за отсутствия теоретического описания.

Таблицы базы данных

Таблица – регулярная структура, состоящая из конечного набора однотипных записей.

Каждая таблица реляционной базы данных состоит из строк и столбцов и предназначена для хранения данных об объектах информационной системы. Столбцы таблиц называют полями.

Поле может содержать данные только одного из допустимых типов (тип данных) для конкретно используемой базы данных, например MS Access, MS SQL Server, Oracle и т. д.

Сущность – объект любого происхождения, данные о котором хранятся в базе данных.

Атрибут – свойство, характеризующее сущность. В таблице представляет собой заголовок столбца.

Ключи и индексы

Ключ представляет собой поле или группу полей, причем таких, которые однозначно определяют любую запись в таблице реляционной базы данных.

Ключ предназначен для:

- идентификации записей в таблице;

- установления связи между таблицами базы данных;
- создания ограничений ссылочной целостности.

В реляционных базах данных ключ реализуется с помощью индекса.

Индекс – указатель на данные, размещенные в реляционной таблице. Он предоставляет информацию о точном физическом их расположении.

Одним из основных требований, предъявляемых к СУБД, является возможность быстрого поиска требуемых записей. В реляционных базах данных для реализации этого требования как раз и служат индексы. Индекс очень похож на алфавитный указатель в книге.

При создании индекса в нем располагается информация о местонахождении записей, относящихся к индексированному полю. При добавлении новых записей в таблицу или удалении имеющихся индексов.

Разработка базы данных

Как и множество других задач, построение базы данных начинается с этапа проектирования. Проектирование реляционной базы данных включает следующие этапы:

- моделирование приложения;
- определение данных, с которыми будет работать приложение;
- распределение данных по таблицам;
- организация связей между таблицами;
- создание необходимых индексов;
- создание механизмов проверки данных;
- создание необходимых запросов к базе данных.

Постановка задачи

В качестве примера рассмотрим разработку прикладного программного обеспечения деятельности отдела по учету недвижимости. Учет недвижимости, а также отслеживание квартплаты в ведении этого отдела.

Необходимо определить задачи, которые будет решать разрабатываемое приложение.

Для разработки приложения будем использовать Microsoft Access 2010 и запланируем перевод базы данных на платформу SQL Server 2008, так как при отслеживании платежей по квартплате число записей в таблице лицевых счетов достигнет сотен тысяч, а это уже несколько превышает возможности Microsoft Access 2010.

Информационная система предназначена для сбора, хранения и обработки информации. База данных – неотъемлемая часть любой информационной системы.

Предстоит решить две задачи:

- разработать базу данных для хранения информации;
- разработать графический интерфейс и само пользовательское приложение, работающее с этой базой данных.

СУБД (система управления базами данных) – программное обеспечение, с помощью которого пользователи могут создавать, модифицировать базу данных и осуществлять к ней контролируемый доступ. СУБД непременно взаимодействует с прикладными программами пользователя и самой базой данных.

Для работы СУБД и прикладных программ необходимо аппаратное обеспечение, которое также является частью информационной системы и может быть

представлено одним компьютером или сетью из многих компьютеров. Приложение, которое мы создадим в начале нашей работы, будет функционировать на одном компьютере или в лучшем случае на уровне рабочей группы в режиме файлового сервера. В этом варианте папка с базой данных и прикладными программами размещается на самом мощном компьютере одноранговой сети, и к ней организуется совместный доступ работников. Нагрузка на локальную вычислительную сеть максимальная. Информационная безопасность – на самом низком уровне.

Дальнейшее развитие нашего приложения и перевод его в архитектуру "клиент-сервер" кардинальным образом изменит положение дел. Корпоративная сеть будет избавлена от излишнего трафика, а применение сервера баз данных поднимет безопасность информационных ресурсов.

В настоящее время существует больше сотни различных СУБД. Подавляющее большинство из них работает с базой данных, в основе которой лежит реляционная модель. На сегодняшний день известны три модели данных: иерархическая, сетевая и реляционная. Microsoft Office Access 2010 и Microsoft SQL Server 2008 – это реляционные СУБД.

Современная реляционная база данных хранит не только сами данные, но и их описания. Такой подход позволяет отделить данные от приложения. Следовательно, добавление поля в таблицу или таблицы в базу данных никак не повлияет на работу приложения. Удаление поля из таблицы, используемой приложением, повлияет на его работу. Приложение придется модифицировать.

Нормализация данных

Приступая к созданию нового приложения, главное – самым тщательным образом спроектировать структуру его таблиц. Если не уделить структуре должного внимания, то в лучшем случае это может проявиться в неэффективной работе приложения, а в худшем – в невозможности реализации некоторых требований к системе в целом. При хорошей организации набора таблиц будут решены не только текущие проблемы, но и потенциальные, которые в данный момент вы не могли предвидеть.

Э.Ф. Кодд доказал, что, следуя при создании таблиц и связей между ними только немногим формализованным правилам, можно обеспечить простоту манипулирования данными. Его методика получила наименование **нормализации данных**. Теория реляционных баз данных основана на концепции использования ключевых полей для определения отношений между таблицами. Чем больше таблиц, тем больше отношений требуется определить, чтобы связать их между собой. Из теории Кодда не следует, что каждая таблица должна быть напрямую связана с любой другой таблицей. Но, поскольку каждая таблица связана хотя бы с одной таблицей в базе данных, можно утверждать, что все таблицы в базе имеют прямые или косвенные отношения друг с другом.

Установили, какие поля будут включены в базу данных. Следующий этап состоит в разделении их на таблицы. Конечно же, можно было бы работать с единственной таблицей, но для каждого проживающего в квартире не имеет смысла повторять всю информацию о здании, квартире, ответственном квартиросъемщике

и лицевом счете, а при переименовании улицы вносить исправления в тысячи записей, содержащих сведения о технических характеристиках квартиры.

Наличие повторяющейся информации приведет к неоправданному увеличению размера базы данных. В результате снизится скорость выполнения запросов. При многократном вводе повторяющихся данных возрастет вероятность ошибки.

Несколько советов по включению полей в таблицы.

- Включайте поля, относящиеся только к предметной области таблицы. Поле, представляющее факт из другой предметной области, должно принадлежать другой таблице. Убедитесь, что каждое поле таблицы описывает только одну предметную область. Если вы обнаружите, что в разных таблицах встречается однотипная информация, значит, какие-то таблицы содержат лишние поля.
- Не включайте производные или вычисляемые данные. В большинстве случаев вам нет необходимости хранить результаты вычислений в таблице. С помощью Microsoft Access вы всегда сможете выполнить данные вычисления в нужный момент. Не имеет смысла хранить итоговые поля в таблице.
- Включите всю необходимую информацию. Подумайте о вопросах, которые вы будете формулировать к базе данных. Сможете ли вы получить на них ответ, используя данные из ваших таблиц? Включили ли вы поля, в которых будут храниться уникальные данные типа кода клиента? Какие таблицы содержат информацию, обязательную для создания отчета или формы?
- Разделите информацию на наименьшие логические единицы.

Первая нормальная форма

Таблица находится в первой нормальной форме, если значения всех ее полей атомарные и в ней отсутствуют повторяющиеся группы полей.

Приведем данные к первой нормальной форме. Выделим самостоятельные группы полей и поместим их в отдельные таблицы.

Таблица ***Улица***

№	Поле	Тип	Размер	Описание
1	НомерУлицы	Числовой	4	
2	НазваниеУлицы	Текстовый	30	
3	ПризнакАдреса	Текстовый	10	
4	ПорядокУлицы	Логический	1	Порядок следования в документах

Таблица ***Здание***

№	Поле	Тип	Размер	Описание
1	НомерУлицы	Числовой	4	Ссылка на номер улицы
2	НомерДома	Текстовый	10	Номер дома
3	НомерРайона	Числовой	1	Ссылка на район города
4	Площадь-Участка	Числовой	10	Площадь земельного участка
5	ГодПостройки	Числовой	4	Год постройки здания
6	Номер-Материала	Числовой	1	Ссылка на материал стен здания
7	Примечания	Поле	Авто	Примечания
8	Износ	Числовой	2	Износ в процентах
9	Стоимость	Денежный	15	Стоимость здания в рублях

10	Расстояние-Центр	Числовой	5	Расстояние от центра города
11	ПлощадьНежилая	Числовой	10	Площадь нежилых помещений
12	Фото	Поле OLE	Авто	Фото здания
13	ВидСобственности	Числовой	1	Вид собственности
14	НаличиеЛифта	Логический	1	Наличие лифта

Таблица ***Квартира***

№	Поле	Тип	Размер	Описание
1	НомерУлицы	Числовой	4	Ссылка на номер улицы
2	НомерДома	Текстовый	10	Номер дома
3	НомерКвартиры	Числовой	4	Номер квартиры
4	НомерЭтажа	Числовой	2	Номер этажа
5	КоличествоКомнат	Числовой	1	Количество комнат
6	Площадь-Квартиры	Числовой	Авто	Общая площадь квартиры
7	ПлощадьЖилая	Числовой	Авто	Жилая площадь квартиры
8	Площадь-Вспомогательная	Числовой	Авто	Вспомогательная площадь
9	ПлощадьБалкона	Числовой	Авто	Площадь балкона
10	ВысотаКвартиры	Числовой	Авто	Высота квартиры
11	НомерЛицСчета	Числовой	5	Номер лицевого счета
12	ФамилияКвартиросъемщика	Текстовый	20	Фамилия квартиросъемшика
13	ИмяКвартиросъемщика	Текстовый	20	Имя квартиросъемщика
14	ОтчествоКвартиросъемщика	Текстовый	20	Отчество квартиросъемшика
15	ПаспортКвартиросъемщика	Поле	Авто	Данные его паспорта

Таблица ***ВладельцыКвартир***

№	Поле	Тип	Размер	Описание
1	НомерУлицы	Числовой	4	Ссылка на номер улицы
2	НомерДома	Текстовый	10	Номер дома
3	НомерКвартиры	Числовой	4	Номер квартиры
4	Номер-Проживающего	Числовой	2	Порядковый номер
5	Фамилия-Проживающего	Текстовый	20	Фамилия проживающего
6	ИмяПроживающего	Текстовый	20	Имя проживающего
7	Отчество-Проживающего	Текстовый	20	Отчество проживающего
8	ГодРождения-Проживающего	Числовой	4	Год рождения проживающего
9	Льготы-Проживающего	Текстовый	20	Льготы и статус

Таблица ***Районы***

№	Поле	Тип	Размер	Описание
1	НомерРайона	Числовой	1	Номер района
2	НазваниеРайона	Текстовый	15	Название района

Таблица ***МатериалСтен***

№	Поле	Тип	Размер	Описание
1	НомерМатериала	Числовой	1	Номер материала
2	НазваниеМатериала	Текстовый	15	Название материала

Таблица Квартиросъемщик

№	Поле	Тип	Размер	Описание
1	НомерЛиц-СчетаNT	Числовой	5	Номер лицевого счета
2	Фамилия-Квартиросъемщика	Текстовый	20	Фамилия
3	ИмяКвартиро-съемщика	Текстовый	20	Имя квартиросъемщика
4	ОтчествоКвартиро-съемщика	Текстовый	20	Отчество
5	ПаспортКвартиро-съемщика	Поле МЕМО	Авто	Данные его паспорта

Вторая нормальная форма

Таблица находится во второй нормальной форме, если она удовлетворяет условиям первой нормальной формы и любое неключевое поле однозначно идентифицируется полным набором ключевых полей.

Каждая таблица должна содержать одно или несколько полей, однозначно определяющих каждую запись в таблице. Такие поля называют **первичным ключом таблицы**.

Если для таблицы определен первичный ключ, то Microsoft Access предотвращает дублирование значений полей или ввод значений Null в эти поля.

В Microsoft Access можно выделить три типа ключевых полей: простой ключ, составной ключ и счетчик. Если поле содержит уникальные значения, то его можно определить как **ключевое** или **простой ключ**.

Поле **НомерУлицы** в таблице **Улица** также можно определить как простой ключ. Этим же требованиям

отвечают поля **НомерРайона** и **НомерМатериала** таблиц **Районы** и **МатериалСтен**. Можно смело гарантировать их уникальность в пределах нашего программного комплекса. С таблицей **Здание** при определении первичного ключа нужно использовать **составной ключ**. Связка полей – номер улицы плюс номер дома – однозначно определит положение записи, относящейся к одному зданию в этой таблице. С однозначным определением квартиры в таблице **Квартира** составной первичный ключ выглядит так: номер улицы + номер дома + номер квартиры.

В очень редких случаях с определением первичного ключа для таблицы может сложиться тупиковая ситуация. В этом случае добавьте в таблицу поле и определите его тип как счетчик. В это поле будет автоматически вноситься уникальное число даже при работе с вашей базой с нескольких компьютеров одновременно. Если до сохранения созданной таблицы ключевые поля не были определены, Microsoft Access 2010 предложит создать ключевое поле.

Третья нормальная форма

Таблица находится в третьей нормальной форме, если она удовлетворяет условиям второй нормальной формы и ни одно из неключевых полей таблицы не идентифицируется с помощью другого неключевого поля.

Посмотрите внимательно на таблицу **Квартира**. Она содержит неключевое поле **НомерЛицСчета**, которое однозначно определяет ответственного квартиросъемщика (поля: **ФамилияКвартиросъемщика**, **ИмяКвартиросъемщика**, **ОтчествоКвартиросъемщика** и

ПаспортКвартиросъемщика) в этой таблице. Уберем все эти поля в еще одну таблицу **Квартиросъемщик** и назначим в ней в качестве простого первичного ключа поле **НомерЛицСчета**.

Таблица ***Квартиросъемщик***

№	Поле	Тип	Размер	Описание
1	НомерЛиц-Счета	Числовой	5	Номер лицевого счета
2	Фамилия-Квартиросъемщика	Текстовый	20	Фамилия
3	ИмяКвартиро-съемщика	Текстовый	20	Имя квартиросъемщика
4	ОтчествоКвартиро-съемщика	Текстовый	20	Отчество
5	ПаспортКвартиро-съемщика	Поле МЕМО	Авто	Данные его паспорта

Связи между таблицами

Microsoft Access поддерживает четыре типа связей: «один-к-одному», «один-ко-многим», «много-к-одному» и «много-ко-многим».

- Связь **«один-к-одному»** означает, что каждой записи одной таблицы соответствует только одна запись другой таблицы и наоборот. Одна квартира – один ответственный квартиросъемщик. Связь между таблицами устанавливается на основании значений совпадающих полей, а не их наименований.
- Связь **«один-ко-многим»**. Одной улице в таблице **Улица** соответствует несколько зданий из таблицы зданий **Здание**.

• Связь **«многие-к-одному»** аналогична ранее рассмотренному типу «один-ко-многим». Тип связи между объектами полностью зависит от вашей точки зрения. Например, если вы будете рассматривать связь между собственниками и квартирой, то получите «много-к-одному». Несколько собственников проживают в одной квартире.

• Связь **«многие-ко-многим»** возникает между двумя таблицами в тех случаях, когда одна запись из первой таблицы может быть связана более чем с одной записью из второй таблицы, а одна запись из второй таблицы может быть связана более чем с одной записью из первой таблицы. Таких связей следует избегать, так как реляционная модель не позволяет непосредственно работать с ними. Всегда можно ввести в базу данных еще одну-две промежуточные таблицы.

Создание проекта MS Access

1. Запустите Microsoft Access 2010. На экране появится стартовое окно Microsoft Access.

2. Определитесь с именем файла проекта MS Access, который будет работать в качестве клиента с базой MS SQL Server, и папкой, в которой он будет расположен. Обязательно укажите после имени файла расширение **.adp**.

3. Щелкните кнопку **Создать**. Появится окно запроса на подключение к серверу.

4. Выберите кнопку **Да**. Появится диалоговое окно **Свойства канала передачи данных**. Укажите в нем имя экземпляра сервера, имя пользователя, его пароль

и выберите базу данных на сервере. Не забудьте проверить соединение, нажав соответствующую кнопку.

Создание новой базы данных

1. Запустите Microsoft Access 2010. На экране появится стартовое окно с названием Microsoft Access. В этом окне можно создать новую пустую базу данных, создать базу данных с помощью шаблона или открыть одну из последних баз данных, с которыми уже велась работа.

2. Определитесь с именем файла базы данных и папкой, в которой он будет расположен. Пусть имя файла – **Недвижимость**.

3. Щелкните кнопку Создать. Будет создана новая база данных и открыта таблица **Таблица1** в режиме таблицы.

В MS Access 2010 применяется несколько расширений файлов. Основные из них:

- **accdb** – расширение файла базы данных формата MS Access 2007/2010;
- **accde** – расширение файлов MS Access 2007/2010, которые работают в режиме «исполнения». В accde-файлах удален весь исходный код. Работающий с accde-файлом может только выполнять код VBA, но не может изменять его;
- **accdt** – расширение файлов шаблонов баз данных MS Access 2007/2010;
- **adp** – расширение файла проекта MS Access 2007/2010, который предназначен для работы с базой данных MS SQL Server;
- **ade** – расширение файлов проекта MS Access 2002/2003/2007/2010, которые работают в режиме

«исполнения». В ade-файлах удален весь исходный код. Работающий с ade-файлом может только выполнять код VBA, но не может изменять его.

Создание таблиц

Существует несколько способов создания таблиц в Microsoft Access 2010:

- создание новой пустой таблицы;
- создание таблицы на основе списка на узле SharePoint;
- создание таблицы с помощью импорта внешних данных;
- создание таблицы при помощи конструктора.

Для создания новой пустой таблицы:

1. Выберите на вкладке **Создание** пункт **Таблица**. На экране появится новая пустая таблица. В ней два столбца: **Код** и **Добавить поле**. Поле **Код** содержит уникальный номер каждой строки этой таблицы и заполняется автоматически по мере добавления записей в таблицу.

2. Замените название столбца **Добавить** поле на свое, например **НазваниеУлицы**, и можете сразу составить весь список улиц города. Тип поля **НазваниеУлицы** будет назначен по умолчанию – текстовый с длиной 255 символов, если в настройках Microsoft Access 2010 не установлен другой размер.

Для создания новой таблицы на основе списка на узле **SharePoint** выберите на второй вкладке ленты **Создание** пункт **Списки SharePoint**. Откроется меню шаблонов с пунктами Контакты, Задачи, Вопросы и т. д. Выбирайте нужные, указывая URL-адрес, и проектируйте свою таблицу.

Создание таблицы с помощью импорта внешних данных позволяет импортировать таблицу, расположенную в другом месте. В этом случае в новой таблице текущей базы данных будет создана копия импортированных данных.

Для создания таблицы конструктором выберите на вкладке ленты **Создание** пункт **Конструктор таблиц**. Откроется окно конструктора с именем новой таблицы **Таблица1**. В окне три колонки. Первые две (**Имя поля** и **Тип данных**) будут использоваться приложением, а третья (**Описание**) предназначена только для разработчика. В ней опишите поля таблицы.

Каждое поле таблицы должно иметь уникальное имя, но в различных таблицах можно использовать одинаковые имена полей.

Имена полей должны содержать не более 64 символов и могут включать любые комбинации символов за исключением точки, восклицательного знака и квадратных скобок. Используйте в именах полей только буквы и не применяйте пробелы.

Длину текстового поля по умолчанию можно изменить. Для этого выберите кнопку **Файл**. В нижней части открывшегося окна щелкните по кнопке **Параметры**. Откроется диалоговое окно **Параметры Access**. Щелкните по пункту **Конструкторы объектов**. В разделе **Конструктор таблиц** установите требуемую длину текстового поля. В этом же разделе можно назначить по умолчанию и длину числового поля.

Типы полей:

Вид	Тип	Описание
Символьный	Текстовый	Текст или числа, не требующие проведения расчетов. Максимальная длина – 255 символов. По умолчанию длина текстового поля устанавливается равной максимальной длине
	Поле MEMO	Поля типа MEMO предназначены для хранения больших текстовых данных. Длина поля может достигать 64 Кбайт. Поле не может быть ключевым или индексированным. Поля MEMO полезны для хранения больших объемов информации. При работе с Access 2010 можно задать свойство (**Только добавление**), при котором приложение Access сохраняет историю всех изменений поля MEMO. Историю изменений затем можно просмотреть
Числовой	Числовой	Содержит множество подтипов (размеров). От выбора размера зависит точность вычислений. Используйте целый тип для полей, которые используются в ссылках на другие таблицы базы
	Счетчик	Уникальные, последовательно возрастающие числа, автоматически вводящиеся в таблицу при добавлении каждой новой записи
	Логический	Содержит одно из двух возможных значений 0 – для представления значения “нет” и -1 (обратите внимание: минус 1) для “да”
	Денежный	Позволяет выполнять расчеты с точностью до 15 знаков в целой и до 4 знаков в дробной части
Дата и время	Дата/Время	Семь видов форматов для отображения даты и времени
Произвольный	Поле объекта OLE	Включает рисунок, фотографию, звукозапись, диаграммы, векторную графику, форматированный текст и т. п.
Адреса Web	Гиперсвязь	Содержит адреса Web-страниц

Произ-вольный	Вложение	Позволяет хранить документы и двоичные файлы любых типов в базе данных без излишнего увеличения ее объема. Для уменьшения общего объема данных вложения автоматически сжимаются. Этот тип данных используется, например, если нужно вложить в запись документ Microsoft Office Word 2007 или сохранить в базе данных набор цифровых изображений. В одной записи можно хранить несколько вложений

Тип поля **Вложение** можно использовать для хранения нескольких файлов в одном поле таблицы, причем в этом поле можно хранить файлы самых разных типов. Например, в поле документов, подтверждающих оплату за услуги, можно добавить к записи каждого лицевого счета одно или несколько платежных поручений, а также фотографию владельца и проживающих. При использовании вложений документы и другие файлы, не являющиеся изображениями, открываются в соответствующих программах, так что эти файлы можно редактировать непосредственно в приложении MS Access 2010.

После того как в таблицу добавлено поле типа Вложение, можно вкладывать файлы в записи этой таблицы, не создавая форму для ввода данных. Кроме того, можно просматривать вложения без помощи формы. Однако помните, что для просмотра непосредственно из таблиц используются программы, в которых создавались эти файлы, или программы, поддерживающие файлы такого типа. Например, при открытии вложенного в таблицу документа MS Office Word запускается также приложение MS Word, и просмотр документа происходит в этом приложении, а не в Access.

Создайте структуры таблиц. Данные в таблицы не заносите.

Создание первичных ключей и индексов

Простой первичный ключ – это индекс, созданный по ключевому полю таблицы.

Составной первичный ключ – это индекс, созданный по ключевой связке полей таблицы.

Первичный ключ у любой таблицы может быть только один. Этого требует теория нормализации.

Кроме первичного ключа таблица может иметь любое количество обычных индексов. Среди них могут быть и уникальные, не допускающие повторяющихся значений. Их принято называть **индексами-кандидатами на роль первичного ключа**.

Индекс можно построить по полю почти любого типа. Достаточно отметить поле как индексированное.

Создание обычного индекса по полю таблицы

Порядок создания, как простого индекса, так и уникального (индекса-кандидата) – один и тот же. Порядок действий следующий:

1. Откройте таблицу в режиме конструктора таблиц.
2. Выделите поле, для которого будем создавать индекс.
3. Сделайте активным свойство **Индексированное поле**.
4. Если поле не может содержать повторяющиеся данные, выбрать значение **Да (Совпадения не допускаются)**.

Создание простого первичного ключа

Создадим простой первичный ключ для таблицы **Квартиросъемщик**. Ключевое поле, однозначно определяющее положение любой записи в этой таблице, носит название **НомерЛицСчета**.

Для его создания выполните следующие действия:

1. Откройте таблицу **Квартиросъемщик** в режиме конструктора.
2. Выделите поле **НомерЛицСчета**.
3. Сделайте щелчок мышью по инструменту **Ключевое поле** вкладки **Конструктор**. Ключ появится возле поля **НомерЛицСчета**.

Создание составного первичного ключа

Создадим составной первичный ключ для таблицы **Квартира**. Ранее ключевая связка полей для этой таблицы была определена так: **НомерУлицы** + **НомерДома** + **НомерКвартиры**:

1. Откройте таблицу в режиме конструктора.
2. Выделите поле **НомерУлицы**, нажмите клавишу <Ctrl> и, удерживая ее, щелкните последовательно кнопки выделения напротив полей **НомерДома** и **НомерКвартиры**.
3. Сделайте щелчок мышью по инструменту **Ключевое поле** вкладки **Конструктор**. Изображение ключа появится напротив всех трех выбранных полей. Имейте в виду: ключиков три, а первичный ключ таблицы – один, составной. Откройте **Индексы**, щелкнув по кнопке **Индексы**.

Контроль правильности ввода данных

Информация, накапливаемая в базе данных, должна обладать абсолютной достоверностью.

Добавление условия на значение поля

Это условие позволяет проверить корректность данных только в одном поле, независимо от значений других полей. Рассмотрим пример, в котором на номер района наложено ограничение. Этот номер не может находиться вне диапазона от 1 до 9.

Чтобы добавить условие на значение поля таблицы **НомерРайона**, выполните следующие действия:

1. Откройте таблицу **Районы** в режиме конструктора.
2. Поместите текстовый курсор в поле **Условие на значение**.
3. Наберите на клавиатуре **>=1 And <10** и нажмите клавишу **<Enter>**. Курсор переместится в поле **Сообщение об ошибке**.
4. Введите текст сообщения: Номер района должен быть в пределах от 1 до 9 включительно.

При попытке ввода номера района, который не находится в пределах диапазона 1–9, получим сообщение об ошибке и отказ программного комплекса от записи в таблицу сделанных изменений.

Добавление условия на значение записи

Это условие позволяет сравнить значения нескольких полей сразу. Рассмотрим пример, в котором производится проверка соответствия общей площади квартиры сумме составляющих: жилой, вспомогательной и при-

веденной площади балкона. Для того чтобы добавить условие на значение записи, необходимо проделать следующие действия:

1. Откройте таблицу **Квартира** в режиме конструктора.

2. Сделайте щелчок мышью по инструменту **Страница свойств** вкладки **Конструктор**. Появится диалоговое окно свойств таблицы.

3. Введите в поле **Описание** краткое назначение выполняемой проверки: Проверка общей площади квартиры.

4. Переместите курсор в поле **Условие на значение**. Появится кнопка **Мастер**.

5. Нажмите кнопку **Мастер**. Появится диалоговое окно **Построитель выражений**. Выберите поле **ПлощадьКвартиры**.

6. Введите с клавиатуры знак равенства и выполните двойной щелчок по элементу **ПлощадьЖилая**, чтобы добавить его в выражение. Аналогичные действия по отношению к элементам **ПлощадьВспомогательная** и **ПлощадьБалкона** приведут вас к окончательному виду:

[ПлощадьКвартиры] = [ПлощадьЖилая] + [ПлощадьВспомогательная] + [ПлощадьБалкона]

7. Закройте окно **Построитель выражений**.

8. Поместите текстовый курсор в поле **Сообщение об ошибке** в Окне свойств. Введите текст, который будет появляться всякий раз при нарушении условия равенства площадей: Общая площадь квартиры не равна сумме составляющих.

В случае неравенства площадей при работе программного комплекса появится сообщение.

Создание связей между таблицами

Таблицы доведены до третьей нормальной формы и помещены в базу. Первичный ключ есть у каждой таблицы. Индексы созданы. Типы связей между таблицами определены. Настало время создания связей между таблицами непосредственно в базе данных. Связи между таблицами назначают и просматривают в специальном окне **Схема данных**.

Создание связи «один-ко-многим»

Определим связь между таблицами **Районы и Здание**. Это связь «один-ко-многим». В одном районе города расположено несколько зданий.

1. Выберите инструмент **Схема данных** на вкладке **Работа с базами**.
2. Выполните щелчок правой кнопкой мыши в любом свободном месте появившегося окна **Схема данных**.
3. В появившемся меню выберите первый пункт – **Добавить таблицу**. Появится диалоговое окно **Добавление таблицы**. Раскройте вкладку **Таблицы**.
4. Выберите **Районы и Здание**. Нажмите кнопку **Добавить**. Нажмите кнопку **Закрыть**.
5. Связь между таблицами **Районы и Здание** строится по значению одноименных полей **НомерРайона**. Поместите указатель мыши над полем **НомерРайона** (оно ключевое и поэтому выделено в списке полей стилизованным изображением ключа), нажмите левую кнопку мыши и, не отпуская ее, «перетащите» появившийся значок поля на поле **НомерРайона** таблицы **Здание**. Отпустите левую кнопку мыши. Появится диалоговое окно **Изменение связей**.

6. Поставьте флажок **Обеспечение целостности данных** и нажмите кнопку **Создать** для подтверждения создания связи и перехода в окно **Схема данных**. Microsoft Access 2010 использует назначенные связи при создании форм, запросов и отчетов, которые требуют данных из рассмотренных выше таблиц.

7. Со стороны таблицы «один» началом «перетаскивания» обязательно должно быть ключевое поле, а со стороны «многие» – индексированное. Причем индексированное поле в атрибуте **Индексированное поле** должно иметь атрибут: **Да (Допускаются совпадения)**.

8. Важной особенностью MS Access 2010 является автоматическое обеспечение ссылочной целостности данных. Если на связь между таблицами наложены условия ссылочной целостности, то добавление в связанную таблицу записи, для которой нет соответствующих записей в главной таблице, становится невозможным.

9. Проверка целостности данных может осуществляться и программными средствами. Например, при добавлении в таблицу **Здание** описания нового здания вы можете проверить, имеется ли в таблице **Районы** район, в котором расположено это здание. Однако более правильным является определение условия целостности данных на уровне базы данных, так как в этом случае ни одно приложение не может нарушить целостность данных. С базой данных может работать несколько приложений, в том числе и не только ваших.

10. Если разработчик поставит флажок каскадное обновление связанных полей, то у него появится возможность исправить номер района лишь в таблице **Районы**, а в таблице **Здание** все связанные записи система MS Access исправит автоматически.

11. Поставленный флажок **Каскадное удаление связанных записей** позволит вам смело удалить район в таблице **Районы**, а все описания зданий этого района Access 2010 удалит без вашего участия.

Рассмотрим создание связи между таблицами в случае, когда одна из них имеет составной первичный ключ. Для этой цели подходят таблицы **Квартира** и таблица **ВладельцыКвартир**. У таблицы **Квартира** ключевая связка полей выглядит так: **НомерУлицы + НомерДома + НомерКвартиры**.

Для построения связи между этими таблицами выполните следующие действия:

1. Поместите указатель мыши над полем **НомерУлицы** таблицы **Квартира** и сделайте щелчок левой кнопкой мыши.

2. Нажмите клавишу <Shift> и, не отпуская ее, сделайте сначала щелчок по полю **НомерДома**, а затем **НомерКвартиры**. Отпустите клавишу <Shift>. Будет выделена группа из трех полей: **НомерУлицы + НомерДома + НомерКвартиры**.

3. Поместите указатель мыши над выделенной группой из трех полей таблицы **Квартира**, нажмите левую кнопку мыши и, не отпуская ее, «перетащите» появившийся значок связки полей в любое место таблицы **ВладельцыКвартир**. Отпустите левую кнопку мыши. Появится диалоговое окно **Изменение связей**. В отличие от связывания таблицы, имеющей простой первичный ключ, где связь была установлена автоматически, в данном случае необходимо явно указать, какие связки полей из таблиц **Квартира** и **ВладельцыКвартир** участвуют при создании связи между этими таблицами.

4. Используя поле с раскрывающимся списком, установите нужные вам связи.

5. Поставьте все три флажка, обеспечивающие ссылочную целостность. Каскадное удаление связанных записей здесь вполне уместно. В последнее время часто две соседние квартиры приобретает одна семья. Без третьего флажка удалить данные по квартире и одновременно по проживающим нельзя. Поставьте его.

Создание связи «один-к-одному»

Такая связь должна быть установлена между таблицами **Квартиросъемщик** и **Квартира**, так как каждая квартира имеет свой уникальный номер лицевого счета. Для построения связи выполните следующие действия:

1. Поместите указатель мыши над полем **НомерЛицСчета** (оно ключевое). Нажмите левую кнопку мыши и, не отпуская ее, перетащите появившийся значок поля на поле **НомерЛицСчета** таблицы **Квартира**. Отпустите левую кнопку мыши. Появится диалоговое окно **Изменение связей.**

2. Поставьте флажок **Обеспечение целостности данных** и нажмите кнопку **Создать** для подтверждения создания связи и перехода в окно **Схема данных**.

3. Началом перетаскивания обязательно должно быть ключевое поле (таблица **Квартиросъемщик**). Конечная цель – индексированное поле (таблица **Квартира**). Индексированное поле в атрибуте Индексированное поле должно иметь значение: **Да (Совпадения не допускаются)**.

Устранение проблем, возникающих при создании ключей

Завершите создание базы данных, назначьте первичные ключи и установите связи между таблицами, а потом вносите данные. Нарушение этого порядка может доставить вам массу неприятностей. Вот одна из них.

В таблицу **Квартира** занесена 356 491 запись. Первичный ключ у этой таблицы, конечно же, не создан. Информация обо всех квартирах занесена, вы пытаетесь создать первичный ключу этой таблицы, а в ней есть повторяющиеся записи. Необходимо построить запрос **Повторяющиеся записи** следующим образом: **Вкладка Создание / группа Запросы / Мастер запросов / Повторяющиеся записи**.

Устранение связи «многие-ко-многим»

Преобразование связи «многие-ко-многим» в несколько связей «один-ко-многим» и «многие-к-одному» можно сделать следующим образом. Вместо того чтобы рассматривать множество поставщиков, поставляющих товары многим клиентам, будем рассматривать одного поставщика и множество поставляемых им товаров («один-ко-многим»). Далее будем рассматривать деление товара одной партии на множество частей, подлежащих продаже («один-ко-многим»). В результате получим множество товаров, приобретаемых одним покупателем («многие-к-одному»).

Создание форм и отчетов MS Access 2010

Форма MS Access 2010 – это объект базы данных, который можно использовать для ввода, изменения или отображения данных из таблицы или запроса.

Форма может использоваться как стартовая точка вашего приложения. Для автоматизации часто выполняемых действий формы содержат так называемые элементы управления, с помощью которых осуществляется доступ к данным. Формы можно рассматривать как окна, через которые пользователи могут просматривать и изменять базу данных. Рационально построенная форма ускоряет работу с базой данных, поскольку пользователям не требуется искать то, что им нужно. Внешне привлекательная форма – достойный элемент интерфейса. Она делает работу с базой данных более приятной и эффективной, кроме того, она может помочь в предотвращении неверного ввода данных.

Автоматическое создание формы на основе таблицы

На вкладке **Создание** в группе **Формы** 6 способов создания форм:

- создание формы с помощью инструмента **Форма**;
- создание формы при помощи инструмента **Пустая форма**;
- создание **Web-формы**, в которой отображаются несколько записей, при помощи инструмента **Несколько элементов**;
- создание разделенной формы при помощи инструмента **Разделенная форма**;

- создание формы в виде сводной диаграммы при помощи инструмента **Сводная диаграмма**;
- создание формы в виде сводной таблицы при помощи инструмента **Сводная таблица**.

Для создания формы с помощью инструмента **Форма** в области навигации выберите таблицу с данными, которые должны отображаться в форме, и сделайте щелчок мышью по пиктограмме этого инструмента. MS Access 2010 создаст форму и отобразит ее в режиме макета. В режиме макета можно внести изменения в структуру формы при одновременном отображении данных. При необходимости можно настроить размер полей в соответствии с данными. Созданная форма со стандартной Access-линейкой навигации по записям готова к работе.

Заслуживает особого внимания инструмент **Разделенная форма**. Созданная им форма одновременно отображает данные в режиме формы и в режиме таблицы.

Эти два отображения связаны с одним и тем же источником данных и всегда синхронизированы друг с другом. При выделении поля в одной части формы выделяется то же поле в другой части. Данные можно добавлять, изменять или удалять в каждой части формы. Работа с разделенной формой дает преимущества обоих типов формы в одной форме. Например, можно воспользоваться нижней (табличной) частью формы, чтобы быстро найти запись, а затем просмотреть или изменить запись в верхней части формы. Для отображения нужных записей можно воспользоваться их сортировкой.

Применение мастера для создания формы

MS Access 2010 имеет в своем арсенале еще одно средство для быстрого создания формы – мастер форм. С его помощью можно создавать формы как на основе одной таблицы или запроса, так и на основе нескольких связанных таблиц. Более солидные результаты дает создание формы с помощью мастера с последующим усовершенствованием ее в режиме конструктора.

Мастер форм разбивает процесс создания формы на несколько этапов. На каждом из них выбираются определенные параметры в предложенном диалоговом окне. Если на одном из этапов сделана ошибка и необходимо изменение уже выбранных параметров, то мастер всегда позволяет вернуться к предыдущему шагу.

Создадим при помощи мастера форму **Здание**. Она отображает информацию из главной таблицы **Здание** и двух таблиц-справочников: **Улица** и **Районы**:

1. Для запуска мастера форм выберите пункт **Мастер форм**, расположенный на вкладке ленты **Создание** в разделе **Формы**. Появится первое окно мастера. Выберите таблицу **Здание**.

2. В поле со списком **Доступные поля** отображены все поля выбранной таблицы **Здание**. Выберите только те из них, которые следует отобразить в создаваемой форме.

3. Для перехода ко второму шагу работы мастера форм нажмите кнопку **Далее**. Появится второе окно мастера. Существует несколько видов форм в зависимости от представления на них данных. Некоторые из них: в один столбец, ленточный, табличный, выровненный, сводная таблица, сводная диаграмма. Мастер

предлагает нам выбрать один, но только из четырех. Пусть это будет первый вид: **в один столбец**.

4. На третьем шаге требуется указать название формы и выбрать опцию переключателя:

- Открыть форму для просмотра и ввода данных;
- Изменить макет формы.

5. Если вы хотите внести свои изменения в форму, созданную мастером, то выберите **Изменить макет формы** и нажмите кнопку **Готово**. Рядом с созданной формой появится окно **Список полей**. Двойной щелчок по нужному полю из связанной таблицы – и оно появится в форме. Добавим **НазваниеРайона** и **НазваниеУлицы**.

В результате получили форму в соответствии с выбранными параметрами.

6. Остается заменить названия полей надписями. Щелкните по нужному полю и вводите текст надписи.

Если требуется определенная доработка, ее можно выполнить в режиме конструктора форм.

Любая форма, так же как и таблица базы данных MS Access 2010, может быть выведена на печать. Работа с формами может вестись в пяти режимах: в режиме формы; в режиме конструктора; в режиме таблицы; в режиме сводной таблицы; в режиме сводной диаграммы.

Создание простой формы в режиме конструктора

Создадим в режиме конструктора форму, связанную с таблицей **Здание**, содержащую элементы различных типов. В любой момент времени в этой форме будет отображаться информация только по одному зданию.

1. На вкладке **Создание** щелкните по кнопке **Конструктор форм**.

На экране появятся три дополнительные вкладки ленты MS Access 2010: **Конструктор**, **Упорядочить** и **Формат**, а также форма в режиме конструктора. На вкладке ленты в разделе **Элементы управления** расположена панель инструментов. Она предназначена для размещения в форме выбранных элементов и содержит их изображения.

2. На вкладке **Конструктор** выберите элемент **Страница свойств**, и окно свойств появится на экране. Каждый объект в MS Access, включая непосредственно базу данных, имеет свойства. Имеются различные категории свойств формы. В MS Access 2010 они представлены на пяти вкладках:

- **Макет** – свойства, которые принадлежат способу отображения объекта;
- **Данные** – свойства, которые принадлежат данным объекта, независимо от того, каким способом они получены;
- **События** – свойства, которые принадлежат событиям и связанным с ними процедурам;
- **Другие** – свойства, которые принадлежат характеристикам объекта или его признакам;
- **Все** – все категории и свойства объекта.

3. В окне свойств перейдите на вкладку **Данные**. Перейдите на первую строчку **Источник записей**. Выберите таблицу **Здание**.

4. В окне свойств перейдите на вкладку **Макет**. Установите свойство **Кнопки размеров** окна в **Отсутствуют**.

5. Цвет фона формы является основным параметром, определяющим ее внешний вид. Для изменения цвета фона формы или объекта:

- Выделите раздел формы, например **Область данных** или объект, цвет фона которого нужно изменить.
- В окне свойств появятся значения всех свойств этого объекта. Перейдите на вкладку **Макет**.
- Щелкните левой кнопкой мыши по свойству **Цвет фона**. Выберите цвет.

6. Чтобы задать фоновый рисунок для формы:

- В режиме конструктора выделите всю форму.
- Сделайте доступным окно свойств формы.
- Перейдите на первую вкладку **Макет**.
- Выберите свойство **Рисунок**.
- Выберите рисунок. Действие фонового рисунка распространяется, в отличие от цвета фона, на все разделы формы.
- Используя свойство **Масштабы** рисунка, можно установить его размеры.
- Чтобы определить положение рисунка, воспользуйтесь свойством Выравнивание рисунка.
- Чтобы фоновый рисунок центрировался относительно формы, а не окна формы, установите его значение **По центру формы**.
- Используя свойство Мозаичное заполнение, можно добиться повторяющегося изображения рисунка по всей форме.

7. Чтобы создать поле со списком:

- В разделе **Элементы управления** вкладки **Конструктор** нажмите кнопку **Мастера**.
- Нажмите кнопку **Поле со списком** на **Панели элементов**.

• Сделайте щелчок левой кнопкой мыши. Одновременно с этим откроется первое диалоговое окно мастера списков **Создание полей со списком**. Вам будет предложено выбрать источник значений.

• После выбора нажмите кнопку **Далее**. Появится второе диалоговое окно, в котором отображен список всех таблиц, входящих в базу данных.

• Выберите таблицу и нажмите кнопку **Далее**.

• В появившемся третьем диалоговом окне MS Access предлагает выбрать поля таблицы, значения которых будут отображаться в раскрывающемся поле со списком.

• Четвертое диалоговое окно предлагает определить порядок отображения элементов в поле со списком. Если строк в таблице более десятка, то сортировка – обязательный момент. Отсортировать записи можно максимум по четырем полям. Для текстовых полей порядок сортировки – по алфавиту.

• В пятом диалоговом окне поставьте флажок **Скрыть ключевой столбец** (рекомендуется). Установите ширину колонок.

• Нажмите кнопку **Далее**. Надо указать поле таблицы, в которое будет записываться ссылка.

• В последнем диалоговом окне предлагается ввести надпись, которая будет стоять рядом с раскрывающимся списком.

• Нажмите кнопку **Готово**.

8. Для создания поля типа **Группа переключателей**:

• Убедитесь, что кнопка с подсказкой **«Использовать мастера»** нажата.

• На этой же панели выберите пиктограмму **Группа переключателей**.

• Поместите указатель мыши над активной областью формы. Он превратится в значок группы переключателей с крестиком в левом верхнем углу.

• Нажмите левую кнопку мыши и, удерживая ее в нажатом состоянии, переместите курсор по диагонали так, чтобы получилась рамка требуемого размера. Отпустите левую кнопку мыши. Автоматически запустится построитель группы переключателей.

• Сделайте подписи у переключателей и нажмите кнопку **Далее**. Теперь назначим переключатель, используемый по умолчанию.

• Выбирайте тип оформления.

• На последнем шаге необходимо ввести подпись для созданной группы переключателей.

9. Для хранения фотографий зданий добавим присоединенную рамку объекта в форму. Этот элемент управления представляет собой контейнер OLE, в котором могут отображаться растровые и векторные изображения, звуковые объекты, рисунки с анимацией, видеообъекты и т. п.

• Выберите на панели элементов пиктограмму **Присоединенная рамка объекта**.

• Нажмите левую кнопку мыши и, удерживая ее в нажатом состоянии, переместите курсор по диагонали так, чтобы получилась рамка требуемого размера. Отпустите левую кнопку мыши.

• Удалите подпись возле этого элемента управления, которую сгенерировал компьютер: **«Присоединенный OLE»**.

• Сделайте щелчок правой кнопкой мыши по только что созданному объекту и в появившемся меню выберите пункт **Свойства**.

• Привяжите к объекту поле таблицы.

10. Для масштабирования графических объектов внутри присоединенной рамки объекта предназначено свойство **Установка размеров** на вкладке **Макет** окна свойств. Если в качестве значения этого свойства выбрать:

• **Фрагмент**. Фотография будет отображена в ее исходной пропорции. Если фото не помещается в рамку целиком, то оно урезается снизу и справа.

• **Вписать в рамку**. Фотография будет вписана в очерченное ранее пространство. Масштаб по высоте и ширине фотографии будет установлен отдельно, чтобы заполнить рамку полностью.

• **По размеру рамки**. Масштаб фотографии будет увеличен или уменьшен по ширине и по высоте так, чтобы фото целиком поместилось в рамку, и при этом его исходная пропорция сохранилась бы.

11. После создания формы можно добавить запись в таблицу **Здание** с помощью формы:

• Сделайте активной Область навигации базы данных **Недвижимость**.

• Запустите на выполнение форму **Здание**.

• Сделайте щелчок правой кнопкой мыши по месту, где должна располагаться фотография.

• В появившемся меню выберите пункт **Вставить объект**. Появится диалоговое окно Microsoft Access для вставки объекта.

• Выберите **Создать из файла**, нажмите кнопку **Обзор** и укажите файл, в котором хранится фото здания. Не устанавливайте флажок **Связь**.

12. Для изменения последовательности перехода между элементами формы сделайте следующее:

- Откройте форму в режиме конструктора.
- В появившемся контекстном меню выберите пункт **Конструктор**.
- На экране появятся три дополнительные вкладки: **Конструктор**, **Упорядочить** и **Формат**. Перейдите на вкладку **Конструктор**.
- Сделайте щелчок левой кнопкой мыши по значку **Переходы**. Появится окно с названием **Последовательность перехода**.
- Для изменения очередности обхода элементов выберите из них нужный и сделайте щелчок по кнопке, находящейся слева от названия.
- Перетащите его в нужное место последовательности.

Создание сложной формы

Форма **Здание** дает возможность работать с таблицей **Здание**. Необходимо создать формы для работы с квартирами, расположенными в здании, и с проживающими в них собственниками. Можно создать еще три формы для раздельного отображения информации по квартирам, лицевым счетам и по проживающим в этих квартирах собственникам, но ввиду небольшого количества полей, содержащихся в таблицах **Квартира, Квартиросъемщик** и **ВладельцыКвартир**, есть смысл поместить всю эту информацию в одной сложной форме. Дадим ей имя **Квартиры**.

Первая проблема, которую необходимо решить на пути создания сложной формы, заключается в следующем. В нашей форме должна отображаться информация не обо всех квартирах в городе, а только о тех, которые

расположены в выбранном для просмотра здании. Поэтому форма **Квартиры** будет связана не с таблицей **Квартира**, а с запросом **Квартиры**, в который мы поместим данные о квартирах, находящихся в одном конкретном здании, и лицевых счетах ответственных квартиросъемщиков.

Порядок создания запроса:

1. Выберите вкладку **Создание**.
2. Сделайте щелчок левой кнопкой мыши по значку **Конструктор запросов**. Появится окно конструктора запросов и окно с названием **Добавление таблицы**.
3. Выберите таблицу **Квартира** и нажмите кнопку **Добавить**, а затем таблицу **Квартиросъемщик**. Закройте окно **Добавление таблицы**. Появится дополнительная вкладка **Конструктор**.
4. Расположите последовательно все поля из таблиц **Квартира** и **Квартиросъемщик** в запросе. Поле **НомерЛицСчета**, связывающее эти таблицы, заносится только один раз.
5. Определите условия отбора записей из таблиц **Квартира** и **Квартиросъемщик** в запрос. Он должен содержать данные только по одному конкретному зданию.
6. Установите порядок сортировки записей, попавших в запрос. Лучше всего, если квартиры в нашей форме будут отображаться в порядке возрастания номеров. Это поле **НомерКвартиры**.
7. Сохраните созданный запрос под именем **Квартиры**.
8. Сделайте щелчок правой кнопкой мыши в свободном месте окна конструктора запросов. Появится контекстное меню. Первым пунктом в нем должен быть

пункт **Режим SQL**. Щелкните по нему правой кнопкой мыши. Появится текст запроса на языке SQL, сгенерированный конструктором запросов.

9. Обратите внимание на сложность текста запроса и на то, с какой легкостью мы сформировали его с помощью конструктора запросов, не зная ни одной конструкции и ни одного служебного слова языка SQL. Начинающему пользователю MS Access вовсе необязательно знать этот язык, но мы к нему обязательно вернемся.

Создание подчиненной формы

Подчиненная форма – это форма, находящаяся внутри другой формы. Первичная форма (в нашем случае **Квартиры**) называется главной. Подчиненные формы очень удобны для вывода информации из таблиц или запросов, связанных отношением «один-ко-многим». Одна квартира – несколько проживающих. При использовании формы с подчиненной формой для ввода новых записей текущая запись в главной форме сохраняется при входе в подчиненную форму. Это гарантирует, что записи из таблицы или запроса на стороне «многие» будут иметь связанную запись в таблице или запросе на стороне «один». MS Access автоматически сохраняет каждую запись, добавляемую в подчиненную форму, и никакие специальные приемы типа обработки события До вставки не требуются.

Порядок создания подчиненной формы следующий:

1. Откройте первичную форму **Квартиры** в режиме конструктора. Убедитесь, что на вкладке **Конструктор** главной ленты в разделе **Элементы**

управления кнопка **Использовать мастера** нажата. Если нет – выделите ее щелчком левой кнопки мыши.

2. Нажмите на панели элементов кнопку **Указатель мыши** и наведите указатель мыши на то место первичной формы, где вы планируете поместить левый верхний угол подчиненной формы. Указатель мыши превратится в значок подчиненной формы с крестиком в левом верхнем углу.

3. Нажмите левую кнопку мыши и, удерживая ее в нажатом состоянии, переместите курсор по диагонали так, чтобы получилась рамка требуемого размера. Отпустите левую кнопку мыши. Автоматически запустится построитель подчиненной формы.

4. Первый шаг работы мастера подчиненных форм – определение данных, которые надо включить в подчиненную форму. Установите переключатель **Имеющиеся** таблицы и запросы и щелкните по кнопке **Далее**. Появится окно для выбора таблиц и полей. Выберите все поля из таблицы **Квартиросъемщик**.

5. Второй шаг – определение полей связи между главной и подчиненной формами. Основа главной формы **Квартиры** – одноименный запрос, в который попали данные из двух таблиц: **Квартира** и **Квартиросъемщик**. Подчиненная форма основана на данных из таблицы **Квартиросъемщик**. Никогда ранее мы не устанавливали отношений между запросом Квартиры и таблицей **Квартиросъемщик**, поэтому все, что может предложить мастер подчиненных форм на этом шаге, нам не подходит. Сделайте щелчок по кнопке **Самостоятельное определение**.

6. Третий шаг – выбор имени для подчиненной формы. Под этим именем она появится в списке форм

базы данных **Недвижимость**. Мы уже условились, что назовем ее **Квартиросъемщики**. Введите это имя и щелкните по кнопке **Готово**.

7. Закройте первичную форму **Квартиры**. Дальнейшая работа будет происходить с формой **Квартиросъемщики**. Откройте ее в режиме конструктора. То, что сгенерировал мастер подчиненных форм, необходимо подвергнуть некоторому улучшению.

В заголовки колонок вынесены названия полей таблицы **Квартиросъемщики**. Конечному пользователю они ни о чем не говорят. Наличие ключевых полей **НомерУлицы**, **НомерДома** и **НомерКвартиры** в подчиненной форме вносит настоящую путаницу. Кнопки перехода по записям подчиненной формы в данном случае просто мешают работе.

Выполните доработку формы:

1. Сделайте подчиненную форму ленточной. В окне свойств этой формы выберите вкладку **Макет**, найдите свойство **Режим по умолчанию** и измените его значение: *Ленточные формы*.

2. Удалите заголовок формы и ключевые поля **НомерУлицы**, **НомерДома** и **НомерКвартиры**. Для удаления элемента сделайте по нему щелчок левой кнопкой мыши и нажмите клавишу **<Delete>**.

3. Уберите совсем кнопки перехода по записям. Для этого на вкладке **Макет** в качестве значения свойства **Кнопки навигации** поставьте *Нет*.

4. Отформатируйте объекты формы. Для этого поместите указатель мыши в любую точку на границе выделенного элемента, отличную от маркеров изменения размеров. Нажмите левую кнопку мыши и, не отпуская ее, перетащите элемент на новое место. Более точно

выставить элемент управления на форме можно при помощи клавиш-стрелок при нажатой клавише **<Ctrl>**, а поточнее изменить размеры элемента можно при помощи клавиш-стрелок при нажатой клавише **<Shift>**.

Добавление кнопки в форму для вызова другой формы

В предыдущем разделе была создана сложная форма **Квартиры** для отображения информации о квартирах, расположенных в здании.

Запускать на выполнение форму **Квартиры** имеет смысл только из формы **Здание**, так как два параметра адреса – улицу и номер дома – запрос по квартирам берет именно из этой формы. Добавьте кнопку с названием Квартиры в форму.

1. Откройте форму **Здание** в режиме конструктора.
2. Убедитесь, что на панели элементов кнопка с подсказкой **Использовать мастера** нажата. Если нет – выделите ее щелчком левой кнопки мыши.
3. Сделайте щелчок левой кнопкой мыши по инструменту **Кнопка** на панели элементов.
4. Наведите указатель мыши на то место формы **Здание**, где вы планируете поместить левый верхний угол кнопки. Указатель мыши превратится в значок кнопки с крестиком в левом верхнем углу.
5. Нажмите левую кнопку мыши и, удерживая ее в нажатом состоянии, переместите курсор по диагонали так, чтобы получилась рамка требуемого размера. Отпустите левую кнопку мыши. Автоматически запустится построитель кнопки.

6. Появится диалоговое окно. Выберите категорию и действия. Нажмите кнопку **Далее**.

7. На втором шаге работы мастера кнопок необходимо выбрать форму, которая будет запущена на выполнение после щелчка мышью по кнопке. Это форма **Квартиры**.

8. На следующем шаге надо выполнить отбор сведений для отображения в форме. Так как форма **Квартиры** отображает данные, полученные в результате выполнения запроса по одному конкретному зданию, и никакого дополнительного отбора не требует – оставьте появившийся переключатель в исходном положении: **Открыть форму и показать все записи**.

9. На следующем шаге выполняется оформление внешнего вида кнопки. На ней можно разместить как надпись, так и рисунок, который находится в файле. При размещении рисунка Access предложит стандартное окно для поиска файла с рисунком.

10. На последнем шаге предлагается задать имя кнопки как объекта для дальнейшей ссылки на нее. Введите **КнопкаКвартиры**. Сделайте щелчок левой кнопки мыши по кнопке **Готово**. Если последний шаг пропустить, то компьютер сам присвоит созданной кнопке имя, под которым она и будет фигурировать как объект VBA.

11. Запустите форму **Здание** в режиме конструктора и сделайте двойной щелчок по созданной кнопке **Квартиры**. Откроется окно свойств этого объекта. Перейдите на вкладку **События** и выберите **Нажатие кнопки**.

Код процедуры нажатия кнопки:

```
Private Sub КнопкаКвартиры Click()
On Error GoTo Err КнопкаКвартиры Click
```

```
Dim stDocName As String
Dim stLinkCriteria As String stDocName = «Квартиры»
    DoCmd.OpenForm stDocName, , , stLinkCriteria
Exit КнопкаКвартиры Click:
    Exit Sub Err КнопкаКвартиры Click:
    MsgBox Err.Description Resume Exit КнопкаКвартиры
Click End Sub
```

Код процедуры можно несколько упростить.

Код процедуры нажатия кнопки после доработки:

```
Private Sub КнопкаКвартиры Click()
    DoCmd.OpenForm «Квартиры»
End Sub
```

Единственное отличие в работе этих процедур заключается в том, что в случае возникновения ошибки при работе формы **Квартиры** об этом в первой процедуре сообщит MS Access 2010, а во второй – Visual Basic for Applications 7.0. В обоих случаях будет выведен один и тот же текст, причем VBA укажет еще и номер ошибки.

Итак, кнопка для вызова формы **Квартиры** из формы **Здание** создана. Основная часть программного комплекса готова к работе.

Попробуйте полностью от начала до конца заполнить карточку здания. После заполнения очередного поля и нажатия клавиши **<Enter>** курсор перепрыгивает вовсе не на следующее поле, а совсем в другой угол формы, и приходится прибегать к помощи мыши, чтобы вернуть его в нужное место.

Надо исправить сложившееся положение. Откройте форму **Здание** в режиме конструктора. Сделайте щелчок левой кнопкой мыши по значку **Переходы**

вкладки **Конструктор**. Появится диалоговое окно **Последовательность перехода**, в котором отображен список всех объектов, имеющихся в форме. Причем отображены они в той последовательности, в какой попали в форму. С помощью полосы прокрутки найдите кнопку с именем **КнопкаКвартиры**. Есть возможность перетащить любой объект в любое место окна. Выделите его щелчком левой кнопки мыши и разместите по вашему усмотрению.

Наилучший порядок перехода фокуса между элементами формы – естественный. Если все элементы формы расположены один под другим, то нажмите кнопку **Авто**. После внесения изменений не закрывайте окно щелчком левой кнопкой мыши по крестику, расположенному в правом верхнем углу окна, так как в этом случае вся ваша работа пойдет насмарку. Для сохранения сделанных изменений обязательно нажмите кнопку **ОК**.

Итак, созданы две формы, которые позволяют добавлять и корректировать данные по описаниям зданий, квартир и проживающих в них, но не предусмотрена возможность удаления и поиска информации. Эти функции реализованы в Microsoft Access его создателями. Ни одна другая СУБД не имеет их в своем арсенале в таком виде. Если эта реализация устраивает разработчика программного комплекса, то больше ничего делать не требуется.

Удаление записи, отображаемой в форме

Для удаления записи из таблицы, которая связана с формой:

1. Сделайте эту запись активной. Для этого откройте нужную форму в режиме формы и при помощи линейки навигации по записям найдите требуемую.

2. На ленте MS Access 2010 выберите вкладку **Главная**, а в ней в разделе **Записи** пункт **Удалить**. Откроется дополнительное меню.

3. В появившемся меню выберите пункт **Удалить запись**.

4. Появится запрос на подтверждение удаляемой записи. Нажмите кнопку **Да**.

Если при формировании связей между таблицами вы указали режим **Каскадное удаление связанных записей**, то вместе со зданием будут удалены все квартиры и проживающие в них, если нет – то MS Access выдаст сообщение о невозможности удаления этого здания, но только при наличии в нем квартир и проживающих.

Поиск в Microsoft Access 2010

Возможности поиска, предоставляемые Microsoft Access, необычайно широки. Рассмотрим лишь некоторые из них.

Задача:

Требуется найти все здания, расположенные не в Центральном районе, 1963 года постройки с износом до 30%. Площадь земельного участка должна быть до 2500 м2. Здания должны находиться в административной собственности и иметь лифт. Наличие фотографии обязательно.

Решение:

1. Запустите на выполнение форму **Здание**. На ленте MS Access 2010 выберите вкладку **Главная**, а на ней

в разделе **Сортировка и фильтр** – значок **Параметры расширенного фильтра**. Появится контекстное меню. Вторым в нем должен быть пункт **Изменить фильтр**. Выберите его. Появится окно фильтра. В этом окне мы можем задать условие, которому должно отвечать значение любого поля, находящегося в форме.

- В список, из которого вы выбираете значение поля, занесены только те значения, которые встречаются в базе данных. Например, если в базе нет зданий 1909 года постройки, то среди предложенных вариантов выбора этот год отсутствует.
- Переключатель, который при занесении здания позволял выбрать лишь один вид собственности, сейчас доступен полностью.
- Наличие фотографии – значение Is Not Null.
- Для формирования очень сложных условий внизу окна имеется вкладка **Или**. Сделайте по ней щелчок левой кнопкой мыши, если в этом есть необходимость, и задайте еще группу условий. Таких групп условий может быть сколько угодно.

2. После ввода всех нужных условий снова выберите значок **Параметры расширенного фильтра**, а в открывшемся меню – пункт **Применить фильтр**. Снова появится форма Здание, но отображаться в ней будут только два здания, удовлетворяющие условиям нашего примера, и надпись: **С фильтром**.

Снимите фильтр, когда поставленная задача будет решена. Для этого предназначен тот же пункт меню, только теперь он носит название **Удалить фильтр**.

Проверка орфографии

Чтобы найти ошибки:

1. Поместите указатель мыши в поле **Примечания**.
2. На ленте MS Access 2010 выберите вкладку **Главная**. В разделе Записи сделайте щелчок по инструменту **Орфография**.
3. Появится диалоговое окно с информацией об ошибке. Исправьте ошибку.

Если конкретное поле не указано, то проверка орфографии будет проведена по всем полям формы.

Создание стартовой формы

Для запуска приложения **Недвижимость** служит форма-заставка. **Форма Здание** активизируется после щелчка по кнопке ее запуска.

Код события **Нажатие кнопки** содержит код процедуры.

Код процедуры запуска формы Здание:

```
Private Sub Кнопка2 Click()
' Кнопка Далее
On Error GoTo Err Кнопка2 Click Dim
    stDocName As String Dim
    stLinkCriteria As String
    DoCmd.Close ' Закрытие формы stDocName = «Здание»
    DoCmd.OpenForm stDocName, , , stLinkCriteria
Exit Кнопка2 Click:
    Exit Sub
Err Кнопка2 Click:
    MsgBox Err.Description Resume Exit
Кнопка2 Click End Sub
```

Код процедуры выхода из программного комплекса:

```
Private Sub Кнопка1 Click()
    ' Кнопка СТОП
    ' Выход из программного комплекса ' Возврат в MS Access DoCmd.Close '
Закрытие формы End Sub
```

Окно MS Access 2010 остается после этого открытым.

Если в параметрах запуска MS Access 2010 сделать ссылку на эту форму, то после щелчка мышью по файлу Недвижимость.accdb пользователь сразу увидит приложение в работе. Для этого выберите в главном окне MS Access 2010 кнопку **Файл**, а в левом нижнем углу открывшегося окна – кнопку **Параметры**.

Поле со списком **Форма просмотра** позволяет выбрать форму, которая будет выводиться на экран при открытии базы данных. Задайте заголовок и значок приложения. Здесь же можно выполнить настройку главного окна MS Access 2010 при работе с базой данных. Просто поставьте или снимите флажки в нужном месте.

' Создание простого отчета

Основная сфера применения форм – обеспечение возможности просмотра отдельных или небольших групп связанных записей. Отчеты же представляют собой наилучшее средство отображения информации из базы данных в виде печатного документа.

Разработка отчета очень похожа на разработку формы. Используется та же панель элементов, тот же список полей и окно свойств.

Создайте отчет для вывода на печать всей информации, которая имеется в базе данных по отдельно взятой квартире: адрес, технические характеристики, ответственный квартиросъемщик, номер лицевого счета, список проживающих. Отчет будет запускаться из формы **Квартиры**.

Работа по созданию отчета всегда начинается с выбора источника, из которого будут извлекаться записи отчета. Отчет может представлять собой как простой список, так и подробную сводку данных, представленных в виде официального документа того или иного ведомства. Однако в любом случае сначала определяют поля, которые должны войти в отчет, и в каких таблицах или запросах находятся эти поля.

Стандартное средство MS Access **Отчет** – самый быстрый способ создания отчета, потому что с его помощью отчет формируется сразу же, без уточнения дополнительной информации. В отчете будут представлены все записи базовой таблицы или запроса.

1. В области переходов щелкните таблицу **Улица**, на основе которой будет создаваться отчет.

2. На вкладке **Создание** ленты MS Access 2010 в разделе **Отчеты** щелкните инструмент **Отчет**. MS Access немедленно создаст отчет и отобразит его в режиме макета.

3. После просмотра отчет можно сохранить, а затем закрыть и его, и источник записей – таблицу или запрос. В следующий раз при его открытии MS Access отобразит в нем самые последние данные из таблицы **Улица**.

Полученный отчет очень далек от совершенства, он позволяет лишь быстро просмотреть базовые данные.

Вторая возможность – **мастер отчетов** – средство MS Access 2010, помогающее создать отчет на основании ответов, полученных на заданные пользователю вопросы.

Мастер отчетов предоставляет больше возможностей относительно выбора полей для включения в отчет. При этом разработчик может указать способ группировки и сортировки данных, а также включить в отчет поля из нескольких таблиц или запросов, но только в том случае, если отношения между этими таблицами и запросами заданы заранее.

Создание отчета в режиме конструктора

В MS Access 2010 отчет разбит на разделы. Разделы отчета можно увидеть только в режиме конструктора. Чтобы созданные отчеты правильно работали, необходимо четко представлять назначение каждого раздела. Перечислим типы разделов и укажем назначение каждого из них.

- **Заголовок отчета**. Выводится на печать только один раз в начале отчета. В заголовок включается информация, обычно помещаемая на обложке: название отчета и дата. Заголовок отчета печатается перед верхним колонтитулом.
- **Верхний колонтитул**. Печатается вверху каждой страницы. Верхний колонтитул используется в тех случаях, когда нужно, чтобы название отчета повторялось на каждой странице.
- **Заголовок группы**. Размещается перед каждой новой группой записей. Используется для печати названия группы. Например, если отчет сгруппирован по зданиям, в заголовках групп можно указать их адрес.

• **Область данных**. Этот раздел печатается один раз для каждой строки данных из источника записей. В нем размещаются элементы управления, составляющие основное содержание отчета.

• **Примечание группы**. Печатается в конце каждой группы записей. Примечание группы можно использовать для печати сводной информации по группе.

• **Нижний колонтитул**. Печатается внизу каждой страницы. Используется для нумерации страниц и для печати постраничной информации.

• **Примечание отчета**. Печатается один раз в конце отчета. Примечание отчета можно использовать для печати итогов и другой сводной информации по всему отчету.

На вкладке **Создание** ленты MS Access 2010 в разделе **Отчеты** найдите конструктор отчетов. После его запуска на ленте появятся четыре дополнитель-ные вкладки: **Конструктор**, **Упорядочить**, **Формат** и **Параметры страницы**, а в центре экрана – окно конструктора отчетов. Новый пустой отчет содержит три раздела: верхний и нижний колонтитулы, между которыми находится область данных. Вы можете изменить размер любого раздела. Линейки с сантиметровыми делениями по верхнему и левому краям отчета помогают расположить данные на странице.

Если линейки отсутствуют, то для их появления на экране выберите вкладку ленты **Упорядочить**, в разделе **Размер** и порядок щелкните значок **Размер** или интервал. Появится меню. Найдите в нем в разделе **Сетка** пункт **Линейка**.

Верхний и нижний колонтитулы будут напечатаны соответственно вверху и внизу каждой страницы. Их

можно убрать совсем с помощью пункта меню **Обработка событий**..., которое появится на экране, если сделать щелчок правой кнопкой мыши в любом месте отчета. Выберите в нем пункт **Колонтитулы страницы**. Расположенный чуть ниже пункт **Заголовок/примечание отчета** даст возможность создать заголовок, который будет напечатан только в начале отчета на первой странице.

В рабочей области могут быть расположены четыре окна:

- окно конструктора отчета;
- окно **Обработка событий** вызывается щелчком правой кнопкой мыши в любом месте окна конструктора отчета;
- окно **Список полей** вызывается при помощи значка **Добавить поля** из раздела **Сервис** вкладки **Конструктор**;
- **Окно свойств** появляется на экране после выбора значка **Страница свойств** из раздела **Сервис** вкладки **Конструктор**.

Создание запроса

Конструктор отчетов практически всегда использует в своей работе данные базового запроса. При создании запроса нам понадобится информация из запроса **Квартиры** и таблиц **Улица** и **ВладельцыКвартир**. Запрос **Квартиры** содержит данные по техническим характеристикам всех квартир, находящихся в отдельном конкретном здании, которое отображается в форме **Здание**.

1. Выберите вкладку **Создание** ленты MS Access 2010.

2. Сделайте щелчок левой кнопкой мыши по значку **Конструктор запросов**, который находится в разделе **Макросы и код**. Появится окно конструктора запросов и окно с названием **Добавление таблицы**.

3. Первая вкладка этого окна содержит список таблиц текущей базы данных. Вторая – список запросов. Выберите в ней запрос **Квартиры** и нажмите кнопку **Добавить**. Таким же образом включите в новый запрос таблицы **Улица** и **ВладельцыКвартир**, расположенные на первой вкладке. Закройте окно **Добавление таблицы**.

4. Параметры объединения таблицы **Улица** и запроса **Квартиры** MS Access установил сам. В запрос будут включены только те записи из запроса **Квартиры** и таблицы **Улица**, в которых связанные поля совпадают. Надо объединить **Квартиры** и **ВладельцыКвартир**.

5. Поместите указатель мыши над полем **НомерУлицы** запроса **Квартиры**. Нажмите левую кнопку мыши и, не отпуская ее, перетащите появившийся значок поля на поле **НомерУлицы** таблицы **ВладельцыКвартир**. Проделайте аналогичную операцию с полями **НомерДома** и **НомерКвартиры** этого же запроса и таблицы. Щелчок правой кнопкой мыши по линии, соединяющей поля из двух таблиц, приведет к появлению окна **Параметры объединения**. Проверьте точность своих действий по перетаскиванию и оставьте переключатель, расположенный в нижней части окна, в первом положении.

6. Расположите последовательно все поля, которые хотите включить в запрос из таблиц **Улица**, **ВладельцыКвартир** и запроса **Квартиры**, в первой строчке нижней части окна конструктора запросов.

Воспользуйтесь инструментом **Вставить столбцы**. Его можно найти в разделе **Настройка запроса** вкладки **Конструктор**.

7. В запрос должны попасть данные только по одной конкретной квартире. Ее номер отображен в объекте **НомерКвартиры** формы **Квартиры**. Для этого добавим условие отбора по полю **НомерКвартиры**: [Forms]![Квартиры]![НомерКвартиры]

8. Установите порядок сортировки записей, попавших в запрос. Они должны отображаться в документе по возрастанию порядковых номеров проживающих.

9. Сохраните созданный запрос под именем **Документ**. Не забудьте, что корректно он будет запускаться только из формы **Квартиры**, которая, в свою очередь, не может быть запущена без формы **Здание**.

10. Сделайте щелчок правой кнопкой мыши в свободном месте окна конструктора запросов. Появится контекстное меню. Первым пунктом в нем должен быть пункт **Режим SQL**. Щелкните по нему правой кнопкой мыши. Появится текст запроса на языке SQL, сгенерированный конструктором запросов. Обратите внимание на сложность текста запроса и на то, с какой легкостью мы сформировали его с помощью конструктора запросов, не зная ни одной конструкции и ни одного служебного слова языка SQL.

Добавление элементов в отчет

В отчетах MS Access 2010 применяются три типа элементов управления:

- *Присоединенные элементы управления*, связанные с полем источника данных для отчета. Это может

быть поле таблицы, запрос и даже значение другого элемента управления. Все присоединенные элементы получают связанные с ними метки. Значение метки представляет собой значение свойства **Подпись**, относящегося к вкладке **Макет**. Метку всегда можно удалить.

- *Свободные элементы управления* не зависят от источника данных отчета. Прямоугольники и линии – для оформления внешнего вида, а OLE – для добавления графики в отчет. Не все свободные элементы имеют метки.
- *Вычисляемые элементы управления* используют в качестве источника данных в выражении. В выражениях могут использоваться как поля таблиц, так и свободные элементы.

Для создания отчета выполните следующие действия:

1. Откройте окно свойств отчета. Для этого перейдите на вкладку **Конструктор** и сделайте щелчок по значку **Страница свойств**, расположенному в разделе **Сервис**.
2. В окне свойств перейдите на вкладку **Данные**.
3. Выберите свойство **Источник записей**. Раскройте поле со списком и найдите в нем запрос **Документ**.
4. Сделайте щелчок правой кнопкой мыши в любом месте окна конструктора отчетов. Появится меню **Обработка событий**. Выберите пункт **Заголовок/примечание отчета**.
5. В этом же меню щелкните пункты **Сетка** и **Линейка**.
6. Перейдите на вкладку **Конструктор** для добавления в отчет полей запроса **Документ**. Их список откроется в окне **Список полей** после щелчка по значку **Добавить поля**.

7. Поместите надпись в самом верху заголовка отчета и введите в нее текст **Справка**. Выделите надпись щелчком мыши. Вокруг нее появятся маркеры.

8. Перейдите на вкладку **Формат**. В разделе **Шрифт** установите шрифт Arial Cyr размером 10 пунктов. Подчеркните текст и сделайте его выделенным. Здесь же можно назначить цвет текста и цвет фона.

9. Перейдите на вкладку **Упорядочить**. В разделе **Размер и порядок** щелкните по значку **Размер или интервал**. Откроется меню.

10. Выберите в нем пункт **по размеру данных**. Размер элемента управления будет настроен в соответствии с назначенным шрифтом.

11. В этом же меню выберите пункт **по узлам сетки**. Он предназначен для изменения места расположения одного или нескольких элементов путем выравнивания по узлам сетки.

12. Выполните аналогичные действия по размещению второй строки заголовка отчета **Данные по квартире, расположенной по адресу**.

13. Перетащите поля **НомерПроживающего**, **НомерЛицСчета**, **НомерДома** и **НомерКвартиры** из списка полей (вкладка **Конструктор**, значок **Добавить поля**) в заголовок отчета.

14. Выполните форматирование. Для этого выберите нужные элементы управления, щелкая по ним левой кнопкой мыши при нажатой клавише **<Shift>**. Откройте вкладку ленты **Упорядочить**. Выберите значок **Размер** или интервал. Откроется меню. В разделах **Размер** и **Интервал** вы найдете 12 пунктов меню для форматирования.

15. Перетащите поля квартиросъемщика в область данных. Удалите подписи, которые к ним сгенерирует Access, и выполните форматирование.

Включение в отчет даты, времени и номеров страниц

Любой документ, выдаваемый организацией, обязательно должен иметь в своем составе дату, а в некоторых случаях и время выдачи. Это единственный показатель актуальности сведений, которые он содержит. Большие отчеты должны иметь пронумерованные страницы.

Для добавления номера страницы в область верхнего или нижнего колонтитулов выполните следующие действия:

1. Откройте отчет в режиме конструктора.
2. На вкладке ленты **Конструктор** выберите значок **Номера страниц**. Он находится в разделе **Колонтитулы**. Откроется диалоговое окно **Номера страниц**.
3. Выберите формат, расположение и выравнивание для номеров страниц.
4. Снимите флажок **Отображать номер на первой странице**, если номер на первой странице не нужен.
5. Нажмите кнопку **OK**. Номера страниц будут добавлены в отчет.

Добавьте в заголовок отчета надпись: **по состоянию на**, после которой поместите дату создания отчета. Для этого:

1. Откройте отчет в режиме конструктора.

2. На вкладке **Конструктор** ленты MS Access 2010 выберите кнопку **Поле**. Она находится в разделе **Элементы управления**. Расположите курсор в области заголовка отчета.

3. Сделайте щелчок левой кнопкой мыши и, удерживая ее, добейтесь требуемого размера поля. Кнопку отпустите.

4. Откройте окно свойств. Для этого сделайте щелчок по кнопке **Страница свойств** вкладки **Конструктор**.

5. В окне свойств выберите вторую вкладку **Данные**.

6. Перейдите к первой строчке **Данные** и нажмите кнопку ... Появится окно построителя выражений.

7. В левом списке **Элементы выражений** выберите строчку **Общие выражения**, а в среднем **Категории выражений – Текущая дата и время**. В правом списке появится стандартная функция MS Access – **Now()**. Щелкните по кнопке **OK**. Эта функция со знаком = появится в поле значений свойства **Данные** создаваемого элемента.

8. Вместо названия надписи Поле49: напишите: **по состоянию на**. Для этого щелчком мыши выделите надпись и внесите изменения прямо на месте.

9. Перейдите на вкладку **Формат**. В разделе **Шрифт** установите шрифт Arial Cyr размером 10 пунктов или любой другой. Подчеркните текст и сделайте его выделенным. Здесь же можно назначить цвет текста и цвет фона.

10. Перейдите на вкладку **Упорядочить**. В разделе **Размер и порядок** щелкните по значку **Размер или интервал**. Откроется меню. Выберите в нем пункт

по размеру данных. Размер элемента управления будет настроен в соответствии с назначенным шрифтом.

11. В этом же меню выберите пункт **по узлам сетки**. Он предназначен для изменения места расположения одного или нескольких элементов путем выравнивания по узлам сетки. Выполните окончательное форматирование.

Добавление кнопки в форму для запуска отчета

Отчет построен на базе запроса **Документ**, который берет данные из формы **Квартиры**, а та в свою очередь запускается на основе данных формы **Здание**, так как два параметра адреса – улицу и номер дома берет именно из этой формы.

Для запуска отчета добавьте кнопку **Документ** в форму **Квартиры** и запускайте отчет только из этой формы.

Текст процедуры VBA, сгенерированный мастером кнопок, для запуска этого отчета:

```
Private Sub Кнопка4 0 Click()
    On Error GoTo Err Кнопка4 0 Click Dim stDocName
As String
  stDocName = «Документ» DoCmd.OpenReport
  stDocName, acPreview
Exit Кнопка40 Click:
  Exit Sub
Err Кнопка4 0 Click:
  MsgBox Err.Description Resume Exit Кнопка4 0 Click
End Sub
```

Имя кнопки – Кнопка4 0 сгенерировано автоматически. Кнопка, скорее всего, будет иметь другой

порядковый номер. Имя отчета – **Документ**. Отчет запускается в режиме предварительного просмотра – acPreview. В случае возникновения ошибки при выполнении отчета о нем сообщит Microsoft Access, а не VBA – MsgBox Err.Description.

Вывод отчета MS Access на печать

При предварительном просмотре отчета в режиме целой страницы Microsoft Access выводит его так, как сделал бы принтер. По умолчанию стандартные поля с трех сторон составляют по 1,5 см, а слева – 1 см. Есть возможность изменить параметры печати отчета. Для этого:

1. Откройте отчет в режиме конструктора.
2. Перейдите на последнюю вкладку ленты MS Access 2010 – **Параметры страницы** и сделайте щелчок по значку **Параметры страницы**.
3. Откроется диалоговое окно **Параметры страницы**. Для того чтобы свести к минимуму расход бумаги при печати рабочих отчетов, установите флажок **Печатать только данные**. Верхний и нижний колонтитулы, а также заголовок отчета и примечание отчета печататься не будут.
4. Задайте в окне **Параметры страницы** поля страницы, выводимой на принтер.

Для того чтобы распечатать отчет, находясь в режиме предварительного просмотра, сделайте щелчок правой кнопкой мыши в любом месте отчета. Появится окно **Печать** вашей операционной системы. Можете распечатать отчет целиком или только необходимые страницы. Также имеется возможность указать количе-

ство экземпляров отчета и отправить отчет в файл для последующей распечатки.

Изменение отчета

Чтобы работать с Microsoft Access более эффективно, необходимо научиться создавать простые выражения с использованием функций и операций. Выражения применяются, как правило, для проверки условий или для арифметических вычислений.

Выражение создается с помощью комбинации идентификаторов, операторов и значений, обеспечивающих получение необходимого результата. Выражения можно создавать самостоятельно или с помощью построителя выражений.

Создайте в отчете вычисляемое поле **Адрес**. Для этого в окне свойств найдите свойство **Данные** и щелчком по кнопке ... запустите построитель выражений.

Окно построителя выражений состоит из четырех разделов:

- **Поле выражения**. В верхней части окна построителя расположено поле, в котором создается выражение. Допускается непосредственный ввод части выражения в поле выражения.

Ниже находится раздел, предназначенный для создания элементов выражения и их последующей вставки в поле выражения. Он содержит 3 списка.

- В левом списке **Элементы выражений** выводятся папки, содержащие таблицы, запросы, формы, объекты базы данных, встроенные и определенные пользователем функции, константы, операторы и общие выражения.

• В среднем списке **Категории выражений** задаются определенные элементы или типы элементов для папки, заданной в левом поле. Например, если выбрать в левом списке **Встроенные функции**, то в среднем списке появится перечень всех типов функций Microsoft Access.

• В правом списке **Значения выражений** выводится перечень значений (если они существуют) для элементов, заданных в левом и среднем списках. Например, если выбрать в левом списке **Встроенные функции** и тип функции в среднем, то в правом списке будет выведен перечень всех встроенных функций выбранного типа.

При вставке идентификатора в выражение построитель вставляет только те его части, которые требуются в текущем контексте. Например, при запуске построителя выражений из окна свойств формы **Здание** и вставке идентификатора для свойства **Вывод на экран** (Visible) будет вставлено только имя свойства – Visible. При использовании данного выражения вне контекста формы необходимо включать полный идентификатор: Forms![Здание].Visible.

Нажмите клавишу =. Выражение, создаваемое при помощи построителя, как правило, начинается со знака присваивания.

Далее используем встроенную функцию: IIF(expr; truepart; falsepart), которая в зависимости от значения логического выражения назначит в нашем случае значением поля **Адрес** либо первую цепочку (название + признак + дом + квартира), либо вторую (признак + название + дом + квартира).

В таблицах Microsoft Access логическое поле имеет тип Числовой, значению True соответствует число -1

(минус один), а значению False – число 0 (ноль). В таблицах же Microsoft SQL Server логическое поле имеет тип Bit (True один, False ноль).

Чтобы избежать проблем при переводе приложения Microsoft Access на платформу SQL Server, не привязывайтесь к конкретным значениям полей таблиц, а используйте константы True и False, которые построитель выражений Microsoft Access сразу же переведет на русский язык (Истина и Ложь) в русскоязычной версии Microsoft Access 2010.

Для окончательного построения выражения потребуется текстовая функция: Trim(stringexpr), убирающая концевые пробелы, и функция преобразования Str(number), преобразующая число в символы.

Обратите внимание на правила записи выражений. Имена полей должны быть заключены в квадратные скобки, а текстовые константы – в апострофы или кавычки. Параметры функций – в круглые скобки. Если параметров несколько, то они отделяются друг от друга точкой с запятой. В выражении общее число открывающих круглых скобок обязательно должно быть равно числу закрывающих.

Рассмотрим процесс создания выражения без помощи построителя.

Введите комбинацию идентификаторов, операторов и значений в элемент управления, в котором требуется получить результат. Например, следующее выражение увеличивает значение элемента управления **Стоимость** формы **Здание** на 20%: = [Forms]![Здание]![Стоимостьt]*1,2

Дополнительные возможности Microsoft Access

Сжатие базы данных

При удалении объектов из базы данных Microsoft Access размер файла этой базы данных не меняется. При удалении записи из таблицы место, которое она занимала в базе, также автоматически не освобождается и не используется для хранения новой записи. Чтобы уменьшить размер файла базы данных и увеличить ее быстродействие, воспользуйтесь служебной программой. **Средства сжатия и восстановления**, начиная с MS Access 2002, усовершенствованы и интегрированы в единый процесс, что делает их более защищенными и эффективными.

Для запуска служебной программы сжатия базы данных:

1. Запустите MS Access 2010.
2. Откройте базу данных. Для этого на вкладке **Файл** стартового окна MS Access найдите нужную базу, если она там есть, или воспользуйтесь пунктом **Открыть**. Появится окно базы данных.
3. Щелкните заголовок вкладки **Файл**. В появившемся меню система сделает активным пункт **Сведения**, который содержит несколько значков.
4. Выберите среди них **Сжать и восстановить**.

После окончания процесса сжатия активизируется окно базы данных.

Имеет смысл включить эту операцию в настройки MS Access. Тогда, закрывая базу данных, вы автоматически запустите утилиту сжатия. Время ее работы при таком размере составляет 2-3 секунды. Для этого:

1. На вкладке **Файл** окна MS Access 2010 выберите пункт **Параметры**.

2. Появится окно **Параметры Access**. Выберите в нем пункт **Текущая база данных**.

3. В разделе **Параметры приложений** поставьте флажок **Сжимать при закрытии**.

4. Для завершения процесса настройки нажмите кнопку **ОК**.

Преобразование базы данных в формат MS Access 2007/2010

Функция преобразования базы данных в новый формат появилась впервые только в Microsoft Access 2000. Более ранние версии не позволяли выполнять такие преобразования, а разработчикам настоятельно рекомендовалось иметь в своем распоряжении несколько версий этого продукта и вести разработку на той, которая есть у заказчика.

Для преобразования файла базы данных предыдущих версий (2000 и 2002–2003) к виду Microsoft Access 2007/2010:

1. Запустите MS Access 2010.

2. Откройте базу данных. Для этого на вкладке **Файл** стартового окна MS Access найдите нужную базу, если она там есть, или воспользуйтесь пунктом **Открыть**.

3. Появится окно **Улучшение базы данных**. Щелкните по кнопке **Да**.

4. Появится диалоговое окно Сохранение, в котором надо указать название файла базы данных и папку, в которую будет помещена приведенная к формату MS Access 2007/2010 база.

Преобразование базы данных формата MS Access 2007/2010 к предыдущим версиям (2000 и 2002–2003) возможно, но корректно воспользоваться им можно только в случае, если при разработке не использовались новые возможности последней версии.

Для преобразования выполните следующие действия:

1. Откройте базу данных. Для этого на вкладке **Файл** стартового окна MS Access найдите нужную базу, если она там есть, или воспользуйтесь пунктом **Открыть**.

2. Щелкните на заголовке вкладки **Файл**. В появившемся меню система сделает активным пункт **Сведения**, который содержит несколько значков.

3. Выберите пункт меню **Доступ**, а в изменившейся правой части окна тип формата базы данных Access 2002/2003 или Access 2000.

4. Появится окно **Сохранение** операционной системы вашего компьютера. Выберите папку и укажите имя файла, в котором требуется сохранить преобразованную базу данных, и нажмите кнопку **Сохранить**.

Анализ быстродействия базы данных

Для оптимизации быстродействия базы данных Microsoft Access 2010 применяется специальная служебная программа – анализатор быстродействия.

Для ее запуска:

1. Откройте базу данных.

2. Перейдите в окне MS Access 2010 на вкладку **Работа с базами данных**.

3. В разделе Анализ выберите значок **Анализ**

быстродействия. Появится одноименное диалоговое окно.

4. Выберите вкладку, соответствующую типу объекта базы данных.

Анализатор быстродействия выдает три типа рекомендаций по оптимизации производительности: советы, предложения и мысли. При выделении элемента в списке **Результаты анализа** сведения о предлагаемом решении выводятся в отдельном окне **Анализ быстродействия**. Операции оптимизации, как правило, подразумевают определенные компромиссы, которые следует иметь в виду, приступая к оптимизации. Для получения дополнительных сведений о рекомендации выберите ее в списке и просмотрите информацию в области **Примечания**.

Microsoft Access 2010 может автоматически выполнять рекомендации типа **совет и предложение**. Рекомендации типа **мысль** выполняются вручную. Очень часто они кажутся примитивными. Например, анализатор советует заменить тип поля.

Сохранение базы данных в виде accde-файла

Если база данных содержит программы Microsoft Visual Basic, то при сохранении ее в виде accde-файла будут скомпилированы все модули, удалены все изменяемые исходные программы, а конечная база данных будет сжата.

Программы Visual Basic будут по-прежнему выполняться, но их нельзя будет просматривать или изменять, благодаря чему уменьшится размер базы данных. Кроме того, будет оптимизировано использование памяти, что повысит быстродействие.

Сохранение базы данных как accde-файла делает невозможным выполнение следующих действий:

- просмотр, изменение или создание форм, отчетов или модулей в режиме конструктора;
- добавление, удаление или изменение ссылок на библиотеки объектов или базы данных;
- изменение программы с помощью свойств или методов Microsoft Access или модели объектов VBA (accde-файл не содержит текстов исходных программ);
- импорт и экспорт форм, отчетов или модулей.

Любые таблицы, запросы, страницы доступа к данным или макросы в базах данных, являющихся accde-файлами, могут быть импортированы в другую базу данных MS Access.

Обязательно сохраните копию исходной базы данных MS Access 2010. В базе данных MS Access, сохраненной как accde-файл, нельзя изменять структуру форм, отчетов или модулей. Чтобы изменить структуру этих объектов, следует сделать это в исходной базе данных, а затем снова сохранить ее как accde-файл. При сохранении в виде accde-файла базы данных, содержащей таблицы, возникают сложности согласования различных версий данных в случае последующего изменения структуры форм, отчетов или модулей.

Для создания Accde-файла:

1. Откройте базу данных, которую требуется сохранить в виде файла accde. Для этого на вкладке **Файл** стартового окна MS Access найдите нужную базу, если она там есть, или воспользуйтесь пунктом **Открыть**.

2. Щелкните на заголовке вкладки **Файл**. В появившемся меню система сделает активным пункт **Сведения**, который содержит несколько значков.

3. Выберите пункт меню **Доступ**, а в изменившейся правой части окна пункт **Создать ACCDE**.

4. Появится окно **Сохранение** операционной системы вашего компьютера. Выберите папку и укажите имя файла, в котором требуется сохранить преобразованную базу данных, и нажмите кнопку **Сохранить**.

Анализ данных в Microsoft Excel

Практически любой объект базы данных MS Access можно передать в MS Excel и там проанализировать. В этом случае можно использовать всю мощь электронных таблиц от богатейшего набора стандартных функций до построения прекрасных диаграмм. Более всего это средство подходит для работы с запросами.

Рассмотрим пример. Отдел социальной защиты администрации Центрального района прислал запрос с просьбой предоставить списки всех граждан, стоящих на учете, старше 55 лет.

Для создания такой выборки:

1. В области навигации базы данных откройте раздел **Запросы**.

2. Щелчком правой кнопки мыши выделите объект, который следует передать в Excel.

3. В открывшемся контекстном меню выберите пункт **Экспорт**.

4. Появится еще одно меню со списком назначения. Найдите в нем Excel.

5. Откроется диалоговое окно **Экспорт – Электронная таблица Excel**.

6. В разделе параметров экспорта поставьте флажки **Экспортировать данные с макетом и**

форматированием и **Открыть целевой файл после завершения операции экспорта**. Нажмите кнопку **ОК**.

7. Автоматически запустится MS Excel 2010, в виде таблицы будут отображены данные по выбранному объекту.

Повышение быстродействия Microsoft Access

Соблюдение следующих правил поможет повысить производительность MS Access 2010.

- При работе с базами данных, которые не применяются другими пользователями, устанавливайте Microsoft Access и все свои базы данных на собственном жестком диске, а не на сетевом сервере.
- Чтобы быть единственным пользователем базы данных, откройте ее в монопольном режиме. Для этого в диалоговом окне **Открытие файла базы данных** нажмите стрелку рядом с кнопкой **Открыть** и выберите в списке вариант **Монопольно**.
- Чтобы освободить память, закройте неиспользуемые приложения.
- Увеличьте оперативную память компьютера.
- Регулярно удаляйте ненужные файлы, выполняйте сжатие базы данных, а после этого проводите дефрагментацию диска с помощью служебной программы дефрагментации. Для запуска этой программы нажмите кнопку **Пуск** в Windows, последовательно выберите пункты **Программы / Стандартные / Служебные / Дефрагментация диска**.
- Замените рисунок или фоновый узор, выбранный для рабочего стола Windows, на однородный экран.

• Не используйте программы сжатия диска или переместите базы данных на несжатый диск.

Если с вашим программным обеспечением работает несколько компьютеров, соединенных в локальную вычислительную сеть, то большинство рекомендаций не даст желаемого результата. Необходимо разделить данные и приложение.

Разделение данных и приложения

Изначально MS Access 2010 хранит все объекты данных и элементы интерфейса в одном файле accdb. При разработке однопользовательского приложения, которое размещается целиком на локальном компьютере, это очень удобно. Если перед вами стоит задача обеспечить работу нескольких пользователей с программным комплексом, то поместите этот файл на один компьютер и обеспечьте доступ к нему всех работающих. Каждый раз, когда пользователю понадобится какой-нибудь объект, например форма или отчет, MS Access будет пересылать по локальной вычислительной сети весь файл accdb. В реальных условиях действующего предприятия – это неоправданное увеличение нагрузки на сеть, снизить которую довольно просто. Разделим базу данных на две части. Для этого:

1. Откройте базу данных, которую требуется разделить на две составляющие.
2. На вкладке **Работа с базами данных** в разделе **Переместить данные** выберите значок **База данных Access**.
3. Появится окно мастера, сообщающее о том, что операция может потребовать много времени и

5. Правильно ответьте на запрос системы для подтверждения удаления.

6. Перейдите на вкладку **Внешние данные**. В разделе **Импортировать и связать** выберите значок Access.

7. Появится окно **Внешние данные – база данных Access**.

8. Укажите полный путь к файлу базы данных сервера. Лучше всего воспользоваться для этого кнопкой Обзор.

9. В разделе окна с названием **Укажите когда и где сохранять данные в текущей базе** данных отметьте переключатель **Создать связанную таблицу для связи с источником данных**. Изменения, сделанные на сервере, будут отображаться у клиента, и наоборот.

10. Появится окно **Связь с таблицами**. Щелкните по кнопке **Выделить** все и нажмите кнопку **ОК**.

11. Все таблицы появятся в области переходов. Это связанные таблицы с правильными ссылками на сервер.

Если в области навигации у значка таблицы отсутствует стрелочка в левом верхнем углу, то это означает, что вы просто импортировали ее на рабочую станцию. Все изменения отныне останутся только у клиента. Выполните набор инструкций 1–11 заново.

Достоинства разделения данных и приложения MS Access 2010 в развернутом виде:

- пользовательский интерфейс работает значительно быстрее;
- у разработчика появляется возможность создавать временные таблицы-выборки на рабочей станции, не беспокоясь о конфликтах и блокировках;

• значительно упрощается установка новых версий приложения;

• файл с данными, расположенный на сервере, потерять теперь значительно сложнее.

Просмотр и изменение свойств документа MS Access 2010

Свойства документа – это метаданные (данные, которые описывают другие данные). Например, записи в таблице являются данными, а их число может служить примером метаданных. К свойствам документа MS Access относятся заголовок, имя автора, тема и ключевые слова, определяющие раздел или содержимое документа. Указав соответствующие значения для полей свойств документа, эти документы можно будет упорядочить и легко находить в дальнейшем.

Свойства документа делятся на следующие типы:

• стандартные (автор, название, тема и др.);

• автоматически обновляемые свойства (размер файла, дата создания или последнего изменения файла) и статистические сведения. По этим свойствам можно, например, найти все файлы, созданные после конкретной даты;

• пользовательские свойства. Имя пользовательского свойства можно выбрать из предлагаемого списка или определить самостоятельно;

• свойства для организации. Если в организации настроена область сведений о документе, то документы пользователя могут иметь определенные для его организации свойства.

Для просмотра и изменения свойств текущего документа сделайте следующее:

1. Откройте базу данных.
2. Нажмите кнопку **Файл**. Появится меню. Выберите в нем пункт **Сведения**.
3. В правом верхнем углу окна найдите ссылку **Просмотр** и изменение свойств базы данных.
4. Появится диалоговое окно **Свойства**. В нем пять вкладок.
5. Откройте вкладки для выбора свойств, которые требуется просмотреть или изменить.

Для создания пользовательских свойств документа перейдите на вкладку **Прочие**. Этим свойствам можно назначать текстовые, числовые значения или значения даты/времени, а также значения **да** или **нет**. Имя пользовательского свойства можно выбрать из предлагаемого списка или определить самостоятельно.

Импортирование объекта в свою базу данных

Любой объект (таблица, форма, отчет, запрос, модуль и т. д.) может быть импортирован во внешнюю базу данных. Для этого:

1. Откройте базу-источник в MS Access 2010.
2. Найдите нужный объект в области навигации.
3. Откройте вкладку **Внешние данные**.
4. В разделе **Экспорт** выберите пункт **Access**. Откроется диалоговое окно **Экспорт – База данных Access.**
5. Щелкните по кнопке **Обзор**. Найдите в нем файл базы-назначения.

Основные сведения о Visual Basic for Applications

Visual Basic for Applications (VBA) – это мощный инструмент разработки приложений. Как и другие средства, например, MS Visual C++, MS Visual C#, MS VB, MS Visual FoxPro, Borland Delphi, он предоставляет разработчику возможность создать полностью законченные программные продукты. VBA встроен во все приложения Microsoft Office, AutoCAD, CorelDRAW и множество других. В MS Office 2010 используется версия VBA с номером 7.0. Изучив VBA, можно создавать свои программные комплексы не только в Microsoft Access. Необходимо лишь дополнительно освоить иерархию объектов выбранного приложения.

Среда Visual Basic for Applications

Создав базу данных Недвижимость, мы автоматически получили в свое распоряжение проект VBA с таким же именем.

На вкладке **Работа с базами данных** расположен инструмент Visual Basic, который позволяет перейти в среду VBA. Среда разработки VBA имеет меню, окна и другие элементы, использующиеся при работе с проектом. Проект VBA носит имя базы данных и содержит формы, модули класса и модули кода.

Четыре основных окна среды VBA: окно проекта (Project); окно свойств (Properties); окно кода; окно отладки (Immediate).

В *окне проекта* можно двойным щелчком выбрать требуемый объект. В окне кода появится его текст, а в окне Properties – свойства объекта.

Окно свойств предназначено для задания свойств объектов. Например, можно указать в окне свойств формы фон, заголовок, номер темы в файле справки и т. д. Всего свойств несколько десятков. На первой вкладке свойства расположены в алфавитном порядке (Alphabetic). Выбор второй вкладки этого окна (Categorized) отсортирует свойства по категориям.

Окно кода предназначено для ввода текста процедур VBA. Код внутри модуля сгруппирован в отдельные разделы – процедуры. Редактор VBA может отображать текст в двух режимах: просмотр всего модуля или просмотр отдельной процедуры. Для переключения этих режимов применяются кнопки, расположенные в левом нижнем углу окна редактора кода.

В режиме просмотра всего модуля процедуры отделяются одна от другой горизонтальной чертой. Если этот разделитель отсутствует, то вернуть его можно следующим способом:

1. В меню VBA выберите пункт **Tools**. Появится всплывающее меню.
2. Щелкните по команде **Options**. Откроется диалоговое окно.
3. Установите флажок **Procedure separator**.

Два раскрывающихся списка General и StartMainMenu предназначены для выбора объекта и его составляющих соответственно. Если в VBA идет работа с кодом обработки событий формы, то левый список содержит объект формы, а правый – перечень событий, допустимых для выбранного объекта.

Выбор объекта и события добавляет в редакторе кода первую и последнюю строчки процедуры.

Редактор кода VBA автоматически завершает написание составных частей строки текста (операторов, параметров, свойств).

Если подсказки для выбора свойств мешают, можно их отключить:

1. В меню VBA выберите пункт **Tools**. Появится всплывающее меню.
2. Щелкните по команде **Options**. Откроется диалоговое окно.
3. Выключите флажок **Auto Data Tips**.

Редактор кода выполняет автоматическую проверку синтаксиса введенной строки после нажатия клавиши **<Enter>**. Если строка выделяется красным цветом – ищите ошибку.

Еще одна возможность редактора кода VBA заключается в вызове нужного в данный момент раздела справки нажатием клавиши **<F1>**.

Переменные, типы данных и константы

Переменная – это область памяти компьютера, имеющая имя. В Visual Basic for Applications переменные используются для хранения данных в оперативной памяти. После создания переменная указывает на одну и ту же область памяти до тех пор, пока не будет уничтожена. Разработчику не нужно знать, где находится эта область. Достаточно лишь сослаться на имя переменной. При выборе имени переменной следует руководствоваться правилом:

- имя должно начинаться с буквы;
- имя не может содержать пробел и точку, а также &, !, $, #;

- имя не должно быть длиннее 255 символов;
- не рекомендуется назначать имена, совпадающие с названием функций и методов VBA.

Перед началом работы с переменной ее нужно объявить. Сделать это можно при помощи операторов **Dim**, **Public**, **Private** и **Static**. В зависимости от используемых операторов и месте их появления в коде программы, переменной будет назначена область видимости и время жизни. В VBA существуют три области видимости переменной.

1. *Переменная уровня проекта.* Описывается с помощью инструкции **Public**. Также называется **открытой**. Является доступной для всех процедур проекта VBA.

Пример:

Public Account As Integer

Объявлена переменная с именем **Account** типа **Integer**. Ее значением может быть только целое число.

2. *Переменная уровня модуля.* Описывается с помощью инструкции **Dim** или **Private**. Эти инструкции равнозначны. Переменная будет доступной только в том модуле, в котором описана. Описание такой переменной необходимо разместить перед описанием всех процедур модуля.

Пример:

Private Фамилия As String

Dim Фамилия As String

Объявлена переменная с именем Фамилия типа String. Ее значением может быть только строка символов.

3. *Переменная уровня процедуры.* Описывается с помощью инструкции **Dim** или **Static**. Также называется **локальной**. Является доступной только в той

процедуре, в которой описана. Если используется конструкция **Static**, то переменная сохраняет свое значение, пока выполняются другие процедуры этого модуля. Если применяется конструкция **Dim**, то после завершения работы процедуры значение такой переменной всегда теряется.

Пример:

Static Period As Long

Dim Period As long

Объявлена переменная с именем **Period** типа **Long**. Ее значением может быть только длинное целое.

Увидеть значение переменной в тот или иной период времени можно в окне отладки, в окне VBA.

Для этого, находясь в главном окне MS Access 2010:

1. Выберите вкладку **Работа с базами данных**.
2. Щелкните по инструменту **Visual Basic**. Появится окно **VBA**.
3. Щелкните правой кнопкой мыши по команде **Debug** меню VBA. Выберите пункт **Debug**. Появится панель инструментов отладчика.
4. Щелкните по инструменту **Immediate Window** (Ctrl+G). В появившееся окно **Immediate** будут выводиться все значения, которые вы укажете после фразы **Debug.Print**.

Для обязательного объявления всех переменных в начало модуля надо поместить директиву: **Option Explicit**.

Использование этой директивы не допускает возможности работы с необъявленными переменными.

VBA дает возможность описывать и использовать переменные различных типов. Выбор типа переменной основан на требованиях разрабатываемого приложения.

В VBA имеются строки, числа, даты, объекты, логические значения, а также общий тип данных – **Variant**. Он применяется по умолчанию и может содержать данные любого типа, кроме строк фиксированной длины.

Переменные **Byte**, **Integer** и **Long** не могут иметь дробных значений. Переменные типа **Byte** не могут иметь отрицательных значений. Тип переменных **Currency** (денежный) предпочтительнее использовать в валютных операциях. Скорость работы с этим типом выше, чем с **Single** или **Double**.

Большинство типов переменных VBA находится в полном соответствии с типами полей базы данных MS Access 2010.

Основные типы данных в Visual Basic for Applications:

Категория	Тип данных	Размер	Диапазон
Числовые типы	Byte	1 байт	От 0 до 255
	Integer	2 байта	От -32 1 байт 768 до 32 767
	Long	4 байта	От 2 147 483 648 до 2 147 483 647
	Single	4 байта	Для отрицательных чисел: от -3.402823×10^{38} до -1.401298×10^{45} Для положительных чисел: от 1.401298×10^{5} до 3.402823×10^{38}
	Double	8 байтов	Для отрицательных чисел: от $-1.79769313486232 \times 10^{308}$ до $-4.94065645841247 \times 10^{-324}$ Для положительных чисел: от $4.94065645841247 \times 10^{324}$ до $1.79769313486232 \times 10^{308}$
	Currency	8 байтов	От -922 337 203 685 477.5808 до 922 337 203 685 477.5807

Категория	Тип данных	Размер	Диапазон
Логический тип	Boolean	2 байта	False или True. Если используется 0, то он интерпретируется как False. Любое другое значение – True. При обратном преобразовании True рассматривается как -1
Тип даты	Date	8 байтов	От 1 января 100 года до 31 декабря 9999 года
Тип объект	Object	4 байта	Ссылка на объект
Строковый тип	String	Длина строки плюс 10 байтов	Строки переменной и фиксированной длины

Константа – это область памяти компьютера, имеющая имя. В Visual Basic for Applications константы, как и переменные, используются для хранения данных в оперативной памяти. Различие в том, что изменить значение константы нельзя. В VBA можно как описать собственные константы, так и применять стандартные, которых в языке великое множество. При описании константы необходимо сразу присвоить ей значение, например:

Public Const Pi As Double=3.141592653589793.

Описав константу один раз, можно использовать ее в разных выражениях столько раз, сколько это требуется. Область видимости констант VBA регламентируется теми же правилами, что и у переменных.

В подавляющем своем большинстве стандартные константы Visual Basic for Applications – это числа, использующиеся в качестве аргументов функций или методов.

Рассмотрим функцию **MsgBox()**. Она работает с 21 константой.

Пример текста процедуры, в которой используется функция **MsgBox()**:

```
Public Sub StopMenu()
' Выход из программного комплекса
' Завершение работы MS Access
If MsgBox(«Вы действительно хотите закрыть базу»&Chr(13)&» и выйти _
из MS Access? «,VbInformation+VbOKCancel+VbDefaultButton1,» _
Выход»)=VbOK Then Application.Quit acQuitSaveAll
' Выходим из приложения
End If
End Sub
```

Текст, следующий за символом ' до конца строки, представляет собой **комментарий** и игнорируется транслятором.

Применение символов пробела и знака подчеркивания означает, что следующая строка кода является продолжением предыдущей.

В этом примере используется четыре константы:

VbOK	
vbInformation=64	используется значок информации
vbOKCancel=1	отображаются кнопки OK и Отмена
VbDefaulButton1=0	номер выделенной (OK) кнопки

Если нажата кнопка **OK**, то значение VbOK=1.

Текст вызова функции **MsgBox()** можно написать короче, указав в коде непосредственно цифры, без констант VBA:

```
If MsgBox(«Вы действительно хотите закрыть базу»&Chr(13)&» и выйти _
из MS Access?»,65,»Выход»)=1 Then Application.Quit acQuitSaveAll
```

'Выходим из приложения
End If

Стандартные функции и выражения VBA

Функции VBA предназначены для возврата значений в точку вызова. В VBA имеется более 150 различных стандартных функций.

Выражение VBA представляет собой комбинацию ключевых слов, операторов, переменных и констант. Результат выражения – число, строка или логическое значение.

Ключевое слово – это набор символов, распознаваемый как элемент языка программирования VBA. Например: Case, If, Sub, Step и т. д.

Оператор – это набор символов, предназначенный для объединения простых выражений в более сложные. Различают арифметические операторы, операторы сравнения и логические операторы.

Перед выражениями, определяющими вычисляемые элементы управления, всегда расположен знак присваивания =.

Массивы

В отличие от переменной, которая содержит один объект (строку, число, дату и т. д.), массив может содержать целый набор связанных между собой данных. Эти данные имеют общее имя. Обратиться к конкретному значению массива позволяет его номер. Например, можно применить переменную Зарплата для хранения величины заработной платы сотрудника:

Зарплата = 19800

Для хранения сведений о зарплате служащих всего отдела из 10 человек можно объявить десять переменных (Зарплата1, Зарплата2, Зарплата3, ...), а можно – один массив из десяти элементов:

```
Dim Зарплата(10) As Integer
```

Все значения в массиве должны обязательно принадлежать к одному типу данных. В скобках после имени указывают количество элементов. В этом примере объявлен **одномерный массив**. Нумерация элементов начинается с нуля (по умолчанию). Первый элемент массива может иметь номер, отличный от нуля. Если поместить в начало модуля оператор **Option Base 1**, то нумерация элементов массива начнется с единицы. В операторе **Dim** можно явно задать нижнюю границу. Это значение может быть любым при условии, что оно меньше верхней границы:

```
Dim Зарплата(1 To 10) As Integer
Зарплата(5) = 19800 'Зарплата пятого работника
```

Массив может быть не только одномерным. В VBA применяются двумерные, трехмерные массивы и массивы с большим числом размерностей. Допускается использование 60 индексов. В следующем примере объявлен двумерный массив с именем Матрица из ста целых чисел: десять строк и десять столбцов. Нумерация элементов начинается с единицы.

```
Dim Матрица(1 To 10, 1 To 10) As Integer
```

Часто размер массива заранее неизвестен. В этом случае в операторе Dim после имени массива ставят пустые скобки и описание типа, а количество элементов и диапазон не задаются. Перед использованием массива выполняется оператор **ReDim**.

При выполнении оператора **ReDim** все значения, хранящиеся в массиве, теряются. Однако VBA предоставляет возможность изменить размеры массива в сторону увеличения без потери данных. Для этого применяется ключевое слово **Preserve**. Стандартная функция **Ubound** возвращает число элементов массива.

Операторы Visual Basic for Applications

Оператор присваивания

Оператор присваивания назначает значение выражения переменной, константе или свойству объекта. Оператор присваивания обязательно содержит в своем составе знак равенства.

Для присваивания переменной ссылки на объект в операторе присваивания используется служебное слово **Set**. В следующем примере переменной **cBar** присваивается ссылка на новое головное меню программного комплекса, созданное методом **Add** (добавить) коллекции **CommandBars** (панели команд):

```
Set cbar=CommandBars.Add(Name:=«MainMenu», _
Position:=msoBarTop,MenuBar:=True,Temporary:=False)
‘ MainMenu – имя главного меню программного комплекса
‘ Значение константы msoBarTop=1. Меню появится вверху окна
‘ Если MenuBar:=True – будет создана строка меню
‘ Если MenuBar:=False – будет создана панель инструментов
‘ Temporary:=False – строка меню не временная и
‘ останется после завершения работы
```

Оператор With

Инструкция **With** освобождает разработчика от необходимости использовать большое количество повторений имени одного и того же объекта при работе с его свойствами и методами. Код становится более понятным.

С использованием **With**:

```
With oExcel.Selection.Borders(xlEdgeLeft)
   .LineStyle = xlContinuous
   .Weight = xlThin
   .ColorIndex = xlAutomatic
EndWith
```

Без использования With:

```
oExcel.Selection.Borders(xlEdgeLeft).LineStyle =
xlContinuous
oExcel.Selection.Borders(xlEdgeLeft).Weight = xlThin
oExcel.Selection.Borders(xlEdgeLeft).ColorIndex =
xlAutomatic
```

Управление выполнением программы

Поведение программы при выполнении изменяется в зависимости от входных данных, внешних условий (клавиатура, мышь и др.) и аварийных ситуаций.

VBA предоставляет в распоряжение разработчика три инструкции, позволяющие управлять ходом выполнения программы. Они носят название **управляющих конструкций ветвления** и дают возможность проверить некоторое условие и в зависимости от результата проверки выполнить ту или иную группу операторов.

Инструкция **If. . .Then** вычисляет условие, и если оно равно **True**, то выполняется оператор, следующий за ключевым словом **Then**. Синтаксис оператора **If**

допускает запись как в одну, так и в несколько строк. Для выполнения одного оператора, следующего за конструкцией **If...Then**, лучше использовать запись в одну строку:

```
If Количество=1 Then Новый=Новый+1
```

Хотя можно применить и другой вариант. Читать такой код легче, но длина его больше:

```
If Количество=1 Then
Новый=Новый +1
EndIf
```

Инструкция **If...Then...Else** выполняет блок операторов, следующий за ключевым словом **Then**, если значение условия после **If** равно **True**. В противном случае выполняется блок операторов, стоящих после ключевого слова **Else**.

Инструкция **Select Case** имеет определенные преимущества перед несколькими операторами **If...Then...Else**. Она облегчает чтение и сопровождение кода. Оператор **Select Case** вычисляет выражение, расположенное вверху всей структуры. Результат вычисления сравнивается с несколькими значениями, и если совпадет с одним из них, то выполняется соответствующий блок операторов.

Если выражению соответствуют значения более чем одного оператора **Case**, то выполняется только первый блок операторов.

Если выражению не соответствует ни одно из значений оператора **Case**, то выполняется блок операторов, стоящих после **Else**.

Операторы цикла

Операторы цикла предназначены для повторного выполнения одной и более строк кода. В VBA имеется

богатый выбор средств организации циклов, которые можно разделить на две основные группы:

1. циклы с перечислением **For...Next**;
2. циклы с условием **Do...Loop**.

Оператор **For...Next** обеспечивает выполнение группы инструкций указанное число раз, пока переменная цикла изменяется от начального значения до конечного с заданным шагом. Если шаг не указан, то он считается равным единице.

Например, трехмерный массив заполняется случайными числами:

```
Option Base 1
Dim SampleArray(10,10,10) As Single
Dim i As Integer, j As Integer, k As Integer
Randomize
For i=1 To 10
For j=1 To 10
For k=1 To 10
SampleArray(i,j,k) = Rnd()
Next k Next j Next i
```

Эту же задачу можно решить, применив вместо трех вложенных циклов со счетчиками всего один **For Each...Next**:

```
Dim SampleArray(10,10,10) As Single
Dim ijk As Variant
Randomize
For Each ijk In SampleArray
‘ Для каждого элемента массива
ijk = Rnd ()
Next
```

В следующем примере вычисляется сумма четных элементов одномерного массива при помощи

For...To...Step...Next:

```
Dim DemoArray(20) As Single
Dim i As Integer, Summa As Single
‘ Здесь массив должен быть заполнен
Summa = 0
For i=2 To UBound(DemoArray) Step 2
Summa=Summa+DemoArray(i)
Next
```

Как правило, выполнение цикла завершается, когда достигнуто условие прекращения его работы. Однако в некоторых случаях необходимо прекратить выполнение цикла досрочно, избежав выполнения лишних операторов. Например, если главное меню программного комплекса обнаружено в коллекции **CommandBars** (панели команд), то дальнейший перебор следует прервать. Для этого применяется связка ключевых слов **Exit For**:

```
For Each cbar In CommandBars
If cbar.Name=«MainMenu» Then
‘ Главное меню с именем MainMenu существует.
‘ Выйти из цикла Exit For End If Next cbar
```

Оператор **Do** повторяет выполнение группы инструкций, пока условие имеет значение **True** (случай While) или пока оно не примет значение **True** (случай Until).

В первом случае используется следующий синтаксис:	Во втором случае синтаксис оператора имеет вид:
Do While <условие>	Do Until <условие>
<операторVBA1>	<операторVBA1>
Exit Do	Exit DO
<операторVBA2>	<операторVBA2>
Loop	Loop

В любом месте управляющей структуры **Do** может быть размещено любое число инструкций **Exit Do**, которые обеспечивают альтернативные возможности выхода из цикла.

В следующем примере рассмотрено применение еще одного оператора цикла, реализованного в VBA. Это оператор **While...Wend**. Генерируется случайное число в диапазоне 1 – 10 до тех пор, пока не выпадет значение 7. Переменная **attempt** после завершения цикла будет содержать количество попыток:

```
Dim rank As Integer   ‘ Количество очков
Dim attempt As Integer’ Попытка
Randomize
rank=Int(10*Rnd())+1
attempt=1
While rank<>7
attempt=attempt+1
‘ Генерация случайного числа от 1 до 10
rank=Int(10)*Rnd())+1
Wend
```

Оператор безусловного перехода

Можно задать переход на любую строку внутри процедуры без проверки каких-либо условий. Этой строке надо присвоить **метку**. Метка должна обязательно начинаться с буквы и заканчиваться двоеточием:

LabelEnd: <очередной ОператорVBA>

Процедуры и функции

Процедуры и функции VBA представляют собой законченные фрагменты программного кода. Текст

процедуры VBA заключается между операторами **Sub** и **End Sub**:

```
Sub <имяПроцедуры> (<Аргумент1>, <Аргумент2>,
...)
   <ОператорVBA1>
   <ОператорVBA2>
End Sub
```

Существуют два способа вызова процедуры. В первом случае указывается имя вызываемой процедуры и список значений аргументов без скобок. Этот способ применяется, если вызывающая процедура расположена в том же модуле, что и вызываемая. Второй способ заключается в использовании оператора VBA Call: Call Account(453,CountPages+1).

VBA допускает две разновидности передачи переменных процедуре или функции: по ссылке и по значению. При передаче по ссылке фактический и формальный параметры отождествляются. Это означает, что любое изменение формального параметра в процедуре повлечет за собой изменение фактического параметра. Другими словами, значение переменной, переданной в процедуру, может быть там изменено. Для этой цели применяется описатель **ByRef**. Он же подразумевается по умолчанию. При передаче по значению в процедуру передается само значение и, следовательно, величина переменной в процедуре изменена быть не может. Такую форму передачи обеспечивает описатель **ByVal**.

Имя функции VBA, в отличие от имени процедуры VBA, выступает в качестве переменной и используется для возвращения значения в точку вызова. Текст функции заключается между операторами **Function** и **End Function**:

Function <имяФункции> (<Аргумент1>, <Аргумент2>, ...)
<ОператорVBA1>
<ОператорVBA2>
<ИмяФункции>=<ВозвращаемоеЗначение>
End Function

Функцию можно вызвать с помощью оператора **VBA Call**:

Call <имяФункции> (< Список фактических значений>)

Для вызова функции применяется еще один способ: напишите ее имя со списком фактических значений аргументов непосредственно в нужном выражении VBA или в формуле в вычисляемых полях форм, запросов и отчетов MS Access 2010.

Преобразование базы данных MS Access 2010 в базу MS SQL Server 2008

С целью оптимизации производительности приложения и обеспечения информационной безопасности приложения на более высоком уровне необходимо выполнить переход на платформу «клиент-сервер».

Перед преобразованием базы данных Access в формат SQL Server рекомендуется выполнить следующие действия:

1. Создайте резервную копию базы данных.
2. Убедитесь в наличии достаточного места на диске, где будет храниться преобразованная база данных SQL Server.
3. Создайте уникальные индексы в таблицах.

Для построения новой базы данных необходимо разрешение на создание базы данных, а также разрешение на доступ к системным таблицам в главной базе данных.

Для запуска мастера преобразования сделайте следующее:

1. Откройте базу данных Microsoft Access 2010.
2. Перейдите на вкладку ленты **Работа с базами данных**.
3. Выберите в разделе **Переместить данные** пункт **SQL Server**.
4. Появится стартовое окно мастера преобразования.
5. На первом шаге мастер предлагает воспользоваться существующей базой данных SQL Server или создать новую. Если после генерации SQL Server вы не создавали никаких баз данных на этом сервере, то установите переключатель в положение «создать базу данных».
6. Второе окно предназначено для сбора сведений об SQL Server, порядке соединения с ним и выбора названия базы данных, которая будет создана на нем. Имя новой базы данных мастер преобразования сформирует сам, добавив окончание **SQL** к имени базы данных MS Access 2010.
7. Некоторое замедление в работе перед третьим шагом связано с проверкой заданного имени базы данных. Если база с таким именем уже существует, то система изменит его, добавив цифру 1 в самый конец.
8. Мастер преобразования предлагает выбрать таблицы базы данных Microsoft Access, которые будут помещены в базу данных на сервере.
9. В данном случае необходимо выбрать все таблицы.

10. Перейдите к четвертому шагу мастера преобразования. Реляционная база данных включает в себя не только таблицы, которые могут содержать условия проверки корректности данных. В ней хранятся индексы, связи между таблицами, триггеры, хранимые процедуры и т. д. Надо определиться с объектами, которые следует конвертировать в базу SQL Server.

11. На пятом шаге надо выбрать способ преобразования в формат SQL Server 2008.

12. Мастер преобразования получил всю необходимую информацию для преобразования базы данных. Сделайте щелчок по кнопке Готово для запуска процесса.

13. Последним окном при работе мастера преобразования в формат MS SQL Server во всех трех случаях будет окно отчета. Оно сообщает об ошибках, возникших в процессе конвертации. Полное описание этого процесса Microsoft Access 2010 может поместить в специальный файл. Сохраните эту информацию.

Литература

1. Браст, Э. Дж. Разработка приложений на основе Microsoft SQL Server 2005. Мастер-класс / Э.Дж. Браст, С. Форте. – М.: Русская редакция, 2007. – 880 с.

2. Дейт, К.Дж. Введение в системы баз данных / К. Дж. Дейт. – М.: Вильямс, 2005. – 1316 с.

3. Жилинский, А.А. Самоучитель Microsoft SQL Server / А.А. Жилинский. – СПб.: БХВ-Петербург, 2009. – 240 с.

4. Хабибуллин, И.Ш. Самоучитель T-SQL / И.Ш. Хабибуллин. – СПб.: БХВ-Петербург, 2003. – 336 с.

5. Хотек, М. Microsoft SQL Server 2008: Реализация и обслуживание. Учебный курс Microsoft / М. Хотек. – М.: Русская редакция, 2012. – 576 с.

Учебное издание

Виноградова Ирина Владимировна
Щупак Светлана Станиславовна

РАБОТА В MICROSOFT ACCESS 2010

Пособие для слушателей переподготовки

Ответственный за выпуск *Е. С. Патей*

Подписано в печать 09.11.2017. Формат $60 \times 84\,^{1}/_{16}$.
Бумага офсетная. Печать цифровая. Усл. печ. л. 6,28.
Уч.-изд. л. 3,26. Тираж 50 экз. Заказ 12919.

Издатель и полиграфическое исполнение:
частное производственно-торговое
унитарное предприятие «Колорград».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/471 от 23.12.2015.

Пер. Велосипедный, 5-904, 220033, г. Минск,
www.сегмент.бел