Учреждение образования

«Белорусский государственный университет культуры и искусств»

Факультет культурологии и социокультурной деятельности

Кафедра информационных технологий в культуре

СОГЛАСОВАНО
Заведующий кафедрой

_____ П.В. Гляков
« 29 » мая 2017 г.

СОГЛАСОВАНО
Декан факультета

_____ Н.Н. Королев
« 30 » мая 2017 г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС
ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ И СИСТЕМЫ: БАЗЫ ДАННЫХ

для специальности 1–21 04 01 Культурология,
направление специальности 1–21 04 01–02 Культурология (прикладная),
специализации 1–21 04 01–02 04 Информационные системы в культуре

7 семестр обучения

Составитель:
П.В. Гляков, заведующий кафедрой информационных технологий в культуре учреждения образования “Белорусский государственный университет культуры и искусств”

Рассмотрен и утвержден
на заседании Совета университета 20.06. 2017 г., протокол № 10.

Составитель:

Гляков Петр Владимирович, заведующий кафедрой информационных технологий в культуре учреждения образования «Белорусский государственный университет культуры и искусств», кандидат физико-математических наук, доцент

Рецензенты:

В.С. Романчик,заведующий кафедрой веб-технологийи компьютерного моделированияБелорусского государственного университета,кандидатфизико-математических наук, доцент;

В.В. Нешитой, профессор кафедры информационныхресурсов Белорусского государственного университетакультуры и искусств, доктор технических наук,профессор

Рассмотрен и рекомендован к утверждению:

Кафедрой информационных технологий в культуре
(протокол от 29.05.2017 г., № 9);

Советом факультета культурологии и социокультурной деятельности
(протокол от 30.05.2017 г., № 9)

Оглавление

1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

В системе дисциплин, предусмотренных для студентов специальности "Культурология"специализации "Информационные системы в культуре", центральное место занимает дисциплина «Базы данных». Она призвана стать основой для понимания процессов автоматизации управления в сфере образования, культуры и искусств.Дисциплина «Базы данных» имеет межпредметную связь с такими дисциплинами, как «Основы информационных дисциплин» и «Информационные технологии в культуре». Эта дисциплина является составной частью интегрированной дисциплины «Информационные процессы и системы», в которую кроме нее входят еще две дисциплины: «Основы информационных систем» и «Аналитическая обработка источников информации».

Целью изучения дисциплины «Базы данных» является формирование знаний и умений для проектирования и разработки баз данных, позволяющих автоматизировать процессы управления в сфере образования, культуры и искусств.

*Задачи*курса:

– ознакомление с современным состоянием проектирования, разработки и ведения баз данных;

– изучение основных способов проектирования и разработки баз данных с помощью системы управления базами данных реляционного типа;

– приобретение умений разрабатывать базы банных в сфере образования, культуры и искусств.

В результате изучения дисциплины студенты должны *знать:*

– понятия информационные системы и автоматизированные информационные системы (АИС);
– понятия базы данных и системы управления базами данных (СУБД);
– иерархическую, сетевую и реляционную модели баз данных;
– понятия знания и базы знаний;
– базовую структуру экспертной системы;
– нормальные формы отношений;
– жизненный цикл АИС;
– концептуальную модель базы данных;
– инфологическую модель базы данных;
– технологию создания базы данных;
– виды и способы создания запросов;
– способы представления данных в виде форм;
– способы разработки отчетов;

– алгоритмы импорта, экспорта и связывания данных;
– способ подготовки серийных писем;
– основные операции поддержки баз данных;
– средства защиты базы данных.

После изучения дисциплины студенты должны *уметь:*

– строить концептуальную и инфологическую модели базы данных;
– создавать запросы для получения информации из базы данных;
– представлять данные в виде форм и отчетов;
– проектировать и разрабатывать учебные базы данных;
– разрабатывать пользовательский интерфейс базы данных;
– выполнять импорт, экспорт и связывание данных;
– подготавливать серийные письма;
– выполнять операции по поддержке базы данных;
– использовать средства защиты базы данных.

Методы обучения. Материал излагается на основе современных методических требований с учетом педагогических целей на уровнях представления, понимания, знания, применения и творчества. При чтении лекций особое внимание уделяется рассмотрениюпримеров, иллюстрирующих то или иное понятие, приводятся различные способы интерпретации понятий.

Лабораторные занятия направлены на формирование умений практического использования полученных знаний при решении конкретных задач. Методика их проведения содействует развитию творческих способностей каждого студента и приобретению навыков самостоятельной работы. Используются такие новые формы активизации учебного процесса, как игры, викторины, работа в командах с распределением ролей и т.п. Хорошо зарекомендовал себя при разработке баз данных метод проектов. Он поддерживает педагогические цели на уровнях представления, понимания, знания, применения и творчества.

Самостоятельная работа студентов ориентирована на изучение отдельных вспомогательных тем дисциплины, решение дополнительных рекомендованных задач и подбор практических примеров, иллюстрирующих теоретические основы баз данных. Результаты самостоятельной работы выявляются как при ответах на теоретические вопросы, так и при выполнении заданий на компьютере.

2 ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

2.1 Учебные издания

1. Гляков, П.В. Система управления базами данных Access 2.0 : учеб.пособие / П.В. Гляков, С.Н. Карачун. – Минск : РИПО, 1998. – 100 с.

2. Гляков, П.В. Импорт, экспорт и связывание данных в Microsoft Access : метод.рекомендации / П.В. Гляков. – Минск : РИПО, 2005. – 34 с.

3. Гляков, П.В. Базы данных: компьютерный практикум : учеб.пособие / П.В. Гляков. – Минск : БГУКИ, 2008. – 130 с.

4. Гляков П.В. Формирование профессиональных компетенций на лабораторных занятиях по базам данных / П.В. Гляков // Компетентностный подход в высшем образовании: проблемы и перспективы : материалы науч.-метод. конф., Минск, 4 февр. 2016 г. / М-во культуры Респ. Беларусь, Белорус. гос. ун-т куль- туры и искусств ; редкол.: Ю. П. Бондарь (пред.) [и др.]. – Минск : БГУКИ, 2016. – 322 с. – С. 153-158.

2.2 Конспект лекций

Лекция 1

Основные понятия баз данных

Основные вопросы

1. Информация и информационные системы.
2. Организация баз данных.
3. Модели баз данных.
4. Реляционные базы данных.

*Цель.*Изучение основных понятий баз данных.

Информация и информационные системы

В деятельности любого предприятия, учреждения, организации существенную роль играют данные. Данные фиксируются в определенной форме, пригодной для последующей обработки, хранения и передачи (на бумаге, магнитных дисках, магнитных лентах, компакт-дисках и т.п.).

Из данных извлекается необходимая информация. Данные в этом смысле можно рассматривать как сырье (ресурс) для производства информации. В результате обработки данные приобретают смысл, т.е. становятся информацией.

Под информацией понимают любые сведения о каком-либо событии, процессе, объекте, которые можно воспринимать, передавать, хранить или использовать.

На предприятии или в учреждении необходима информация для управления финансовыми, трудовыми и материальными ресурсами. Для управления финансовыми ресурсами надо иметь следующую информацию: источники денежных поступлений и их объемы, сколько денег израсходовано и на что, сколько средств предстоит получить и сколько осталось.

Управлять трудовыми ресурсами можно, когда известна информация о числе сотрудников, их специальности или профессии, должностном окладе, местонахождения рабочего места, прошлых достижениях сотрудников, сегодняшнем положении, возможности продвижения по службе.

Для управления материальными ресурсами требуется знать: какие материалы есть в наличии, откуда они поступают, какое количество их требуется, сколько уже израсходовано, сроки поставки материалов и другую информацию.

Информация, предназначенная для управления учреждением, условно может быть разбита на три уровня. К нижнему уровню управления

относится оперативная информация. Эта информация используется сотрудниками подразделений учреждения в повседневной работе. Она представляет собой часто обновляемую, первичную, рутинную информацию. Поэтому в информационных системах в первую очередь подлежит автоматизации обработка оперативной информации.

На среднем уровне управления имеют дело с тактической информацией. Эта информация предназначена для руководителей среднего звена. Тактическая информация получается путем обобщения оперативной информации и может быть представлена в виде отчетов или различных вариантов решения.

К верхнему уровню управления относят стратегическую информацию. Она получается в результате обработки оперативной и тактической информации. Эта информация содержит краткие, но содержательные сводки, отчеты, прогнозы. На ее основе осуществляется долгосрочное планирование работы учреждения.

Каждое предприятие, учреждение, организацию можно рассматривать как информационную систему, состоящую из элементов, связей между ними, по которым циркулирует некоторая информация. Информационная система функционирует на базе некоторой информационной технологии. В понятие информационной технологии входят устройства, носители информации, методы хранения, переработки и обмена информацией. Например, информационные технологии 30-40 г.г. строились на базе телефона, почты, устных сообщений, отчетов, различных форм и бланков, бумаги и т.п.

Задачей разработчиков автоматизированных информационных систем (АИС) является включение в существующие информационные системы со своей информационной технологией элементов автоматизации на всех уровнях и создание на базе персональных компьютеров и вычислительных сетей новой информационной технологии.

Организация баз данных

Первые АИС создавались на основе так называемого позадачного метода. При таком методе решались вопросы автоматизации задач оперативного уровня. Выбирались самые очевидные, формализуемые (имеющие алгоритмы решения) задачи, автоматизация которых сразу давала максимальную эффективность. Это были задачи расчета заработной платы, снабжения, кадры и т.п.

Но вскоре разработчики АИС столкнулись с трудностями и проблемами, которые оказались непреодолимыми при таком подходе. Требовался новый более качественный метод. Были сформулированы

стандартные требования к организации данных в АИС. Основные из них приведены ниже:

– интеграция данных. Это означает, что все данные должны накапливаться и храниться централизованно, создавая в реальном масштабе времени обновляемую модель предметной области. Предметная область (ПО) – часть реального мира, подлежащая автоматизации;

– максимально возможная независимость прикладных программ от данных, т.е. отделение логической модели ПОот физического представления в памяти компьютера или, говоря другими словами, обеспечение логической и физической независимости данных;

– безопасность данных. Под безопасностью данных понимают защиту данных от случайного или преднамеренного доступа к ним лиц, не имеющих на это право;

– минимальная избыточность данных, т.е. требования новых приложений должны удовлетворяться за счет существующих данных, а не путем создания новых файлов;

– обработка непредсказуемых запросов должна быть обеспечена с помощью высокоуровневого языка запроса или пользовательского интерфейса генерации отчетов.

– Выполнение этих требований привело к созданию единого для многих приложений блока данных, который мы будем называть базой данных – БД, и разработке одной управляющей программы для манипулирования данными на физическом уровне, называемой системой управления базами данных – СУБД.

Отметим, что не каждый блок данных является БД. БД – это совокупность данных, обладающих следующими качествами:

– интегрированностью, направленной на решение общих задач;

– модельностью, т.е. структурированностью, отражающей некоторую часть реального мира;

– взаимосвязанностью;

– независимостью описания данных от прикладных программ, поскольку данные и их описания хранятся совместно в БД.

Аналогично не всякая управляющая программа работы с БД является СУБД. СУБД – это пакет программ, позволяющий:

– обеспечить пользователей языковыми средствами описания и манипулирования данными;

– обеспечить поддержку логических моделей данных. Модель данных определяет логическое представление физических данных;

– обеспечить операции создания и манипулирования логическими данными (выбор, вставка, обновление, удаление и т.п.) и одновременное выполнение этих операций над физическими данными;

– обеспечить защиту и целостность данных, поскольку при коллективном режиме работы многих пользователей возможно использование общих физических данных. Поэтому необходимо обеспечить защиту от некорректных обновлений пользователями, защиту от несанкционированного доступа, защиту данных от разрушений при сбоях оборудования.

Иногда для совокупности баз данных употребляют термин банк данных. Некоторые специалисты вместо базы данных используют термин банк данных. Часто в литературе термин банк данных дается без четкого определения. В нашем пособии термин банк данных включает в себя БД, СУБД, аппаратные средства, обслуживающие службы и некоторые другие компоненты.

Пользователя БД интересует ее информационное и смысловое содержание. Детали организации физического хранения данных нас интересовать не будут. Поэтому в представлении данных в БД мы будем выделять два уровня абстракции: информационную модель и физическую модель.

Информационная модель должна отображать ПО в терминах, понятных и привычных для пользователя. Обычно это информация об интересующих его фактах, явлениях, событиях, предметах и связях между ними.

Проектировщики информационной модели термином сущность называют объект любой природы, о котором надо хранить информацию в БД. Например, множество студентов можно назвать сущностью ‘студент’. Один объект сущности называют экземпляром сущности. Свойства, характеризующие сущность, называют атрибутами. Примерами атрибутов сущности ‘студент’ являются Номер зачетной книжки, Фамилия, Имя, Отчество, Факультет, Курс, Группа и т.п. Между различными сущностями ПО и их атрибутами могут существовать межсущностные и межатрибутные связи, информационно важные для пользователя БД.

Информационная модель ПО с выделенными в ней сущностями, атрибутами и связями должна быть описана для представления в компьютере. Это описание делается средствами модели данных, которую поддерживает СУБД, и называется внутренней схемой информационной модели.

СУБД поддерживает некоторую модель данных и отображает ее в соответствующие структуры физической базы данных. Средствами модели данных СУБД строится логическая внутренняя схема информационной модели ПО. Прикладная программа, которая пишется в терминах модели данных СУБД, поддерживает логику внешней информационной модели ПО для пользователя, основываясь на внутренней схеме ПО.

Модели данных, поддерживаемые СУБД, делят на сетевые, иерархические и реляционные. Соответственно различают иерархические, сетевые и реляционные СУБД. Сетевые и иерархические СУБД получили наибольшее распространение на больших и мини- ЭВМ.

В иерархических СУБД данные представляются в виде древовидной структуры. Дерево представляет собой иерархию элементов, называемых узлами. На самом верхнем уровне иерархии имеется только один узел – корень. Каждый узел, кроме корня, связан с одним узлом на более высоком уровне, называемым исходным узлом для данного узла. Ни один элемент не имеет более одного исходного. Каждый элемент может быть связан с одним или несколькими элементами на более низком уровне. Они называются порожденными.

Если порожденный элемент в отношении между данными имеет более одного исходного элемента, то это отношение нельзя описать как древовидную структуру. Его описывают в виде сетевой структуры. Любой элемент в сетевой структуре может быть связан с любым другим элементом. Такие структуры используются для представления данных в сетевых СУБД.

Иерархические и сетевые модели данных имеют тенденцию к усложнению описания ПОпо мере роста БД и требуют высокой квалификации пользователя. Была найдена другая модель для представления данных - реляционная. Более того было показано, что при помощи нормализации иерархические и сетевые модели данных могут быть приведены к реляционной. Практически все СУБД, предназначенные для персональных компьютеров, поддерживают реляционную модель данных.

Реляционные базы данных

В основе реляционной модели данных лежит понятие отношения (англ. relation). Отношение удобно представляется в виде двумерной таблицы при соблюдении определенных ограничивающих условий. Таблица привычна для пользователя, понятна и обозрима, ее легко запомнить.

Основоположник теории реляционных баз данных Е.Ф.Кодд показал, что набор отношений (таблиц) может быть использован для хранения данных об объектах реального мира и моделирования связей между ними. Например, для хранения сущности ‘слушатель’ используют отношение СТУДЕНТ, в котором свойства сущности располагаются в столбцах таблицы:

СТУДЕНТ

ФИО	Факультет	Курс	Группа
Сидорова	ФКиСКД	3	308
Василевский	ФКиСКД	2	212
Дубова	ТБКиСИ	3	314
Алексеенко	ФМИ	1	116
Королева	ТБКиСИ	2	213

Столбцы отношения называют атрибутами, атрибутам присваивают имена. Список имен атрибутов отношения называют схемой отношения. Список значений атрибутов отношения называют картежом. Схема отношения СТУДЕНТ записывается так:

СТУДЕНТ (Фамилия, Факультет, Курс, Группа).

Реляционная БД – это набор взаимосвязанных таблиц. Набор взаимосвязанных таблиц на физическом уровне (на внешних носителях информации) хранится в виде файла БД. Соответствие между элементами файла БД, таблицы, отношения и сущности, когда файл БД содержит одну таблицу, может быть показано следующим образом:

Файл БД	Таблица	Отношение	Сущность
запись	строка	кортеж	экземпляр сущности
поле	столбец	атрибут	атрибут

Важным достоинством реляционной модели данных является то, что к схемам отношений можно применять соответствующие операции и тем самым получать новые схемы отношения, которые ранее в файле БД не были представлены. Таким способом СУБД позволяет получать ответы на незапланированные прикладными программами информационные запросы.

Лекция 2

Жизненный цикл баз данных

Основные вопросы

1. Комбинированная модель жизненного циклаАИС.
2. Основополагающие принципы технологии SSADM.
3. Методическое обеспечение технологии SSADM.

Цель. Изучение этапов комбинированной модели жизненного циклаАИС.

По мнению специалистов, из-за многообразия типов АИС единой модели, которая описывала бы их жизненный цикл, не существует. В литературе был предложен целый спектр моделей, на противоположных концах которых находятся так называемые каскадная и спиральные модели. По-видимому, оптимум находится где-то посередине между ними. Такая комбинированная модель жизненного цикла АИС содержит следующие стадии:

1) разработка стратегии автоматизации;
2) оценивание реализуемости;
3) анализ требований;
4) разработка технического задания;
5) логическое проектирование;
6) физическое проектирование;
7) программирование;
8) отладка и испытание;
9) внедрение;
10) сопровождение;
11) анализ опыта эксплуатации.

Результатом выполнения стадии 1 являются документы, инициирующие разработку, в которых должны быть сформулированы стратегические цели и общий замысел автоматизации. Стадии 1, 7-11 являются каскадными, т.е. они выполняются последовательно друг за другом. Стадии 2-6 являются спиральными - в процессе их выполнения происходят многократные встречи проектировщиков АИС с заказчиками, сопровождающиеся многократными уточнениями и корректировками разрабатываемых документов. Далее стадии 2-6 будем называть технологией проектирования SSADM (StructuredSystemsAnalysisandDesignMethod). Разработка этой технологии началась с середины 70-х годов в Великобритании и в 1993 году она была официально принята в качестве государственного стандарта

Великобритании. Сейчас она широко используется в странах Западной Европы и в Японии.

Основополагающие принципы технологии SSADM приведены ниже:

– постоянное вовлечение представителей будущих пользователей в процесс выработки решений на протяжении всего проектирования АИС;

– четкая структуризация технологического процесса, взаимная увязка всех стадий, этапов и проектных процедур, явная регламентация ролей всех участников разработки;

– эффективный контроль хода разработки со стороны руководителей проекта, встроенный контроль качества проектирования по формализованным критериям, возможность применения существующих технологий автоматизированного управления разработкой;

– стыковка с технологиями, реализованными в существующих системах программирования и управления БД;

– формализация процесса разработки, обеспечивающая широкое применение средств автоматизации проектирования.

Технологическая стадия 2 не является обязательной и может быть пропущена, если ранее были проведены достаточно глубокие исследования при выработке стратегии автоматизации. Основной целью стадии является предварительное технико-экономическое обоснование проекта и разработка концепции будущей АИС.

Стадия 3 состоит из двух этапов: предпроектного обследования и выбора варианта автоматизации. На этапе предпроектного обследования определяют основные требования к новой АИС. Для этого изучают существующую систему обработки информации, составляя с участием пользователей ее логическое описание в терминах потоков данных, задач и информационных объектов. Определяют границы существующей системы, внешние объекты и функции пользователей. При этом, анализируя ее недостатки, формулируют основные требования к новой АИС, которые отражают в Каталоге требований.

На этапе выбора варианта автоматизации путем упорядочения требований по важности выбирают различные их подмножества и составляют описания нескольких вариантов разрабатываемой АИС. При этом также выполняют технико-экономические расчеты, позволяющие на этой стадии сравнить варианты и при активном участии будущих пользователей обоснованно выбрать среди них оптимальный.

На стадии 4 полностью определяют требования к выбранному варианту построения АИС, которые, как и на этапе предпроектного обследования стадии 3, формулируют в терминах потоков данных, задач и

информационных объектов. Отличие состоит в том, что речь идет не о существующей, а о новой системе, причем ее описывают значительно подробнее. Кроме того, в терминах событий и данных разрабатывают требования к динамике функционирования АИС. Характерно, что на этой стадии предусмотрена также разработка демонстрационного прототипа. Это позволяет частично реализовать достоинства спиральной модели жизненного цикла без присущего ей усложнения контроля хода разработки со стороны проекта.

На стадии 5 обосновывают выбор технической и программной сред реализации создаваемой АИС и параллельно с этим проектируют диалоговое взаимодействие пользователей с системой, а также разрабатывают и увязывают между собой постановки задач.

Стадия 6 является заключительной в технологии проектирования SSADM. На ней разрабатывают описания данных на физическом уровне, выполняют их оптимизацию, уточняют постановки задач и готовят руководящие указания по генерации баз данных и разработке программного обеспечения к выбранной среде реализации.

Таким образом, основным продуктом, создаваемым по технологии SSADM, является комплект документов, на основе которых может быть реализована разрабатываемая АИС на компьютере с использованием системы программирования и СУБД, выбранных на стадии 5.

Последовательность выполнения технологических стадий и состав разрабатываемых на них проектных документов четко регламентирован 13 методиками, составляющими методическое обеспечение технологии SSADM. Перечень их с краткой характеристикой приводится ниже:

№п/п	Название методики	Характер основныхпроектных документов	Номера стадий, на которых применяется методика
1.	Определение требований к АИС	Табличная форма	2, 3, 4, 5, 6
2.	Моделирование информационных потоков	Схема, табличная форма	2, 3
3.	Логическое моделирование данных	Схема, табличная форма	2, 3, 4, 5
4.	Определение функций и задач	Схема, табличная форма	3, 4
5.	Динамическое моделирование данных	Схема, табличная форма	4, 5

6.	Реляционный анализ данных	Табличная форма	3, 4, 5
7.	Выбор вариантов автоматизации	Пояснительная записка, схема	3
8.	Разработка демонстрационного прототипа	Видеограмма, схема	4
9.	Выбор вариантов технической реализации	Пояснительная записка, табличная форма	5
10.	Проектирование диалогового взаимодействия	Табличная форма, схема	4, 5
11.	Логическое проектирование процедур обработки	Табличная форма, схема	5
12.	Физическое проектирование БД	Табличная форма, схема	6
13.	Физическое проектирование процедур обработки	Табличная форма, схема	6

В рамках типового технологического процесса применение отдельных методик, например, “Выбор вариантов автоматизации”, ограничено одной технологической стадией. Однако вследствие итерационного характера разработки основных проектных документов большинство методик используется на протяжении нескольких стадий.

Основное назначение рассматриваемой технологии проектирования АИС заключается в том, чтобы разработчики могли выявить скрытые в проекте противоречия и устранить ошибки задолго до стадии отладки и испытания.

Лекция 3
Классификация и функции СУБД

Основные вопросы

1. Виды классификаций СУБД.
2. Основные функции и операции СУБД.
3. Возможности реляционных систем управления базами данных.
4. Поддержка целостности данных.

Цель. Изучение основных видов классификации СУБД и знакомство с возможностями СУБД.

Классификация СУБД по модели данных

Различают следующие модели баз данных: иерархические, сетевые, иерархические,

В *иерархических* СУБД используется представление базы данных в виде древовидной (иерархической) структуры, состоящей из объектов (данных) различных уровней.

Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка (объект более близкий к корню) к потомку (объект более низкого уровня), при этом возможна ситуация, когда объект-предок не имеет потомков или имеет их несколько, тогда как у объекта-потомка обязательно только один предок. Объекты, имеющие общего предка, называются близнецами (в программировании применительно к структуре данных дерево устоялось название братья).

Иерархической базой данных является файловая система, состоящая из корневого каталога, в котором имеется иерархия подкаталогов и файлов.Примерами таких баз данных являются:Cache иGoogleAppEngineDatastoreAPI.

Сетевые базы данных подобны иерархическим, за исключением того, что в них имеются указатели в обоих направлениях, которые соединяют родственную информацию.Пример –сетевая база данных Cache.

Практически все разработчики современных приложений, предусматривающих связь с системами баз данных, ориентируются на *реляционные* СУБД. По оценке Gartner в 2013 году рынок реляционных СУБД составлял 26 млрд. долларов с годовым приростом около 9%, а к 2018 году рынок реляционных СУБД достигнет 40 млрд. долларов. В настоящее время абсолютными лидерами рынка СУБД являются компании Oracle, IBM и Microsoft, с общей совокупной долей рынка около 90%, поставляя такие системы как Oracle Database, IBM DB2 и Microsoft SQL Server.

Объектно-ориентированные СУБД управляют базами данных, в которых данные моделируются в виде объектов, их атрибутов, методов и классов.Этот вид СУБД позволяет работать с объектами баз данных так же, как с объектами в программировании в объектно-ориентированных языках программирования. Объектно-ориентированные СУБД расширяет языки программирования, прозрачно вводя долговременные данные, управление параллелизмом, восстановление данных, ассоциированные запросы и другие возможности.Примером такой СУБД является GemStone.

*Объектно-реляционный*тип СУБД позволяет через расширенные структуры баз данных и язык запросов использовать возможности объектно-ориентированного подхода: бъекты, классы и наследование.Зачастую все те СУБД, которые называются реляционными, являются, по факту, объектно-реляционными.Примерами являются – PostgreSQL, DB2, Oracle, Microsoft SQL Server.

Классификация СУБД по степени распределённости

По степени распределённости различают:

– локальные СУБД (все части локальной СУБД размещаются на одном компьютере)

– распределённые СУБД (части СУБД могут размещаться на двух и более компьютерах).

Классификация по способу доступа к базе данных

По способу доступа к базе данныхвыделяют следующие СУБД: файл-серверные, **Клиент-серверные.,** Встраиваемая

В *файл-серверных* СУБД файлы данных располагаются централизованно на файл-сервере. СУБД располагается на каждом клиентском компьютере (рабочей станции). Доступ СУБД к данным осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок. Преимуществом этой архитектуры является низкая нагрузка на процессор файлового сервера. Недостатки: потенциально высокая загрузка локальной сети; затруднённость или невозможность централизованного управления; затруднённость или невозможность обеспечения таких важных характеристик как высокая надёжность, высокая доступность и высокая безопасность. Применяются чаще всего в локальных приложениях, которые используют функции управления БД; в системах с низкой интенсивностью обработки данных и низкими пиковыми нагрузками на БД.

На данный момент файл-серверная технология считается устаревшей, а её использование в крупных информационных системах –

недостатком.Примерами файл-серверных СУБД являются:Microsoft Access, Paradox, dBase, FoxPro, VisualFoxPro.

Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно. Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу. Достоинства: потенциально более низкая загрузка локальной сети; удобство централизованного управления; удобство обеспечения таких важных характеристик как высокая надёжность, высокая доступность и высокая безопасность.

Примеры: Oracle, Firebird, Interbase, IBM DB2, Informix, MS SQL Server, Sybase Adaptive Server Enterprise, PostgreSQL, MySQL, Cache, ЛИНТЕР.

Встраиваемая СУБД – СУБД, которая может поставляться как составная часть некоторого программного продукта, не требуя процедуры самостоятельной установки. Встраиваемая СУБД предназначена для локального хранения данных своего приложения и не рассчитана на коллективное использование в сети. Физически встраиваемая СУБД чаще всего реализована в виде подключаемой библиотеки. Доступ к данным со стороны приложения может происходить через SQL либо через специальные программные интерфейсы (API).

ПримерамивстраиваемыхСУБД являются: OpenEdge, SQLite, BerkeleyDB, FirebirdEmbedded, MicrosoftSQLServerCompact, ЛИНТЕР.

Функции СУБД

Основными функциями СУБД являются:

– определение данных (описание структуры баз данных);

– обработка данных;

– управление данными во внешней памяти (на дисках);

– управление данными в оперативной памяти с использованием дискового кэша;

– журнализация изменений, резервное копирование и восстановление базы данных после сбоев;

– поддержка языков баз данных (языка определения данных, языка манипулирования данными).

Прежде чем заносить данные в таблицы, нужно *определить структуру этих таблиц.* Под этим понимается не только описание наименований и типов полей, но и ряд других характеристик (например, формат, критерии проверки вводимых данных). Кроме описания структуры таблиц, обычно задаются связимежду таблицами. Связи в реляционных

базах данных определяются по совпадению значений полей в разных таблицах. Например, клиенты и заказы связаны отношением "один-ко-многим",т. к. одной записи в таблице, содержащей сведения о клиентах, может соответствовать несколько записей в таблице заказов этих клиентов.

Если же рассмотреть отношение между преподавателями и курсами лекций, которые они читают, это будет отношение "многие-ко-многим",т. к. один преподаватель может читать несколько курсов, но и один курс может читаться несколькими преподавателями. И последний тип связей между таблицами – это отношение "один-к-одному".Такой тип отношений встречается гораздо реже. Как правило, это бывает в двух случаях: запись имеет большое количество полей, и тогда данные об одном типе объектов разносятся по двум связанным таблицам, или нужно определить дополнительные атрибуты для некоторого количества записей в таблице, тогда создается отдельная таблица для этих дополнительных атрибутов, которая связывается отношением "один-к-одному" с основной таблицей.

Любая СУБД позволяет выполнять четыре *простейшие операции с данными*:

– добавлять в таблицу одну или несколько записей;

– удалять из таблицы одну или несколько записей;

– обновлять значения некоторых полей в одной или нескольких записях;

– находитьодну или несколько записей, удовлетворяющих заданному условию.

Для выполнения этих операций используется механизм запросов. Результатом выполнения запросов является либо отобранное по определенным критериям множество записей, либо изменения в таблицах. Запросы к базе формируются на специально созданном для этого языке, который так и называется язык структурированных запросов (SQL – StructuredQueryLanguage).

Под *управлением данными* обычно понимают защиту данных от несанкционированного доступа, поддержку многопользовательского режима работы с данными и обеспечение целостности и согласованности данных.

Защита от несанкционированного доступа обычно позволяет каждому пользователю видеть и изменять только те данные, которые ему разрешено видеть или менять. Средства, обеспечивающие многопользовательскую работу, не позволяют нескольким пользователям одновременно изменять одни и те же данные. Средства обеспечения целостности и согласованности данных не дают выполнять такие

изменения, после которых данные могут оказаться несогласованными. Например, когда две таблицы связаны отношением "один-ко-многим", нельзя внести запись в таблицу на стороне "многие" (ее обычно называют подчиненной), если в таблице на стороне "один" (главной) отсутствует соответствующая запись.

В большинстве реляционных систем реализованы следующие возможности:

1. Представление информации в виде таблиц.

2. Поддержка логической структуры данных, независимо от их физического представления.т.е. представление данных абсолютно не зависит от способа их физического хранения и изменение взаимосвязей между таблицами и строками не влияет на правильное функционирование программных приложений и текущих запросов.

3. Использование языков высокого уровня для структурирования, выполнения запросов и изменения информации в базах данных. Для работы с БД используются специальные языки, в целом называемые языками баз данных. В СУБД обычно поддерживается единый язык, содержащий все необходимые средства – от создания БД до обеспечения пользовательского интерфейса при работе с данными. Наиболее распространенным в настоящее время языком СУБД является язык SQL **(**StructuredQueryLanguage**).**

4. Поддержка основных реляционных операций (выбор, проектирование и объединение), а также теоретико-множественные операции, такие как объединение, пересечение и дополнение. Существует три операции по выборке данных – проектирование, выбор и объединение, которые позволяют строго указать системе, какие данные необходимо показать. Операция проектирования выбирает столбцы, операция выбора – строки, а операция объединения собирает вместе данные из связанных таблиц.

5. Возможность различать в таблицах неизвестные значения (nulls), нулевые значения и пропуски в данных. Чтобы сохранить целостность данных в реляционной модели для обработки пропущенной информации используется понятие нуля. «Нуль» не означает пустое поле или обычный математический нуль. Он отображает тот факт, что значение неизвестно, недоступно или неприменимо. Существенно, что использование нулей инициирует переход с двухзначной логики (да/нет) на трехзначную (да/нет/может быть). «Нули» являются составной частью большинства официальных стандартов различных реляционных СУБД.

6. Обеспечение механизмов для поддержки целостности, авторизации,

транзакций и восстановления данных.

Целостность – очень сложный и серьезный вопрос при управлении реляционными базами данных. *Несогласованность* между данными может возникать по целому ряду причин. Несогласованность или противоречивость данных может возникать вследствие сбоя системы – проблемы с аппаратным обеспечением, ошибки в программном обеспечении или логической ошибки в приложениях. Реляционные системы управления базами данных защищают данные от такого типа несогласованности, гарантируя, что команда либо будет исполнена до конца, либо будет полностью отменена. Этот процесс обычно называют управлением транзакциями.

Другой тип целостности, называемый объектной целостностью, связан с корректным проектированием базы данных. *Объектная целостность* требует, чтобы ни один первичный ключ не имел нулевого значения.

Третий тип целостности, называемой *ссылочной целостностью*, означает непротиворечивость между частями информации, повторяющимися в разных таблицах. Чрезвычайно важно, чтобы при изменении информации в одном месте, она соответственно изменялась и во всех других местах.

Лекция 4

Нормализация отношений

Основные вопросы

1. Объектные и связные отношения.
2. Первая нормальная форма отношений.
3. Функциональная зависимость и вторая нормальная форма отношений.
4. Транзитивная зависимость и третья нормальная форма отношений.

Цель. Изучение основных нормальных форм отношений и способов нормализации отношений.

В зависимости от содержания отношений реляционных БД мы будем различать два типа отношений: объектные и связные. Объектное отношение хранит данные об объектах. Рассмотренное ранее отношение СТУДЕНТ является примером объектного отношения.

В объектном отношении один из атрибутов однозначно идентифицирует объект. Такой атрибут называют ключом отношения. В отношении СТУДЕНТ ключом может быть атрибут Фамилия. Для удобства ключ размещают в первом столбце таблицы. Ключ может состоять из нескольких атрибутов или быть частью одного атрибута.

Основное ограничение реляционной модели БД состоит в следующем. В объектном отношении не должно быть строк с одинаковыми ключами. Это ограничение позволяет обеспечить целостность БД.

Связное отношение хранит ключи двух или более объектных отношений, по этим ключам устанавливаются связи между объектами. Пусть в БД имеются два отношения: ФИРМА (Название, Адрес) и ТОВАР (Наименование, Цена). Из них может быть получено связное отношение ПОСТАВЛЯЕТ (Фирма,Товар).

Связное отношение кроме связываемых ключей может иметь и другие атрибуты, которые будут функционально зависеть от этой связи. Например, отношение ПОСТАВЛЯЕТ может иметь следующий вид: ПОСТАВЛЯЕТ(Фирма,Товар,Цена). Ключи в связных отношениях называются внешними ключами, поскольку они являются первичными ключами других отношений.

Каждому внешнему ключу должна соответствовать строка какого-либо объектного отношения. Невыполнение этого ограничения может привести к нарушению ссылочной целостности данных. Ключ должен ссылаться на объект, который существует.

Отношения в БД должны быть нормализованы. Это означает, что каждый атрибут должен быть простым – содержать неделимые значения. В приведенном ниже отношении условие нормализации не выполнено.

ПРЕПОДАВАТЕЛЬ

Фамилия	Кафедра	Предмет	Телефон	
			рабочий	домашний
Голубев	Информатика	Гипертекстовые системы	20-08-35	70-01-32
Погорелова	Психология	Практическая психология	20-08-35	50-85-30
Михайлов	Методика воспитания	Методика воспитания	20-15-91	53-33-45
Новожилов	Информатика	Информационные технологии	20-08-35	50-85-30
Антипов	Педагогика	Педагогика	20-13-04	55-93-14

Это отношение можно нормализовать разбиением сложного атрибута Телефон на два простых: Телефон рабочий и Телефон домашний. Схема нормализованного отношения выглядит так:

ПРЕПОДАВАТЕЛЬ (Фамилия, Кафедра, Предмет, Телефон рабочий, Телефон домашний).

Отношение, у которого все атрибуты простые, называется приведенным к первой нормальной форме (1НФ).

Перечислим условия и ограничения, накладываемые на отношения реляционной моделью данных, которые позволяют таблицы считать отношениями:

1. Не может быть одинаковых первичных ключей, т.е. все строки таблицы должны быть уникальны.

2. Все строки таблицы должны иметь одну и ту же структуру, т.е. одно и то же количество атрибутов с соответственно совпадающими именами.

3. Имена столбцов таблицы должны быть различны, а значения столбцов должны быть однотипными.

4. Значения атрибутов должны быть простыми, следовательно, отношения не могут иметь в качестве компонент другие отношения.

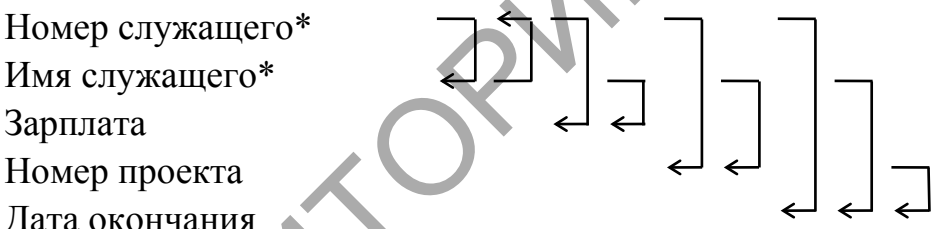
5. Должна соблюдаться ссылочная целостность для внешних ключей.

6. Порядок следования строк в таблице несущественен - он лишь влияет на скорость доступа к строке.

Задавая отношения над элементами данных, проектировщик БД определяет, какие из атрибутов объекта являются зависимыми. Термин функциональная зависимость означает следующее. Пусть имеется два атрибута: A и B. Если в любой момент времени каждому значению A соответствует не более чем одно значение атрибута B, говорят, что B функционально зависит от A. Функциональная зависимость обозначается так:

A - > B.

Рассмотрим следующее отношение: СЛУЖАЩИЙ (Номер служащего, Имя служащего, Зарплата, Номер проекта, Дата окончания). Функциональные зависимости между атрибутами СЛУЖАЩИЙ для наглядности представим в виде диаграммы.



В нашем примере звездочками отмечены основные атрибуты. Основные атрибуты являются элементами возможных ключей отношения. Атрибут Имя служащего функционально зависит от атрибута Номер служащего и, в свою очередь, атрибут Номер служащего функционально зависит от атрибута Имя служащего. Атрибут Зарплата функционально зависит от атрибута Номер служащего или Имя служащего. Аналогично атрибут Номер проекта функционально зависит от атрибута Номер служащего или Имя служащего. Атрибут Дата окончания обладает функциональной зависимостью от атрибутов: Номер служащего, Имя служащего и Номер проекта.

Атрибут может функционально зависеть не от одного какого-то атрибута, а от целой группы атрибутов. Рассмотрим, например, отношение, содержащее информацию о расходе рабочего времени программиста:

РАБОТА ПРОГРАММИСТА (Номер программиста, Номер программы, Имя программиста, Имя программы, Количество часов).

Атрибут Количество часов функционально зависит от составного ключа (Номер программиста, Номер программы) или от одного из следующих возможных ключей: (Номер программиста, Имя программы), (Имя программиста, Номер программы) или (Имя программиста, Имя программы). Заранее предполагается, что среди программистов нет однофамильцев и что две программы не могут иметь одинаковых имен.

Атрибут (или набор атрибутов) B из отношения R называется полностью зависимым от другого набора атрибутов A отношения R, если B функционально зависит от всего множества A, но не зависит ни от какого подмножества A.

Например, в отношении РАБОТА ПРОГРАММИСТА атрибут Количество часов является полностью зависимым от составного ключа (Номер программиста, Номер программы), так как он задает количество рабочего времени, затраченного данным программистом на конкретную программу. При этом ни один из атрибутов Номер программиста или Номер программы в отдельности не определяет значение атрибута Количество часов.

Говорят, что отношение задано во второй нормальной форме (2НФ), если оно является отношением в первой нормальной форме, и каждый атрибут, не являющийся основным атрибутом в этом отношении, полностью зависим от любого возможного ключа этого отношения.

Если все возможные ключи отношения содержат по одному атрибуту, как это было в ранее рассмотренном отношении СЛУЖАЩИЙ, то это отношение задано во второй нормальной форме. Если ключи состоят более чем из одного атрибута, отношение, заданное в 1НФ, может не быть отношением в 2НФ. Отношение РАБОТА ПРОГРАММИСТА задано в 2НФ, потому что единственный атрибут Количество часов, не являющийся основным, полностью зависит от каждого возможного ключа.

Рассмотрим пример отношения, которое не является отношением в 2НФ:

ИСТОЧНИК СНАБЖЕНИЯ (Номер поставщика, Номер партии товара, Имя поставщика, Сведения о поставщике, Цена).

У этого отношения один возможный ключ, который является составным: (Номер поставщика, Номер парии товара). Атрибут Имя поставщика не входит в ключ, так как одной и той же фирме в разных районах могут быть присвоены различные номера поставщика. Таким образом, атрибут Номер поставщика не определяется значением атрибута Имя поставщика. В этом отношении атрибуты Имя поставщика и Сведения

о поставщике, не будучи основными, функционально зависят от атрибута Номер поставщика, который является подмножеством составного ключа.

Нарушение условий 2НФ приводит к ряду неудобств:

1. Графа Сведения о поставщике не может быть заполнена до фактической поставки конкретной партии товара поставщиком, либо необходимо задать какой-нибудь фиктивный номер партии.

2. Если поставщик временно задержал поставку некоторой партии, то удаление кортежа, соответствующего данному значению атрибута Номер поставщика, вызовет удаление сведений о нем.

3. Если требуется изменить значение атрибута Сведения о поставщике, то придется внести одни и те же изменения сразу в несколько кортежей.

Устранение подобных трудностей достигается разложением отношения на два отношения, представленных в 2НФ, говоря другими словами, приведением отношения к 2НФ. Наше отношение ИСТОЧНИК СНАБЖЕНИЯ можно разложить на два отношения:

ПОСТАВЩИК (Номер поставщика, Имя поставщика, Сведения о поставщике) и ТОВАР (Номер поставщика, Номер партии товара, Цена). Легко убедиться, что отношения ПОСТАВЩИК и ТОВАР находятся в 2НФ.

В ряде случаев и вторая нормальная форма порождает неудобства. Для их устранения используется еще один шаг нормализации, преобразующий 2НФ в третью нормальную форму (3НФ). На этом шаге ликвидируется так называемая транзитивная зависимость атрибутов.

Если для атрибутов A, B, C выполняются условия A -> B и B -> C, но обратная зависимость отсутствует, то говорят, что C зависит от A транзитивно. Отношение находится в 3НФ, если оно находится в 2НФ и в нем отсутствуют транзитивные зависимости неосновных атрибутов от каждого возможного ключа.

В отношении СЛУЖАЩИЙ, ранее представленном диаграммой, атрибут Дата окончания зависит от атрибута Номер проекта, который в свою очередь зависит от атрибута Номер служащего. Таким образом, Дата окончания транзитивно зависит от атрибута Номер служащего. Это отношение можно привести к 3НФ, расщепив его на два отношения:

СЛУЖАЩИЙ (Номер служащего, Имя служащего, Зарплата, Номер проекта), ПРОЕКТ (Номер проекта, Дата окончания).

Следует отметить, что уже известны и другие нормальные формы отношений: усиленная 3НФ – нормальная форма Бойса-Кодда, четвертая нормальная форма и пятая нормальная форма. Они позволяют устранить

зависимости ключей от неосновных атрибутов, независимые многозначные зависимости и т.д. Рассмотрение этих форм выходит за пределы данной публикации.

При проектировании БД процессу нормализации отношений отводится следующее место:

– вначале составляются исходные отношения проекта БД с использованием объектно-связной модели для отображения объектов предметной области и связей между ними;

– затем производится нормализация, т.е. разложение исходных отношений и назначение ключей новых отношений в соответствии с правилами нормализации;

– далее схемы нормализованных отношений описываются средствами СУБД и вводятся в компьютер.

Лекция 5

Основные характеристики и возможности MS Access

Основные вопросы

1. Средства для разработки приложений.
2. Мастера для выполнения работ в MicrosoftAccess.
3. Объекты базы данных в MicrosoftAccess.
4. Достоинства средств MicrosoftAccess.

Цель. Знакомство с основными характеристиками и возможностямиMicrosoftAccess.

Группа реляционных СУБД представлена на рынке программных продуктов очень широко. Это, например, такие системы, какSQLite, MySQL, PostgreSQL, Microsoft Access,Oracle, Firebird, Interbase, IBMDB2, Informix, MSSQLServer, SybaseAdaptiveServerEnterprise, Cache, ЛИНТЕР.

СУБД Microsoft Access имеет достаточно высокие скоростные характеристики и входит в состав чрезвычайно популярного в нашей стране и за рубежом пакета Microsoft Office. Набор команд и функций, предлагаемых разработчикам программных продуктов в среде Access, по мощи и гибкости отвечает большинству современных требований к представлению и обработке данных. В Access поддерживаются разнообразные всплывающие и многоуровневые меню, работа с окнами и мышью, реализованы функции низкоуровневого доступа к файлам, управления цветами, настройки принтера, представления данных в виде электронных таблиц и т.п. Система также обладает средствами быстрой генерации экранов, отчетов и меню, поддерживает язык управления запросами SQL, имеет встроенный язык VisualBasicforApplication (VBA), хорошо работает в сети. СУБД Access позволяет использовать другие компоненты пакета Microsoft Office, такие как текстовый процессор WordforWindows , электронные таблицы Excel и т. д.

В отличие от других настольных СУБД, Access хранит все данные в одном файле, хотя и распределяет их по разным таблицам. Для пользователя имеется возможность создания множества таблиц. Самым важным правилом, которое необходимо соблюдать, является то, что в базе данных нужно хранить только необходимую информацию, и при этом все данные должны храниться только в одном месте.

Хранение информации в таблицах, которые имеют поля с повторяющимися данными очень неэффективный способ хранения данных. И не только потому, что они занимают лишнее место в памяти. Этот аргумент в последнее время не является таким сильным, как раньше

из-за значительного снижения цен на микросхемы памяти. Основная причина — это то, что такие данные долго вводить и трудно анализировать. Если случайно, при вводе значения пользователь сделал грамматическую ошибку или даже просто ввел лишний пробел, то при запросах и группировках такое значение будет рассматриваться как самостоятельное, и строка, содержащая это значение, не попадет в нужную группу или просто не будет выведена на экран. Именно поэтому при проектировании структуры баз данных стараются избегать повторения данных и создают для них отдельные таблицы. Этот процесс называется нормализацией.

СредстваMicrosoft Access, существенно упрощающие разработку приложений:

1. Процедуры обработки событий и модули форм и отчетов. На встроенном языке VBA можно писать процедуры обработки событий, возникающих в формах и отчетах. Процедуры обработки событий хранятся в модулях, связанных с конкретными формами и отчетами, в результате чего код становится частью макета формы или отчета. Кроме того, существует возможность вызова функции VBA свойством события.

2. Свойства, определяемые в процессе выполнения. С помощью макроса или процедуры обработки событий можно определить практически любое свойство формы или отчета в процессе выполнения в ответ на возникновение события в форме или отчете.

3. Модель событий.Модель событий, похожая на используемую в языке MicrosoftVisualBasic, позволяет приложениям реагировать на возникновение различных событий, например, нажатие клавиши на клавиатуре, перемещение мыши или истечение определенного интервала времени.

4. Использование обработки данных с помощью VBA. С помощью языка VBA можно определять и обрабатывать различные объекты, в том числе, таблицы, запросы, поля, индексы, связи, формы, отчеты и элементы управления.

5. Построитель меню. Предназначен для помощи при создании специальных меню в приложениях. Кроме того, специальные меню могут содержать подменю.

6. Улучшенные средства отладки. Помимо установки точек прерывания и пошагового выполнения программ на языке VBA, можно вывести на экран список всех активных процедур. Для этого следует выбрать команду Вызовы в меню Вид или нажать кнопку [Вызовы] на панели инструментов.

7. Процедура обработки ошибок. Помимо традиционных способов обработки ошибок возможно использование процедуры обработки события Error для перехвата ошибок при выполнении программ и макросов.

8. Улучшенный интерфейс защиты. Команды и окна диалога защиты упрощают процедуру защиты и смены владельца объекта.

9. Программная поддержка механизма OLE. С помощью механизма OLE можно обрабатывать объекты из других приложений.

10. Программы-надстройки. С помощью VBA можно создавать программы-надстройки, например нестандартные мастера и построители. Мастер – средство Microsoft Access, которое сначала задает пользователю вопросы, а затем создает объект (таблицу, запрос, форму, отчет и т.д.) в соответствии с его указаниями.

Диспетчер надстроек существенно упрощает процедуру установки программ-надстроек в Microsoft Access.

Access позволяет даже мало подготовленному пользователю создать свою БД, обрабатывать данные с помощью форм, запросов и отчетов, проводить анализ таблиц БД и выполнять ряд других работ. Практически для любых работ с БД в Access имеется свой мастер, который помогает их выполнять.

Мастер по анализу таблиц позволяет повысить эффективность базы данных за счет нормализации данных. Он разделяет ненормализованную таблицу на две или несколько таблиц меньшего размера, в которых данные сохраняются без повторения.

Мастера по созданию форм и отчетов упрощают и ускоряют процесс создания многотабличных форм и отчетов. Новые форма и отчет могут наследовать примененный к таблице-источнику записей фильтр. Мастера по разработке форм и отчетов автоматически создают инструкцию SQL определяющую источник записей для формы или отчета, поэтому отпадает необходимость в создании запроса.

Для изменения вида формы, отчета или отдельных элементов может быть использован *мастер, вызываемый кнопкой [Автоформат]*.

Мастер подстановок создает в поле таблицы раскрывающийся список значений из другой таблицы для выбора и ввода нужного значения. Для создания такого поля со списком достаточно в режиме конструктора таблицы выбрать тип данных этого поля – Мастер подстановок. Мастер подстановок можно вызвать в режиме таблицы командой меню Вставка|Столбец подстановок. Созданный в данном поле таблицы список наследуется при включении этого поля в форму.

Мастера по импорту/экспорту позволяют просматривать данные при импорте/экспорте текста или электронных таблиц, а также при экспорте данных Microsoft Access в текстовые файлы.

Мастер защиты при необходимости эвакуирует данные, для чего создает новую базу данных, копирует в нее все объекты из исходной базы данных, снимает все права, присвоенные членам группы пользователей, и шифрует новую базу данных. После завершения работы мастера администратор может присвоить новые права доступа пользователям и группам пользователей.

Мастер по разделению базы данных позволяет разделить ее на два файла, в первый из которых помещаются таблицы, а во второй – запросы, формы, отчеты, макросы и модули. При этом пользователи, работающие в сети, имея общий источник данных, смогут устраивать формы, отчеты и другие объекты, используемые для обработки данных, по своему усмотрению.

Мастера MSAccess как средство создания реляционных БД использует все достоинства технологии MS Windows.

Достоинства средств MS Access:

1. СУБД MS Access полностью совместима с такими компонентами пакета MicrosoftOffice, как электронные таблицы Excel и текстовый процессор Word.

2. MS Access обеспечивает возможность динамического обмена данными DDT (DynamicDataExchange) с любым приложением Windows, поддерживающим DDE.

3. MS Access поддерживает также механизм OLE, обеспечивающий связь и внедрение объектов различных приложений, т.е. установление связи с объектами другого приложения и внедрение объекта в данное приложение базы данных. Причем достоинством внедренного объекта является то, что при его активизации открывается программа, которая его создала, поэтому новый пользователь имеет возможность изменить объект по своему усмотрению. При использовании механизма OLE как связи с объектом для другого приложения, объект по-прежнему сохраняется в файле приложения-источника. Следовательно, такой объект может обновляться независимо от приложения-потребителя, вызвавшего его, а в базе данных при этом можно всегда иметь последнюю версию объекта.

Внедряемыми или связываемыми объектами могут быть документы различных приложений – рисунки, графики, электронные таблицы или звуковые файлы. Например, в таблице наряду с обычными реквизитами, характеризующими информационный объект, может храниться любая

графическая информация о нем Windows схемы, чертежи, диаграммы и т. п. Таким образом, в MS Access расширяется традиционное понятие данных, хранимых в базе.

5. MS Access распространил широко используемый в Windows метод drag-and-drop (перетащить и отпустить) на работу с формами и отчетами. Например, для создания подчиненных формы и отчета можно заранее перетащить подготовленные форму и отчет из окна базы данных. Также можно перетащить таблицу и запрос, из которых автоматически создаются подчиненная форма и запрос.

MS Access может использовать данные других СУБД, т.е. в ней непосредственно могут обрабатываться файлы систем Paradox, dBase, PoxPro, Btrieve.

6. MS Access может использовать все файлы СУБД, поддерживающие стандарт открытого доступа к данным ODBC (OpenDatabaseConnectivity) – Orac1e, MSSQLServer, SybaseSQLServer и др.

Так, ODBC определяет язык и набор протоколов для обмена между пользовательским приложением и самими данными, хранящимися в сервере, т.е. используется как средство коммуникации между настольным персональным компьютером (клиентом) и сервером.

Основными компонентами (объектами) базы данных являются таблицы, запросы, формы, отчеты, страницы, макросы и модули.

1. *Таблицы.* Это фундаментальная структура системы управления реляционными базами данных. В базе данных информация хранится в виде двумерных таблиц, которые создаются пользователем для хранения информации о предметах или субъектах в определенной структуре. Можно так же импортировать и связывать таблицы из других СУБД или систем управления электронными таблицами. Каждая таблица характеризуется наличием записей (строки) и полей (столбцы).

2. *Запросы.* Требование на отбор данных, хранящихся в таблицах, или требование на выполнение определенных действий с данными. Запрос позволяет создать общий набор записей из данных, находящихся в разных таблицах, который будет служить источником данных для формы, отчета или страницы доступа к данным. Как правило, при создании запросов используется язык SQL. С помощью запросов выполняются операции по извлечению, созданию, изменению или удалению данных в базе данных.

Типы запросов, которые могут быть созданы с помощью Microsoft Access:

– запрос-выборка, задающий вопрос о данных, хранящихся в таблицах, и представляющий полученный динамический набор в режиме

формы или таблицы без изменения данных. Изменения, внесенные в динамический набор, отражаются в базовых таблицах;

– запрос-изменение, изменяющий или перемещающий данные. К этому типу относятся запрос на добавление записей, запрос на удаление записей, запрос на создание таблицы или запрос на ее обновление;

– перекрестные запросы, предназначенные для группирования данных и представления их в компактном виде;

– запрос с параметрами, позволяющий определить одно или несколько условий отбора во время выполнения запроса;

– запросы SQL, которые могут быть созданы только с помощью инструкций SQL в режиме SQL: запрос-объединение, запрос к серверу и управляющий запрос.

3. *Формы*. Объект базы данных Microsoft Access, в котором разрабочик размещает элементы управления, принимающие действия пользователей или служащие для ввода, отображения и изменения данных в полях. Формы позволяют отображать данные из таблиц и запросов в более удобном для восприятия виде. С помощью форм можно добавлять и изменять данные, содержащиеся в таблицах. В формы позволяется включать модули. Кроме этого, формы применяются для управления разработанным приложением, например, для выполнения какого-либо действия при возникновений определенного события.

4. *Отчёты*. Объект базы данных Microsoft Access, предназначенный для вывода на печать данных, организованных и отформатированных в соответствии со спецификациями пользователя. С помощью отчетов составляются коммерческие сводки, списки телефонов или списки рассылки.Отчеты представляют информацию, содержащуюся в таблицах базы данных в виде итогов, а также выводят информацию в определенном формате на печать.

5. *Страницы*. Представляют собой объекты, которые обеспечивают доступ к информации, имеющейся в БД, из сети Internet посредством браузера InternetExplorer. Каждая страница, как правило, представляет собой HTML-файл, посредством которого пользователи Internet получают доступ к имеющейся БД.

Страницы доступа к данным имеют следующие преимущества перед печатными отчетами:

– Страницы, присоединенные к данным, отображают текущие данные благодаря наличию подключения к базе данных.

– Страницы интерактивны. Пользователи имеют возможность фильтровать, сортировать и просматривать нужные записи.

– Страницы могут распространяться в электронном виде с помощью электронной почты. Получатели будут видеть текущие данные при каждом открытии сообщения.

6. *Макросы* – одна или несколько макрокоманд, которые можно использовать для автоматизации конкретной задачи.

7. *Модуль* – набор описаний, инструкций и процедур, сохраненных под одним именем. В Microsoft Access имеется три типа модулей: формы, отчета и общий. Модули форм и отчетов содержат локальную программу для форм или отчетов. Если процедуры общего модуля явным образом не объявлены личными для модуля, в котором они появляются, значит, они распознаются и могут вызываться процедурами из других модулей этой базы данных.

База данных может содержать несколько модулей, в том числе общие модули, модули форм и модули отчетов.

8. *Модули форм и отчетов*. Предназначены для сохранения процедур, связанных с обработкой событий в форме или отчете (щелчок мыши на элементе формы или отчета, двойной щелчок мыши, получение элементов фокуса и т. д.), а также обычных процедур для типовых действий.

В состав Access входит множество мастеров, построителей и надстроек, которые позволяют упростить процесс создания объектов базы данных.

Лекция 6

Этапы проектирования базы данных Microsoft Access

Основные вопросы

1. Определение цели создания базы данных.
2. Определение таблиц, которые должна содержать база данных.
3. Определение необходимых в таблице полей.
4. Задание индивидуального значения каждому полю.
5. Определение связей между таблицами.
6. Обновление структуры базы данных.
7. Добавление данных и создание других объектов базы данных.
8. Использование средств анализа в Microsoft Access.

Цель. Изучение основных этапов проектирования базы данных Microsoft Access.

В Microsoft Access, прежде чем создавать таблицы, формы и другие объекты необходимо задать структуру базы данных. Хорошая структура базы данных является основой для создания адекватной требованиям, эффективной базы данных.

К этапам проектирования баз данных можно отнести следующие:

1.Определение цели создания базы данных.
2.Определение таблиц, которые должна содержать база данных.
3. Определение необходимых в таблице полей.
4. Задание индивидуального значения каждому полю.
5. Определение связей между таблицами.
6. Обновление структуры базы данных.
7. Добавление данных и создание других объектов базы данных.
8. Использование средств анализа в Microsoft Access.

Определение цели создания базы данных

На первом этапе проектирования базы данных необходимо определить цель создания базы данных, основные ее функции и информацию, которую она должна содержать. То есть нужно определить основные темы таблиц базы данных и информацию, которую будут содержать поля таблиц.

База данных должна отвечать требованиям тех, кто будет непосредственно с ней работать. Для этого нужно определить темы, которые должна покрывать база данных, отчеты, которые она должна выдавать, проанализировать формы, которые в настоящий момент используются для

записи данных, сравнить создаваемую базу данных с хорошо спроектированной, подобной ей базой.

Определение таблиц, которые должна содержать база данных.

Одним из наиболее сложных этапов в процессе проектирования базы данных является разработка таблиц, так как результаты, которые должна выдавать база данных (отчеты, выходные формы и др.) не всегда дают полное представление о структуре таблицы.

При проектировании таблиц вовсе не обязательно использовать Microsoft Access. Сначала лучше разработать структуру на бумаге. При проектировании таблиц рекомендуется руководствоваться следующими основными принципами:

– Информация в таблице не должна дублироваться. Не должно быть повторений и между таблицами.

Когда определенная информация храниться только в одной таблице, то и изменять ее придется только в одном месте. Это делает работу более эффективной, а также исключает возможность несовпадения информации в разных таблицах. Например, в одной таблице должны содержаться адреса и телефоны клиентов.

– Каждая таблица должна содержать информацию только на одну тему.

Сведения на каждую тему обрабатываются намного легче, если содержаться они в независимых друг от друга таблицах. Например, адреса и заказы клиентов хранятся в разных таблицах, с тем, чтобы при удалении заказа информация о клиенте осталась в базе данных.

3. Определение необходимых в таблице полей.

Каждая таблица содержит информацию на отдельную тему, а каждое поле в таблице содержит отдельные сведения по теме таблицы. Например, в таблице с данными о клиенте могут содержаться поля с названием компании, адресом, городом, страной и номером телефона. При разработке полей для каждой таблицы необходимо помнить:

– Каждое поле должно быть связано с темой таблицы.

– Не рекомендуется включать в таблицу данные, которые являются результатом выражения.

– В таблице должна присутствовать вся необходимая информация.

– Информацию следует разбивать на наименьшие логические единицы (Например, поля «Имя» и «Фамилия», а не общее поле «Имя»).

4. Задание индивидуального значения каждому полю.

С тем чтобы Microsoft Access мог связать данные из разных таблиц, например, данные о клиенте и его заказы, каждая таблица должна

содержать поле или набор полей, которые будут задавать индивидуальное значение каждой записи в таблице. Такое поле или набор полей называют основным ключом.

5. Определение связей между таблицами.

После распределения данных по таблицам и определения ключевых полей необходимо выбрать схему для связи данных в разных таблицах. Для этого нужно определить связи между таблицами.

Желательно изучить связи между таблицами в уже существующей базе данных.

6. Обновление структуры базы данных.

После проектирования таблиц, полей и связей необходимо еще раз просмотреть структуру базы данных и выявить возможные недочеты. Желательно это сделать на данном этапе, пока таблицы не заполнены данными.

Для проверки необходимо создать несколько таблиц, определить связи между ними и ввести несколько записей в каждую таблицу, затем посмотреть, отвечает ли база данных поставленным требованиям. Рекомендуется также создать черновые выходные формы и отчеты и проверить, выдают ли они требуемую информацию. Кроме того, необходимо исключить из таблиц все возможные повторения данных.

7. Добавление данных и создание других объектов базы данных.

Если структуры таблиц отвечают поставленным требованиям, то можно вводить все данные. Затем можно создавать любые запросы, формы, отчеты, макросы и модули.

8. Использование средств анализа в Microsoft Access.

В Microsoft Access существует два инструмента для усовершенствования структуры баз данных. Мастер анализа таблиц исследует таблицу, в случае необходимости предлагает новую ее структуру и связи, а также переделывает ее.

Анализатор быстродействия исследует всю базу данных, дает рекомендации по ее улучшению, а также осуществляет их.

Защита информации в базах данных

Существуют различные приемы управления доступом к базе данных Microsoft Access и ее объектам. Эти приемы кратко описаны ниже в порядке повышения уровня безопасности.

Кодирование и декодирование

Кодирование базы данных – это простейший способ защиты. При кодировании базы данных ее файл сжимается и становится недоступным для чтения с помощью служебных программ или текстовых редакторов.

Кодирование незащищенной базы данных неэффективно, поскольку каждый сможет открыть такую базу данных и получить полный доступ ко всем ее объектам. Кодирование обычно применяется при электронной передаче базы данных или сохранении ее на дискету, кассету или компакт-диск.

Чтобы приступить к кодированию базы данных Microsoft Access, необходимо быть либо ее владельцем, либо, если база данных использует средства защиты, членом группы «Admins» в файле рабочей группы, который содержит учетные записи, используемые для защиты базы данных. Кроме того, базу данных надо открыть в монопольном режиме, для чего необходимо иметь разрешения «открытие/запуск» и «монопольный доступ».Декодирование базы данных является операцией, обратной кодированию.

Отображение и скрытие объектов в окне базы данных

Другим способом защиты объектов в базе данных от посторонних пользователей является скрытие объектов в окне базы данных. Этот способ защиты является наименее надежным, поскольку относительно просто можно отобразить любые скрытые объекты.

Использование параметров запуска

Параметры запуска позволяют задать такие настройки, как стартовая форма, которая автоматически открывается при открытии базы данных, а также заголовок и значок приложения базы данных. Кроме того, можно скрыть окно базы данных и установить собственную кнопочную форму. В новой базе данных параметры запуска отсутствуют до тех пор, пока не внесены изменения в диалоговом окне Параметры запуска.

Использование пароля

Другим простейшим способом защиты является установка пароля для открытия базы данных (.mdb). После установки пароля при каждом открытии базы данных будет появляться диалоговое окно, в которое требуется ввести пароль. Только те пользователи, которые введут правильный пароль, смогут открыть базу данных. После открытия базы данных все объекты становятся доступными для пользователя (пока не определены другие типы защиты, описанные ниже в этом разделе). Для базы данных, которая совместно используется небольшой группой пользователей или на автономном компьютере, обычно оказывается достаточно установки пароля.

Microsoft Access хранит пароль базы данных в незашифрованном виде. Если это нарушает безопасность защищаемой паролем базы данных, то для защиты базы данных не следует использовать пароль. Вместо этого

определите защиту на уровне пользователей, которая помогает управлять доступом к важной информации в базе данных.

Использование защиты на уровне пользователя

Наиболее гибкий и распространенный способ реализации средств защиты базы данных называют защитой на уровне пользователя. Защита на уровне пользователя позволяет установить различные уровни доступа к важным данным и объектам в базе данных. Чтобы воспользоваться базой данных, защищенной на уровне пользователя, необходимо ввести пароль при запуске Microsoft Access. После этого анализируется файл рабочей группы, в котором каждый пользователь идентифицируется уникальным кодом. Уровень доступа и объекты, доступ к которым получает пользователь, зависят от кода и пароля.

Хотя установка защиты на уровне пользователей для большинства баз данных является сложной задачей, мастер защиты позволит быстро и легко защитить базу данных. Более того, благодаря использованию общих схем защиты мастер позволяет уменьшить или даже вообще исключить необходимость использования команды Защита в меню Сервис.

После запуска мастера защиты можно создать собственные группы пользователей и определить разрешения на работу с базой данных и ее таблицами, запросами, формами, отчетами и макросами для различных пользователей или групп пользователей. Также могут быть установлены разрешения на доступ, по умолчанию присваиваемые вновь создаваемым объектам базы данных. Группам и пользователям предоставляются разрешения, определяющие возможность их доступа к каждому объекту базы данных.

Запрещение репликации базы данных, установки паролей и настройки параметров запуска пользователями

В многопользовательской среде часто возникают ситуации, требующие использования средств защиты базы данных. Возможно, потребуется запретить репликацию базы данных. Репликация позволяет пользователям создавать копию общей базы данных, а также добавлять поля и вноситьдругие изменения в текущую базу данных. Кроме того, может потребоваться запрещение установки пароля базы данных пользователями, поскольку, если это произойдет, никто, не зная пароля, не сможет открыть базу данных. Также следует рассмотреть возможность установки запрета на изменение параметров запуска, которые определяют такие свойства, как настраиваемые меню, настраиваемые панели инструментов и стартовую форму.

Если общая база данных не имеет защиты на уровне пользователей, невозможно запретить пользователям вносить подобные изменения. При установке защиты на уровне пользователей пользователь или группа для репликации базы данных, установления пароля базы данных и изменения параметров запуска должны иметь такое же разрешение на доступ, как и администратор. Только члены группы «Admins» текущей рабочей группы имеют права администратора.

Если пользователь или группа имеют в настоящий момент разрешение на доступ к базе данных, соответствующее полномочиям администратора, удаление разрешения запретит пользователю или группе внесение изменений. Можно присвоить соответствующее разрешение на доступ пользователю или группе, что позволит им выполнять эти задачи. Невозможно независимо управлять доступом для каждой из таких задач.

Защита страниц доступа к данным

Страницей доступа к данным называют файл HTML, содержащий ссылки на данные в файле Microsoft Access. Однако страницы доступа к данным фактически сохраняются не в файле Microsoft Access, а в виде файла HTML в локальной файловой системе, в папке на общем сетевом ресурсе или на HTTP-сервере. Из-за этого не обеспечивается контроль за безопасностью файлов страниц доступа к данным. Чтобы защитить страницу доступа к данным, необходимо применить средства защиты для ссылки и файла HTML с помощью средств защиты файловой системы компьютера, на котором эти файлы хранятся. Для защиты данных, доступ к которым осуществляется со страницы, необходимо либо применить средства защиты базы данных, к которой подключена страница, либо задать настройки безопасности Microsoft InternetExplorer для предотвращения несанкционированного доступа.

Лекция 7

Инфологическое моделирование баз данных

Основные вопросы

1. Понятие концептуальной модели.
2. Основные компоненты концептуальной модели.
3. Требования, предъявляемые к концептуальной модели.
4. Преимущества использования ER-модели.
4. Описание базовой ER-модели.
5. Разновидности объектовконцептуальной модели.

Цель. Изучение основных понятий инфологического моделирования баз данных.

Понятие концептуальной модели

Концептуальное проектирование является центральной частью, ядром всего процесса проектирования баз данных. Подходы к концептуальному проектированию, излагаемые в разных литературных источниках и реализованные в разнообразных CASE-системах, отличаются друг от друга. Многие из подходов имеют существенные недостатки, что не позволяет рекомендовать именно их как основные. В связи с этим опишем некоторую базовую модель, с которой можно проводить сравнение других систем.

Так как в настоящее время CASE-систем достаточно много, то неизвестно, с какой именно из систем придется проектировщику столкнуться на практике. Поэтому приведем некоторые критерии, по которым следует сравнивать CASE-системы, и обобщенные рекомендации по построению ER-моделей в зависимости от доступных изобразительных средств и алгоритмов проектирования логической структуры базы данных.

В базе данных отображается какая-то часть реального мира. Естественно, что полнота ее описания будет зависеть от целей создаваемой информационной системы. Как указано выше, часть реального мира, представляющая интерес для данного исследования, называется предметной областью. Для того чтобы база данных адекватно отражала предметную область, проектировщик должен хорошо представлять себе все нюансы, присущие ей, и уметь отобразить их в базе данных.

Предметная область должна быть предварительно описана. Для этого в принципе может использоваться и естественный язык, но его применение имеет много недостатков, основные из которых – громоздкость описания и неоднозначность его трактовки. Поэтому обычно для этих целей используют искусственные формализованные (чаще всего – графические)

языковые средства.

Формализованное описание предметной области будем называть ее концептуальной моделью. Предметные области могут быть различными, и для их моделирования могут потребоваться специфические средства, соответствующие особенностям этих областей. Мы будем ориентироваться в основном на экономико-организационные системы, хотя описываемые далее подходы к проектированию являются более универсальными и могут быть использованы и в других предметных областях.

Моделирование предметных областей выполняется с разными целями, например для реинжиниринга бизнесс-процессов, для прогнозирования развития предметной области, при проектировании баз данных и программного обеспечения и т.п. Используемые средства и методы моделирования при этом будут различаться. Мы остановимся здесь только на средствах и методах моделирования, которые находят наибольшее использование при проектировании баз данных.

Поскольку существует большое разнообразие видов БД, то подходы к проектированию баз данных разных классов будут существенно различаться. В настоящее время основную часть баз данных представляют структурированные базы данных, поэтому основное внимание будем уделять проектированию именно таких систем.

Изучение предметной области складывается из непосредственного наблюдения протекающих в ней процессов, изучения документов, циркулирующих в системе, а также интервьюирования участников этих процессов (рис. 1).

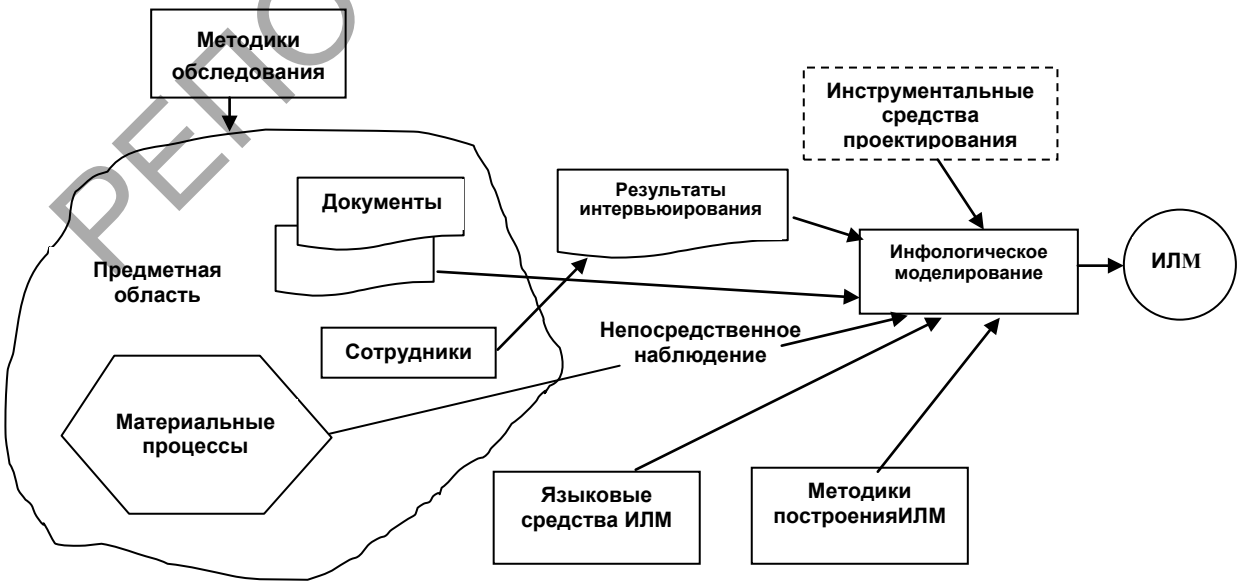


Рис. 1. Исходная и результирующая информация инфологического

моделирования

Так как описание инфологической модели выполняется на специализированном языке, то необходимо владение этим языком. Следует обратить внимание на то, что возможности языка описания ИЛМ оказывают влияние на методику построения модели с использованием данных языковых средств. Построение концептуальной модели может выполняться и вручную, и с использованием автоматизированных средств проектирования. Средства автоматизации проектирования отличаются как нотациями используемых языковых средств, так и алгоритмами преобразования концептуальной модели в модели базы данных. Это, в свою очередь, скажется на методике построения модели в их среде.

Основные компоненты концептуальной модели

Основными компонентами концептуальной модели ПО являются (рис. 2):

– описание объектов ПО и связей между ними;

– описание информационных потребностей пользователей;

– описание существующей информационной системы (документы.документооборот, при наличии автоматизированной информационной системы - ее описание);

– описание алгоритмических зависимостей показателей;

– описание ограничений целостности;

– описание функциональной структуры системы, для которой создается АИС;

– требования к ИС и существующие ограничения;

– лингвистические отношения.

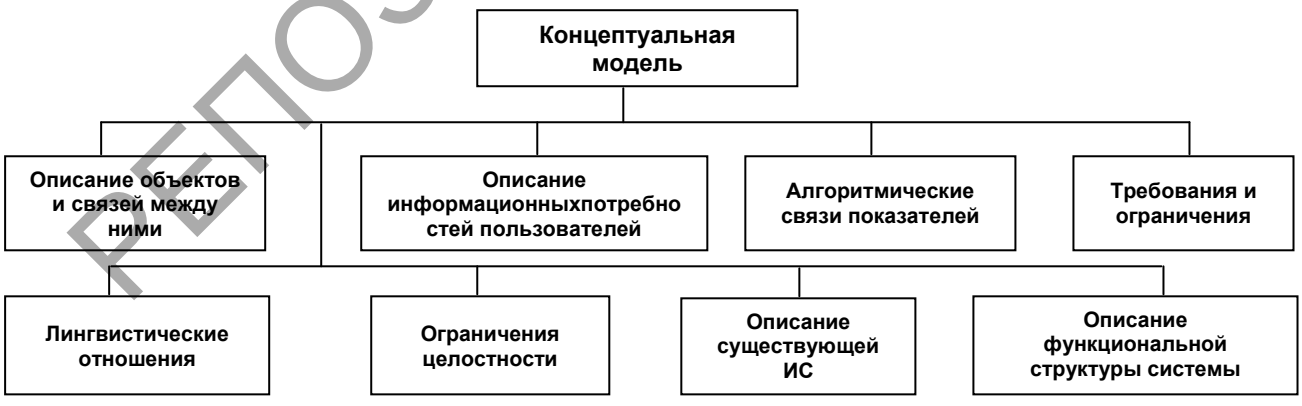


Рис. 2. Компоненты концептуальной модели

Далее более подробно остановимся на первом из перечисленных компонентов, так как именно он оказывает наибольшее влияние на проектирование структуры базы данных. Чаще всего описание объектов ПО и связей между ними представляется в виде так называемых ER-

моделей (или ER-диаграмм).

Требования, предъявляемые к концептуальной модели

К концептуальной модели предъявляются следующие требования:

– адекватное отображение предметной области (язык для представления модели должен обладать достаточными выразительными возможностями для отображения явлений, имеющих место в предметной области, а сама модель должна содержать всю необходимую и достаточную информацию для дальнейшего проектирования системы);

– непротиворечивость (модель отражает взгляды и потребности всех пользователей системы, а также обычно является результатом работы многих специалистов, поэтому целостное описание ПО должно быть проверено на непротиворечивость);

– однозначная трактовка модели всеми ее пользователями (обеспечивается формализованностью языка и четким его пониманием всеми участниками процесса создания ИС);

– легкость восприятия разными категориями пользователей (обеспечивается выбором соответствующего языка моделирования);

– конечность модели (несмотря на то, что реальный мир, отображаемый в КМ, является по своей природе бесконечным, инфологическая модель является конечной, что обеспечивается четким ограничением предметной области);

– легкость модификации (в концептуальную модель по разным причинам часто приходится вводить новые объекты или модифицировать существующие; КМ должна в связи с этим обладать свойством легкой расширяемости, обеспечивающим ввод новых данных без изменения раннее определенных.То же самое можно сказать и об удалении и корректировке данных);

– возможность композиции и декомпозиции модели. Желательно, чтобы язык спецификации концептуальной модели был одинаковым как при ручном, так и при автоматизированном проектировании информационных систем. Последнее предъявляет к языку дополнительные требования, а именно, он должен:

– быть вычисляемым, т.е. восприниматься и обрабатываться ЭВМ;

– использовать «дружелюбные» пользователю интерфейсы, в частности графические;

– быть независимым от оборудования и других ресурсов, которые подвержены частым изменениям;

– использовать средства тестирования КМ, а также иметь аппарат

для указания того, что спецификация завершена и по ней может быть выполнена генерация структур баз данных.

При автоматизированном проектировании все изменения, внесенные в КМ, должны быть автоматически отражены в связанных с модифицируемым элементом компонентах банка данных.

Желательно, чтобы КМ строили специалисты, работающие в предметной области, для которой создается АИС, а не проектировщики систем машинной обработки данных. Если в силу определенных причин это невозможно, то необходимо, чтобы первые могли хотя бы проверить сделанное другим специалистом описание, чтобы убедиться в том, что специфика предметной области воспринята и отображена правильно.

Концептуальная модель является средством коммуникации разнообразных коллективов как конечных пользователей, так и разработчиков. Информация из КМ корреспондирует со словарной системой и другими компонентами банка данных.

Преимущества использования ER-моделирования

ER-модель представляет собой графическое описание предметной области в терминах «обьект-свойство-связь». ER-модель является одним из элементов концептуальной модели. Использование ER-моделирования (особенно в сочетании с автоматизированными средствами проектирования – CASE-средствами) дает много преимуществ:

– предписывая определенную методологию моделирования, делает анализ предметной области более целенаправленным и конкретным;

– является удобным средством документирования проекта;

– позволяет вести проектирование АИС без привязки к конкретной целевой СУБД и осуществлять выбор последней в любой момент времени (чем ближе к концу проектирования это будет сделано, тем точнее может быть выбор).

При использовании ER-моделирования в составе CASE-средств появляются дополнительные преимущества:

– снижаются требования к знанию деталей языков описания данных (DDL –DataDefinitionLanguage) и диалектов SQL конкретных СУБД;

– при смене используемой СУБД не нужно проводить проектирование заново; следует только осуществить шаг по переводу ER-модели в целевую (если выбранная вами целевая СУБД поддерживается данным CASE-средством, то такой переход вообще будет выполнен автоматически);

– наличие в CASE-средстве возможности «обратного проектирования» (т.е. получения ER-диаграммы по имеющимся описаниям данных)

позволяет использовать существовавшие ранее наработки для «реверс-инжиниринга» системы;

– указание связи объектов в ER-модели и соответствующая миграция ключа при преобразовании этой модели в целевую не только позволяют задавать контроль целостности связи при ведении БД, но и автоматически обеспечивают согласованное описание схемы (внешний ключ мигрирует в связанное отношение; при этом имя, тип и длина соответствующего атрибута повторяются в зависимой сущности);

– сокращается время проектирования системы;

– появляется возможность автоматизированного тестирования проекта на всех этапах проектирования;

– повышается качество документирования проекта;

– мощные современные CASE-средства позволяют вести коллективную разработку проекта.

Описание базовой ER-модели

В предметной области имеется множество разнообразных объектов. Объект – понятие широкое. Его трудно точно определить. Обычно под объектом понимают некую сущность (реальную или абстрактную), о которой собирается какая-то информация. Объекты группируются в классы. Классом объектов называют совокупность объектов, обладающих одинаковым набором свойств. Например, для объектов класса СТУДЕНТ таким набором свойств являются: «Год-рождения», «Пол» и др.

Объекты могут быть реальными, как названный выше объект СТУДЕНТ, и абстрактными, как, например, ПРЕДМЕТЫ, которые изучают студенты.

ER-модель строится на уровне классов объектов, а не отдельных экземпляров объектов.

Каждому классу объектов в ER-модели присваивается уникальное имя. Именем класса объекта является грамматический оборот существительного (существительное, у которого могут быть прилагательные и предлоги). Если имя состоит из нескольких слов, то желательно, чтобы первым стояло существительное. Существительное должно употребляться в единственном, а не во множественном числе (например, ДИСЦИПЛИНА_ИЗУЧАЕМАЯ). Если в предметной области традиционно используются разные имена для обозначения какого-либо класса объектов (т.е. имеет место синонимия), то все они должны быть зафиксированы при описании системы, и затем одно из них выбирается за основное, и только оно должно в дальнейшем использоваться в ER-модели. Помимо имени класса объектов в ER-модели может использоваться его

короткое кодовое обозначение; для дальнейшего перехода к даталогической модели еще может указываться имя, которое будет использоваться при описании структуры базы данных.

При построении ER-модели желательно дать словесную интерпретацию каждой сущности, особенно если возможно неоднозначное толкование понятия.

Вместо термина «объект» часто используется термин «сущность». В дальнейшем будем рассматривать эти термины как синонимы.

При отражении в информационной системе каждый объект (имеется в виду уже экземпляр объекта, а не весь класс) представляется своим именем, которое называет конкретный объект и отличает один объект от другого. Чтобы выполнять свою роль, имя должно быть уникальным, но иногда в реальной жизни это бывает не так (явление синонимии). Поэтому в концептуальной модели должны быть каким-то образом обозначены случаи, когда естественное имя объекта является неуникальным. Уникальное имя объекта будем называть идентификатором (ИО). Каждый объект должен иметь, по крайней мере, один идентификатор.

Каждый объект обладает определенным набором свойств - характеристик, описывающих состояние каждой сущности. Набор (перечень) свойств для всех объектов данного класса будет одинаковым, но конкретные значения этих характеристик и особенно сочетание этих значений будут отличаться от объекта к объекту, что, собственно, и отличает один экземпляр объекта от другого.

Может случиться, что в данной предметной области свойства объекта не представляют для нас интереса. В связи с этим в ER-модели возможно наличие объектов, не имеющих свойств и представленных только своими идентификаторами.

Разновидности объектов

Различают несколько разновидностей объектов. Прежде всего, это простые и сложные объекты. Объект называется простым, если он рассматривается в данном исследовании как неделимый.

Выделяют несколько разновидностей сложных объектов: составные, обобщенные и агрегированные объекты.

Сложный объект представляет собой объединение других объектов, простых или сложных, также отображаемых в информационной системе. Понятия «простой» и «сложный» объект являются относительными. В одном случае объект может считаться простым, а в другом - этот же объект может рассматриваться как сложный. Так, например, объект АУДИТОРИЯ, если АИС строится только для управления учебным

процессом, будет рассматриваться как простой. Если же АИС будет включать подсистемы для служб энергетика, материально-технического снабжения и др., то АУДИТОРИЯ будет рассматриваться как составной объект

Составной объект соответствует отображению отношения «целое-часть». Примерами составных объектов являются УЗЕЛ-ДЕТАЛИ, КЛАСС-УЧЕНИКИ и т.п.

Обобщенный объект отражает наличие связи «род-вид» между объектами предметной области. Например, объекты СТУДЕНТ, ШКОЛЬНИК, АСПИРАНТ, УЧАЩИЙСЯ_ТЕХНИКУМА образуют обобщенный объект УЧАЩИЙСЯ. Объекты, составляющие обобщенный объект, называются его категориями.

Как родовой объект, так и видовые объекты могут обладать определенным набором свойств. Причем наблюдается так называемое наследование свойств, Т.е. видовой объект обладает всеми теми свойствами, которыми обладает родовой объект, плюс свойствами, присущими только объектам этого вида.

Определение родовидовых связей означает классификацию объектов предметной области по тем или иным признакам. Естественно, что классификация может быть многоуровневой.

Агрегированные объекты соответствуют обычно какому-либо процессу, в который оказываются вовлеченными другие объекты. Например, агрегированный объект ПОСТАВКА объединяет в себе объекты ПОСТАВЩИК, ПОТРЕБИТЕЛЬ, а также саму поставляемую ПРОДУКЦИЮ. Своеобразным объектом является ДАТА_ПОСТАВКИ. Агрегированный объект может, так же как и простой объект, иметь характеризующие его свойства. В рассматриваемом примере таким свойством может быть «Размер_поставки». Агрегированные объекты обычно выражаются отглагольными существительными.

3 ПРАКТИЧЕСКИЙ РАЗДЕЛ

3.1 Лабораторные работы

Лабораторная работа 1

Создание базы данных

*Цель работы:*Сформировать умения создавать структуры таблиц и устанавливать связи между ними.

Создание файла пустой базы данных выполняют следующим образом. ЗапускаютMSAccess и нажимают на кнопку**Создать** на панели **База данных** (можно выполнить команду **Создать** в меню **Файл**). На панели **Область задач** выбирают задачу **Новая база данных**. В появившемся диалоговом окне**Файл новой базы данных** указывают имя создаваемого файла базы данных и открывают папку, в которой будет храниться база данных. После этого нажимают на кнопку**Создать**. На экране появится окно созданной пустой базы данных.

Для создания структуры (макета) таблицы MSAccess во вкладке**Таблицы**окна базы данных предлагает три способа: в режиме конструктора, с помощью мастера и путем ввода данных. Если надо осуществить импорт таблицы из другого приложения или установить связь с таблицей, созданной другим приложением, можно поступить следующим образом: во вкладке **Таблицы**окна базы данных нажать кнопку**Создать**и из появившегося окна **Новая таблица** выбрать команду **Импорт таблиц** или **Связь с таблицами** соответственно. После этого в диалоговом окне **Новая таблица** надо указать режим создания таблицы.

Наиболее удобным способом создания структуры таблицы является способ, использующий режим конструктора. При его использовании выполняют двойной щелчок мышью на команде **Создание таблицы в режиме конструктора** во вкладке **Таблицы**окна базы данных. На экране появится окно таблицы в режиме конструктора (см. рис. 1).

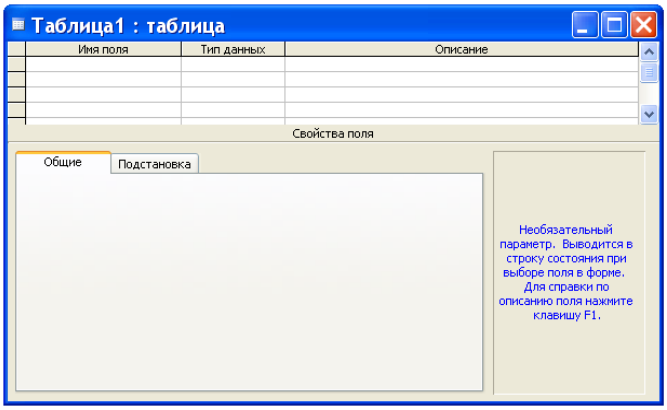


Рис.1. Окно таблицы в режиме конструктора.

В области проекта таблицы указывают имена полей, типы данных и описание. Колонка **Описание** является не обязательной – в ней можно записывать комментарии. Нижняя часть окна таблицы в режиме таблицы может быть использована для установки свойств данных, отличных от принципа умолчания.

После того, как в таблицу, представленную в режиме таблицы, внесена необходимая информация, ее следует сохранить. Для этого в меню **Файл** можно выбрать команду**Сохранить** или на панели **База данных** нажать на кнопку **Сохранить**. В появившемся окне **Сохранение** указывают имя сохраняемой таблицы и нажимают кнопку **ОК**.

Когда все макеты таблиц для проектируемой базы данныхсозданы, приступают к установке связей между таблицами. Это делается именно в этот период для того, чтобы при вводе данных в таблицы MSAccess проверял целостность данных.

Установка связей между таблицами выполняется в окне **Схема данных**, которое выводится на экран нажатием кнопки **Схема данных** на панели **База данных**. При этом появляется диалоговое окно **Добавление таблицы**, в котором надо выделить имена тех таблиц, между которыми будут устанавливаться связи. После этого нажимают кнопки**Добавить** и **Закрыть**. Затем в окне **Схема данных** с помощью мыши перетаскивают ключевое поле одной таблицы на соответствующее поле в другой таблице. В появившемся окне **Связи** задают режим **Обеспечение целостности данных** и его подрежимы: **каскадное обновление связанных полей** и **каскадное удаление связанных записей** и нажимают кнопку**Создать**.

Задание

1. Создайте файл пустой базы данных **Библиотека**.

2. Создайте структуру таблицы **Издательства**, которая содержит следующие поля: **Код издательства**, **Наименование**, **Город** (см. таблицу 1).Имена полей таблиц из базы данных **Библиотека**: **Издательства**, **Книги** и **Темы**, типы данных, свойства полей, отличные от принципа умолчания, а также поля, являющиеся ключами, приведены в таблице 1.

Таблица 1

Поля таблиц базы данных **Библиотека**

Название таблицы	**Имя поля**	**Тип данных**	**Свойства поля**	**Ключ**
Издательства	Код издательства	**Числовой**	Размер поля – целое, обязательное поле	Да
Издательства	Наименование	Текстовый	Размер поля – 20	

Издательства	Город	Текстовый	Размер поля – 15	
Книги	Код книги	Числовой	Размер поля – целое, обязательное поле	Да
Книги	Название	Текстовый	Размер поля – 25	
Книги	Автор	Текстовый	Размер поля – 15	
Книги	Код издательства	Числовой	Размер поля – целое, обязательное поле	
Книги	Объем	Числовой	Размер поля – целое	
Книги	Год издания	Числовой	Размер поля – целое	
Книги	Стоимость	Денежный	Формат поля – денежный	
Темы	Код книги	Числовой	Размер поля – целое, индексированное поле (совпадения допускаются)	
Темы	Тема	Текстовый	Размер поля – 30	

После того, как вы набрали имена полей таблицы **Издательства**, указали для них типы данных, установили требуемые свойства и определили ключ, нажмите на кнопку**Закрыть** окна таблицы в режиме конструктора. При этом появится сообщение: **Сохранить изменения макета или структуры таблицы «Таблица1» ?**Ответьте на него утвердительно. После этого появится диалоговое окно **Сохранение**, в котором наберите имя таблицы **Издательства** и нажмите копку **ОК**.

3. Описанные выше действия выполните для создания таблиц **Книги** и **Темы** базы данных **Библиотека**.

4. Установите связи между таблицами **Издательства**, **Книги** и **Темы** базы данных **Библиотека**так, как это показано на рис. 2. Обратите внимание на то, что между таблицами базы данных **Библиотека**будут связи одного типа – один ко многим.

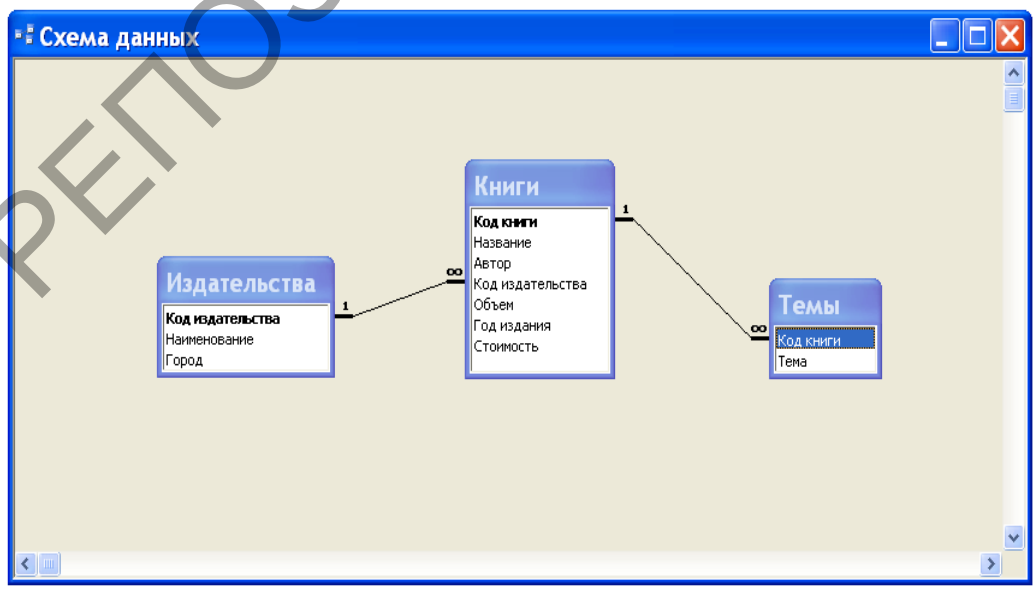


Рис. 2. Схема базы данных **Библиотека.**

Поскольку при установке связей между таблицами мы указали режим обеспечения целостности данных с его подрежимами, то тем самым мы задали порядок, в котором надо вводить данные в таблицы. Вначале надо вводить данные в таблицу **Издательства**, затем – в таблицу **Книги** и лишь после этого – в таблицу **Темы**.

5. Введите в таблицы **Издательства**, **Книги** и **Темы** базы данных **Библиотека** данные, приведенные ниже в таблицах 2-4. Ввод данных в таблицу осуществляется в режиме таблицы. Чтобы открыть таблицу в режиме таблицы достаточно в окне базы данных во вкладке **Таблицы** выполнить двойной щелчок мышью на имени таблицы. Быстро перевести таблицу из режима конструктора в режим таблицы и наоборот можно нажатием кнопки **Вид** на панели **База данных**. Для перехода от одного поля к другому, когда осуществляется ввод данных в таблицу, удобно использовать клавишу **Tab**.

Таблица 2

Издательства

Код издательства	**Наименование**	**Город**
1	Наука	Москва
2	Мир	Москва
3	Радио и связь	Минск
4	Машиностроение	Киев
5	Финансы и статистика	Москва

Таблица 3

Книги

Код книги	**Название**	**Автор**	**Код издательства**	**Объем**	**Год издания**	**Стоимость**
1	Педагогика	Беспалько	2	340	1994	24 000,00р.
2	Сборник задач	Сканави	2	634	1992	60 000,00р.
3	Программирование	Арсак	1	273	1989	18 000,00р.
4	Язык Ада	Перминов	3	278	1987	16 000,00р.
5	Операционные системы	Грибанов	3	446	1991	23 000,00р.
6	БД на Паскале	Ульман	4	563	1992	32 000,00р.
7	IBM PC для	Фигурнов	5	368	1994	22 000,00р.

Код книги	Название	Автор	Код издательства	Объем	Год издания	Стоимость
	пользователя					

Таблица 4

Темы

Код книги	Тема	Код книги	Тема
1	Личность человека	5	Управление заданиями
1	Проектирование ППС	5	Управление задачами
1	Технология обучения	5	Управление данными
1	Анализ учебного процесса	6	Операции с поставщиками
2	Уравнения	6	Бухгалтерская книга
2	Прогрессии	6	Платежная ведомость
2	Геометрические задачи	6	Реляционная алгебра
3	Игры с числами	6	Правила нормализации
3	Игры без стратегии	7	Устройства компьютера
3	Комбинаторные задачи	7	Файлы и каталоги
3	Стратегия без игры	7	Диалог пользователя
4	Программные модули	7	Работа с дисками
4	Лексика	7	Программы архивации
4	Предопределенные типы	7	Конфигурирование системы
4	Операторы	7	Обслуживание дисков
5	Структура ОС ЕС	7	Редактирование текстов

Лабораторная работа 2

Работа с базой данных

Цель работы: Сформировать умения добавлять таблицы в базу данных с целью расширения ее функциональных возможностей.

Иногда в процессе разработки базы данных или в процессе опытной эксплуатациие возникает необходимость добавления в нее новых таблиц. Очевидно, что спроектированная нами в предыдущей работе база данных **Библиотека** обладает очень ограниченными возможностями. Эта база данных, состоящая из трех таблиц: **Издательства**, **Книги** и **Темы**, не

позволяет автоматизировать работу с читателями. В ней отсутствует информация о читателях.

В данной работе мы научимся добавлять таблицы в базу данных с целью расширения ее функциональных возможностей. Создание новых таблиц осуществляется точно так же, как это мы делали в предыдущей работе. Для добавления таблиц в ранее созданную схему данных и установления связи между таблицами используется кнопка**Отобразить таблицу**, размещенная на панели инструментов **Связь**.

Задание

1. Откройте базу данных **Библиотека**.Создайте в ней структуру таблицы**Читатели**, которая будет содержать следующие поля: **Код читателя**, **Фамилию**, **Имя**, **Отчество**, **Домашний телефон**, **Домашний адрес.**Типы данных для полей таблицы, их свойства определите самостоятельно по смыслу. В качестве ключа укажите поле **Код читателя**.

2. Аналогичным способом создайте структуру таблицы **Выдача книг**. В эту структуру включите три поля: **Код читателя**, **Код книги**, **Дата заказа**. В этой таблице ключевое поле не задавайте. Для поля **Дата заказа** укажите тип данных – **Дата/время**. Обратите внимание на то, что в последствии ключ **Код читатели** в таблице **Читатели** будет связываться с полем **Код читателя** в таблице **Выдача книг**. Поэтому эти поля должны иметь соответствующие типы данных и свойства.

3. Установите между добавленными таблицами:**Читатели** и **Выдача книг**, а также ранее созданными таблицами: **Издательства**, **Книги** и **Темы**, связи так, как это показано в окне **Схема данных** на рис. 1.

Напомним, что для установления связи между таблицами надо открыть окно**Схема данных.** При его открытии появляется диалоговое окно **Добавление таблицы**, в котором надо выделить имена тех таблиц, между которыми будут устанавливаться связи. После этого нажимают кнопки**Добавить** и **Закрыть**. Затем в окне **Схема данных** с помощью мыши перетаскивают ключевое поле одной таблицы на соответствующее поле в другой таблице. В появившемся окне **Связи** задают режим **Обеспечение целостности данных** и его подрежимы: **каскадное обновление связанных полей** и **каскадное удаление связанных записей** и нажимают кнопку**Создать**.

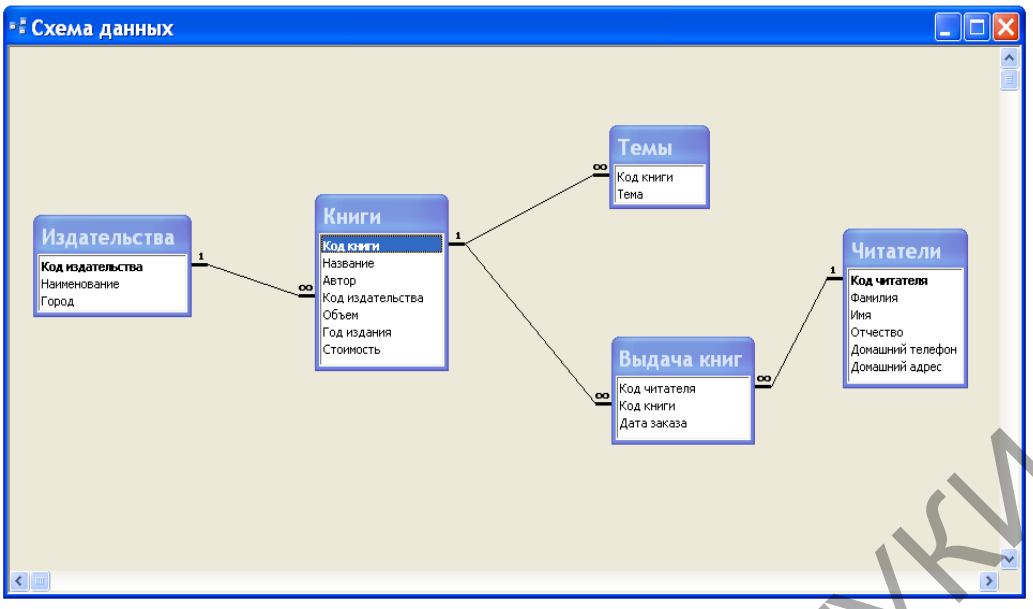


Рис. 3. Схема расширенной базы данных **Библиотека**.

4. Откройте таблицу **Читатели** и введите в нее данные, приведенные в таблице 1.

Таблица 1

Данные для ввода в таблицу **Читатели**

Кодчи тателя	Фамилия	Имя	Отчество	Домашни йтелефон	Домашнийадрес
1	Аксенов	Виктор	Сергеевич	252-88-13	ул. Есенина, 15-19
2	Голубева	Елена	Андреевна	220-99-29	ул. Чкалова, 7-38
3	Васильев	Игорь	Петрович	232-64-78	ул. Богдановича, 102-34
4	Кучеров	Валентин	Степанович	266-24-95	ул. Кнорина, 27-5
5	Мастяница	Вячеслав	Иванович	246-42-25	ул. Плеханова, 34-98
6	Победимская	Лариса	Анатольевна		ул. Чкалова, 9-10
7	Литвин	Борис	Николаевич	239-55-76	пр. Независимости, 46-54
8	Германович	Рита	Мироновна	278-31-51	ул. Казинца, 26-9
9	Бинцаровский	Теодор	Петрович		ул. Корженевская, 1-288

5. Введите в таблицу **Выдача книг** данные, приведенные в таблице 2.

Таблица 2

Данные для ввода в таблицу **Выдача книг**

Код читателя	Код книги	Дата заказа	Код читателя	Код книги	Дата заказа
1	1	1.09.07	4	3	7.01.08
1	3	5.07.08	4	4	25.10.07
1	4	21.10.07	5	2	23.04.08
2	1	4.11.07	6	1	18.06.08
3	2	3.08.08	7	3	20.01.08
8	7	25.12.07	9	6	2.02.08

Обратите внимание на то, что, если бы вы попробовали вначале ввести данные в таблицу **Выдача книг**, а затем в таблицу **Читатели**, то MSAccessэто не позволил бы сделать. Поэтому мы специально раньше установили связи между таблицами, а затем уже вводили данные в таблицы. В этом случае MSAccess будет проверять целостность данных.

Лабораторная работа 3

Создание простых запросов

*Цель работы:*Сформировать умения создавать простые запросы для выбора данных.

Запрос в MSAccess – это требование предоставить информацию, накопленную в таблицах базы данных. Запрос можно получить с помощью с помощью инструментов запроса. Запрос может относиться к одной или к нескольким связанным таблицам. На основании запроса MS Access формирует динамический набор записей. Физически он выглядит как таблица, хотя фактически не является ею. Динамический набор записей является временным (или виртуальным) набором записей и не хранится в базе данных. После закрытия запроса динамический набор записей этого запроса прекращает свое существование.

MS Access поддерживает различные типы запросов, которые можно разбить на шесть основных категорий.

Запрос на выборку. Извлекает данные из одной или нескольких таблиц (основываясь на заданных критериях) и результаты представляет в виде динамического набора записей.

Групповой запрос. Представляет специальную версию запроса на выборку. Позволяет вычислять суммы, подсчитывать количество записей и выполнять расчет итоговых значений. Для этого запроса MS Access добавляет в бланк запроса строку **Групповая операция.**

Запрос на изменение. Позволяет создавать новые таблицы (команда **Создание таблицы**) или изменять данные в существующих таблицах (команды **Удаление**, **Обновление** и **Добавление**). Если в наборе результатов запроса на выборку можно вносить изменения только в одну запись за раз, то запрос на изменение разрешает вносить изменения в несколько записей сразу при выполнении этой операции.

Перекрестный запрос. Отображает результаты статистических расчетов (такие как суммы, количество записей и средние значения). Эти результаты группируются по двум наборам данных в формате перекрестной таблицы. Первый набор выводится в столбце слева и образует заголовки строк, а второй выводится в верхней строке и образует заголовки столбцов.

Запрос SQL. Существуют три типа запросов SQL: запрос на объединение, запрос к серверу и управляющий запрос, которые используются для манипуляций с базами данных SQL. Создаются эти запросы с помощью написания специальных инструкций SQL.

Запрос с ограничением, или Top(n). Этот ограничитель запроса можно использовать только в паре с одним из предыдущих пяти типов запросов. Он позволяет задавать число первых записей или часть общего количества записей в процентах, которую вы хотели бы получить в любом виде запроса.

С помощью запросов можно выполнять следующее: выбирать таблицы, выбирать поля, выбирать записи, сортировать записи, выполнять вычисления, создавать таблицы, создавать формы и отчеты на основе запроса, создавать диаграммы на основе запроса, использовать запрос в качестве источника данных для других запросов (подчиненных запросов) и вносить изменения в таблицы.

Создание запроса и работа с ним выполняется во вкладке **Запросы** окна базы данных. Для работы с запросом можно воспользоваться панелью инструментов **Конструктор запросов**.

Рис. 1. Панель инструментов **Конструктор запросов**.

MSAccess допускает два способа создания запроса: с помощью мастера и в режиме конструктора. Для того чтобы приступить к созданию запроса с помощью мастера можно выполнить двойной щелчок мышью на строке **Создание запроса с помощью мастера** во вкладке **Запросы** окна базы данных или щелчок мышью на кнопке**Создать**, а затем выбрать вариант **Простой запрос** в окне диалога **Новый запрос**.

Создание запроса на выборку для сортировки информации

В работе далее для создания запросов будем использовать режим конструктора. Самый быстрый способ запустить этот режим – выполнить двойной щелчок мышью на строке **Создание запроса в режиме конструктора**. При этом появится окно диалога **Добавление таблицы** (см. рис. 2).

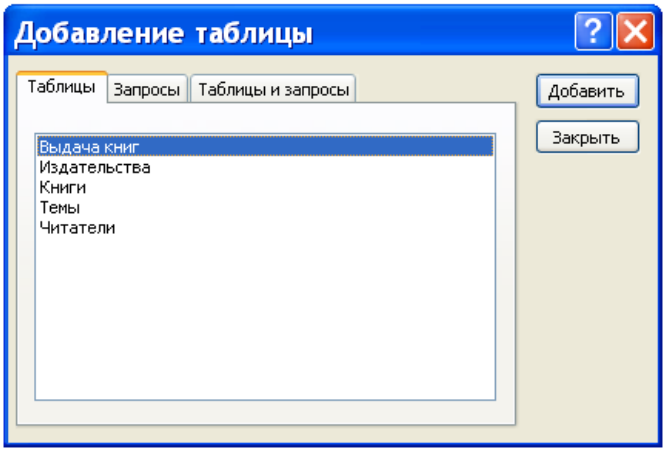


Рис. 2. Окно диалога **Добавление таблицы**.

Создание запроса для сортировки информации рассмотрим на следующем примере. Требуется составить список книг московских издательств, рассортированных по фамилиям авторов. В динамический набор надо включить следующие поля: **Автор**, **Название**, **Наименование** и **Год издания**.

Обратим внимание на то, что в нашем запросе будут использоваться поля из двух таблиц: **Издательства** и **Книги**. Поэтому в окне диалога надо выделить имена этих двух таблиц. Для этих целей щелкните вначале, например, по имени **Издательства**, а затем, удерживая клавишу **CTRL**, щелкните по имени **Книги**. После того как требуемые имена таблиц выделены, надо в окне диалога **Добавление таблицы** щелкнуть мышью по кнопке **Добавить**, а затем– **Закрыть**. В результате выполнения таких действий в верхней части окна запроса в режиме конструктора появятся списки полей для каждой из выбранных таблиц (см. рис. 3).

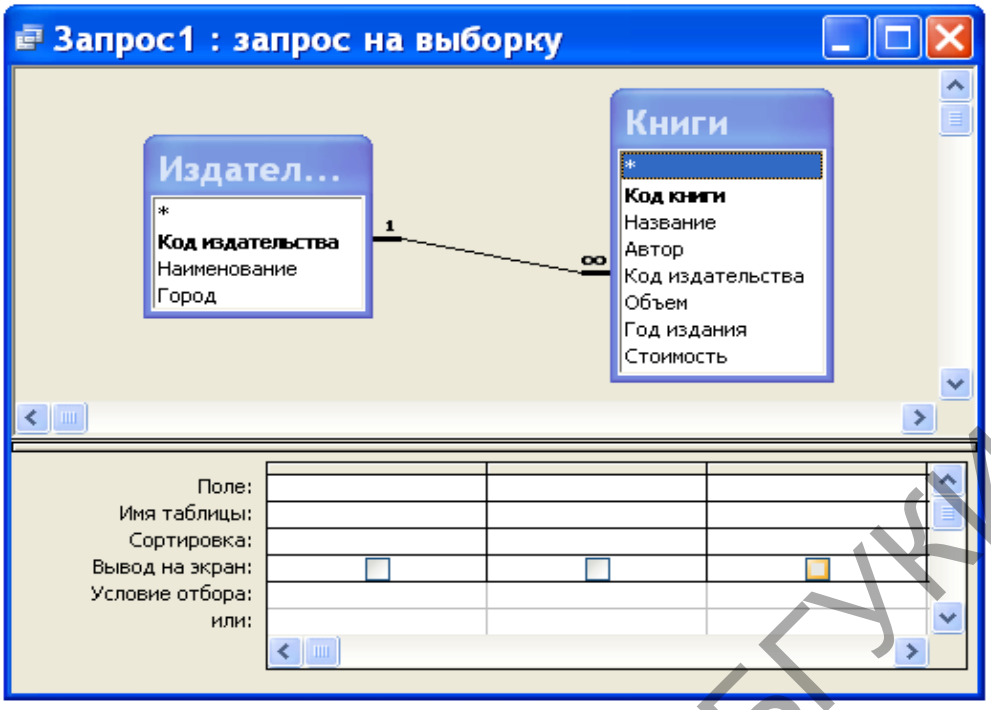


Рис. 3. Окно запроса в режиме конструктора.

Окно запроса в режиме конструктора предназначено для создания новых и изменения существующих запросов. При создании запросов в этом режиме используется механизм запросов по образцу **QBE** (QuerybyExample). Окно в этом случае состоит из двух частей. В верхней части окна размещаются списки полей, из которых будет формироваться запрос. В нижней части окна располагается бланк **QBE**, в который нужные для запроса поля перемещаются при помощи мыши из списков полей, размещенных в верхней части окна.

Для изменения относительной высоты верхней и нижней частей окна используется специальная разделительная линия. При установке курсора на эту линию курсор приобретает вид двунаправленной стрелки. В это момент разделительную линию можно перемещать вверх или вниз.

Имена полей, которые будут образовывать динамический набор, должны быть в соответствующем порядке размещены в строке бланка **QBE**. Сделать это можно несколькими способами. Самый простой способ состоит в двойном щелчке мышью на имени в списке полей. Указанным способом в строке **Поле** бланка **QBE** поместите поля: **Автор**, **Название**, **Наименование, Год издания** и **Город**. Последнее поле нам понадобилось, чтобы задать условие отбора для выбора для выбора книг московских издательств.

В строку **Условие отбора** для поля **Город** наберите текст "Москва" (задание условий отбора подробнее будет рассмотрено ниже). Даже, если вы текст в кавычки не возьмете, MSAccess сам это сделает. Условие отбора

нам понадобилось для того, чтобы в запросе выбирались не все книги, а только книги, изданные в Москве.

Поскольку по условию задачи поле **Город** не надо выводить на экран, то в строке **Вывод на экран** для этого поля уберите щелчком мыши пометку ("птичку").

Для того чтобы в динамическом наборе записи выводились в алфавитном порядке по фамилиям авторов, надо в строке **Сортировка** для поля **Автор** задать направление сортировки. Выполните щелчок мышью на ячейке в строке **Сортировка** для поля **Автор**. При этом справа в этой ячейке появится кнопка раскрытия списка направления сортировки. Выберите в этом списке направление сортировки – по возрастанию.

Порядок обработки полей при сортировке по нескольким полям определяется их положением в бланке **QBE**: сначала сортируются значения в крайнем левом поле и далее слева направо. После указанных действий бланк **QBE** будет иметь вид, представленный на рисунке 4.

Сейчас выполним созданный нами запрос. Для этого нажмите кнопку **Режим таблицы** на панели инструментов **Конструктор запросов** (первая кнопка – см. рис. 1). После нажатия этой кнопки вы увидите список книг московских издательств, рассортированный в алфавитном порядке по фамилиям авторов (см. рис. 5).

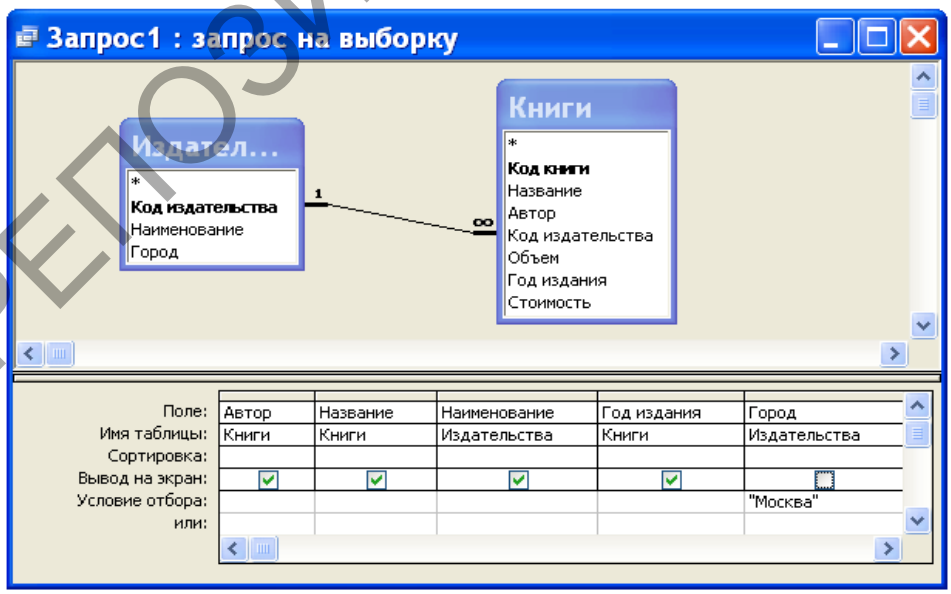


Рис. 4. Вид бланка **QBE** для решения задачи.

Для того чтобы установить оптимальную ширину столбца списка, надо выполнить двойной щелчок мышью на правой границе столбца в строке заголовков полей. Установите оптимальную ширину для всех столбцов списка книг, как это сделано на рис. 5.

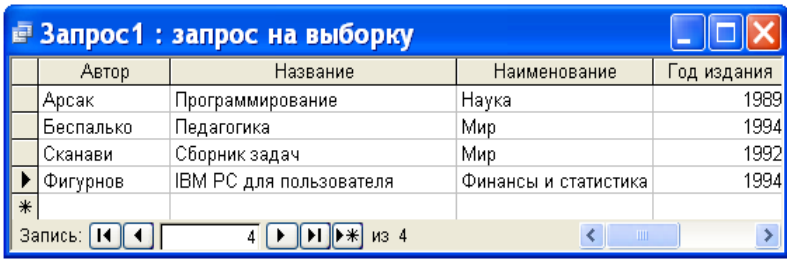


Рис. 5. Результат выполнения запроса.

После того как запрос создан, его можно сохранить. Для этой цели надо выполнить команду**Сохранить запрос** или **Сохранить запрос как** в меню **Файл**. Если мы выполняем сохранение первый раз, то выполнение этих команд приводит к одному и тому же результату – на экране появляется окно диалога, приведенное на рис. 6.

Рис. 6. Окно диалога для сохранения запроса.

Сохраните созданный нами запрос под именем **Список книг московских издательств**. Для этого введите новое имя (старое имя **Запрос1**, которое предложил Access, после нажатия первой клавиши исчезнет, так что нет необходимости специально его убирать) и нажмите кнопку **OK**.

Отбор данных

Основное назначение запроса состоит в формировании динамического набора, записи которого удовлетворяют некоторым условиям. Условия отбора записей вводятся как выражения. Выражение указывает, какие записи следует включить в динамический набор при выполнении запроса. Выражения могут быть простыми (например, <30) или сложными (например, Between 100 And 500).

Определить условия отбора можно самостоятельно, введя нужное выражение в ячейку **Условия отбора**, соответствующую данному полю, или воспользоваться построителем выражения. Для определения условия с помощью построителя выражений вначале устанавливают указатель в

ячейку **Условия отбора** в бланке **QBE**, в которой следует определить выражение, и нажимают кнопку мыши. После этого нажимают кнопку**Построить** на панели инструментов. На экране появляется диалоговое окно **Построитель выражений**, приведенное рис. 7.

Если в ячейке бланка **QBE**, из которой вызывался построитель, содержится значение, то это значение автоматически копируется в поле построения выражения. Используя построитель выражений, можно вводить символы в область ввода или нажимать кнопки для ввода операторов, а также вставлять ссылки на объекты и другие элементы выражения, выбирая их из папок.

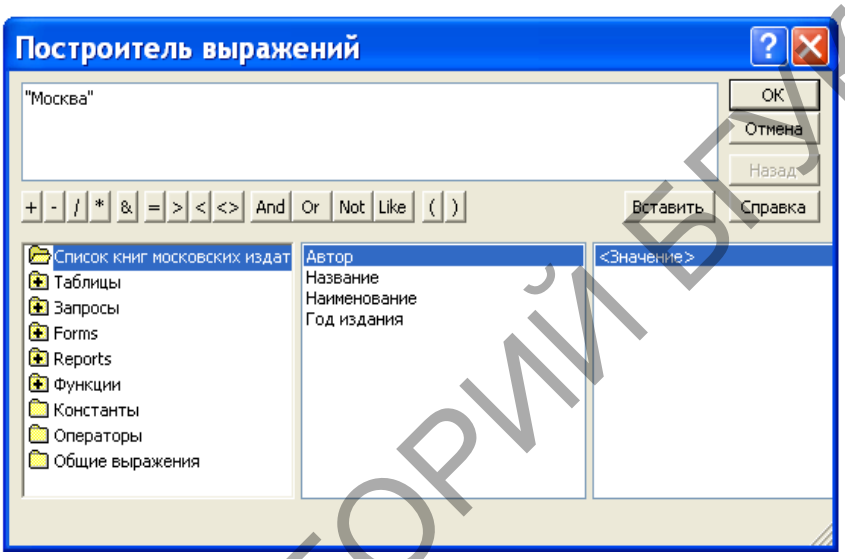


Рис. 7. Окно диалога **Построитель выражений**.

Вставка операторов в выражение из строки операторов, расположенной ниже поля построителя, выполняется щелчком мыши на операторе.

Для вставки элемента поступают следующим образом. В левом нижнем поле построителя выбирают папку, содержащую нужный элемент. В нижнем среднем поле дважды щелкают элемент, чтобы вставить его в поле выражения, или выбирают тип элемента. Если выбран тип в нижнем среднем поле, то значения будут отображаться в нижнем правом поле. Дважды щелкните значение, чтобы вставить его в поле выражения.

Вставьте необходимые операторы в выражение. Для этого поместите указатель мыши в определенную позицию поля выражения и выберите одну из кнопок со знаками операций, расположенных в середине окна построителя. Закончив создание выражения, нажмите кнопку **OK**.

MSAccess скопирует созданное выражение в ту позицию, из которой был вызван построитель выражений. Если в данной позиции уже

содержится значение, то исходное значение будет заменено новым выражением.

Следует иметь в виду, что любая часть выражения или все выражение может быть введено в поле выражения непосредственно с клавиатуры. Может также случиться, что выражение можно быстрее ввести в строку **Условие отбора** без использования построителя выражений.

Выражение – комбинация операторов, констант, литералов, значений, функций, названий свойств, имен полей и элементов управления, при оценке которых получается одно значение. Оператор – это символ или слово (например, > или Or), указывающее на операцию, которую следует выполнить над одним или несколькими элементами. Операторы сгруппированы в классы операторов, например, арифметические, сравнения, логические.

В выражениях для условий отбора допускается использование символов шаблона. Символами шаблона являются звездочка (*), знак вопроса (?), знак номера (#), восклицательный знак (!), дефис (-) и квадратные скобки ([]). Эти символы можно использовать в запросах, командах и выражениях для включения всех записей, имен файлов или других элементов, которые начинаются с определенной последовательности букв или удовлетворяют указанному шаблону. Назначение и примеры использования символов шаблонов приведены в таблице 1. При вводе шаблонов можно использовать как прописные, так и строчные буквы. Например, шаблон "ст*" эквивалентен шаблону "Ст*".

Таблица 1

Символы шаблона

Символ	Назначение	Пример	Результат отбора
*	Заменяет любую группу символов; может быть первым или последним символом в шаблоне.	ст* *иск	"стол", "станок" и т.п. "иск", "диск", "риск" и т.п.
?	Заменяет любой один символ.	ко?а	"кора", "коса", "коза" и т.п.
#	Заменяет любую одну цифру.	5#4	504, 554, 514 и т.п.
[]	Заменяет любой один символ, указанный в скобках.	ко[рс]а	"кора" и "коса", но не коза
!	Заменяет любой один символ, кроме символов, указанных в скобках.	ко[!рс]а	"коза" и "кожа", но не "кора" и "коса"
-	Заменяет любой один символ из указанного диапазона.	ко[к-м]а	"кока", "кола" и "кома"

После завершения ввода выражения в ячейку строки **Условие отбора** (например, нажатием клавиши Enter, клавиш управления курсором или щелчком мыши в другой ячейке) выполняется синтаксический анализ этого выражения и выражение приводится в соответствие с правилами синтаксиса MSAccess. Например, если введено слово Москва, то добавляются прямые кавычки и это слово выводится как "Москва".

Если выражение не содержит оператор, то подразумевается оператор равняется (=). Например, если в ячейку **Условие отбора** для поля **Город**введено слово Москва, то выражение интерпретируется как **Город** = "Москва".

Задание

1. Выведите список книг, цена которых находится в диапазоне от 20 до 30 тыс. рублей. Динамический набор этого запроса должен содержать поля: **Автор**, **Название**, **Год издания**, **Стоимость**. Для задания условия отбора вначале используйте оператор **Between** … **And**, а затем операторы >=, <=, **And**. Записи в динамическом наборе расположите по возрастанию цены книг. Сохраните первый запрос под именем **Цена книг из диапазона**, а второй – под именем **Операторы сравнения для поиска цены**.

2. Выведите список читателей, у которых нет домашнего телефона. В список включите следующие поля: **Фамилия**, **Имя**, **Отчество**, **Домашний адрес**. Список рассортируйте в алфавитном порядке по фамилии, имени и

отчеству. Для поиска требуемых записей в строке **Условие отбора** для поля **Домашний телефон** используйте выражение **IsNull**. Это выражение предназначено для поиска записей, у которых поле не содержит значение (является пустым). Если требуется отобрать записи, у которых поле имеет значение, то можно использовать выражение **IsNotNull**. Запрос сохраните под именем **Читатели без домашних телефонов**.

3. Выведите список читателей, у которых в домашнем телефоне вторая цифра есть 5 или 6. В динамический набор включите поля: **Фамилия**, **Имя**, **Отчество**, **Домашний телефон**. Условие отбора для поля может иметь следующий вид: **Like "?[56]*"**.

Запрос сохраните под именем **Использование символов шаблона**. Измените условие отбора предыдущего запроса так, чтобы он выводил список всех читателей, в номерах телефонов которых вторая цифра не 5 и не 6. Полученный запрос сохраните под именем **Символ отрицания в квадратных скобках**.

4. Создайте запрос, который будет выводить список книг, заказанных читателями в 2007 году. В динамический набор включите следующие поля: **Автор**, **Название**, **Наименование**, **Город**. Для решения задачи вначале используйте функцию DatePart(interval; date; firstweekday; firstweek), а затем Format(expr; fmt; firstweekday; firstweek). Запросы сохраните под именами **Использование функцииDatePart** и **Использование функцииFormat** соответственно.

5. В таблицу **Выдача книг** базы данных **Библиотека** добавьте поле **Дата возврата**. В это поле для записей, приведенных в таблице 2, введите даты возврата. В остальных записях поле **Дата возврата** должно остаться пустым.

Таблица 2

Учет возврата книг

Код читателя	Код книги	Дата заказа	Дата возврата
1	1	01.09.2007	15.10.2007
1	3	05.07.2008	23.09.2008
4	3	07.01.2008	02.03.2008
5	2	23.04.2008	03.05.2008
7	3	20.01.2008	11.04.2008
9	6	02.02.2008	03.03.2008

Составьте запрос, который будет выводить список читателей, которые не сдали своевременно книги (предполагается, что читатель может держать книгу на руках не более 100 дней). В динамический набор включите следующие поля: **Фамилия**, **Имя**, **Отчество**, **Домашний**

телефон, **Автор**, **Название**, **Стоимость**. Для решения задачи воспользуйтесь функцией **Date**(). Запрос сохраните под именем **Читатели, не сдавшие своевременно книги**.

Лабораторная работа 4

Просмотр и изменение динамического набора

*Цель работы:*Сформировать знания о редактировании динамического набора. Сформировать умения изменять свойства запросов и его элементов.

Запрос и его элементы (поля и списки полей) имеют свойства. Свойства запроса определяют поведение запроса в целом. Например, можно определить свойство, запрещающее включение повторяющихся значений в динамический набор. Свойства поля определяют поведение данных в поле. Например, можно определить формат изображения чисел в поле. Свойства можно определять для любого поля, кроме звездочки и полей, для которых не установлен флажок (имеет вид крестика) **Вывод на экран**. Свойства списка полей влияют на один из списков полей таблицы и запроса, включенных в запрос. Например, можно определить свойство, задающее нестандартное название списка полей.

Опишем последовательность действий, которые надо выполнять при просмотре, определении или изменении свойств запроса или его элементов. Чтобы это сделать, надо вначале выделить запрос или его элементы:

– для выделения всего запроса следует выполнить щелчок мышью в любом месте окна запроса вне бланка **QBE** и списков полей;

– для выделения поля следует выполнить щелчок мышью в соответствующей ячейке в строке **Поле**;

– для выделения списка полей следует выполнить щелчок мышью в любом месте этого списка.

После этого надо выбрать команду **Свойства** в меню **Вид** или нажать кнопку **Свойства** на панели инструментов. На экране появится окно свойств выделенного объекта. Окно свойств объекта для случая, когда объектом является поле, приведено на рис. 1.

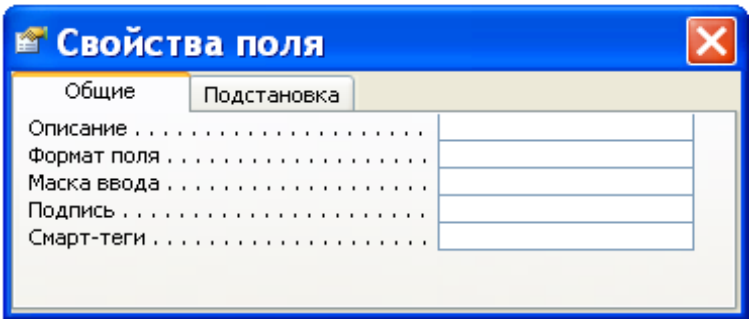


Рис. 1. Окно**Свойства поля**.

Затем, если требуется определить или изменить какое-либо свойство в этом бланке, раскрывают список возможных значений данного свойства, нажав кнопку раскрытия списка (если такая кнопка есть), и выбирают требуемое значение из списка или вводят допустимое значение. На рис. 2 раскрыт список значений свойства **Формат поля** с типом данных дата/время.

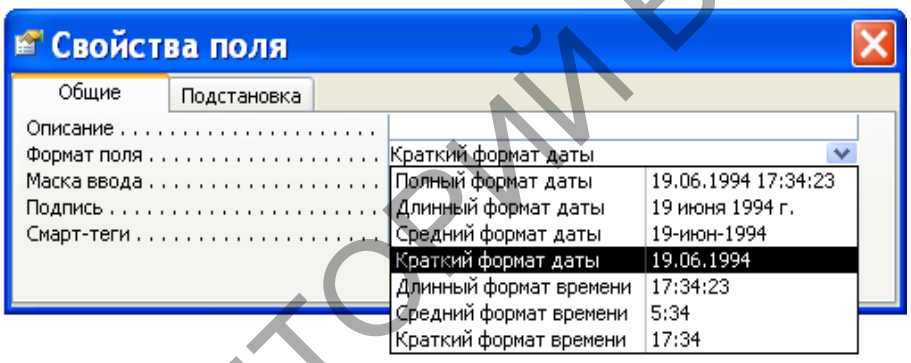


Рис. 2. Определение свойства **Формат поля**.

Для определения значений некоторых свойств, например, **Маска ввода**, можно использовать построители. Справа от ячейки бланка свойств, соответствующей такому свойству, расположена кнопка**Построить**, которую можно нажать для вызова построителя.

Из списка значений свойства **Формат поля**, приведенного на рис. 2, видно, что дата может иметь четыре формата: полный, длинный, средний и краткий. В макете базовой таблицы **Выдача книг** поле **Дата заказа** имеет краткий формат (например, "19.06.1994"). Полный, длинный и средний форматы для этого значения даты имеют соответственно вид: "19.06.1994 17:34:23", "19 июня 1994 г." и "19-июн-1994". На этом же рисунке показано, что время может изображаться в трех форматах: длинном ("17:34:23"), среднем ("5:34") и кратком ("17:34").

Аналогичным образом можно определять свойства полей для изображения других типов данных. Например, числа можно изображать с десятичным разделителем или знаком процента.

В запросе для изображения данных можно устанавливать форматы, отличные от форматов полей базовой таблицы. При этом важно знать следующее. По умолчанию поля в запросе наследуют все свойства соответствующих полей базовой таблицы или запроса. При изменении свойства поля в макете базовой таблицы это изменение будет автоматически отражено в макете запроса, однако после изменения свойства поля в режиме конструктора запросов новое значение заменит установленное для базового поля, и все последующие изменения этого свойства в макете таблицы не будут отражены в запросе.

Рассмотрим, как можно выполнять операции над полями (изменение порядка, вставка и удаление) после включения их в запрос. Перемещение поля в бланке **QBE** запроса выполняется просто: вначале выделяют поле (выполняют щелчок мышью в области маркировки столбца), а затем, не меняя положения указателя, нажимают кнопку мыши и перемещают столбец на новое место.

Для вставки поля в бланк **QBE** запроса надо в списке полей выделить поле, которое следует вставить, и перенести его из списка полей в нужный столбец бланка **QBE**.

Удаление всех полей из бланка **QBE** выполняется командой**Очистить бланк** в меню **Правка**. Чтобы удалить одно поле, надо вначале его выделить, а затем выполнить команду**Удалить** в меню **Правка** или нажать клавишу **Del**.

Улучшить внешний вид запроса можно, изменив ширину столбцов. Для изменения ширины столбца устанавливают указатель на правую границу в области маркировки столбца и перемещают ее влево (для уменьшения ширины) или вправо (для увеличения ширины). Очень быстро можно установить оптимальную ширину столбца. Оптимальной называют такую ширину, при которой в поле помещается самое длинное значение в этом столбце (в расчет принимается и ширина заголовка столбца). Установка оптимальной ширины столбца осуществляется двойным щелчком мышью на правой границе столбца в области маркировки. Если до выполнения этой операции было выделено несколько столбцов, то после ее выполнения для каждого из выделенных столбцов будет установлена оптимальная для него ширина.

Следует иметь в виду, что, если после установки оптимальной ширины в столбец будет добавлено значение, длина которого превышает текущую ширину столбца, то для отображения значения полностью придется снова повторить описанную выше процедуру. Обратите внимание на то, что рассмотренные процедуры изменяют ширину столбцов в бланке **QBE**. При просмотре результата выполнения запроса в режиме таблицы вам еще раз придется изменять ширину столбцов.

Иногда для улучшения наглядности и читабельности запроса в режиме таблицы требуется изменить название поля. Сделать это можно следующим образом. Откройте запрос в режиме конструктора или перейдите в режим конструктора, если текущим является режим таблицы. Установите указатель слева от первой буквы имени поля в бланке QBE и введите перед старым именем новое имя с двоеточием (см. рис. 3).

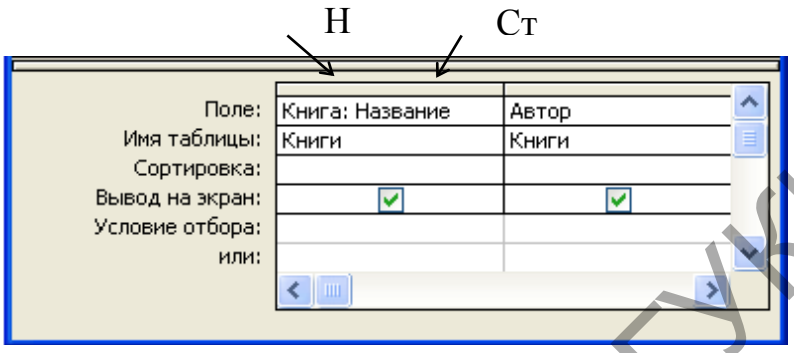


Рис. 3. Переименование поля.

После переименования поля новое имя будет появляться как заголовок столбца запроса в режиме таблицы. Более того, новое имя будет использоваться в любом новом объекте, основанном на этом запросе, а ссылки на это поле в существующих объектах должны быть изменены пользователем.

Если требуется больше места для ввода или редактирования значения свойства, нажмите клавиши **SHIFT+F2** для открытия окна **Область ввода**. Получить дополнительные сведения о свойстве и его значениях можно, нажав клавишу **F1** после выбора свойства.

Задание

1. Откройте запрос **Стоимость книг из диапазона** в режиме конструктора и для поля **Стоимость** установите значение свойства **Число десятичных знаков** равным нулю, а для свойства **Формат поля** выберите из списка значение **Денежный**. Убедитесь после этого в том, что значения этого поля в динамическом наборе не будут больше содержать запятую с двумя нулями (в Беларуси в настоящее время копейки не используются). Сохраните этот запрос с измененным свойством поля **Стоимость** под старым именем. Такие же изменения свойств поля **Стоимость** выполните в запросах **Операторы сравнения для поиска цены** и **Читатели, не сдавшие своевременно книги**.

2. В динамическом наборе запроса **Список книг московских издательств** после поля **Год издания** вставьте поле **Объем**. Измените свойства поля **Объем** в этом запросе таким образом, чтобы значения этого поля в динамическом наборе имели следующий вид: "340 с.". Для этой

цели в свойствах поля **Объем** задайте требуемую маску ввода. Данные изменения сохраните в запросе.

3. Переставьте местами поля **Автор** и **Название** в запросе **Читатели, не сдавшие своевременно книги**. В динамическом наборе этого запроса после поля **Домашний телефон** вставьте поле **Домашний адрес**. Измените значение свойства запроса **Уникальные записи** "Нет" на "Да".

4. Установите оптимальную ширину столбцов в бланке **QBE**для ранее созданных запросов: **Использование символов шаблона**, **Операторы сравнения для поиска цены**, **Список книг московских издательств** и **Читатели без домашних телефонов**, а затем установите оптимальную ширину столбцов для этих же запросов в режиме таблицы.

5. Переименуйте в запросе **Цена книг из диапазона** поле **Название**. Этому полю дайте имя **Книга**. Просмотрите результат выполнения полученного запроса в режиме таблицы. Сохраните этот запрос под именем **Запрос с переименованным полем**.

Лабораторная работа 5

Запросы с параметрами

*Цель работы:*Сформировать умения для создания запросов с параметрами.

Часто встречаются ситуации, когда перед выполнением запроса надо изменять условия отбора. В таком случае целесообразно создать запрос с параметрами. При выполнении запроса с параметрами не требуется открывать окно запроса и вносить изменения в бланк **QBE**. Вместо этого пользователю надо ввести нужное условие отбора в диалоговое окно**Введите значение параметра**. Запрос с параметрами может содержать несколько параметров. Тогда перед каждым выполнением запроса на экране будет появляться определенная пользователем последовательность диалоговых окон, предназначенных для ввода условий отбора.

Параметр имеет имя. Это имя определяет разработчик запроса и записывает его в квадратных скобках. Имя параметра может записываться в строке **Условия отбора** или в строке **Поле** бланка **QBE**. Если в бланке **QBE**Access встречает в квадратных скобках текст, не совпадающий с именем поля, то он автоматически считает его именем параметра.

Для создания запроса с параметрами выполните следующие действия:

– Создайте запрос в режиме конструктора и включите в него нужные таблицы. Перенесите нужные поля в бланк **QBE**.

– В ячейку **Условие отбора** поля, которое планируется использовать для определения параметра, введите текст (см. рис. 1), заключенный в квадратные скобки. Этот и будет именем параметра, он появится на экране при выполнении запроса. Имя параметра должно отличаться от имен полей.

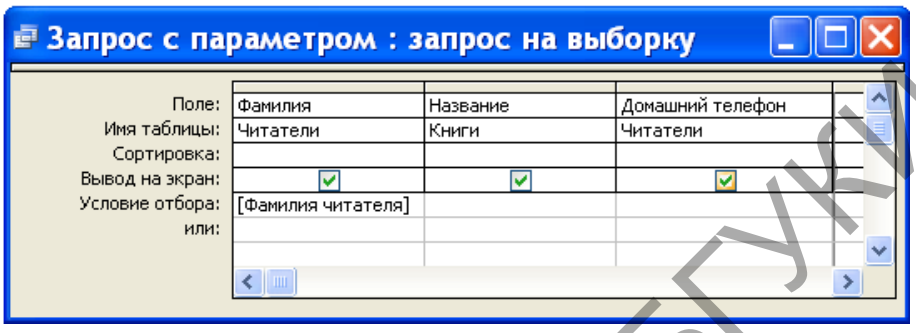


Рис. 1. Пример записи имени параметра.

– Выберите команду **Параметры** в меню **Запрос**. На экране появится диалоговое окно **Параметры запроса** (см. рис. 2). В первую ячейку **Параметры** введите имя параметра, которое было введено в первую ячейку бланка **QBE** при составлении запроса с параметром. В ячейке, расположенной справа от имени параметра, установите нужный тип данных. Если запрос содержит несколько параметров, продолжите ввод имен параметров и установку типов данных. При выполнении запроса пользователю будет предложено ввести значения параметров в том порядке, в котором они расположены в диалоговом окне **Параметры запроса**. Тип данных для параметров можно и не задавать, тогда он наследуется из базовой таблицы.

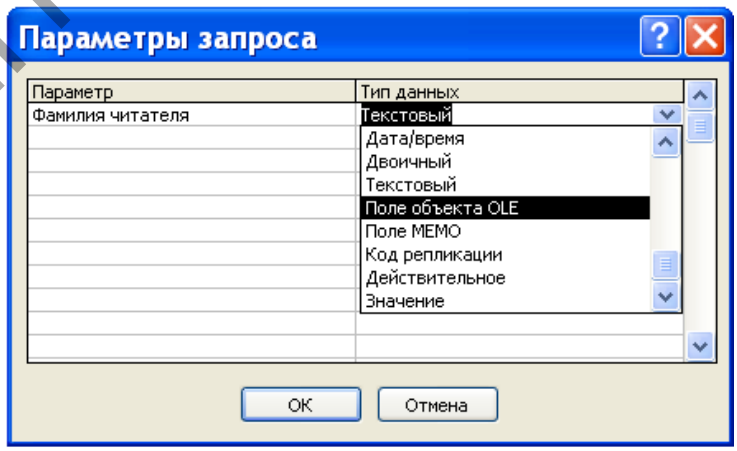


Рис. 2. Установка типа данных для параметров.

– Выберите команду **Таблица** в меню **Вид** или нажмите кнопку

Режим таблицы на панели инструментов. На экране появится диалоговое окно**Введите значение параметра** (см. рис. 3).

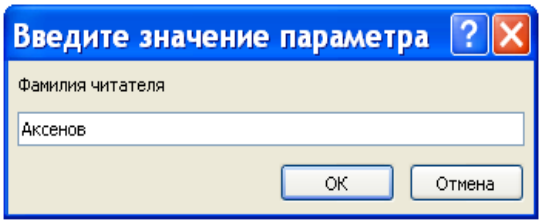


Рис. 3. Диалоговое окно для ввода значения параметра.

– Введите значение параметра и нажмите кнопку **OK**. MSAccess выполнит отбор данных и выведет на экран динамический набор или, если запрос содержит несколько параметров, на экране появится диалоговое окно для ввода значения следующего параметра. После ввода значений всех параметров на экране появится динамический набор.

Задание

1. В качестве упражнения создайте запрос, который будет осуществлять поиск книг по ключевому слову в теме. Назовите данный запрос **Книги с ключевым словом в теме**. Бланк **QBE** для данного запроса приведен на рис. 4. Обратите внимание на то, что при вводе значения параметра в диалоговое окно надо указывать корень слова, а не само слово, иначе некоторые книги могут быть не найдены.

Сократить количество релевантных сведений при поиске книг можно, указав несколько ключевых слов. Создайте запрос, который будет осуществлять поиск книг по двум ключевым словам в теме. Запрос назовите **Поиск по двум ключевым словам**.

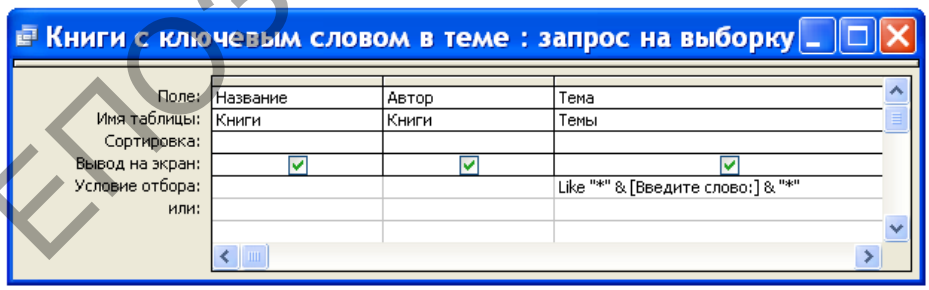


Рис. 4. Поиск книг по ключевому слову в теме.

2. Запросы с параметрами удобно использовать для указания нескольких первых букв искомого значения. На рис. 5 показан бланк **QBE** для запроса, который будет осуществлять поиск книг по нескольким первым буквам фамилии автора.

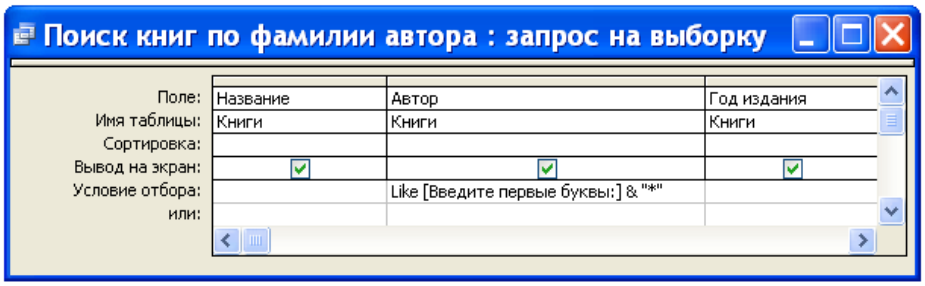


Рис. 5. Поиск книг по первым буквам фамилии автора.

Сохраните данный запрос под именем **Поиск книг по фамилии автора** и выполните его несколько раз для наборов первых букв фамилии автора, содержащих различное количество символов.

3. Создайте запрос, который будет выводить список читателей, которые вовремя не сдали книги. В качестве параметра возьмите период (количество дней), в течение которого читатель может держать книги на руках. Для этих целей измените ранее созданный запрос с именем **Читатели, своевременно не сдавшие книги,** добавив в него параметр. Новый запрос назовите **Список читателей для вызова**.

4. Создайте запрос, который будет подсчитывать количество книг, заказанных в конкретном месяце года. Год и номер месяца возьмите в качестве параметров. Запрос назовите **Заказы книг по месяцам**. В динамический набор включите три поля, которым дайте следующие имена: **Год**, **Номер месяца**, **Количество книг**. Напомним, что для создания этого запроса в бланке **QBE** понадобится строка **Групповая операция**.

Лабораторная работа 6

Запросы с вычисляемыми полями

*Цель работы:*Сформировать умения создавать вычисляемые поля для организации вычислений.

Вычисляемое поле – это такое поле, которое не содержится ни в одной из таблиц базы данных, а создается с помощью выражений. Для расчетов с использованием формул, определяемых пользователем, требуется создать новое вычисляемое поле прямо в бланке запроса. Вычисляемое поле создается с помощью выражения, которое вводится в пустую ячейку **Поле** бланка запроса.

Вычисляемое поле имеет следующий формат:

имя вычисляемого поля: выражение для построения вычисляемого поля

Если при создании вычисляемого поля пользователь не указывает имя, то Access по умолчанию присвоит ему имя **Выражение1**. Имя для вычисляемого поля рекомендуется задавать по двум причинам: во-первых,

для заголовка столбца таблицы, содержащей динамический набор запроса, и, во-вторых, для обращения к этому полю в форме, отчете или другом запросе.

Рассмотрим пример. Требуется вывести список читателей в алфавитном порядке, содержащий фамилии, инициалы и домашний адрес. Поскольку поля с инициалами читателей ни в одной таблице базы данных **Библиотека** нет, нам потребуется создать вычисляемое поле для выделения инициалов из имени и отчества читателей. Вычисляемое поле назовем **Фамилия и инициалы**. Запрос сохраним под именем **Список читателей с инициалами**.

Вид бланка запроса для вывода списка читателей с фамилиями и инициалами представлен на рис. 1.

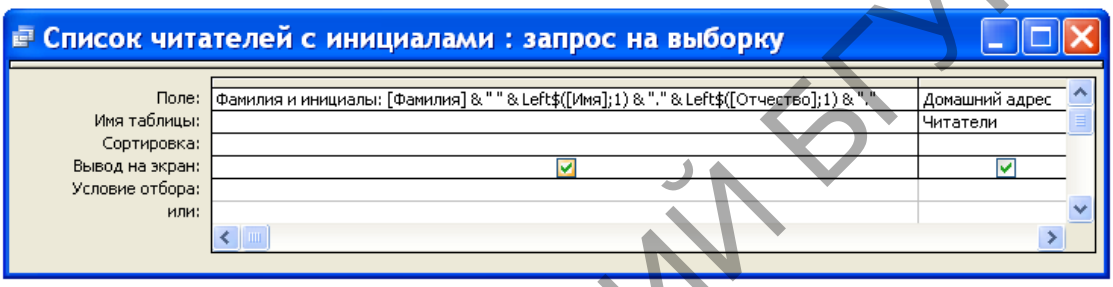


Рис. 1. Бланк запроса для вывода списка читателей.

Выражение в вычисляемом поле, приведенном на рис. 1, является текстовым. Оператор конъюнкция ("&") в нем используется для сцепления строк, так, например, выражение [Фамилия]&" " сцепляет значение поля **Фамилия** с пробелом. Функция Left$([Имя];1) в данном контексте используется для выделения одной левой буквы из значения поля **Имя**. Динамический набор записей в результате выполнения данного запроса будет иметь вид, представленный на рис. 2.

Рис. 2. Динамический набор запроса.

Убедитесь в том, что в рассмотренном примере вместо функции Left$([Имя];1) мы бы могли воспользоваться более общей функцией Mid$([Имя];1;1). Функция Mid$([Имя поля];n;m)позволяет из указанного

поля выделить подряд расположенных mсимволов, начиная с номераn. Более того, тот же результат получился бы, если мы в названиях используемых функций убрали бы символ доллара ("$").

Задание

1. Создайте вычисляемое поле для вычисления новой цены книг. Новая цена должна определяться умножением значения поля **Стоимость** на величину 1,1. Вычисляемое поле назовите **Новая цена**. Для этого поля установите денежный формат. В динамический набор запроса включите следующие поля: **Автор**, **Название**, **Год издания**, **Новая цена**. Запрос сохраните под именем **Стоимость книг с учетом инфляции**.

2. Создайте вычисляемое поле для вычисления цены книг в условных единицах. Курс доллара примите равным 2165 белорусских рублей. В динамический набор включите те же поля, что и в предыдущем примере, но вычисляемое поле назовите **Цена в у_е**. Для вычисляемого поля в его свойствах задайте пользовательский формат, позволяющий отображать информацию так, как это показано на рис. 3.

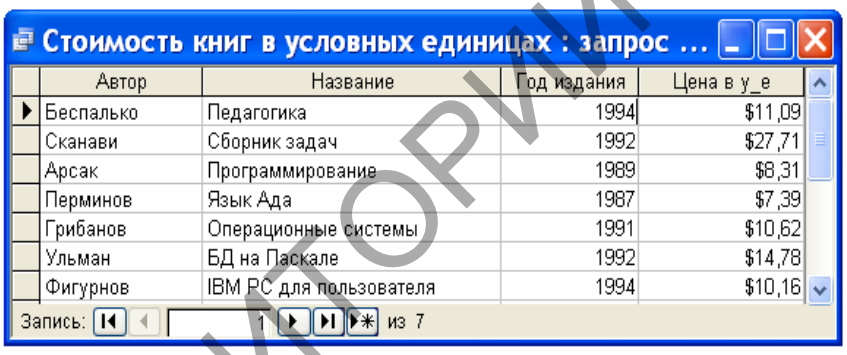


Рис. 3. Форматирование вычисляемого поля.

Получение указанного результата могут обеспечить свойства вычисляемого поля, приведенные на рис. 4. Запрос сохраните под именем **Стоимость книг в условных единицах**.

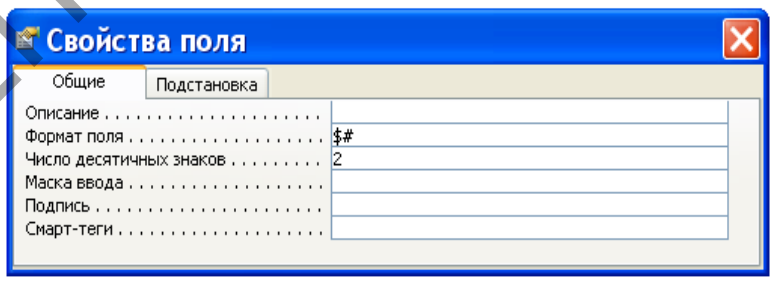


Рис. 4. Свойства вычисляемого поля.

3. Подсчитайте, сколько книг заказывал каждый читатель за весь период использования абонемента библиотеки. В динамический набор включите вычисляемое поле **Фамилия и инициалы** и новое поле **Количество**, значения которого вычисляются в результате применения

групповой операции Count над полем **Код книги** из таблицы **Выдача книг** (см. рис. 5).

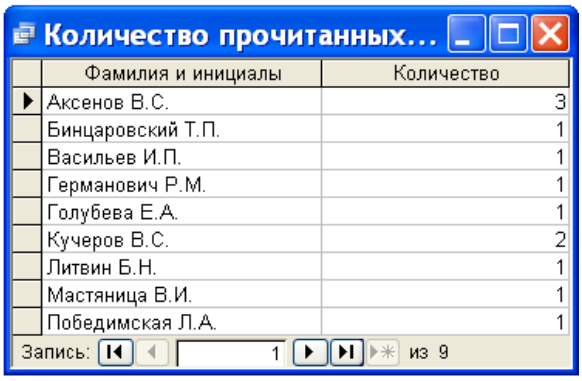


Рис. 5. Использование групповых операций для вычислений.

Для поля **Фамилия и инициалы** в строке **Групповая операция** бланка запроса выберите операцию Группировка, а в строке **Сортировка** выберите направление сортировки: по возрастанию. Запрос сохраните под именем **Количество прочитанных книг**.

4. По аналогии с предыдущим заданием, используя групповые операции, создайте запрос, который позволит получить информацию о том, сколько раз заказывали каждую книгу. В динамический набор запроса включите поля **Автор**, **Название** из таблицы **Книги** и новое поле **Рейтинг**, значение которого указывает, сколько раз заказывалась данная книга (см. рис. 6).

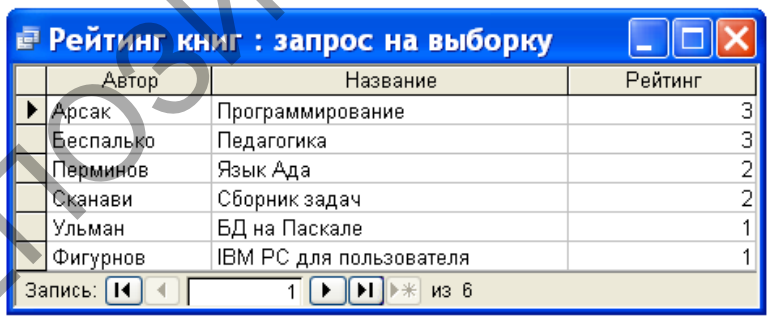


Рис. 6. Результат выполнения запроса **Рейтинг книг**.

Книги, которые не заказывались, в динамический набор запроса включать не надо. Запрос сохраните под именем **Рейтинг книг**.

Лабораторная работа 7

Перекрестный запрос

Цель работы: Сформировать умения для создания перекрестных запросов.

Перекрестные запросы предназначены для группирования данных и представления их в компактном виде, напоминающем электронную таблицу. Перекрестный запрос позволяет представить большой объем данных в виде, удобном для восприятия, анализа и сравнения. Более того, перекрестный запрос удобно использовать в качестве базового при создании отчета. Следует иметь в виду, что в результате выполнения перекрестного запроса получается не динамический, а статический набор записей, то есть такой набор записей, который нельзя обновлять.

Проще всего создать перекрестный запрос с помощью мастера по разработке перекрестных запросов. При необходимости перекрестный запрос можно создать без помощи мастера. Рассмотрим два этих способа создания перекрестного запроса.

Создание перекрестного запроса с помощью мастера выполняют следующим образом:

– Находясь в окне базы данных, выбирают корешок **Запрос** и нажимают кнопку**Создать**.

– В диалоговом окне **Создание запроса** нажимают кнопку **Мастера по разработке запросов**.

– В первом окне **Мастера по разработке запросов** выбирают пункт **Перекрестный запрос**.

– Далее выполняют инструкции, появляющиеся в диалоговых окнах. В последнем диалоговом окне нажимают кнопку **Готово**.

Для создания перекрестного запроса без помощи мастера надо выполнить следующую последовательность действий:

1. В окне базы данных выберите корешок **Запрос** и нажмите кнопку**Создать**. В появившемся диалоговом окне **Создание запроса** нажмите кнопку **Новый запрос**.

2. Выберите таблицы или запросы, содержащие поля, которые следует включить в запрос. Перенесите нужные поля в строку **Поле** бланка **QBE** и задайте условия отбора.

3. Выберите команду **Перекрестный** в меню **Запрос** или нажмите кнопку **Перекрестный** на панели инструментов. В бланке **QBE** появятся строки **Групповые операции** и перекрестная таблица. По умолчанию

ячейки **Групповые операции**, соответствующие каждому полю, включенному в запрос, будут содержать надпись **Группировка**.

4. Установите указатель в ячейку **Перекрестная таблица**, соответствующую полю, которое содержит заголовки строк, и нажмите кнопку мыши, а затем нажмите кнопку раскрытия списка и выберите строку **Заголовки строк**. Можно указать несколько полей с заголовком строк. В ячейке **Групповые операции**, соответствующей по крайней мере одному из этих полей, должна содержаться надпись **Группировка**.

5. Установите указатель в ячейку **Перекрестная таблица**, соответствующую полю, которое содержит заголовки столбцов, и нажмите кнопку мыши, а затем нажмите кнопку раскрытия списка и выберите строку **Заголовки столбцов**. Только одно поле в перекрестном запросе может содержать заголовки столбцов. Ячейка **Групповые операции**, соответствующая этому полю, должна содержать надпись **Группировка**.

6. Установите указатель в ячейку **Перекрестная таблица**, соответствующую полю, которое содержит значения для вычислений, и нажмите кнопку мыши, а затем нажмите кнопку раскрытия списка и выберите строку **Значения**. Только одно поле в перекрестном запросе может содержать значения для вычислений. Если поле следует использовать для группирования, сортировки или размещения условий отбора, но не следует включать в результирующий набор записей, то нажмите кнопку раскрытия списка в ячейке **Перекрестная таблица**, соответствующей этому полю, и выберите строку **не выводить**.

7. Установите указатель в ячейку **Групповые операции**, соответствующую полю, которое содержит значения для вычислений, и нажмите кнопку мыши, а затем нажмите кнопку раскрытия списка и выберите тип групповой операции (например, **Sum**, **Max**или **Count**). Поле, которое содержит значения для вычислений, нельзя использовать для группирования; оно должно содержать результат выполнения групповой операции.

8. Для вывода на экран полученного набора записей выберите команду **Таблица** в меню **Вид** или нажмите кнопку **Режим таблицы** на панели инструментов.

Следует иметь в виду, что перекрестный запрос может содержать несколько полей с заголовками строк, но только одно поле с заголовками столбцов. Если в перекрестный запрос следует включить несколько полей с заголовками столбцов, но только одно поле с заголовками строк, поменяйте местами заголовки строк и столбцов.

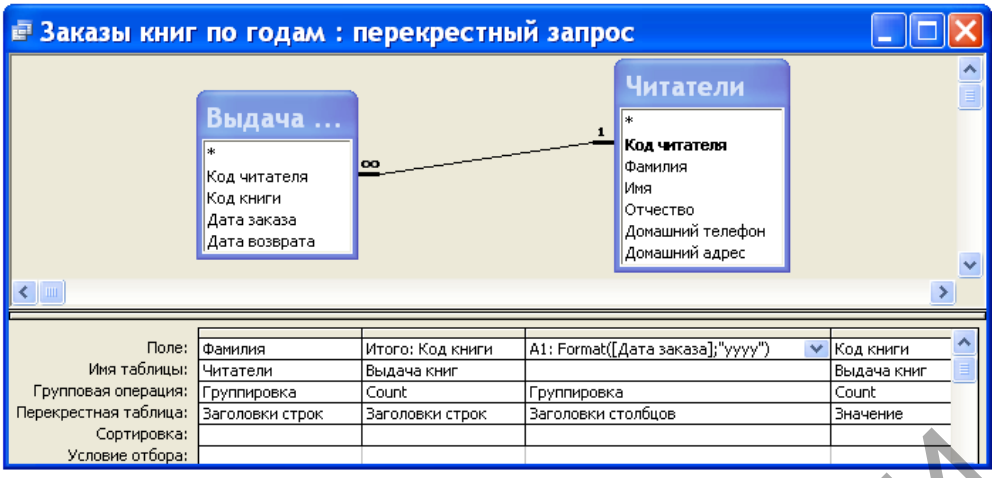


Рис. 1. Окно в режиме конструктора для запроса **Заказы книг по годам**.

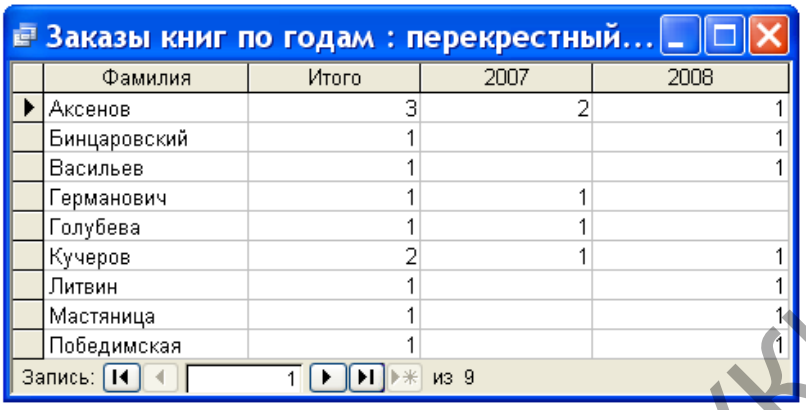
Рассмотрим следующий пример. Требуется создать список фамилий читателей, в котором для каждого читателя будет указано общее количество заказанных книг за весь период пользования услугами библиотеки и количество книг, заказанных в каждом году данного периода.

Для решения этого примера нам понадобятся две таблицы из базы данных **Библиотека**: **Читатели** и **Выдача книг**. Разместите списки этих таблиц в верхней части окна запроса, как это сделано на рис. 1.

В бланке **QBE** окна запроса надо заполнить четыре столбца. Переместите нужные поля из верхней части окна запроса в строку **Поле** бланка **QBE**. Первые два столбца должны определять заголовки строк перекрестной таблицы: **Фамилия** и**Итого**. Для того чтобы появились в бланке **QBE** строки **Групповая операция** и **Перекрестная таблица**, надо выполнить команду **Перекрестная** в меню **Запрос** или нажать кнопку **Перекрестный** на панели инструментов. После этого заполните строки **Групповая операция** и **Перекрестная таблица**. Для первого столбца укажите групповую операцию **Группировка**, а для второго – **Count**.

Третий столбец должен определять заголовки столбцов перекрестной таблицы. В нашем примере в качестве заголовков столбцов должны быть выбраны годы из поля **Дата заказа** таблицы **Выдача книг**. Для того чтобы выбрать элемент год из значений **Дата заказа** можно использовать функцию **Format**. В четвертом столбце бланка **QBE** надо указать, какие данные будут выведены в перекрестной таблице под заголовками столбцов. С этой целью указано вычисляемое значение, которое определяется функцией **Count**.

Набор данных, созданный в результате выполнения перекрестного запроса, приведен на рис. 2. Сохраните его под именем **Заказы книг по годам**.



Заказы книг по годам : перекрестный...

Фамилия	Итого	2007	2008
Аксенов	3	2	1
Бинцаровский	1		1
Васильев	1		1
Германович	1	1	
Голубева	1	1	
Кучеров	2	1	1
Литвин	1		1
Мастяница	1		1
Победимская	1		1

Запись: 1 из 9

Рис. 2. Результат выполнения запроса **Заказы книг по годам**.

Рассмотрим еще один пример перекрестного запроса, иллюстрирующий использование постоянных заголовков столбцов. Требуется создать запрос, который будет выводить информацию о количестве выдач каждой книги на протяжении всего периода работы библиотеки, а также суммарное количество выдач каждой книги по месяцам данного периода.

Вид бланка **QBE** для решения примера приведен на рис. 3. Первые два столбца этого бланка определяют заголовки строк для результирующей таблицы запроса. Третий столбец используется для задания заголовков столбцов. Как и в предыдущем примере для выбора месяца из поля **Дата заказа** используется функция **Format**. Поскольку данный столбец используется для группирования записей, в строке **Групповая операция** бланка **QBE** указана операция **Группировка**. Последний столбец указывает, что для определения количества выдач книг с конкретным кодом книги используется групповая операция **Count** и в требуемой таблице в качестве значений ячеек, расположенных на пересечении строки с кодом книги и столбца с названием месяца, будет располагаться значение этой операции.

Выдача книг по месяцам : перекрестный запрос

Поле:	Код книги	Итого: Код книги	B1: Format([Дата заказа];"mmm")	Код книги
Имя таблицы:	Выдача книг	Выдача книг		Выдача книг
Групповая операция:	Группировка	Count	Группировка	Count
Перекрестная таблица:	Заголовки строк	Заголовки строк	Заголовки столбцов	Значение
Сортировка:				
Условие отбора:				
или:				

Рис. 3. Вид бланка **QBE** для запроса **Выдача книг по месяцам**.

Поскольку мы желаем получить результирующую таблицу перекрестного запроса, содержащую все названия месяцев, даже если некоторые из них отсутствуют в записях базы данных, надо определить эти названия при задании свойств запроса. Заголовки столбцов перекрестного запроса, определенные как свойства запроса, называются постоянными.

Постоянные заголовки столбцов имеют одно важное преимущество по сравнению с обычными заголовками столбцов, рассмотренными в предыдущем примере, использование их повышает скорость выполнения перекрестного запроса. Другое преимущество состоит в следующем: пользователь сам может задать очередность вывода заголовков столбцов, кроме того, он может уменьшить или увеличить количество заголовков столбцов в наборе записей независимо от наличия значений для поля, которое используется при выборе значений для заголовков столбцов.

Определим постоянные заголовки столбцов для нашего примера. Будем считать, что к настоящему времени бланк **QBE** имеет такой вид, как на рис. 3. Выделите поле **Выражение1** (напомним, что оно используется для задания заголовков столбцов) и нажмите кнопку **Свойства**. После этого на экране появится бланк свойства запроса. В ячейку **Заголовки столбцов** введите названия месяцев, которые надо использовать в качестве заголовков столбцов (см. рис. 4).

Введенные заголовки должны полностью совпадать с соответствующими значениями в полях базы данных. Разделяются заголовки столбцов точкой с запятой или другим разделителем, который установлен на **Панели управления**Windows. При этом кавычки можно не вводить – они появятся после нажатия клавиши **Enter** или при перемещении курсора в другую ячейку.

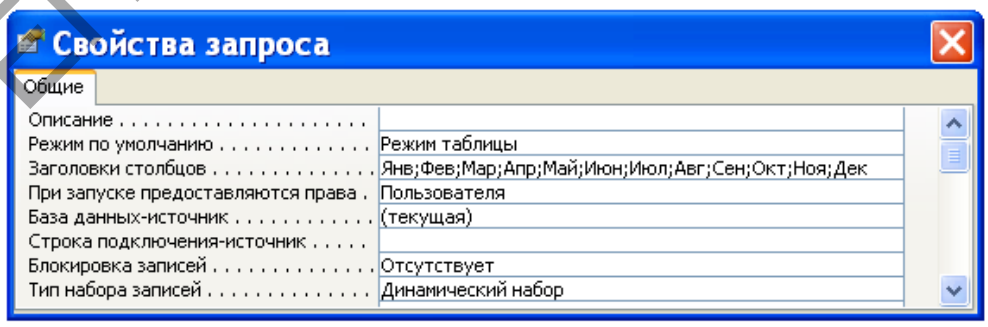
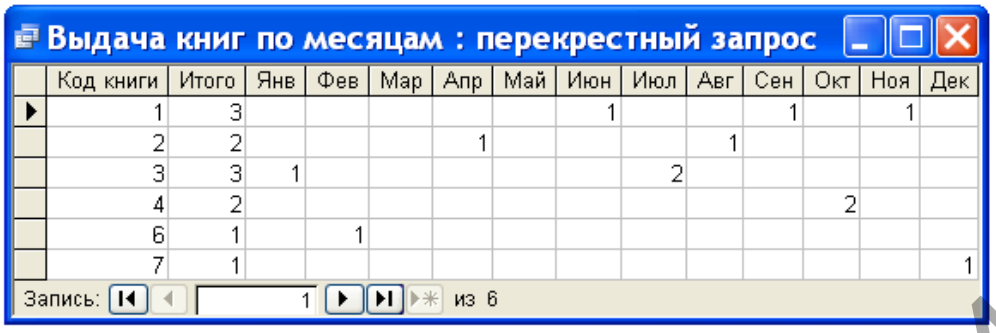


Рис. 4. Вид свойства запроса для постоянных заголовков столбцов.

Для вывода на экран полученного набора записей надо выполнить команду **Таблица** в меню **Вид** или нажать кнопку **Режим таблицы** на панели инструментов. Выполните одно из указанных действий, и вы

увидите на экране набор записей примерно в таком виде, в каком он приведен на рис. 5. Сохраните этот запрос под именем **Выдача книг по месяцам**.


Выдача книг по месяцам : перекрестный запрос

Код книги	Итого	Янв	Фев	Мар	Апр	Май	Июн	Июл	Авг	Сен	Окт	Ноя	Дек
1	3						1			1		1	
2	2				1				1				
3	3	1						2					
4	2										2		
6	1		1										
7	1												1

Запись: 1 из 6

Рис. 5. Результат выполнения запроса с постоянными заголовками столбцов.

Задание

1. Измените решение последнего примера так, чтобы перекрестная таблица содержала информацию о выдаче книг не за весь период работы библиотеки, а только за 2007 год. Полученный запрос назовите **Заказы книг в 2007 году**.

2. Создайте в электронной таблице MSExcel таблицу телефонных звонков, приведенную ниже.

Фамилия абонента	**Город**	**Телефон**	**Количество минут**
Голубев В.И.	Витебск	20-34-85	5
Василевская И.Л.	Гродно	36-67-91	3
Петрович А.А.	Гомель	44-34-29	8
Мухин П.И.	Брест	34-76-89	5
Петрович А.А.	Могиле	87-45-90	2
Мухин П.И.	Гомель	23-87-46	5
Голубев В.И.	Гродно	68-75-84	4
Мухин П.И.	Брест	65-78-34	6
Василевская И.Л.	Гродно	54-90-28	2

Сделайте экспорт данной таблицы в систему управления базами данных MSAccess. В MS Access создайте запрос, который для всех абонентов телефонной сети определяет количество звонков в каждый город и суммарную продолжительность звонков.

Лабораторная работа 8

Язык конструирования запросов SQL

Цель работы: Сформировать умения создавать запросы с помощью языка структурированных запросов SQL**.**

В предыдущих лабораторных работах мы научились создавать запросы с помощью таких средств, как мастер и конструктор. В данной работе мы научимся использовать для этих целей язык структурированных запросов SQL (StructuredQueryLanguage).

Основным оператором языка SQL, позволяющим осуществлять отбор информации из базы данных, является оператор SELECT, который в простейшем виде может быть задан следующим образом:

SELECT <список колонок, включаемых в ответ>FROM <список таблиц>

WHERE <условие>;

Предложения SELECT (отобрать) и FROM (из) должны присутствовать обязательно. Условие WHERE (где) может быть опущено. Тогда в ответ войдут все строки, имеющиеся в таблице (SQL позволяет управлять выводом в ответ повторяющихся строк, и можно добиться как вывода только уникальных строк, так и включения в ответ повторяющихся строк).

Оператор SELECT может включать в себя и другие предложения, позволяющие, в частности, осуществлять упорядоченность ответа, выполнять обобщающие функции. Если в ответ должны войти все колонки, имеющиеся в исходной таблице, то вместо их перечисления в SELECT можно поставить знак «*».

Так, например, запрос «Выдать всю информацию о читателях из таблицы **Читатели**, которые проживают на улице Чкалова» может быть представлен на SQL следующим образом:

```
SELECT Читатели.*
FROM Читатели
WHERE ((Читатели.[Домашний адрес]) Like "ул. Чкалова" & "*");
```

Условие, задаваемое в предложении WHERE, может быть простым и сложным. Для формулирования сложного условия могут быть использованы логические операторы And и Or. Так, например, ранее составленный запрос **Операторы сравнения для поиска цены** может быть представлен на SQL следующим образом:

```
SELECT Книги.Автор, Книги.Название, Книги.[Год издания],
Книги.Стоимость
FROM Книги
WHERE ((Книги.Стоимость)>=20000 And
(Книги.Стоимость)<=30000)
ORDERBYКниги.Стоимость;
```

Оператор SELECT оперирует над множествами и результатом обработки в общем случае является множество строк. К этим множествам могут быть применены теоретико-множественные операции объединение (UNION), пересечение (INTERSECTION), разность (DIFFERENCE, MINUS, EXCEPT) и др. В разных реализациях языка SQL наборы теоретико-множественных операций различаются.

Язык SQL позволяет запрашивать вычисляемые значения. В этом случае в предложении SELECT указывается выражение для вычисления значения колонки. Например, в рассмотренном ранее запросе **Стоимость книг в условных единицах** запрашивается вывод стоимости книг в условных единицах путем ее вычисления на основе хранящейся в таблице **Книги** стоимости по соответствующей формуле:

```
SELECT Книги.Автор, Книги.Название, Книги.[Год издания],
[Стоимость] /2165 AS [Цена в у_е]
FROM Книги;
```

С помощью конструкции AS в этом запросе задано имя столбца-результата.

Запрос может быть простым, состоящим из одного оператора SELECT, и вложенным, когда один оператор SELECT включается в состав другого оператора. Этот включенный оператор называется подзапросом (subselect) или подчиненным запросом. Существуют два типа вложенных подзапросов: обычный и коррелированный. В обычном подзапросе внутренний запрос выполняется первым, и его результат используется для выполнения основного запроса. В коррелированном подзапросе внешний запрос выполняется первым, и его результат используется для выполнения внутреннего запроса. Внутренний запрос выполняется для каждой строки, возвращенной внешним запросом.

В запросе можно указать упорядоченность ответа по определенному признаку (полю, совокупности полей, выражению).

Возможнаподгруппировка данных в целях получения подытогов или других обобщающих величин (среднее, минимум, максимум и др.). Набор агрегатных функций отличается в разных системах. В запросе допускается только один уровень группировки. Группировка может осуществляться как по одному полю, так и по совокупности полей.

В некоторых реализациях языка SQL отобранные оператором SELECT данные могут быть сохранены в виде таблицы базы данных

При выполнении запроса может возникнуть необходимость соединения двух или более таблиц. Возможны разные способы задания условия соединения (вложенные запросы, задание условия соединения в предложении WHERE, операция JOIN в предложении FROM).

Общая характеристика оператора SELECT

Для отбора информации из базы данных служит оператор SELECT. Синтаксис оператора выглядит следующим образом:

```
SELECT [DISTINCT]
        {{функция агрегирования | выражение для вычисления
значения [AS имя столбца]}.,}
| {спецификатор.*}
|*
FROM {{ имя таблицы [AS][имя корреляции].[(имя столбца.,..)]}
| {подзапрос [AS][имя корреляции.[имя столбца.,..]}
| соединенная таблица }.,..
[WHERE предикат]
        [GROUP BY {{[ имя таблицы | имя корреляции]}.| имя
столбца}.,..{] [HAVINGпредикат]
[UNIONIINTERSECT | EXCEPT}[ALL]
[CORRESPONDING [BY (имястолбца.,..)]]
        операторSELECT | TABLEимятаблицы |
конструкторзначенийтаблицы]
[ORDERBY{{столбец-результат [ASC | DESC]}.,..}
| {{положительное число[ASC | DESC]}.,..}]};
```

Оператор состоит из предложений SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY, которые должны быть записаны в команде именно в той последовательности, в которой они перечислены в синтаксической формуле.

Предложение SELECT определяет столбцы таблицы, получаемой в результате выполнения запроса. Столбец результатной таблицы может

быть задан именем столбца исходной таблицы. Если в запросе используется несколько таблиц и в них имеются поля, имеющие одинаковые имена, то для указания такого поля используется конструкция <имя таблицы>.<имя поля>. Кроме того, в предложении SELECT могут использоваться любые допустимые выражения, которые зададут формулу для определения вычисляемого поля. С помощью конструкции [AS <имя столбца>] можно задать имя столбца-результата. КонструкциюAS можно использовать не только тогда, когда определяются вычисляемые поля, но и во всех других случаях, когда нужно задать имя столбца-результата, отличающееся от имени столбца исходной таблицы.

Результат выборки может в принципе содержать повторяющиеся строки. Чтобы избежать вывода повторяющихся строк в ответе, используется параметр DISTINCT.

Запросы могут использовать функции агрегирования. Стандартязыка SQL предусматривает использование следующих функций агрегирования: Count – подсчет, Sum– сумма,Max– максимум, Min– минимум, Avg - среднее.

Чаще всего функции агрегирования используются совместно с предложением GROUP BY, но могут применяться и самостоятельно. В последнем случае результат относится не к какой-то группе, а ко всей выборке.

Существуют два типа функции COUNT. Первый тип использует символ «*». В этом случае функция подсчитывает количество строк в группе. Отдельные значения столбцов при этом не учитываются, и результат не будет зависеть от того, имеются ли в полях значения Null и указан ли параметр DISTINCT. Второй тип функции COUNT игнорирует значения Null.

Если в ответ требуется включить все поля таблицы, то для этого можно использовать символ «*». Если запрос многотабличный, то следует применять конструкцию {спецификатор. *}.

В предложении FROM указываются таблицы, которые используются при формулировании запроса. Кроме этого, в качестве источника данных в запросе могут быть заданы представления.

Начиная со стандарта SQL-92, в предложение FROM можно включать встроенный оператор JOIN, который служит для задания разнообразных условий соединения таблиц, участвующих в запросе.

В предложении WHERE задается условие отбора записей. Предложение может включать одно выражение или несколько. Части сложного условия соединяются логическими операторами AND (И) или

OR (ИЛИ).В выражениях могут использоваться следующие операторы сравнения: = (равно), <> (не равно), < (меньше), <= (меньше или равно), > (больше), >= (больше или равно), которые могут предваряться оператором NOT. Выражение может принимать одно из трех значений: TRUE, FALSE, UNKNOWN. В результатную таблицу переносятся те строки, для которых значение предиката равно TRUE.

Кроме стандартных операторов сравнения в SQL можно использовать специальные операторы предикатов:<интервальный предикат >, <предикат IN>, <предикат проверки на неопределенное значение>, <предикат подобия>.

При использовании интервального предиката диапазон значений можно задавать в виде

WHERE [NOT]<выражение> BETWEEN <нижнее выражение> AND <верхнее выражение>

При использовании предиката IN предложение WHERE будет иметь следующий вид:

WHERE [NOT]<выражение> [NOT] IN <список значений>|<под-запрос>

Предикат подобия применяется для поиска подстроки в указанной строке. Предложение WHERE при использовании предиката этого типа будет иметь следующий вид:

WHERE [NOT] <выражениедлявычислениязначениястроки1> [NOT] LIKE <выражениедлявычислениязначениястроки2>

Предикат проверки на неопределенное значение имеет вид

предикат NULL ::=конструктор значения строки IS [NOT] NULL

При использовании подзапросов в условии WHERE может быть использован квантор существования EXISTS. Формат условия WHERE в этом случае имеет вид

WHERE [NOT] EXISTS <подзапрос>

Предложение GROUP BYиспользуется для определения групп выходных строк, к которым могут применяться те или иные агрегатные функции. Предложение GROUP BY всегда используется со встроенными агрегатными функциями. Обратное утверждение неверно. Агрегатные функции могут использоваться в предложениях SELECT, HAVING. Если агрегатные функции используются без предложения GROUP BY, то они будут применяться ко всему набору строк, удовлетворяющему условию запроса.Конструкция GROUP BY работает только на одном уровне. Нельзя разбить каждую из этих групп на группы более низкого уровня, а затем применять стандартную функцию на каждом уровне подчиненности.

Фраза GROUP BY означает логическую перекомпоновку (группировку) таблицы по указанной колонке (колонкам). Физически таблицы в базе данных не перекомпоновываются. Логика выполнения запроса при использовании GROUP BY несколько отличается от реализации обычного запроса. Фраза SELECT при использовании GROUP BY применяется к каждой группе, а не к каждой строке, как обычно.

Каждое выражение во фразе SELECT должно принимать единственное значение для группы, т.е. оно может быть либо самой колонкой, либо арифметическим выражением, включающим эту колонку, либо агрегатной функцией, которая получает в результате единственное значение для группы. Кроме того, в SELECT может быть включена константа.

Вместе с предложением GROUP BY может использоваться предложение HAVING, которое для групп имеет то же значение, что и фраза WHERE – для строк.

Корректирующие операторы

Оператор INSERT позволяет включить в таблицу новые строки. Он имеет следующий вид:

INSERT INTO имя таблицы [(имя столбца ,...)]

выражение запроса [конструктор значений таблицы] |{DEFAULT VALUES}

Если список столбцов не задан, то значения должны вводиться в каждый столбец таблицы; если список столбцов задан, то значения соответственно должны вводиться в те столбцы, которые перечислены в списке, и в том порядке, в котором они расположены в нем.

Элементы в списке значений могут быть константами, функциями, переменными памяти. Если эти элементы являются константами, то при их задании используются определенные разделители в зависимости от типа вводимых данных: символьные данные заключаются в кавычки, даты – в фигурные скобки, логические – в точки, числовые данные вводятся без разделителей.

Пример использования оператора INSERT:

INSERTINTOИздательстваVALUES (6, "Новое знание", "Минск");

В данном примере значения вводятся во все столбцы таблицы, поэтому <список столбцов> не указан.

Если значения, которые необходимо ввести, являются результатом выполнения запроса, то эти значения также помещаются в специфицированные колонки и должны соответствовать им по типу. При использовании <подзапроса> в указанную таблицу вводятся данные,

отобранные из другой таблицы (или даже нескольких таблиц).

Командой, позволяющей корректировать содержание таблицы, является оператор UPDATE, имеющий следующий формат:

UPDATE <имя таблицы> SET <имя столбца> = <новое значение> [, <имя столбца> = <новое значение> ...] [<предложение WHERE>];

Используя оператор UPDATE, можно изменить значения указанногостолбца для всех записей таблицы, если предложение WHERE не задано, или для записей, удовлетворяющих условию запроса, если используется предложение WHERE.

Оператор UPDATE Книги SET Стоимость=Стоимость*1.1; увеличивает стоимость книг для всех записей в таблице **Книги**на 10 %.

Оператор UPDATE Книги SET Стоимость=Стоимость*0.9 WHERE [Год издания]<2000; уменьшает стоимость книг, изданных до 2000 года, на 10 %.

Оператор DELETEможно использовать для удаления строк таблицы: DELETE FROM <имя таблицы> [<предложение WHERE>];

Следует быть осторожным при использовании оператора DELETE, поскольку, если фраза WHERE в операторе DELETE отсутствует, будут удалены все строки таблицы. То же самое произойдет, если неправильно указать условие отбора и в результате не будет отобрано ни одной строки в таблице. Оператор DELETE физически удаляет строки таблицы.

Задание

1. Просмотрите на SQLранее созданные запросы: **Поиск книг пофамилии автора**, **Рейтинг книг**, **Список читателей с инициалами**. Чтобы увидеть, как выглядит запрос в MSAccessна SQL, надо в окне базы данных во вкладке **Запросы** выделить имя запроса, нажать на кнопку**Открыть** или **Конструктор**, а затем в меню **Вид** выбрать команду **Режим SQL**. Например, запрос **Книги с ключевым словом в теме** на SQLбудет выглядеть так, как показано на рис. 1.

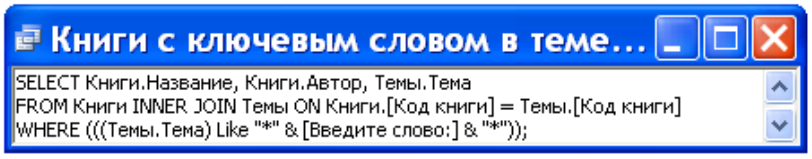


Рис. 1. Окно с текстом запроса на SQL.

2. Чтобы приступить к созданию запроса на SQL, надо открыть вкладку **Запросы** окна базы данных, выполнить двойной щелчок мышью на команде **Создание запроса в режиме конструктора**, закрыть

диалоговое окно **Добавление таблицы** и в меню **Вид** выбрать команду **Режим SQL**. Создайте на SQL запрос, который будет выводить о читателях, заказавших книгу **Язык Ада**, следующую информацию: Фамилию, Имя, Отчество, Домашний адрес. Запрос назовите **Поиск читателей по заказанной книге**.

3. Сформулируйте задачу, которую решает следующий запрос на SQL:

```
SELECT Книги.*
FROM Книги
WHERE Название Like "Я"&"*";
```

4. Выясните, что делает приведенный ниже запросна SQL.

```
SELECT Автор, Название, Наименование, Город, [Год издания]
FROM Издательства,  Книги
WHERE Издательства.[Код издательства]=Книги.[Код издательства];
```

Назовите этот запрос так, чтобы было ясно, что он делает.

5. Выполните следующий запрос на SQL:

```
SELECTCount(*) ASКоличество
FROM Книги;
```

и укажите, что он делает. Дайте ему соответствующее имя.

6. Выясните назначение приведенных ниже двух запросов на SQL.

```
SELECT Читатели.[Код читателя], Читатели.Фамилия,
(SELECT COUNT ([Выдача книг].[Код книги])
FROM [Выдача книг]
WHERE [Выдача книг].[Код читателя]=Читатели.[Код читателя]) AS Количество
FROM Читатели;
```

```
SELECT Читатели.Фамилия, Count([Выдача книг].[Код книги]) AS Количество
FROM [Выдача книг], Читатели
WHERE ([Выдача книг].[Код читателя])=Читатели.[Код читателя]
GROUP BY Читатели.Фамилия;
```

7. Составьте на SQL запрос, который будет вычислять количество прочитанных каждым читателем страниц. Попытайтесь это сделать двумя способами, показанными в предыдущем задании.

Лабораторная работа 9

Представление данных в виде форм

Цель работы: Сформировать умения создавать формы для представления данных.

Форма, также как таблица и запрос, может использоваться в MSAccess для ввода информации в базу данных и для ее обработки. Более того, если база данных постоянно пополняется новыми записями или информация базы данных часто изменяется, удобней использовать форму. При создании формы можно указать, какие поля и в какой последовательности должны быть в ней представлены, разбить поля на логически связанные группы, задать удобное расположение на экране. Любая форма создается на основе таблиц и запросов. Данные из одной таблицы могут быть представлены в нескольких формах, и, наоборот, в одной форме могут содержаться данные из различных таблиц и запросов. Кроме того формы могут содержать иллюстрации, графически представлять хранящуюся в базе данных информацию. Таким образом, формы позволяют создать удобный пользовательский интерфейс для работы с данными.

Конечно, создание форм требует дополнительных усилий. Однако потраченное время будет возмещено за счет уменьшения ошибок при вводе, удобства доступа к хранящейся в базе данных информации, наглядности ее представления и облегчения восприятия. Кроме того, форма может служить защитой базы данных от действий неквалифицированных пользователей.

Так же как и в случае таблиц или запросов, самым простым способом создания форм является использование мастера. Мастер задает пользователю вопросы о структуре и оформлении формы, предлагая на выбор несколько вариантов. В процессе создания формы можно вернуться на несколько шагов назад, чтобы изменить принятые решения или выбрать другой вариант. В результате такого диалога появляется готовая к применению форма.

Для создания формы с помощью мастера необходимо выполнить следующие действия:

– В окне базы данных раскройте вкладку **Форма** или выберите команду **Объекты базы данных |Формы** в меню **Вид.**

– Нажмите кнопку**Создать** в окне базы данных или в меню **Вставка** выберите команду **Форма**. На экране появится диалоговое окно **Новая форма** (см. рис.1).

–В поле ввода с раскрывающимся списком можно ввести имя таблицы или запроса, данные из которых необходимо представить в форме. Это целесообразно сделать лишь в том случае, если в форме будут использоваться данные из одной таблицы или одного запроса. В противном случае источник данных в этом окне можно не задавать. Нажмите кнопку **Мастер форм**, а затем – кнопку **ОК.**

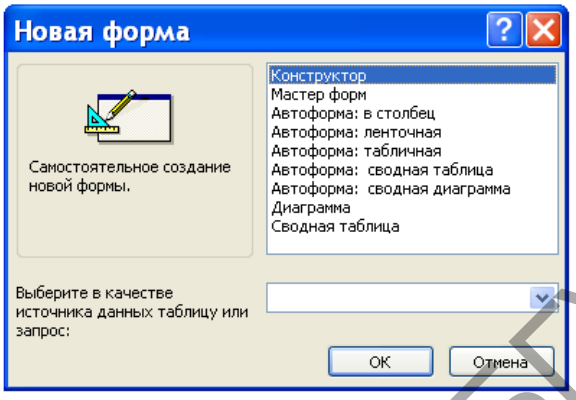


Рис. 1. Диалоговое окно **Новая форма**.

После этих действий на экране появится диалоговое окно **Создание форм**, показанное на рис. 2. Это же окно можно получить на экране более быстрым способом – двойным щелчком мыши на команде **Создание формы с помощью мастера** во вкладке **Формы** окна базы данных.

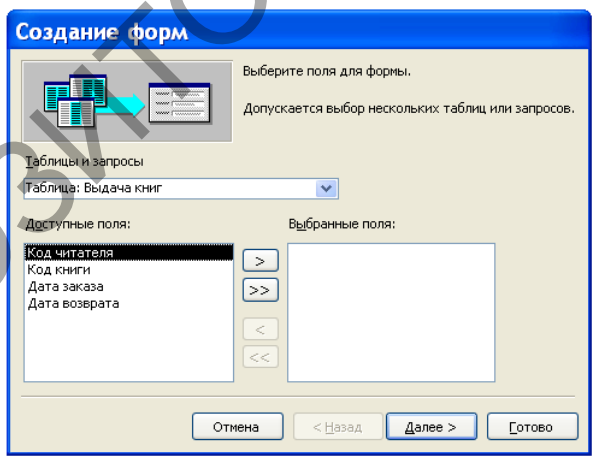


Рис. 2. Диалоговое окно **Мастер форм**.

– В этом окне из списка **Таблицы и запросы** выберите требуемые имена, а затем в нижней части окна из левого столбца перенесите требуемые поля вправый столбец, используя для переноса кнопки с символами >, >>. Копки <, << используются для отмены переноса полей.

– После нажатия кнопки**Далее** появится диалоговое окно, в котором надо выбрать вид представления формы. Допустимы три вида представления: одиночная форма, подчиненная форма или связанная

форма. Выберите требуемый вид представления формы.

– Для перехода к следующему окну нажмите кнопку**Далее**. В этом окне выберите внешний вид формы: в один столбец, ленточную, табличную или выровненную форму.

– После этого действуйте в соответствии с инструкциями, приведенными в ряде диалоговых окон.

Процесс построения формы рассмотрим на следующем примере. Требуется создать форму, которая позволит просматривать содержание книг.

В окне базы данных **Библиотека** во вкладке **Формы** выполните двойной щелчок мышью на команде **Создание формы с помощью мастера**.

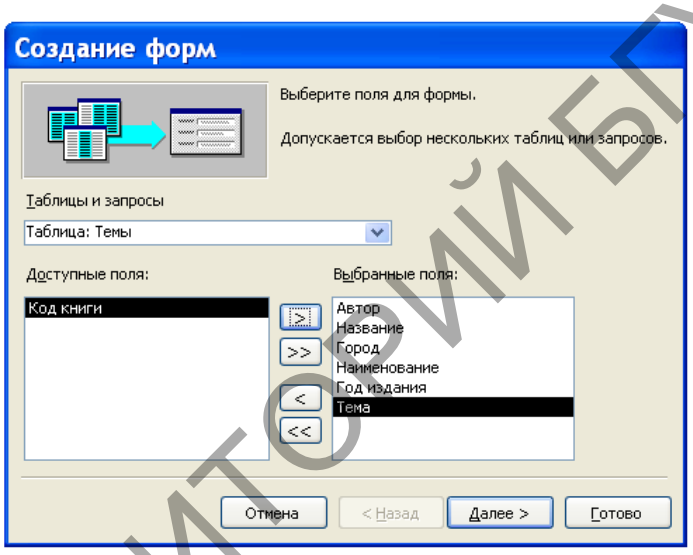


Рис. 3. Диалоговое окно **Создание форм** с выбранными полями.

В появившемся диалоговом окне (см. рис. 3) вначале выберите таблицу **Книги** и из нее в правый столбец перенесите поля **Автор** и **Название**. После этого выберите таблицу **Издательства** для переноса полей **Город** и **Наименование**. Поступая таким образом, в правый столбец перенесите поля **Год издания** и **Тема**. Результат выполнения этих действий мы видим на рис. 3. Нажмите кнопку**Далее**.

Следующее диалоговое окно предназначено для выбора вида представления данных. Оно показано на рис. 4. В нем надо определить таблицу для подчиненной формы. Подчиненная форма в нашем случае должна базироваться на таблице **Темы**. Для этого выполните щелчок мышью на таблице **Книги** в левой части окна и, если кнопка **Подчиненные формы** не активна, выполните на ней щелчок мышью. Затем нажмите кнопку**Далее**.

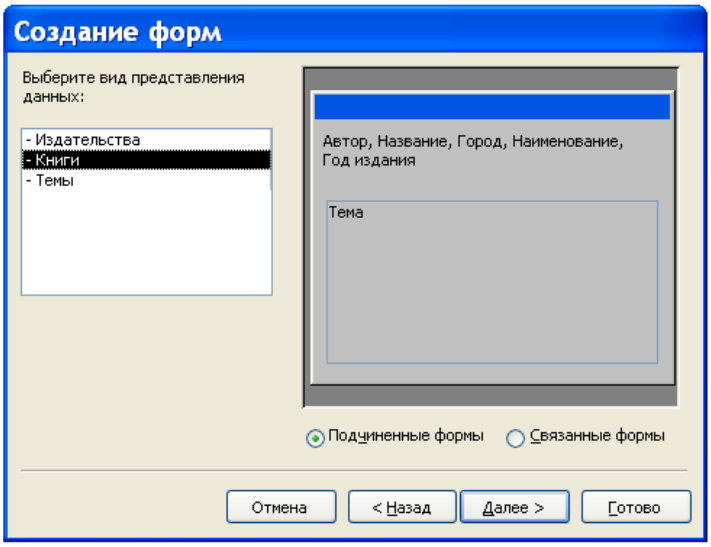


Рис. 4. Диалоговое окно для выбора вида представления данных.

В следующем диалоговом окне (см. рис. 5) для определения внешнего вида подчиненной формы нажмите кнопку **табличный**, а затем кнопку – **Далее**.

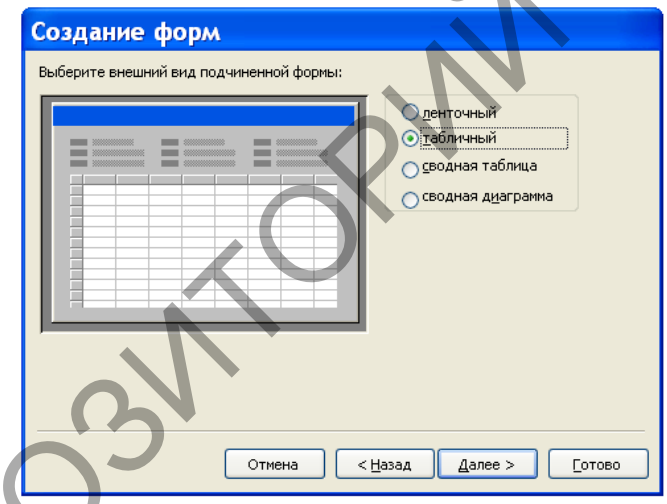


Рис. 5. Диалоговое окно для выбора внешнего вида подчиненной формы.

Диалоговое окно, показанное на рис. 6, предназначено для задания стиля формы. Выберите стиль формы – **Камень** и нажмите на кнопку**Далее**.

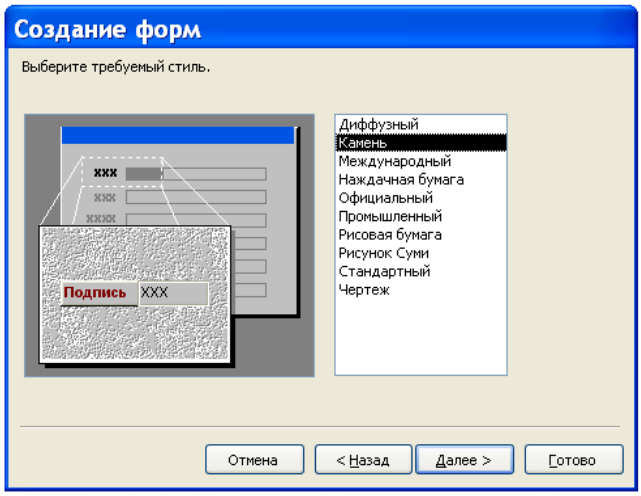


Рис. 6. Диалоговое окно для выбора стиля.

Последнее диалоговое окно при создании формы с помощью мастера представлено на рис.7. В нем укажите имя формы **Содержание книг** и имя подчиненной формы **Подчиненная форма Темы**. Поскольку мы еще не знаем, как изменить макет формы, то сделайте активной кнопку (если она не активна) **Открыть форму для просмотра и ввода данных**, а затем нажмите на кнопку **Готово**.

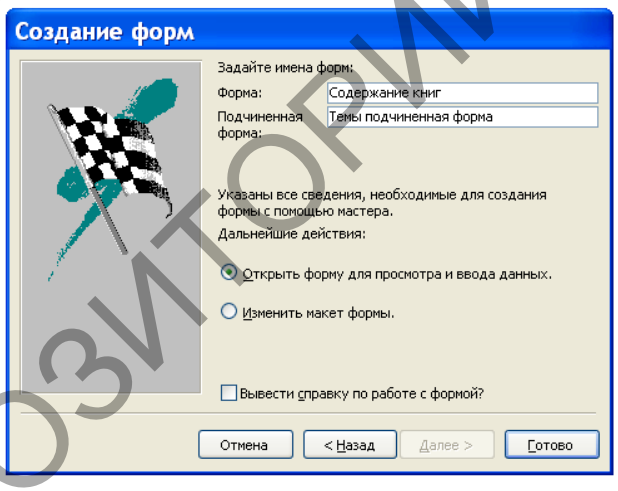


Рис. 7. Последнее диалоговое окно при создании формы.

В результате выполнения описанных действий откроется приведенная на рис. 8 форма **Содержание книг**. Безусловно, эта форма в том виде, как мы ее создали, обладает рядом недостатков, которые можно устранить при выполнении ее редактирования. Важно отметить, что поставленную задачу она все же решает – достаточно удобно позволяет просматривать содержание книг.

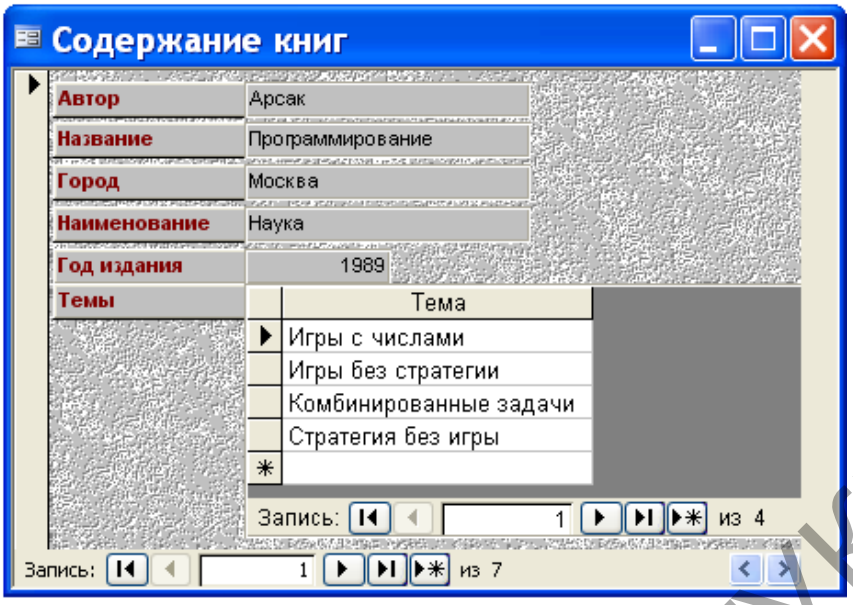


Рис. 8. Составная форма **Содержание книг**.

Каждая форма в MSAccess может быть представлена на экране в одном из четырех видов: в основном режиме работы с формой, в табличном режиме, в режиме конструирования и в режиме предварительного просмотра.

Задание

1. Для запроса **Стоимость книг с учетом инфляции** создайте автоформу в столбец. Напомним, что для создания автоформ надо выполнить следующие действия. В окне базы данных во вкладке **Формы** надо выполнить щелчок мышью на команде**Создать**, а затем в появившемся диалоговом окне **Новая форма** выбрать в качестве источника данных требуемую таблицу или запрос, далее выбрать соответствующую автоформу и нажать кнопку **ОК**. Форму для нашего задания назовите **Автоформа в столбец**.

2. Выясните, что вам напоминает форма, приведенная на рис. 9. Какую функцию она автоматизирует?

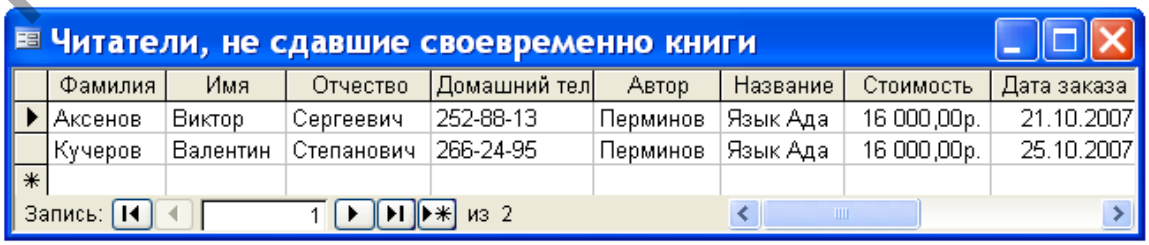


Рис. 9. Табличная автоформа.

Создайте эту автоформу и самостоятельно дайте ей название.

3. Создайте форму, приведенную на рис. 10. Самостоятельно

выясните, какую таблицу или какой запрос для этой формы надо взять в качестве источника данных. Определите, какой стиль выбран для создания этой формы.

Созданную вами форму назовите **Ленточнаяавтоформа**. Перечислите недостатки, которые вы в форме обнаружили.Каким способом можно изменить макет **Ленточнойавтоформы**?

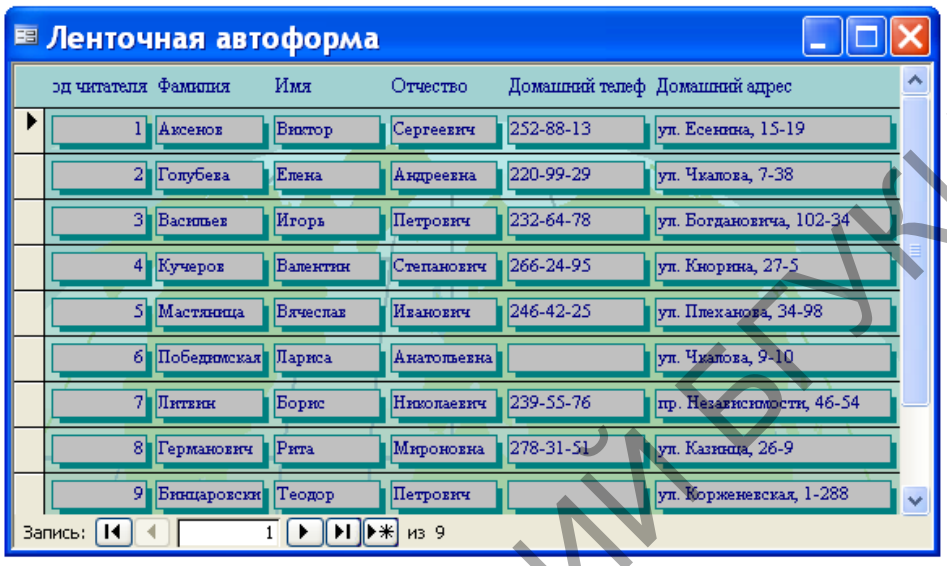


Рис. 10. Пример ленточнойавтоформы.

4. MSAccess имеет возможность представлять числовую информацию из базы данных в графической форме. Эту форму представления числовой информации называют диаграммой. Создайте диаграмму, приведенную на рис. 11.

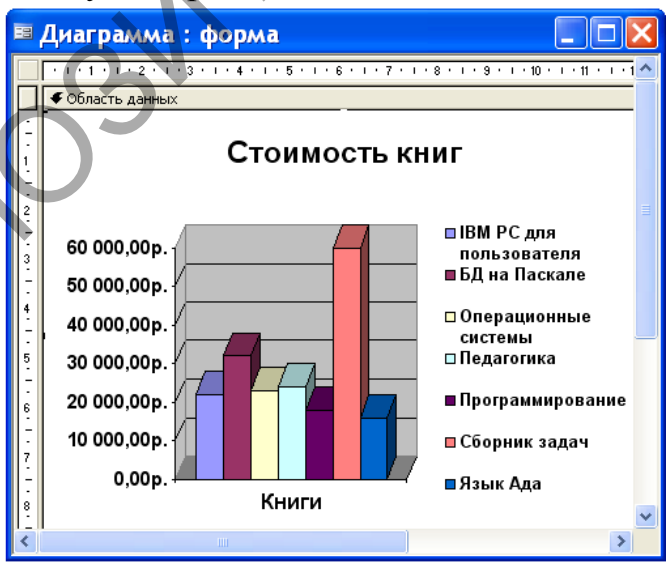


Рис. 11. Графическое представление стоимости книг.

Возможности MSAccess для построения и форматирования диаграмм являются очень ограниченными. Поэтому дополнительную обработку диаграммы можно выполнить с помощью приложения MSGraph. Вызвать

его можно в режиме конструирования формы двойным щелчком мыши в области диаграммы.

Лабораторная работа 10

Обработка данных с помощью отчетов

Цель работы: Сформировать умения для представления данных в виде отчетов.

Наилучшим средством для представления данных в виде печатного документа являются отчеты. Отчет предоставляет возможность наглядно представить извлеченную из базы данных информацию, дополнив ее результатами анализа и вычислений. В них можно отобразить данные в виде диаграммы или графика, использовать другие средства оформления. Отчеты очень похожи на формы, однако между ними имеется существенное различие – отчеты предназначены исключительно для вывода данных на печать. Поэтому в них можно отказаться от элементов управления, предназначенных для ввода данных: списков, полей ввода со списком, флажков, переключателей и т.п.

Сконструируем отчет, который будет выводить имеющуюся в базе данных информацию о книгах, их общем количестве и средней стоимости по каждому издательству. Для этого нам понадобятся данные из двух таблиц: **Издательства** и **Книги**. Поэтому перед созданием отчета необходимо сначала создать запрос.

В окне базы данных перейдите на вкладку **Запрос**, нажмите кнопку**Создать** и выберите вариант **Новый запрос**. В диалоговом окне **Добавление таблицы** выберите **Издательства** и **Книги**. В область конструктора запроса перенесите поля **Название**, **Автор**, **Объем**, **Год издания**, **Стоимость** таблицы **Книги** и поле **Наименование** таблицы **Издательства**. Сохраните запрос с именем **Каталог**.

Теперь можно начать создание отчета. Раскройте вкладку **Отчеты**в окне базы данных **Библиотека**и нажмите кнопку**Создать**. В появившемся диалоговом окне **Новый отчет** в поле ввода с раскрывающимся списком для выбора источника данных введите имя только что созданного запроса. Для создания отчета выберите команду **Конструктор** и нажмите кнопку **ОК** (см. рис. 1).

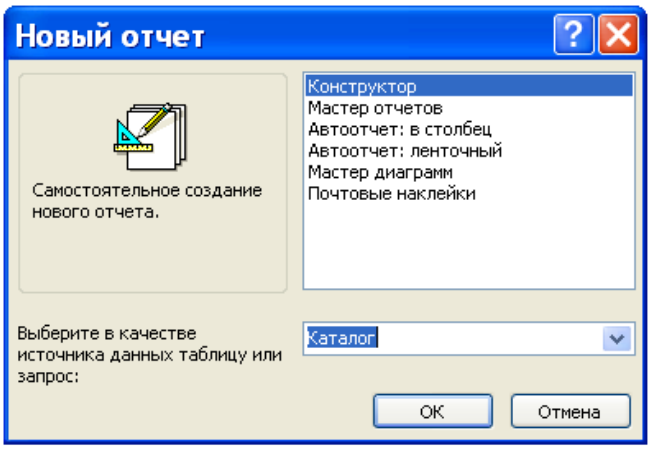


Рис. 1. Диалоговое окно **Новый отчет**.

MSAccessвыведет на экран пустой бланк отчета с разделами **Верхний колонтитул**, **Нижний колонтитул**, в центре между которыми находится **Область данных**.Этот бланк показан на рис. 2.

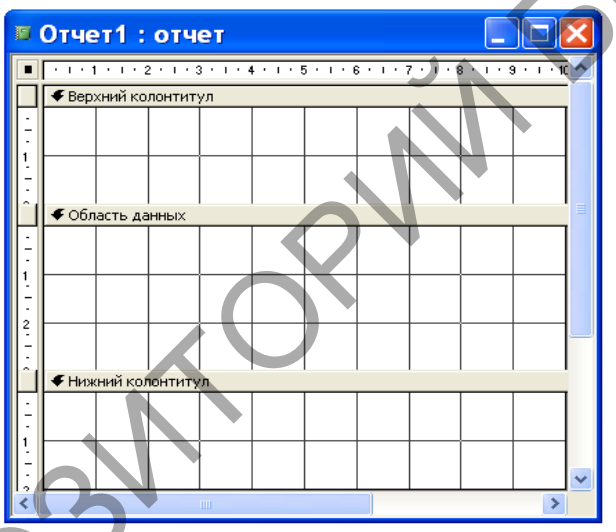


Рис. 2. Пустой бланк для создания отчета.

Линейки с делениями по верхнему и левому краям отчета помогут Вам спланировать расположение данных на странице. Если Вы хотите сделать по бокам печатной страницы поля по 2 см, то вы должны располагать предназначенные для печати элементы отчета в области шириной до 17 см для страницы стандартного размера 210x297 мм. Расположение данных на странице по вертикали определяется тем, как вы зададите верхний и нижний колонтитулы (верхнее и нижнее поля страницы). Как и при работе с формами, вы можете перетащить край любого раздела, чтобы сделать его больше или меньше. Заметьте, что ширина всех разделов должна быть одной и той же, поэтому если вы измените ширину одного из разделов, то MSAccess автоматически изменит ширину и всех других разделов.

Отчет, в отличие от формы, предоставляет пользователю возможность задать группировку данных прямо в отчете с помощью команды **Сортировка и группировка** меню **Вид**. В диалоговом окне можно определить до 10 полей или выражений, которые будут использоваться в отчете для группировки данных. Первый элемент в списке определяет основную группу, а последующие элементы – подгруппы внутри группы предыдущего уровня.

Чтобы подсчитать общее количество имеющихся в настоящее время книг и среднюю стоимость книги по каждому издательству, необходимо сгруппировать данные по полю **Наименование**. Для этого в меню **Вид** выберите команду **Сортировка и группировка.**В появившемся диалоговом окне **Сортировка и группировка**(см. рис. 3) щелкните по первой строке в столбце **Поле/выражение** и раскройте кнопку раскрывающегося списка.

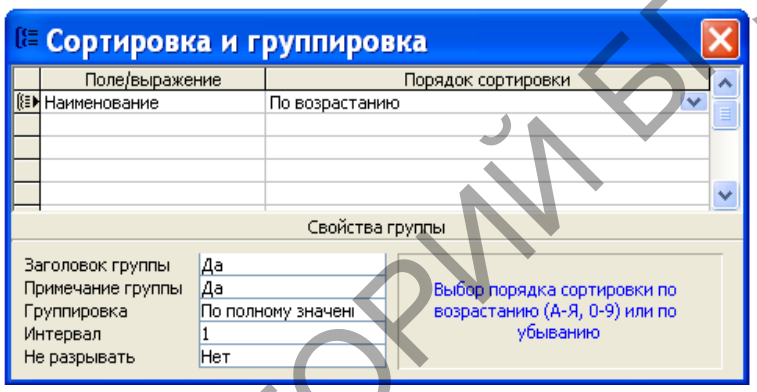


Рис. 3. Диалоговое окно **Сортировка и группировка**

Выберите поле **Наименование** (вы можете также использовать выражение, содержащее ссылку на любое поле таблицы или запроса). По умолчанию MSAccess сортирует значения по возрастанию. В отчете должно быть зарезервировано место, куда вы поместите заголовок для каждой группы и примечание для вычисляемых полей (общего количества и среднего значения). Чтобы добавить эти разделы, установите для свойств **Заголовок группы** и **Примечание группы** значения**Да**. MSAccess добавит в отчет требуемые разделы. Закройте окно сортировки и группировки.

Теперь вы можете закончить построение отчета, выполнив приведенные ниже действия (результат ваших действий должен выглядеть так, как показано на рис. 4).

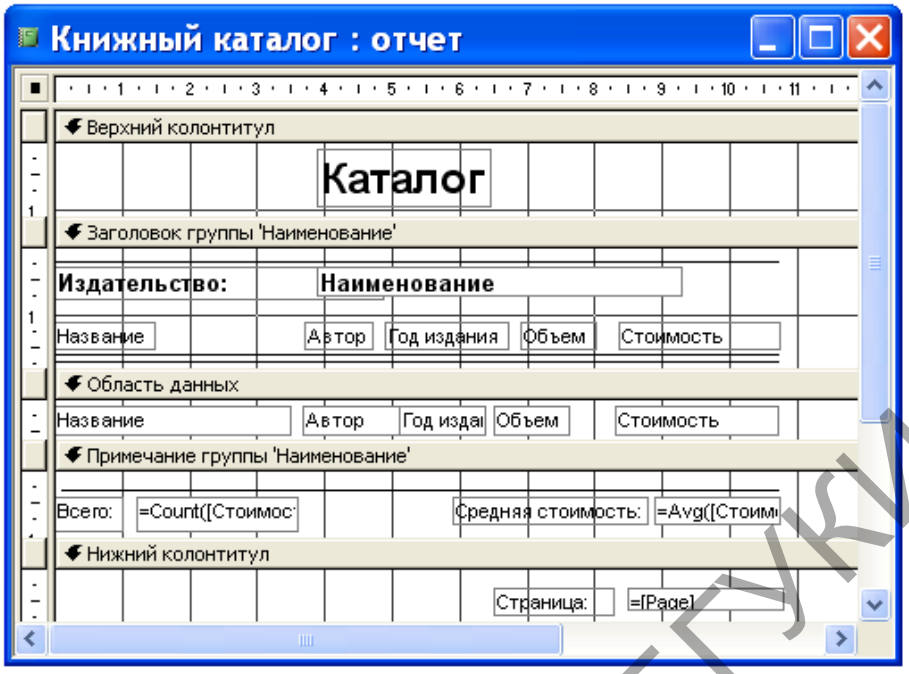


Рис. 4. Отчет **Каталог** в режиме конструктора.

– Разместите несвязанную подпись в области верхнего колонтитула в введите в нее **Каталог**. Выделите эту подпись и установите для нее полужирный шрифт ArialCyr размером 18 пунктов. Чтобы привести размер элемента управления в соответствие с назначенным шрифтом, выполните команду **Размер| по размеру данных**в меню **Формат**.

– Раскройте список полей и перетащите поле **Наименование** в область заголовка группы. Для подписи и элемента управления установите полужирный шрифт ArialCyr размером 10 пунктов. Измените текст подписи **Наименование** на **Издательство**.

– В заголовок группы необходимо поместить в качестве заголовков столбцов несколько несвязанных подписей. Для этого увеличьте область данных и перетащите поля **Название**, **Автор**, **Год издания**, **Объем** и **Стоимость** из списка полей в область данных. Выделите подпись поля **Название** и выполните команду**Вырезать** меню **Правка**, чтобы поместить ее в буфер обмена. Выделите заголовок группы и выполните команду**Вставить** меню **Правка**. Подпись будет вставлена в левый верхний угол области заголовка группы. Теперь она стала несвязанной и ее можно разместить независимо от поля. Аналогичным образом отделите остальные подписи и переместите их в заголовок группы.

– Разместите подписи столбцов в заголовке группы таким образом, чтобы они располагались точно над соответствующими элементами управления. После этого выделите все подписи и выполните команду**Выровнять| по верхнему краю**в меню **Формат**, чтобы выровнять эти подписи по горизонтали.

– Чтобы улучшить внешний вид отчета, с помощью инструмента **Линия** проведите линию вдоль верхней границы заголовка группы и две линии вдоль нижней границы заголовка группы.

– Размер области данных определяет расстояние между строками в отчете. Нам не требуется, чтобы между строками отчета было какое-то расстояние. Поэтому уменьшите размер области данных до высоты размещенных в ней полей.

– Проведите линию вдоль верхней границы примечания группы **Наименование** и разместите под ней два несвязанных поля.

– Измените подпись первого поля на **Всего:**. Откройте бланк свойств элемента управления и в качестве значения свойства **Данные** введите выражение **=Count([Название])**. С помощью этой функции вычисляется количество записей внутри группы, т.е. общее количество книг данного издательства. Поместите это поле под полем **Название**.

– Измените значение подписи второго поля на текст **Средняя стоимость:**. В качестве значения свойства **Данные** элемента управления введите выражение **=Avg([Стоимость])**. По данной формуле вычисляется среднее значение стоимости внутри группы. Чтобы вместе с числовым значением выводился денежный знак, для свойства **Формат поля** установите значение **Денежный**. Поместите это поле под полем **Стоимость**.

– В правый нижний угол нижнего колонтитула поместите несвязанное поле. Измените подпись этого поля на текст **Страница**, а для свойства **Данные** элемента управления введите выражение =[Page].

Иногда отчет должен содержать информацию, относящуюся к заголовку отчета или его примечанию. Увидеть заголовок отчета и его примечание в режиме конструктора можно, выполнив команду **Заголовок/примечание** отчета в меню **Вид**.

Чтобы просмотреть результаты работы, выберите команду **Предварительный просмотр**в меню **Файл** (см. рис. 5). Сохраните созданный отчет под именем **Книжный каталог**.

Рис. 5. Созданный отчет в режиме предварительного просмотра.

Познакомимся более детально со структурой макета отчета.Макет отчета разделен на несколько областей, каждая из которых имеет свое назначение и особенности (см. рис. 6). У верхнего края окна отчета расположена область заголовка. Заголовок появляется в сформированном отчете только один раз – в начале отчета – и будет расположен на первой странице. В нем обычно размещают подпись, а также выражение =Now(). В режиме предварительного просмотра на месте этого выражения будет помещена текущая дата. Благодаря этой функции в заголовке всегда присутствует информация о том, насколько свежи данные отчета. Ниже расположен верхний колонтитул, который в режиме просмотра вы увидите вверху каждой страницы отчета. Если отчет имеет табличную структуру, то в верхнем колонтитуле обычно располагаются имена полей.

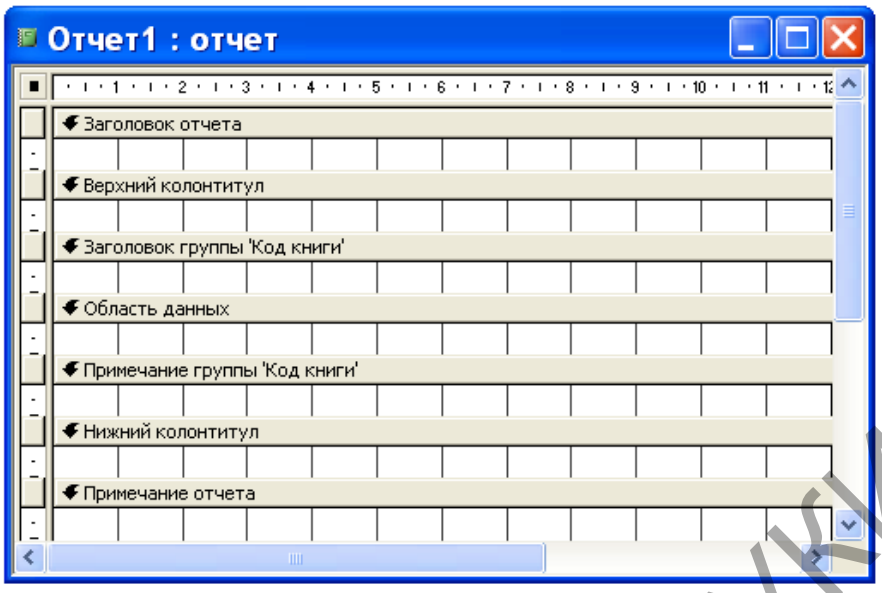


Рис. 6. Макет отчета.

Далее расположен заголовок группы, который MSAccess печатает для каждой группы данных. В случае группировки по нескольким полям в макете отчета будут присутствовать несколько заголовков групп. Ниже заголовка группы расположена область данных, содержащая основные, детальные данные для отчета. В сформированном отчете каждая группа будет представлена детальными данными для каждой записи данных из этой группы.

Далее расположено примечание группы, которое появляется в конце каждой группы записей детальных данных. Оно используется для указания промежуточных итогов по группе. Ниже расположен нижний колонтитул, который появляется в конце каждой страницы отчета. Мастер по разработке отчетов вносит в нижний колонтитул функцию =[Page]. Если отчет займет несколько страниц, то благодаря этой функции они будут пронумерованы автоматически.

И последняя область – примечание отчета. Оно появляется только один раз в конце отчета и содержит общие итоги по всем суммируемым полям. И хотя примечание отчета является последней областью макета отчета, оно появляется в сформированном отчете перед нижним колонтитулом последней страницы.

Задание

1. В нижнюю часть рассмотренного выше отчета включите общее количество книг, среднюю стоимость книги, а также средний объем книги. Полученный отчет назовите **Информация об издательствах**.

2. Создайте отчет, который будет для каждого читателя формировать список прочитанных им книг. В этот список включите поля: **Фамилия**,

Имя, **Отчество**, **Автор**, **Название** и **Стоимость**. В качестве групповых итогов отчет должен для каждого читателя вычислять количество прочитанных книг и их суммарную стоимость. Суммарная стоимость книг должна иметь денежный формат. Примечание отчета должно содержать среднее количество прочитанных книг и среднюю стоимость книги. Отчет сохраните под именем **Рейтинг читателя**.

3. Создайте отчет, который будет вычислять для каждого издательства суммарное количество страниц книг. В отчет включите следующие поля: **Наименование** из таблицы **Издательства**, а также **Название**, **Автор**, **Объем** из таблицы **Книги**. Примечание отчета должно содержать информацию об общем количестве страниц книг. Отчет назовите **Объем книг**.

Лабораторная работа 11

Использование мастера отчетов

Цель работы: Сформировать умения для создания отчетов с помощью мастера.

Простейший путь создания отчета состоит в использовании мастера отчетов. Для создания отчета с помощью мастера необходимо выполнить следующие действия:

– В окне базы данных раскройте вкладку **Отчет** или выберите команду **Отчет** в меню **Вид**.

– Выполните двойной щелчок мышью на команде **Создание отчета с помощью мастера**. На экране появится диалоговое окно **Создание отчетов** (см. рис.1).

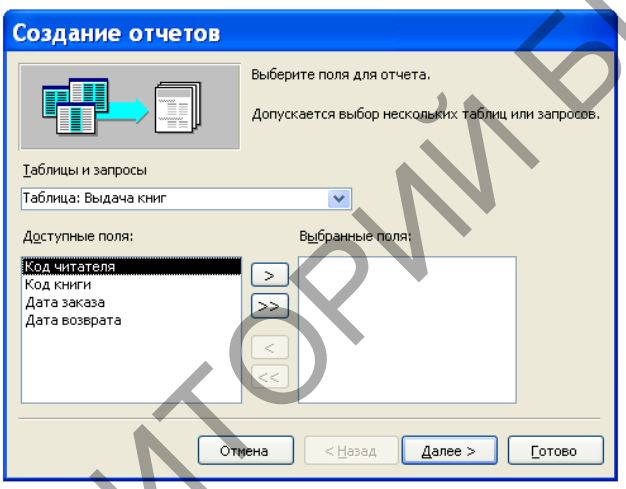


Рис. 1. Диалоговое окно **Создание отчетов**.

– В этом окне из списка **Таблицы и запросы** выберите требуемые имена, а затем в нижней части окна из левого столбца перенесите требуемые поля вправый столбец, используя для переноса кнопки с символами >, >>. Кнопки <, << используются для отмены переноса полей. Выбирать можно несколько таблиц и запросов, при этом в той очередности, которая бы обеспечила требуемую очередность расположения полей в отчете.

– Нажимая кнопку**Далее,**вы будете получать очередное диалоговое окно, в котором надо действовать в соответствии с приведенными инструкциями. Нажатие кнопки **Готово** приведет к завершению построения отчета. Так, например, если вы в диалоговом окне, приведенном на рис. 1, в качестве источника данных выберите таблицу **Выдача книг**, а затем с помощью кнопки >> перенесете все поля из этой

таблицы в отчет и нажмете кнопку **Готово**, то сразу получите отчет, приведенный на рис. 2. Автоматически этому отчету будет присвоено имя **Выдача книг**.

Мы уже познакомились с двумя способами создания отчетов: в режиме конструктора и с помощью конструктора. Рассмотрим другие возможности создания отчетов.

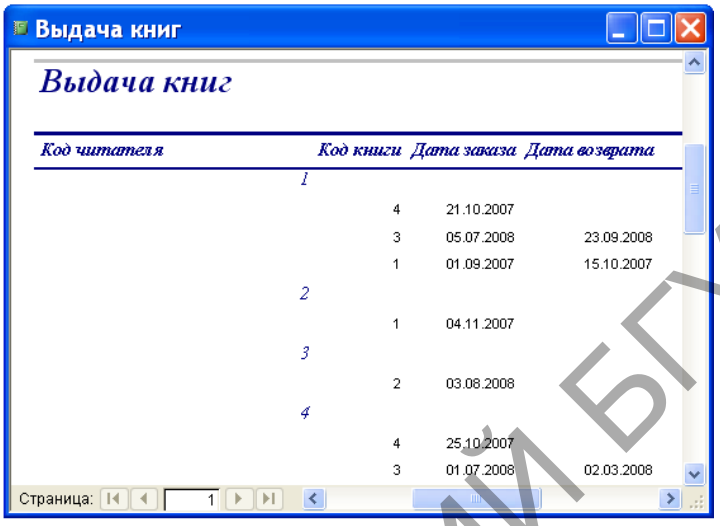


Рис. 2. Отчет **Выдача книг**.

MSAccess позволяет создавать отчет в один столбец автоматически. Созданный таким способом отчет называют автоотчетом в столбец.Для создания такого отчета поступают следующим образом. Во вкладке **Отчеты** базы данных нажимают кнопку**Создать**. В появившемся диалоговом окне **Новый отчет** выбирают команду **Автоотчет: в столбец**, задают источник данных (в нашем примере в качестве источника данных выбрана таблица **Издательства**) и нажимают кнопку **ОК**.

Вид полученного при этом отчета очень похож на форму, выводящую данные в один столбец: каждое поле расположено в отдельной строке, а записи данных строго друг под другом (см. рис. 3). Обратите внимание на то, что в строке заголовка окна отчета автоматически появилось имя отчета, совпадающее с именем источника данных – таблицы **Издательства**. При закрытии отчета мы можем дать ему требуемое имя. Назовите этот отчет в один столбец **Списком издательств**.

Автоотчет ленточный создается точно таким же образом, как и автоотчет в столбец, но вместо команды **Автоотчет: в столбец** в диалоговом окне **Новый отчет** выбирается команда **Автоотчет: ленточный**.Вид ленточного отчета соответствует виду таблицы или запроса базы данных. Поэтому не случайно в предыдущих версиях MSAccessэтот отчет назывался табличным.

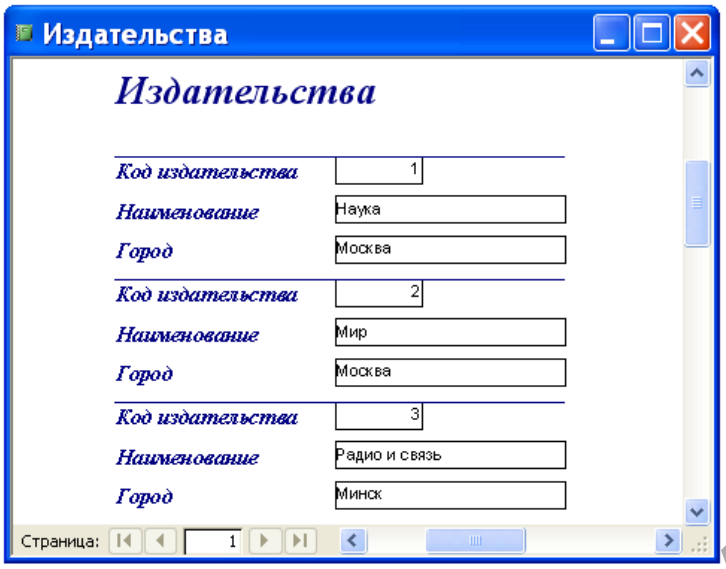


Рис. 3. Автоотчет в столбец.

Вид ленточного отчета для таблицы **Издательства** приведен на рис. 4. Последовательность расположения полей в ленточном отчете соответствует последовательности расположения этих полей в источнике данных.

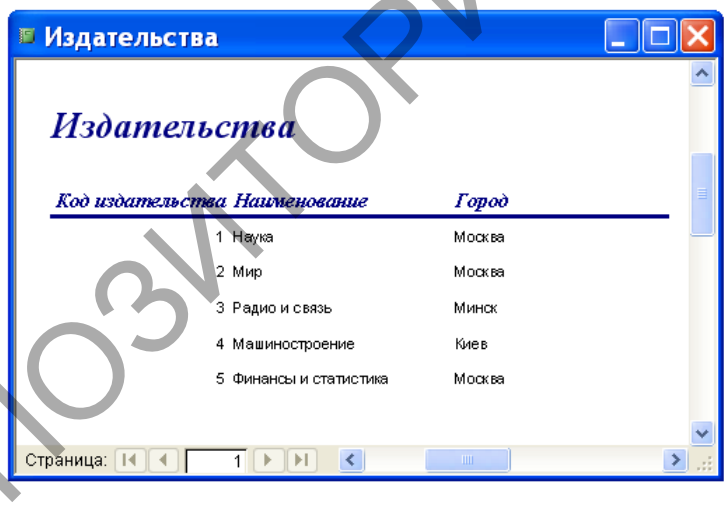


Рис. 4. Автоотчет ленточный.

MSAccess предоставляет удобный способ для создания почтовых наклеек на конверты при рассылке писем.Рассмотрим на примере, как можно создать почтовые наклейки для писем читателям. Для этого выполните следующие действия:

– Во вкладке **Отчеты** окна базы данных **Библиотека** нажмите на кнопку**Создать**.

– В появившемся диалоговом окне **Новый отчет** выберите команду **Почтовые наклейки**, установите в качестве источника данных таблицу **Читатели** и нажмите кнопку **ОК**.

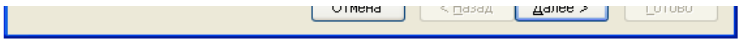


Рис. 5. Диалоговое окно **Создание наклеек**.

– В появившемся диалоговом окне **Создание наклеек** (см. рис. 5) выберите размер наклейки со следующими параметрами: **Avery J8163; 99,0x38,1; 2**. В качестве системы единиц выберите метрическую, задайте тип наклеек – **на листах**, установите фильтр по изготовителю – **Avery** и нажмите кнопку**Далее**.

– В следующем диалоговом окне все оставьте без изменения за исключением размера шрифта. Сделайте его равным 10 пунктам и нажмите кнопку**Далее**.

– В появившемся диалоговом окне для создания прототипа в первой строке поля прототипа наклейки наберите текст **Куда:**, сделайте пробел, выполните щелчок мышью в левой части окна на поле **Домашний адрес**, нажмите на кнопку **>** для переноса домашнего адреса в область прототипа наклейки. Затем нажмите на клавиатуре клавишу **Enter**и приступите к формированию второй, а после и третьей строки прототипа наклейки (см. рис. 6). Когда прототип наклейки будет создан, нажмите кнопку**Далее**.

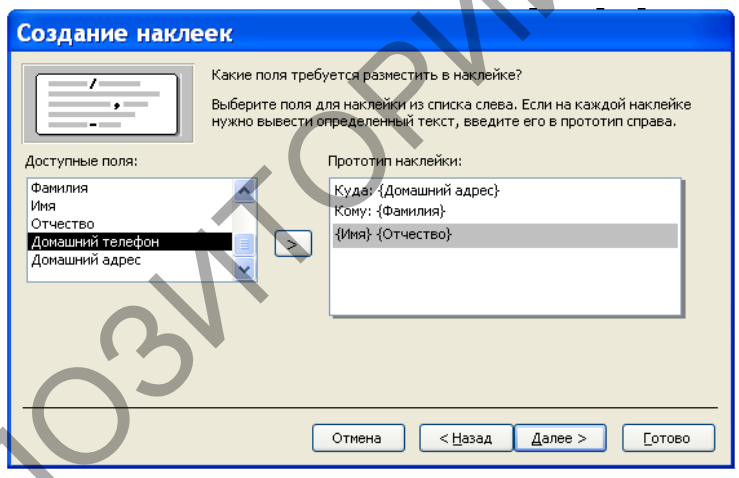


Рис. 6. Диалоговое окно для создания прототипа наклейки.

– Следующее диалоговое окно предлагает возможность отсортировать наклейки по полям таблицы **Читатели**. Эта операция используется только для того, чтобы расположить наклейки на листе в определенном порядке и облегчить работу по отысканию соответствующей наклейки. Поэтому поля, используемые для сортировки, не обязательно должны присутствовать в тексте наклейки. Отсортируйте наклейки по полям: **Фамилия**, **Имя**, **Отчество** и нажмите на кнопку**Далее**.

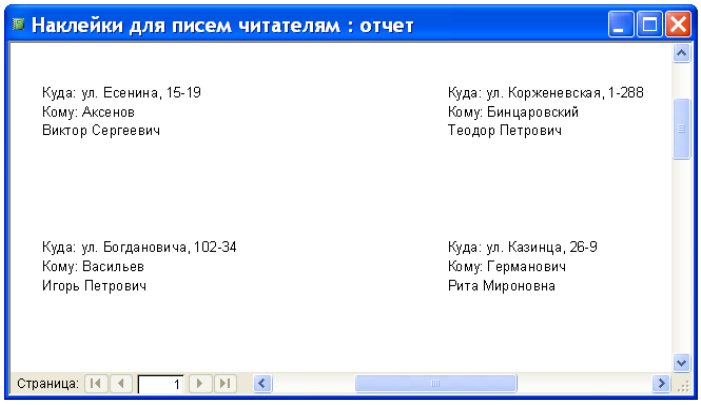


Рис. 7. Отчет с наклейками для писем читателям.

– Заключительное диалоговое окно предлагает ввести имя отчета и определить дальнейшие действия. Дайте имя отчету **Наклейки для писем читателям** и нажмите на кнопку **Готово**. Мы увидим почтовые наклейки, представленные на рис. 7. Для вывода почтовой наклейки на печать войдите в меню **Файл** и выберите команду **Печать**.

В рассмотренных выше случаях отчеты использовались просто для вывода на печать данных, оформленных специальным образом. Мастер отчетов предоставляет возможность для группировки записей и вычисления промежуточных итогов по числовым полям и общего итога для всех групп. В таком отчете записи данных располагаются в строку, в табличной форме, а подписи полей находятся у верхнего края страницы.

Сформируем этот вид отчета на основании таблицы **Книги** и сгруппируем все записи по полю **Год издания** по десятилетиям. Для создания отчета во вкладке **Отчеты** окна базы данных выполните двойной щелчок мышью на команде **Создание отчета с помощью мастера**. В появившемся диалоговом окне **Создание отчетов** задайте в качестве источника данных таблицу **Книги** и выберите из нее поля: **Название**, **Автор**, **Объем**, **Год издания**, **Стоимость** для включения их в отчет, а затем нажмите кнопку**Далее** для перехода к выполнению следующего шага.

В следующем диалоговом окне необходимо определить поля, по которым надо провести группировку (выбрать можно до 4 полей). Порядок выбора полей для группировки определяет иерархию группировки: первой выполняется группировка по первому выбранному полю, затем внутри этих групп проводится группировка по второму полю и т.д. Для данного отчета в качестве поля, по которому будет производиться группировка, выберите поле **Год издания** и нажмите кнопку **Группировка**. В следующем диалоговом окне **Интервалы группировки** необходимо определить, как именно должна производиться группировка данных.

Содержимое раскрывающегося списка **Группировка** зависит от типа данных в поле, по которому производится группировка. Текстовые поля могут быть сгруппированы по полному значению или по нескольким совпадающим первым символам – от одного до пяти. Поля даты и времени могут быть сгруппированы по полному значению, по годам, по кварталам, по месяцам, по неделям, дням, часам или минутам.

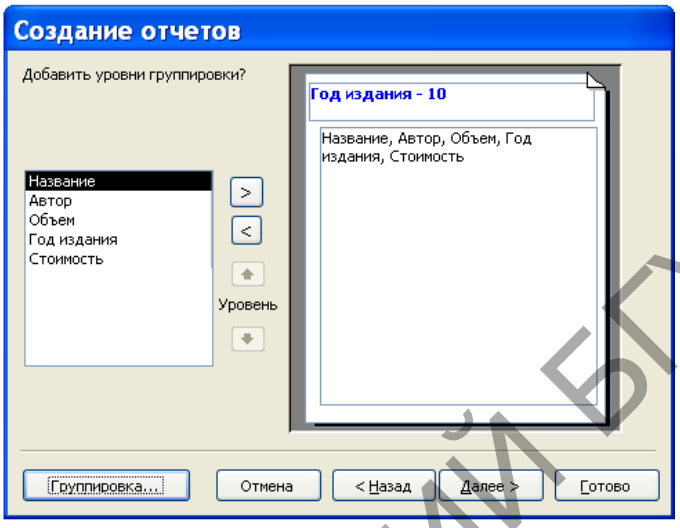


Рис. 8. Диалоговое окно **Создание отчетов** после задания группировки.

Для нашего числового поля **Год издания** предоставляются следующие интервалы группирования**: обычный** (означает по полному значению)**, 10, 50, 100, 500, 1000, 5000, 10000.** Выберите интервал группирования, равный **10**, и нажмите кнопку **ОК**, чтобы вернуться в диалоговое окно **Создание отчетов**. Оно в этот момент времени будет иметь вид, приведенный на рис. 8. Нажмите в нем кнопку**Далее**.

Далее укажите поля, по которым следует произвести дополнительную сортировку в пределах соответствующих групп. Сортировка внутри групп будет также осуществлена в той последовательности полей, которую вы определите. Произведите дополнительную сортировку по полям **Название** и **Автор** и нажмите кнопку **Итоги**. В появившемся диалоговом окне **Итоги** укажите, какие итоговые значения надо вычислить.Установленный в этом окне флажок**Вычислить проценты** означает, что под каждой из итоговых сумм будет выводиться процентная доля каждой группы в общей сумме.Настройте это окно так, как показано на рис. 9, и нажмите кнопку **ОК**. Затемв диалоговом окне **Создание отчетов** нажмите кнопку **Далее**.

В следующем диалоговом окне, предназначенном для определения вида макета отчета выберите макет с названием **по левому краю**

2,снимите флажок**Настроить ширину полей для размещения на одной странице**, ориентацию сохраните книжнуюи нажмите кнопку **Далее**.

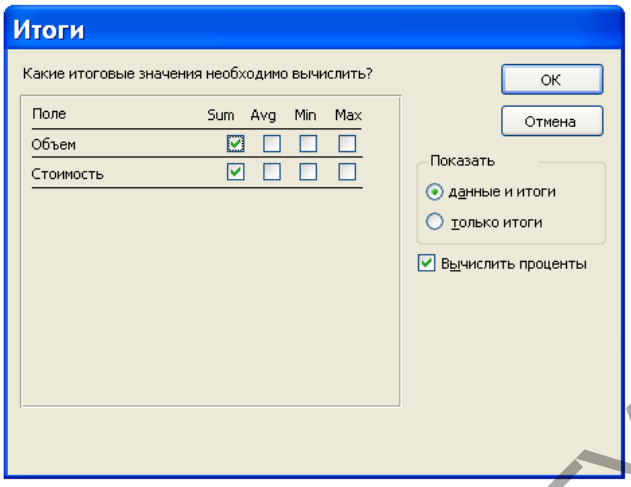


Рис. 9. Диалоговое окно для задания итоговых значений.

Следующее диалоговое окно позволяет выбрать стиль отчета. Выберите для нашего отчета стиль **Деловой** и нажмите кнопку**Далее**.

В последнем диалоговом окне **Создание отчетов** задайте имя отчета **Выпуск книг по десятилетиям**, щелкните мышью по переключателю**Просмотреть отчет** и нажмите кнопку **Готово**. Вы получите отчет, представленный на рис. 10.

Следует заметить, что наиболее предпочтительным способом создания отчетов является комбинированный способ – вначале создается “заготовка” отчета с помощью рассмотренных выше средств, а затем в режиме конструктора выполняется доработка отчета.

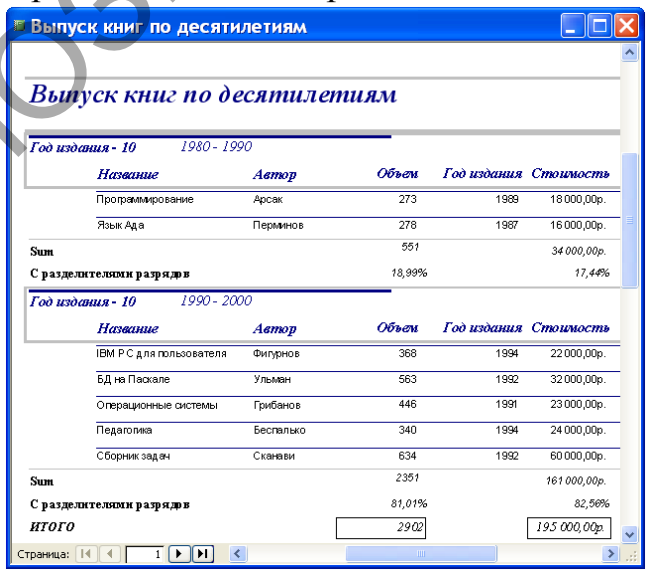


Рис. 10. Отчет с итогами.

Режим конструктора для отчетов почти идентичен аналогичному режиму для форм. Это также касается панели инструментов, панели элементов и средств, предназначенных для размещения и работы с элементами управления. Однако в отчетах отсутствует необходимость наличия управляющих элементов для ввода данных. Поэтому в них можно отказаться от использования списков, полей со списком, флажков и т.п.

Задание

1. В последнем созданном отчете **Выпуск книг по десятилетиям** в режиме конструктора из области данных удалите строку со значениями полей: **Название**, **Автор**, **Объем**, **Год издания** и **Стоимость**. Отчет назовите **Групповые вычисления**.

2. Отчет**Групповые вычисления**, полученный в предыдущем пункте, создайте с помощью мастера отчетов, не привлекая для этих целей режим конструктора. Дайте название отчету**Возможности мастера отчетов**.

3. Создайте отчет с наклейками для писем читателям, которые своевременно не сдали книги. Для этих целей вначале создайте запрос, который будет содержать необходимую информацию, а затем его возьмите в качестве источника данных для создания отчета. Отчет назовите **Наклейки для уведомления читателей.**

4. Создайте отчет, который будет содержать только итоговые строки, показывающие заказы книг по годам. Отчет сохраните под именем **Итоги заказов по годам**.

Лабораторная работа 12

Импорт данных

Цель работы: Сформировать умения для организации импорта электронных таблиц в базу данных MSAccess.

Импорт данных из электронных таблиц

MSAccess позволяет импортировать данные из файлов электронных таблиц, созданных в Lotus 1-2-3, Lotus 1-2-3 для WindowsMSExcel версии 2 и выше. Вы можете импортировать всю электронную таблицу или только ее часть как в новую, так и существующую таблицу MSAccess. Если первая строка электронной таблицы содержит заголовки столбцов, вы можете использовать их в качестве имен полей новой таблицы MSAccess.

Импорт электронной таблицы в базу данных Access можно выполнить следующим образом:

1. Откройте базу данных, в которую вы хотите импортировать электронную таблицу. Если она уже открыта, переключитесь в окно базы данных.

2. Выберите команду **Файл, Внешние данные, Импорт**. При этом откроется диалоговое окно **Импорт**.

3. В раскрывающемся списке **Тип файлов** этого окна выберите тип электронной таблицы, которую вы хотите импортировать. Затем найдите исходную папку и выделите имя файла, содержащего импортируемую электронную таблицу.

4. Нажмите кнопку **Импорт**. Если файл содержит несколько рабочих листов или именованных диапазонов, откроется первое окно мастера импорта электронных таблиц. В этом диалоговом окне выберите нужный рабочий лист или именованный диапазон и нажмите кнопку**Далее**.

5. MSAccess откроет следующее окно мастера импорта электронных таблиц. Установите флажок**Первая строка содержит заголовки столбцов**, если вы хотите использовать значения, находящиеся в первой строке электронной таблицы, в качестве имен полей таблицы MSAccess. После нажатия кнопки**Далее** в появившемся окне мастера вы можете добавить импортируемые данные в существующую таблицу MSAccess, указав ее имя, или сохранить их в новой. Нажмите кнопку**Далее**, чтобы перейти к следующему шагу.

6. Если для сохранения данных выбрана новая таблица, в следующем окне мастера, вы можете изменить определения ее полей, в том числе указать индексные поля. Раскрывающийся список **Индекс** содержит те же значения, что и свойство **Индекс** таблицы в режиме конструктора. В поле

со списком **Тип данных** отображается тип данных текущего поля таблицы, выбранной мастером на основе анализа нескольких первых строк. Некоторые поля можно и не включать в новую таблицу MSAccess. Для таких полей устанавливается флажок**Не импортировать поле**. Нажмите кнопку**Далее**, чтобы перейти к следующему шагу.

7. Очередное окно мастера позволяет определить первичный ключ таблицы MSAccess. Лучше здесь ключ не создавать и не определять, а установить переключатель**Не создавать ключ** и нажать на кнопку **Далее**.

8. В последнем окне мастера можно ввести имя для новой таблицы и попросить MSAccess запустить мастера анализа таблиц после завершения импорта данных. Если указано имя существующей таблицы, MSAccess спросит, хотите ли вы заменить имеющуюся таблицу новой.

9. Чтобы импортировать данные электронной таблицы, нажмите кнопку Готово. MSAccess откроет окно сообщения, информирующее о результате выполнения операции импорта. Если все прошло успешно, то по умолчанию новой таблице MSAccess будет присвоено имя исходной электронной таблицы. Если данные добавляются в существующую таблицу и MSAccess обнаружил ошибки, вы можете завершить импорт электронной таблицы с ошибками или вернуться к мастеру и попытаться устранить причину (например, внести изменения в определения полей). В некоторых случаях вам придется выйти из мастера и исправить данные в исходной электронной таблице. Иногда имеет смысл исправить данные уже в полученной таблице в MSAccess.

Типы данных для полей новой таблицы MSAccess определяет, анализируя значения в первых импортируемых строках. При импорте данных из электронных таблиц MSAccess анализирует значения в первых импортируемых строках. Алфавитно-цифровая информация сохраняется в текстовых полях с размером 255 символов, числовые данные – в числовых полях со свойством **Размер поля**, установленным в значение**С плавающей точкой (8 байт)**, числовые данные в денежном формате – в денежных полях, значения дат или времени – в полях типа **Дата/время**. Если в первых строках столбца MSAccess обнаружит смешанные данные, он импортирует столбец в текстовое поле.

Задание

1. Создайте в папке **Мои документы** электронную таблицу **Сотрудники**, используя для этих целей приложение MSExcel. Первый лист электронной таблицы **Сотрудники** должен содержать информацию, представленную на рис. 1. Закройте приложение MSExcel.

	A	B	C	D	E
1	Код сотрудника	Фамилия и инициалы	Код должности	Стаж	Телефон
2	202	Лагодич Н.В.	101	23	209-46-74
3	205	Онопка Л.Е.	102	15	250-83-65
4	230	Коваленко С.Ю.	102	10	222-83-35
5	368	Турцевич А.К.	110	21	289-32-62
6	125	Васильева В.О.	140	18	209-46-74
7	808	Бычко Д.М.	131	5	284-86-21
8	654	Грабовский В.Ю.	125	2	220-99-07

Рис. 1. Информация о сотрудниках.

2. Запустите MSAccess и откройте базу данных **Библиотека**. Окно базы данных **Библиотека** будет выглядеть так, как это показано на рис. 2.

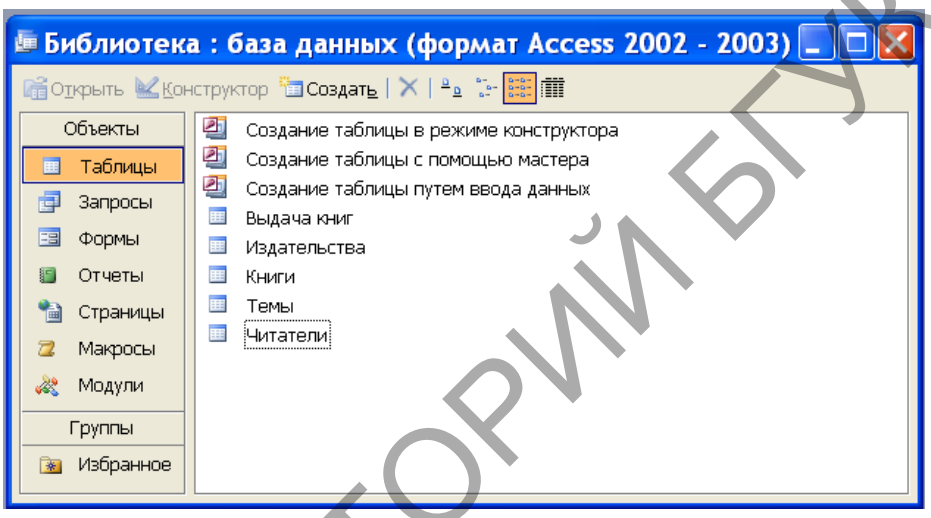


Рис. 2. Окно базы данных **Библиотека**.

3. В пункте меню **Файл** выберите команду **Внешние данные** | **Импорт**. При этом откроется диалоговое окно **Импорт**.

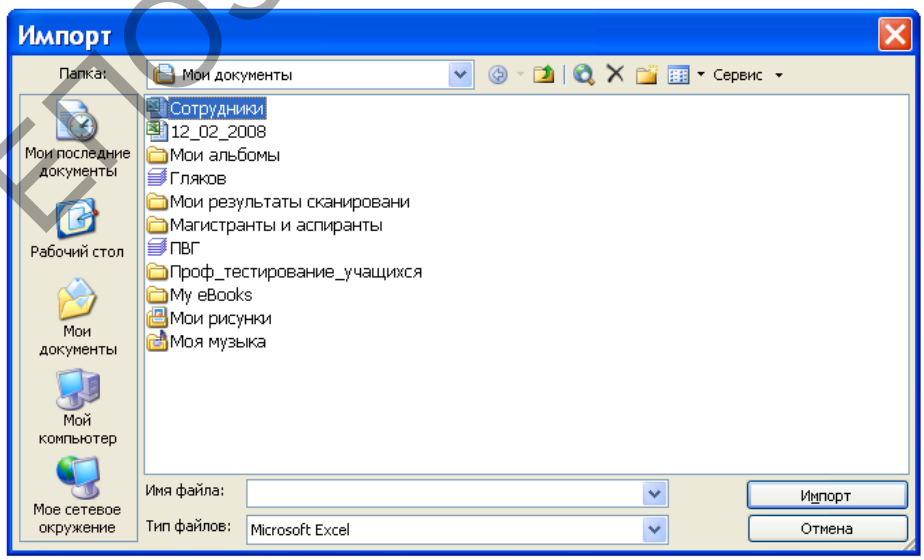


Рис. 3. Диалоговое окно **Импорт** после задания типа файла, адреса таблицы и выбора ее имени.

В раскрывающемся списке **Тип файлов** этого окна выберите тип электронной таблицы, которую вы хотите импортировать. Затем найдите исходную папку и выделите имя файла, содержащего импортируемую электронную таблицу. После этих действийдиалоговое окно будет иметь вид, представленный на рис. 3.

4. Нажмите кнопку **Импорт**. В появившемся диалоговом окне мастера импорта электронных таблиц (см. рис. 4) выберите первый рабочий лист (Лист1) и нажмите кнопку**Далее**.

5. MSAccess откроет следующее окно мастера импорта электронных таблиц. Установите флажок**Первая строка содержит заголовки столбцов**, поскольку в первой строке электронной таблицы **Сотрудники** находятся значения, которые следует взять в качестве имен полей таблицы MSAccess. После нажатия кнопки**Далее** в появившемся окне мастера укажите, что вы будете добавлять импортируемые данные в новую таблицу MSAccess. Нажмите кнопку**Далее**, чтобы перейти к следующему шагу.

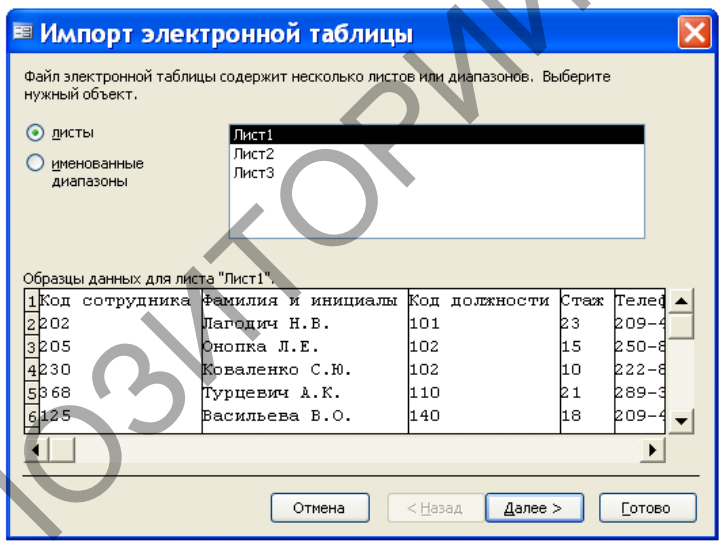


Рис. 4. Окно мастера импорта.

Поскольку для сохранения данных выбрана новая таблица, в следующем окне мастера, вы можете изменить определения ее полей, в том числе указать индексные поля. Раскрывающийся список **Индекс** содержит те же значения, что и свойство **Индекс** таблицы в режиме конструктора. В поле со списком **Тип данных** отображается тип данных текущего поля таблицы, выбранной мастером на основе анализа нескольких первых строк. Некоторые поля можно и не включать в новую таблицу MSAccess. Для таких полей устанавливается флажок**Не**

импортировать поле. Нажмите кнопкуДалее, чтобы перейти к следующему шагу.

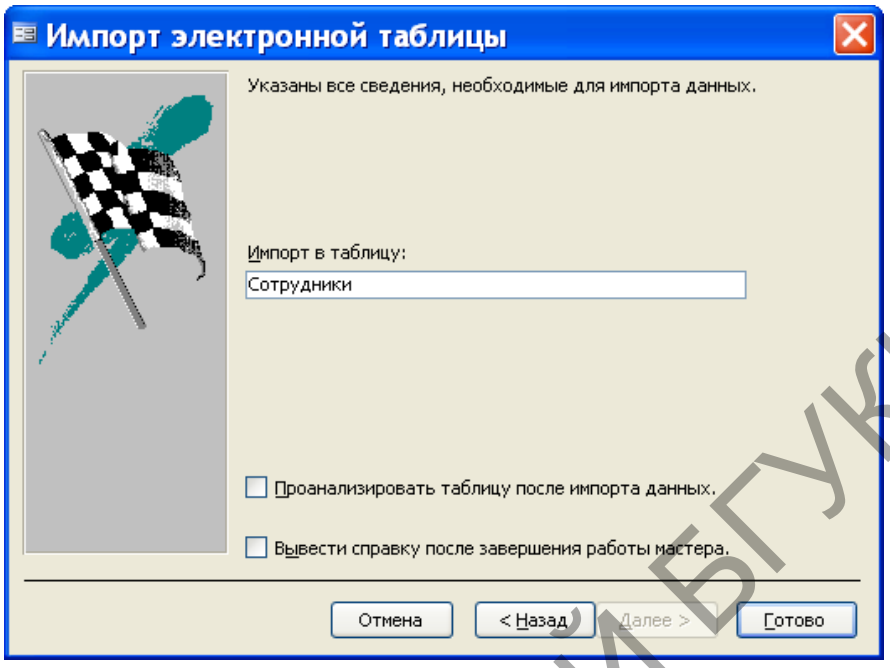


Рис. 5. Последнее окно мастера импорта.

6. Очередное окно мастера позволяет определить первичный ключ таблицы MSAccess. Лучше здесь ключ не создавать и не определять, а установить переключатель**Не создавать ключ** и нажать на кнопку **Далее**. В последнем окне мастера можно введите имя для новой таблицы **Сотрудники**, как это сделано на рис. 5.

7. Чтобы импортировать данные электронной таблицы, нажмите кнопку **Готово**. При успешном выполнении операции импорта MSAccess откроет окно сообщения, показанное на рис. 6.

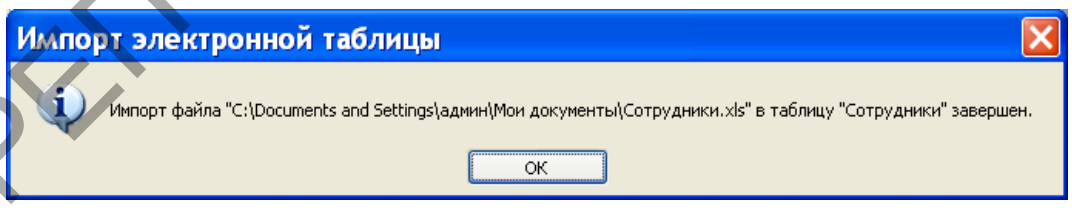


Рис. 6. Окно мастера импорта при успешном импорте таблицы.

После нажатия кнопки **ОК** мастер импорта закроет свои окна. Обратите внимание на то, что в окне базы данных **Библиотека** во вкладке **Таблицы** появилось имя импортируемой нами таблицы **Сотрудники**. Просмотрите эту таблицу в режиме таблицы и в режиме конструктора.

8. Откройте в базе данных **Библиотека** полученную в результате импорта таблицу **Сотрудники** в режиме конструктора. Обратите

внимание, что поле **Код сотрудника** имеет тип данных **Числовой**, а свойство **Размер данных** имеет значение **Двойное с плавающей точкой**. Сделайте это поле ключом, а свойству **Размер данных** дайте значение **Длинное целое**. Аналогичным образом измените значение свойства **Размер данных** у поля **Код должности**. Свойству **Размер данных** поля **Стаж** дайте значение **Целое**.

Поля **Фамилия и инициалы**, **Телефон** имеют тип данных **Текстовый**, а свойство **Размер поля** для них по умолчанию равно 255. Это свойство для поля **Фамилия и инициалы** сделайте равным 20, а для поля **Телефон** – 9.

Импорт текстовых файлов

В MSAccess можно импортировать не произвольный текстовый файл, а специальным образом организованный. MSAccess должен знать, как различить начало и конец значения поля в каждой входной текстовой строке. В качестве стандартных разделителей полей может использоваться три символа: запятая, символ табуляции и пробел. Другой способ подготовки текстового файла для импорта состоит в следующем. Все поля располагаются в фиксированных позициях внутри каждой записи. В этом случае каждое поле во всех записях должно начинаться в одном и том же месте.

После подготовки текстового файла одним из указанных способов вы можете импортировать его в базу данных MSAccess, выполнив следующие действия:

1. Откройте базу данных MSAccess, в которую вы хотите импортировать текстовые данные. Если она уже открыта, переключитесь в окно базы данных.

2. Выберите команду **Файл, Внешние данные, Импорт**. MSAccess откроет окно диалога Импорт.

3. В раскрывающемся списке **Тип файлов** выберите **Текстовые файлы**. Найдите исходную папку, выделите имя импортируемого текстового файла и нажмите кнопку **Импорт**. MSAccess запустит мастера импорта текста и откроет его первое окно.

4. В этом окне мастер в соответствии со своими предположениями о формате файла (с разделителями или с фиксированной длиной записей) выводит несколько строк данных. Просмотрев данные, вы можете согласиться с выбором мастера или предложить другой формат. Если мастер неправильно определил формат файла, то обычно это означает, что данные отформатированы некорректно. В этом случае надо выйти из

мастера и исправить исходный файл. Если мастер правильно выбрал формат, нажмите кнопку**Далее**, чтобы перейти к следующему шагу.

5. Для текстового файла с разделителями мастер импорта выводит окно, в котором надо указать какие символы используются в качестве разделителей полей и ограничителя текста.

Для текстового файла с фиксированной длиной записей мастер импорта выводит окно с графическим представлением разделителей полей. Для создания разделителя надо установить указатель в нужной позиции и нажать кнопку мыши. Чтобы убрать разделитель, надо выполнить двойной щелчок мышью на линии со стрелкой. Переместить разделитель можно, перетащив линию со стрелкой в другую позицию.

После завершения работы на этом шаге нажмите кнопку**Далее**.

6. В появившемся окне вам предлагается сделать выбор: сохранить импортируемые данные в новой таблице или добавить их в существующую. Если вы решите создать новую таблицу, мастер выведет окно, в котором можно изменить имена полей, выбрать типы данных и создать индексы. Нажмите кнопку**Далее**, чтобы перейти в следующее окно мастера импорта текста (оно аналогично соответствующему окну мастера импорта электронных таблиц), которое позволяет определить первичный ключ таблицы.

При добавлении данных в существующую таблицу порядок расположения импортируемых столбцов должен точно совпадать с расположением столбцов в таблице MSAccess или импортируемые данные должны содержаться в текстовом файле с разделителями, в котором имена столбцов в первой строке совпадают с именами полей в существующей таблице.

7. В последнем окне можно изменить предлагаемое мастером имя конечной таблицы. Чтобы импортировать данные, нажмите кнопку **Готово**. MSAccess выведет окно с сообщением о выполнении операции. Если мастер обнаружит ошибку, не позволяющую произвести импорт данных, он снова откроет свое последнее окно. С помощью кнопки**Назад** вернитесь в предыдущие окна мастера и исправьте некоторые установки.

Задание

1. В текстовом редакторе MSWord подготовьте файл **Должности сотрудников**, информация для которого показана на рис. 1.Все поля записей данного файла начинаются в фиксированной позиции.Сохраните его в папке **Мои документы**, причем при сохранении файла выберите тип файла **Обычный текст**.

101 Директор
103 Зам. директора по учебно-воспитательной работе
104 Зам. директора по административно-хозяйственной работе
106 Зам. директора по учебно-методической работе
109 Заведующий отделением
110 Преподаватель
140 Администратор сети
150 Библиотекарь

Рис. 1. Вид текстового файла **Должности сотрудников**.

Как только вы нажмете кнопку сохранить, появится диалоговое окно **Преобразование файла** (см. рис. 2). В этом окне сделайте активной вместо радиокнопки **Windows (по умолчанию)** радиокнопку **MS-DOS**и нажмите на кнопку OK. В результате этих действий **Должности сотрудников** сохранится в текстовом формате **TXT**.

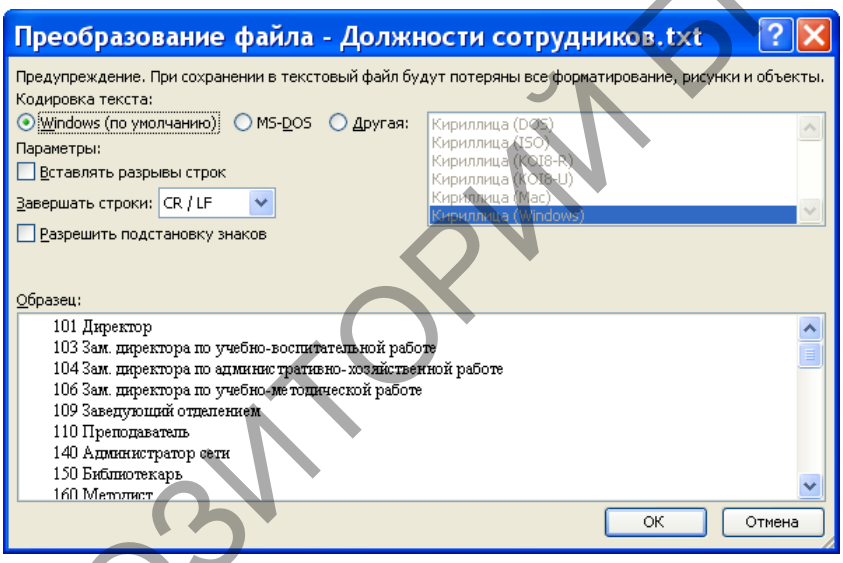


Рис. 2. Диалоговое окно **Преобразование файла**.

Требуется этот файл импортировать в MSAccess в качестве таблицы базы данных **Библиотека**. Импортированной таблице дать прежнее имя **Должности сотрудников**.

2. Откройте в MSAccess базу данных **Библиотека**. Выберите в пункте меню **Файл** команду **Внешние данные | Импорт**. MSAccess откроет окно диалога **Импорт**, которое имеет такой же вид, как и для случая импорта электронных таблиц.

В раскрывающемся списке **Тип файлов** выберите тип **Текстовые файлы**. Найдите исходную папку, выделите имя импортируемого текстового файла **Должности сотрудников** и нажмите кнопку **Импорт**. MSAccess запустит мастера импорта текста и откроет его первое окно. Это окно показано на рис. 3.

В этом окне мастер импорта сделал правильное предположение о формате файла (**фиксированная ширина полей – интервалы заполняются пробелами**) и вывел несколько строк данных. Нажмите кнопку**Далее**, чтобы перейти к следующему шагу.

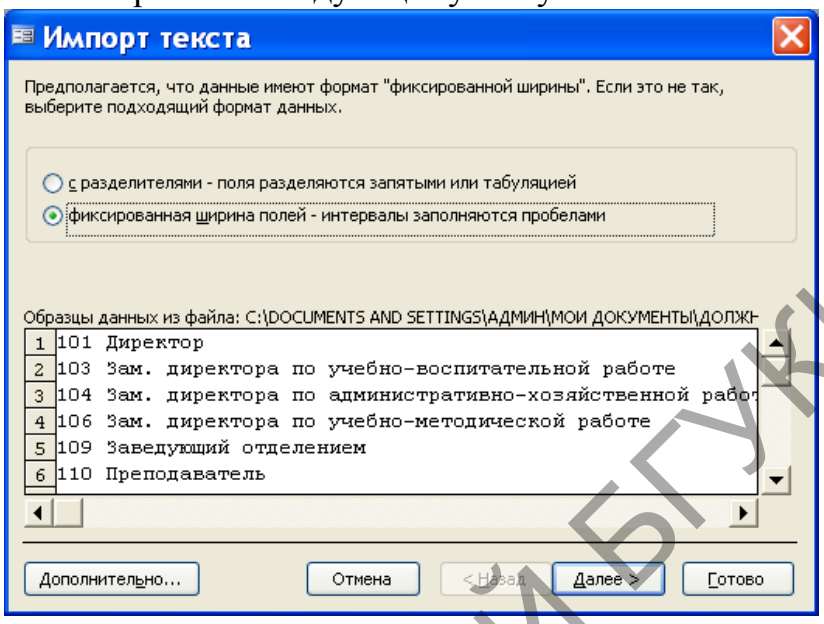


Рис. 3. Первое окно мастера импорта.

3. На этом шаге мастер импорта текста выводит окно с графическим представлением разделителей полей, показанное на рис. 4. Второй разделитель полей оказался лишним. Чтобы убрать лишний разделитель, выполните двойной щелчок мышью на линии со стрелкой. После завершения работы на этом шаге нажмите кнопку**Далее**.

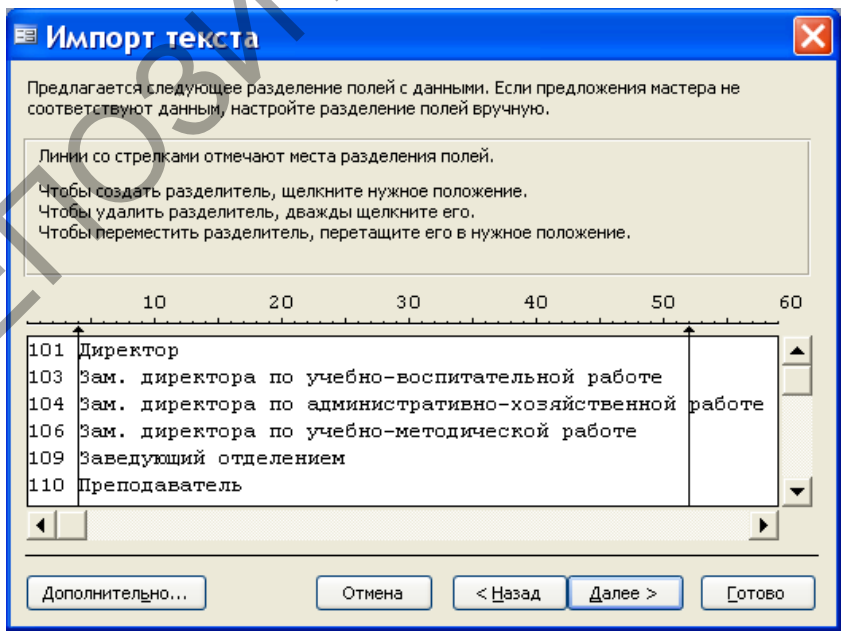


Рис. 4. Окно мастера импорта с лишним разделителем.

В появившемся окне укажите, что импортируемые данные надо сохранить в новой таблице, и нажмите кнопку**Далее**. При этом мастер выведет окно, в котором можно изменить имена полей, выбрать типы данных и создать индексы. В этом окне выполните следующие действия: вместо предложенного мастером импорта имени **Поле1** введите имя **Код должности**, вместо типа данных **Длинное целое** выберите в списке тип данных **Целое**, для свойства индекс выберите в списке значение**Да (Совпадения не допускаются)**. В результате этих операций окно мастера импорта будет иметь вид, показанный на рис. 5.

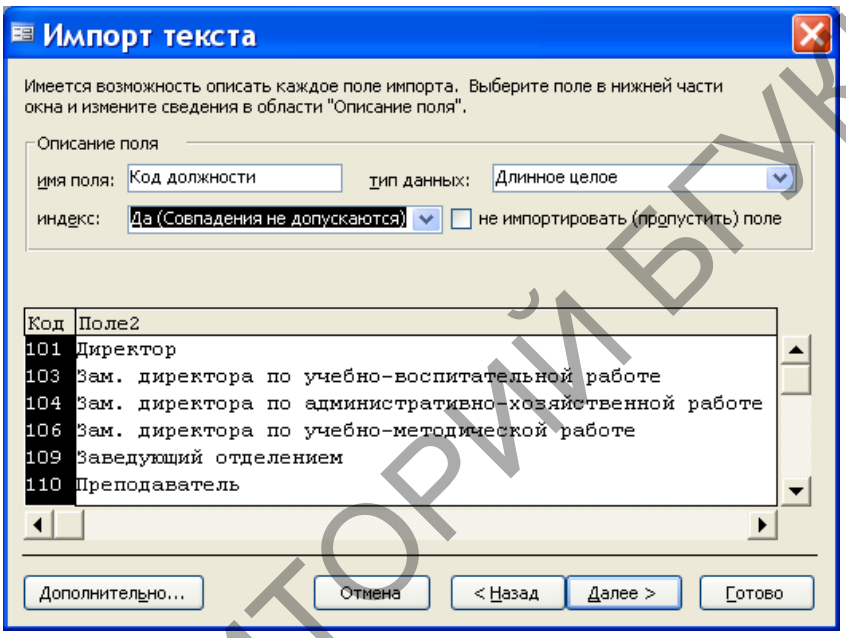


Рис. 5. Окно импорта для описания поля.

Перейдите к изменению описания второго поля, выполнив щелчок мышью на его названии **Поле2**. Для этого поля внесите только одно изменение – дайте ему имя **Должность**. Нажмите кнопку**Далее**, чтобы перейти в следующее окно мастера импорта текста (оно позволяет определить первичный ключ таблицы). В этом окне сделайте активным переключатель **определить ключ** и из списка для него выберите имя **Код должности**. Нажмите после этого кнопку**Далее.**

4. В последнем окне мы согласимся с предлагаемым мастером именем конечной таблицы **Должности сотрудников**. Чтобы импортировать данные, нажмите кнопку **Готово**. MSAccess выведет окно с сообщением о выполнении операции.

Сейчас в окне базы данных Кадры вы обнаружите имя таблицы, которую мы импортировали из текстового файла. Просмотрите эту

таблицу в режимах таблицы и конструктора и убедитесь в том, что она будет иметь вид, показанный на рис. 6 и 7.

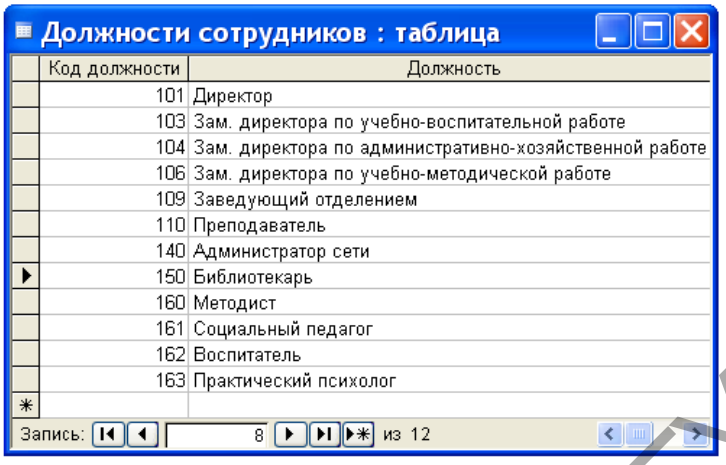


Рис. 6. Таблица **Должности сотрудников** в режиме таблицы.

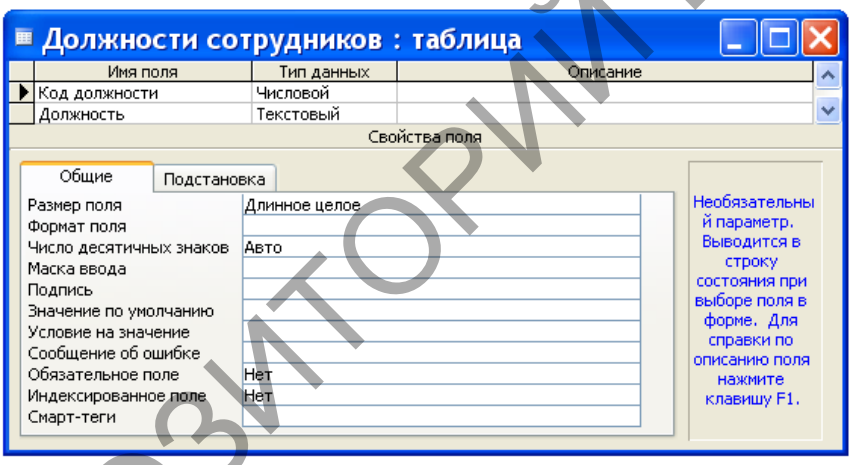


Рис. 7. Таблица **Должности сотрудников** в режиме конструктора.

Импорт объектов MSAccess

Можно создавать таблицы MSAccess, копируя (импортируя) данные из файлов различных форматов, электронных таблиц или текстовых файлов. При преобразовании импортируемых данных и присвоении имени полям создаваемой таблицы MSAccess использует информацию, хранящуюся в исходной базе данных. Импорт данных возможен не только из других баз данных MSAccess, но также из dBASE, Paradox, FoxPro и любых баз данных SQL, поддерживающих специальный стандарт ODBC (OpenDatabaseConnectivity – Открытый доступ к данным).

Вы можете импортировать любой из семи основных типов объектов (таблицы, запросы, формы, отчеты, страницы, макросы и

модули) из одной базы данных MSAccess в другую. Тот же результат можно получить, если открыть исходную базу данных, выбрать нужный объект, выполнить команду**Копировать** в меню **Правка**, открыть конечную базу данных (в которую импортируется объект) и выполнить команду **Вставить** в меню **Правка**. Но с помощью команды **Внешние данные** | **Импорт** в меню **Файл** вы можете копировать несколько объектов, не переходя каждый раз из одной базы данных в другую.

Чтобы импортировать объект из другой базы данных MSAccess, можно выполнить следующие действия:

1. Откройте базу данных MSAccess, в которую вы хотите импортировать объект. Если она уже открыта, переключитесь в окно базы данных.

2. Выберите команду **Внешние данные**|**Импорт** в меню **Файл**. MSAccess откроет окно диалога **Импорт**.

3. В раскрывающемся списке **Тип файлов** этого окна выберите тип файла **MicrosoftAccess**, затем найдите исходную папку и выделите имя файла базы данных, содержащей нужный объект.

4. После нажатия кнопки **Импорт**MSAccess откроет окно диалога **Импорт объектов**. В этом окне на соответствующей вкладке выделите имя импортируемого объекта. Чтобы импортировать все объекты определенного типа, можно нажать кнопку**Выделить все**. Вы можете импортировать несколько типов объектов сразу, переходя поочередно на нужную вкладку и выделяя объекты, которые вы хотите импортировать.

После нажатия кнопки **Параметры** становятся доступными дополнительные параметры. При копировании таблиц можно активизировать параметр **Схема данных**, чтобы импортировать также связи, определенные для выбранных таблиц в исходной базе данных. Если объектом является таблица, вы можете импортировать только структуру или данные вместе со структурой. Параметр **Меню и панели** позволяет импортировать из исходной базы данных все специальные меню и панели инструментов. При активном параметре **Спецификации** импортируются все спецификации импорта/экспорта, определенные в исходной базе данных. Кроме того, вы имеете возможность импортировать наборы записей, а не определения запросов. Чтобы скопировать выбранные вами объекты в текущую базу данных, нажмите кнопку **ОК**.

5. После успешного выполнения операции импорта скопированные объекты сохраняют свои прежние имена. Если MSAccess обнаружит, что в текущей базе некоторое имя уже используется, он

сгенерирует новое имя, добавив в конец исходного имени уникальное целое число. В связи с тем, что объекты в базе данных MSAccess могут ссылаться на другие объекты по имени, вы должны тщательно проверить ранее установленные ссылки на переименованные объекты.

Задание

1. Создайте пустую базу данных с именем **Кадры** в папке **Мои документы**. В эту базу данных будем импортировать две таблицы **Сотрудники** и **Должности сотрудников** из базы данных**Библиотека**.

2. Выберите в меню **Файл** команду **Внешние данные**| **Импорт**. MSAccess откроет диалоговое окно **Импорт**. В раскрывающемся списке **Тип файлов** этого окна выберите тип файла **MicrosoftOfficeAccess**, затем найдите папку, в которой находится файл базы данных **Библиотека**, и выделите имя **Библиотека** (см. рис. 1).

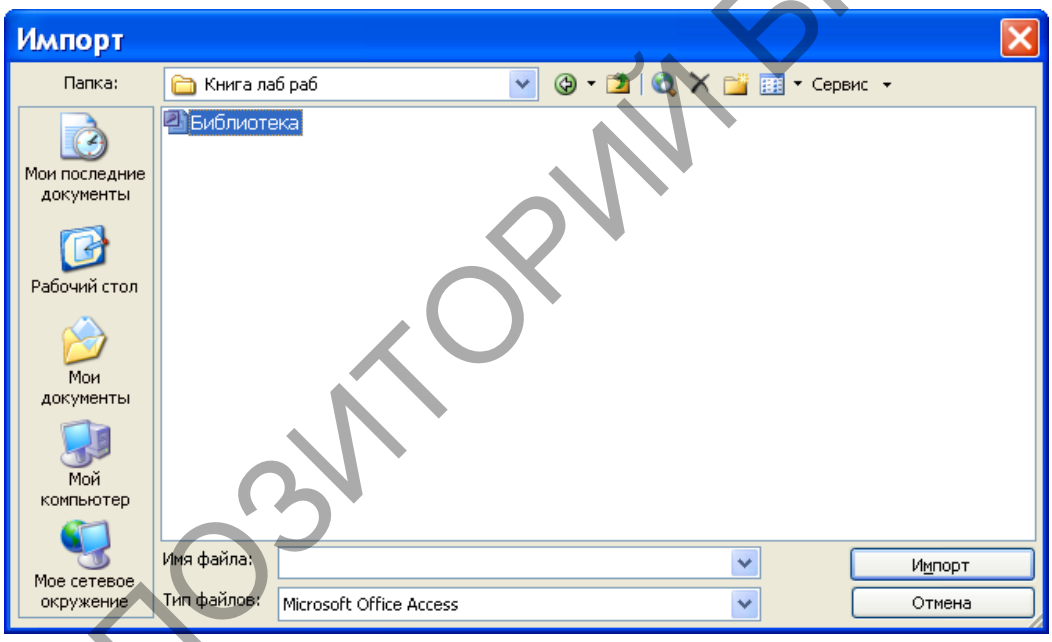


Рис. 1. Окно импорта базы данных.

3. После нажатия кнопки **Импорт**MSAccess откроет диалоговое окно **Импорт объектов**. В этом окне на соответствующей вкладке выделите имена импортируемых таблиц **Сотрудники** и **Должности сотрудников**. Результат выполнения этих действий показан на рис.2.

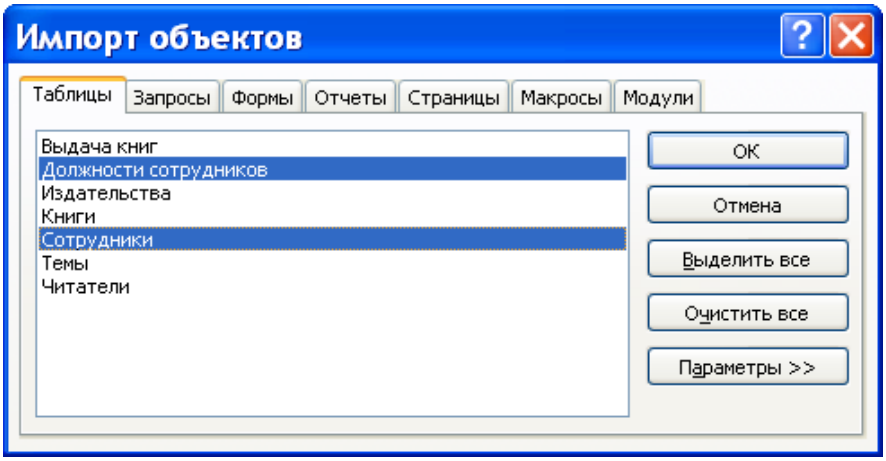


Рис. 2. Диалоговое окно импорта объектов.

4. Чтобы скопировать выбранные вами объекты в текущую базу данных, нажмите кнопку **ОК**. При успешном выполнении операции импорта скопированные объекты сохраняют свои прежние имена.

На рис. 3 вы можете обнаружить таблицы, которые мы импортировали из базы данных **Библиотека**, они имеют имена **Сотрудники** и **Должности сотрудников**.

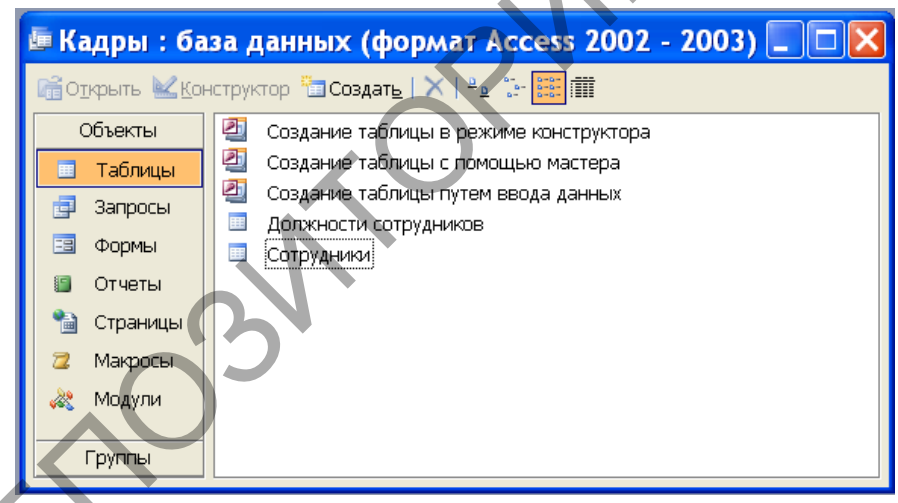


Рис. 3. Результат импорта таблиц.

5. Осуществите импорт всех форм и запросов, созданных в предыдущих лабораторных работах в базе данных **Библиотека**, в базу данных **Кадры**. Укажите, как это можно сделать оптимально.

Обратите внимание на то, что если при импорте MSAccess обнаружит, что в текущей базе такие имена импортируемых объектов уже используется, то он сгенерирует новые имена, добавив в конец исходных имен уникальные целые числа.

Лабораторная работа 13

Связывание файлов и таблиц

Цель работы: Сформировать умения для связывания файлов и таблиц в базе данных MSAccess.

Можно установить связь с таблицами из других баз данных MSAccess или с базами данных в другом формате (например, FoxPro, dBASE или любой базы данных SQL, поддерживающей ODBC). В последнем случае связывание можно использовать как альтернативу импорту. В большинстве случаев можно читать данные, вставлять новые записи, удалять записи или изменять данные так, будто связанный файл является таблицей Access. Можно также установить связь с текстовым файлом или электронной таблицей, что позволит обрабатывать связанные данные с помощью запросов, форм и отчетов в базе данных MSAccess.

Для связывания таблицы из другой базы данных MSAccess с текущей базой данных, выполните следующие действия:

1. Откройте базу данных или переключитесь в окно открытой базы данных.

2. Выберите команду **Внешние данные | Связь с таблицами** в меню **Файл**. MSAccess откроет окно диалога **Связь**, которое очень похоже на окно диалога **Импорт**.

3. В раскрывающемся списке **Тип файлов** выберите тип **MicrosoftAccess**. Затем установите исходную папку, выберите файл с расширением .mdb, содержащий таблицу, с которой надо установить связь. При работе в сети выберите логический диск, назначенный сетевому серверу, содержащему нужную вам базу. Если надо, чтобы при открытии таблицы MSAccess автоматически подключал вас к серверу, то вместо выбора логического диска введите в поле **Имя файла** полный путь к файлу в сети. После выбора файла базы данных нажмите кнопку **Связь**.

4. MSAccess откроет окно диалога **Связь с таблицами** со списком таблиц в выбранной базе данных. Выберите одну или несколько таблиц и нажмите кнопку **ОК**, чтобы связать таблицы с текущей базой данных. Если операция пройдет успешно, то в текущей базе данных появятся новые таблицы с именами выбранных таблиц.

В окне базы данных MSAccess отмечает связанную таблицу значком со стрелкой. Обнаружив, что имя уже используется в текущей базе данных, MSAccess генерирует новое, добавив в конец имени уникально целое число.

Связывание текстовых файлов и электронных таблиц почти идентично импорту файлов этих типов. В связанном текстовом файле данные можно только читать, в то время как в связанной электронной

таблице разрешается обновлять и добавлять новые строки, нельзя только удалять их.

Для связывания текстового файла или файла электронной таблицы с текущей базой данных выполните следующие действия:

1. Откройте базу данных MSAccess или переключитесь в окно открытой базы данных.

2. Выберите команду **Внешние данные**| **Связьс таблицами** в меню **Файл**. Access откроет окно диалога **Связь**, позволяющее выбрать нужный тип файла.

3. В раскрывающемся списке **Тип файлов** выберите тип **MicrosoftExcel** или **Текстовые файлы**. Найдите исходную папку и выберите имя файла. При работе в сети выберите логический диск, назначенный сетевому серверу, содержащему нужную вам базу. Если вы желаете, чтобы при открытии связанного файла MSAccess автоматически подключал вас к серверу, то вместо выбора логического диска введите в поле **Имя файла** полный путь к файлу в сети.

4. Поле нажатия кнопки **Связь**MSAccess запустит соответствующего мастера, помогающего установить связь с текстовым файлом или электронной таблицей.

5. Окна этих мастеров аналогичны соответствующим окнам мастера импорта текста или электронной таблицы, работа с которыми была описана ранее.

Связанные таблицы можно настроить на работу в среде MicrosoftAccess, внеся в их определения некоторые изменения. При открытии связанной таблицы в режиме конструктора MSAccess выведет на экран предупреждение о том, что некоторые свойства связанной таблицы изменять нельзя. Нажмите кнопку**Да**, чтобы открыть связанную таблицу в режиме конструктора.

Для полей связанной таблицы наряду со свойствами подстановки разрешено также изменить **Формат поля**, **Число десятичных знаков**, **Подпись**, **Маску ввода** и **Описание**. Настраивая их, можно облегчить просмотр и обновление данных в формах и отчетах MSAccess. Кроме того, связанной таблице в базе данных можно присвоить новое имя (конечно, в исходной базе данных оно останется неизменным). Новое имя может дать вам возможность использовать таблицу в существующих запросах, формах и отчетах.

Изменение свойств связанной таблицы не влияет на ее структуру в исходной базе данных. Следует учесть, что при изменении определения таблицы в исходной базе данных необходимо снова установить связь с ней.Придется также удалять связи и вновь связывать таблицы, если

изменятся ваше регистрационное имя или пароль доступа к исходной базе данных.

Для удаления связанной таблицы из базы данных выделяют в окне базы данных имя связанной таблицы и нажимают клавишу **Del** или выбирают команду**Удалить** в меню **Правка**. Затем нажмите кнопку**Да** в окне диалога, которое откроет MSAccess для подтверждения операции удаления связи. При удалении связанной таблицы из базы данных исчезают лишь ее имя в окне базы данных и связь, но сама исходная таблица, конечно, остается целой и невредимой.

Если некоторые связанные таблицы перемещаются в другое место, то обновить информацию об их расположении поможет диспетчер связанных таблиц. Для открытия окна Диспетчера связанных таблиц в начале открывают базу данных, содержащую связанные таблицы, информацию о которых необходимо обновить, а затем выбирают команду **Диспетчер связанных таблиц** в подменю **Служебные программы** меню **Сервис**. MSAccess выведет на экран диалоговое окно со списком всех связанных таблиц в текущей базе данных. Отметьте таблицы, местонахождение которых вы считаете необходимым проверить и обновить, а затем нажмите кнопку **ОК**. Если какая-либо из них перемещена, диспетчер связанных таблиц выведет диалоговое окно, с помощью которого вы определите новое местонахождение исходного файла. Для проверки расположения всех связанных таблиц можно установить флажок**Всегда выдавать запрос нового местонахождения**.

Задание

1. Удалите из базы данных **Кадры** все таблицы. В базе данных **Кадры** будем устанавливать связь с таблицами **Читатели**, **Сотрудники** и **Должности сотрудников**, расположенными в базе данных **Библиотека**.

2. Откройте базу данных **Кадры** и выберите команду **Внешние данные**| **Связьс таблицами** в меню **Файл**. Появится диалоговое окно **Связь** (см. рис. 1).

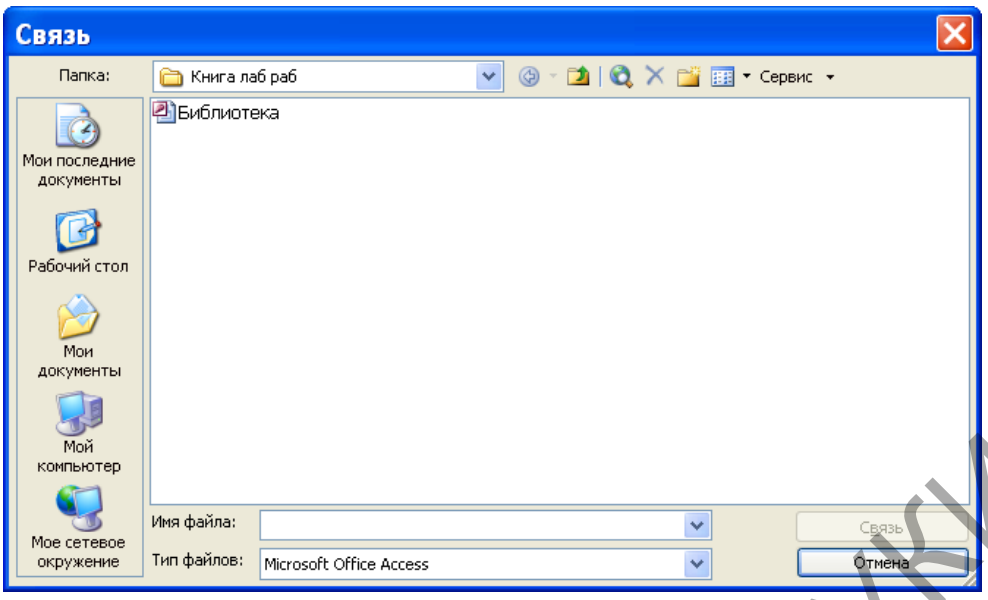


Рис. 1. Диалоговое окно **Связь**.

3. В раскрывающемся списке **Тип файлов** этого окна выберите тип **MicrosoftAccess** и откройте папку, содержащую файл базы данных **Библиотека**. Выделите его имя и нажмите кнопку **Связь**.

4. В появившемся диалоговом окне **Связь с таблицами**, показанном на рис. 2, выберите три таблицы: **Читатели**, **Сотрудники** и **Должности сотрудников**.

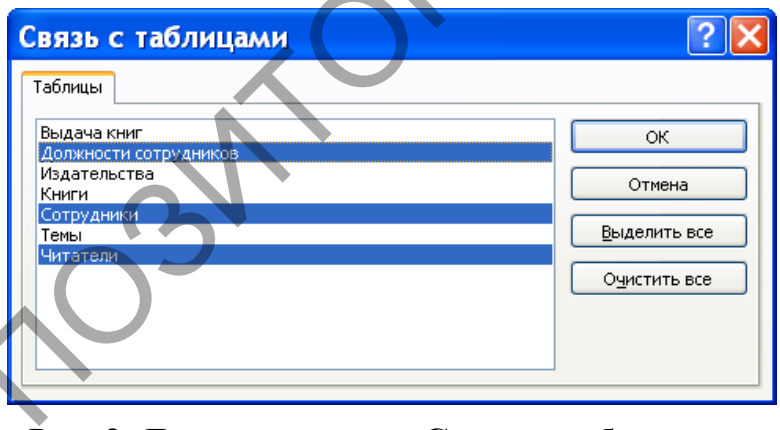


Рис. 2. Диалоговое окно **Связь с таблицами**.

Для связи выбранных таблиц с текущей базой данных нажмите кнопку **ОК**. При этом в окне базы данных **Кадры** появятся таблицы с именами: **Читатели**, **Сотрудники** и **Должности сотрудников** (см. рис. 3). Связанные таблицы в нем будут помечены стрелками.

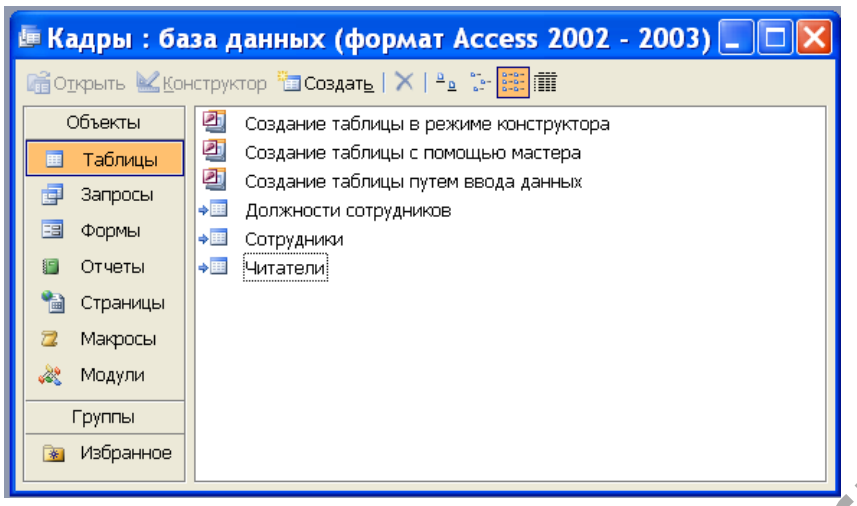


Рис. 3. Окно базы данных **Кадры** с тремя связанными таблицами.

Удалите в базе данных **Кадры** связанную таблицу **Сотрудники** и убедитесь в том, что в базе данных **Кадры** удалилась только ссылка (именно наличие стрелки на имени таблицы **Сотрудники** означает, что это ссылка на таблицу в другой базе данных, а не сама таблица), сама же таблица **Сотрудники** в базе данных **Библиотека** осталась целой и невредимой.

Переместите таблицу **Читатели** из базы данных **Библиотека** в пустую базу данных **Резерв** (предварительно ее создайте). Наиболее быстрый способ сделать это состоит в следующем. Откройте два окна программы MicrosoftAccess в нормальном представлении. В одном из них откройте базу данных **Библиотека**, а в другом пустую базу данных **Резерв** (открыть две базы данных в одном окне MicrosoftAccess не позволяет). После этого при нажатой клавише **Shift** перетащите мышью имя таблицы **Читатели** из окна базы данных **Библиотека** в окно базы данных **Резерв**. Поскольку при этом будет происходить удаление связанной таблицы **Читатели** в базе данных **Библиотека**, то MSAccess выдаст предупреждающее сообщение, показанное на рис. 4.

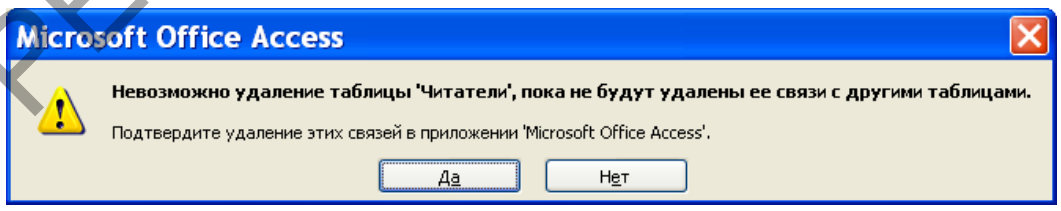


Рис. 4. Предупреждающее сообщение об удалении связей.

Убедитесь в том, что после нажатия на кнопкуДа в этом сообщении имя таблицы **Читатели** исчезнет из окна базы данных **Библиотека** и появится в окне базы данных **Резерв**.

Попробуйте теперь открыть связанную таблицу **Читатели** в базе данных **Кадры**. Вам это, разумеется, не удастся – ведь мы переместили связанную таблицу **Должности** в другую базу данных. О невозможности выполнения этой операции будет свидетельствовать предупреждающее окно (см. рис. 5).

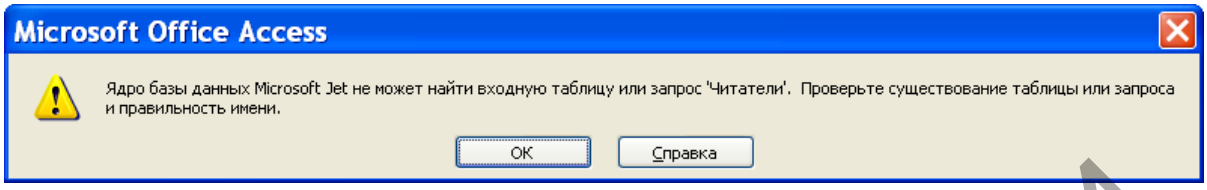


Рис. 5. Предупреждающее окно о безуспешном поиске таблицы.

Справиться с этой задачей можно, воспользовавшись диспетчером связанных таблиц следующим образом:

1. Откройте окно базы данных **Кадры** и выберите команду **Диспетчер связанных таблиц** в подменю **Служебные программы** меню **Сервис**.

2. В диалоговом окне **Диспетчер связанных таблиц**, выведенным на экран MSAccess (см. рис. 6), отметьте таблицу **Должности** (поставьте возле нее флажок), местонахождение которой надо обновить, а затем нажмите кнопку **ОК**.

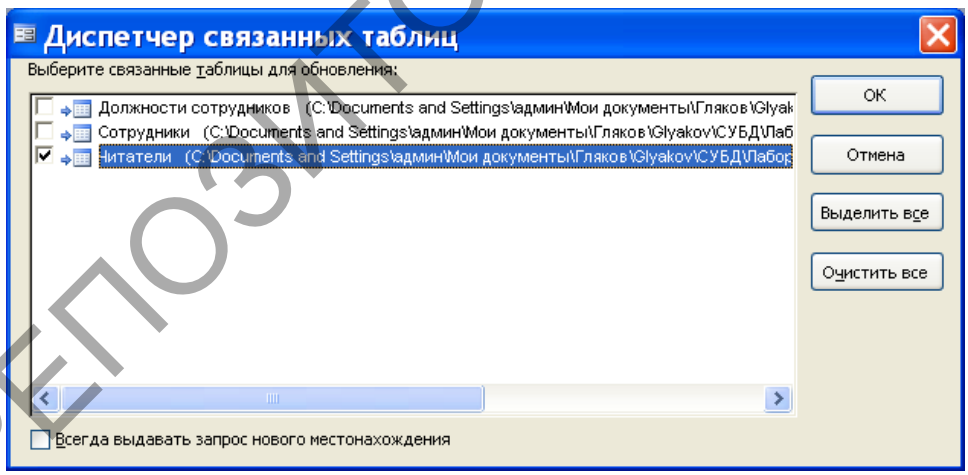


Рис. 6. Диалоговое окно **Диспетчер связанных таблиц**.

3. В результате выполнения предыдущих действий появится диалоговое окно, показанное на рис. 7.

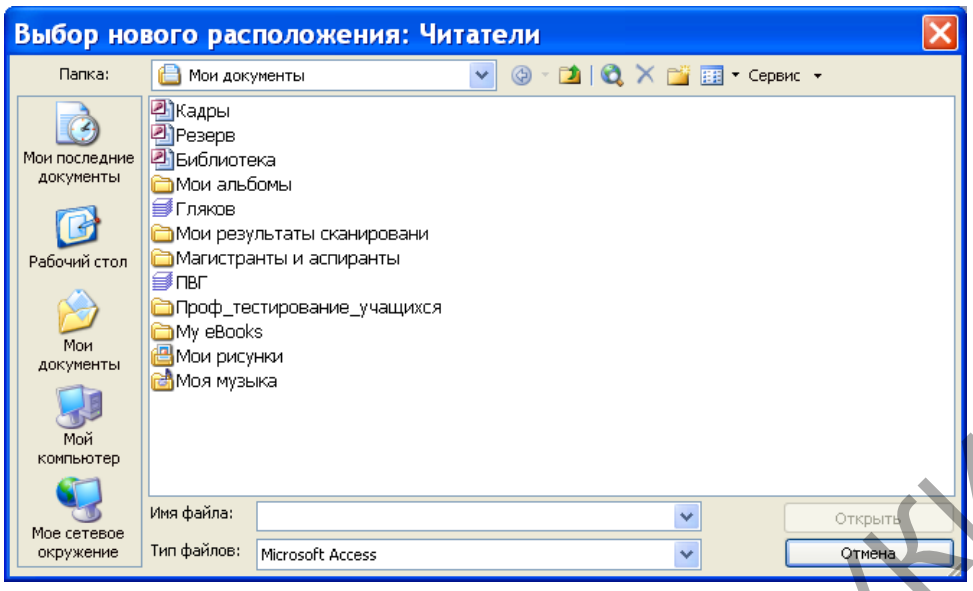


Рис. 7. Диалоговое окно **Выбор нового расположения**.

В этом окне для таблицы **Читатели** выделите имя базы данных **Резерв** (напомним, что в данный момент времени именно в этой базе данных находится таблица **Читатели**) и нажмите кнопку**Открыть**.

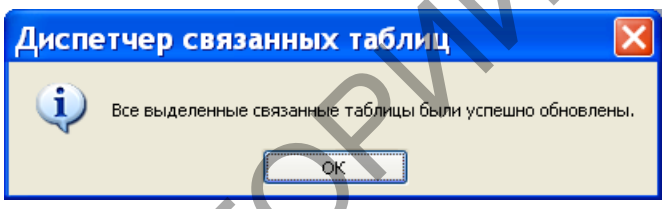


Рис. 8. Информационное сообщение MSAccess.

4. В окне информационного сообщения MSAccess (см. рис.8) об успешном обновлении связанной таблицы **Читатели** нажмите на кнопку **ОК**, а затем нажмите на кнопку**Закрыть** в окне диспетчера связанных таблиц.

После выполнения данных действий убедитесь, что в базе данных **Кадры** связанная таблица **Читатели** может быть открыта.

5. Скопируйте таблицу **Читатели** из базы данных **Резерв** в базу данных **Библиотека**.

Лабораторная работа 14

Экспорт данных

Цель работы: Сформировать умения для организации экспорта объектов из одной базы данных в другую.

Можно экспортировать (копировать) любой объект из одной базы данных MSAccess в другую. Можно также экспортировать данные из таблиц MSAccess в базы данных других СУБД, таблицы SQL, электронные таблицы, текстовые файлы и в документы MicrosoftWord. Любую таблицу, форму или отчет можно экспортировать в документ HTML (язык разметки гипертекста).

Экспорт объектов из одной базы данных MSAccess в другую во многом аналогичен операции импорта объектов.

Чтобы экспортировать объект из одной базы данных Access в другую, выполните следующие действия:

1. Откройте базу данных MSAccess, из которой вы хотите экспортировать объект. Если она уже открыта, переключитесь в окно базы данных.

2. В окне базы данных выделите имя копируемого объекта и выберите команду **Экспорт** в меню **Файл**. MSAccess откроет окно диалога **Экспорт объекта**.

3. В этом окне выберите папку и имя базы данных, в которую вы хотите экспортировать объект, и нажмите кнопку**Сохранить**.

4. MSAccess откроет диалоговое окно **Экспорт**, которое позволяет указать имя, присваиваемое объекту в другой базе данных. Можно оставить предлагаемое имя или изменить его. При экспорте таблицы можно копировать только структуру таблицы или структуру вместе с данными. Чтобы экспортировать объект, нажмите кнопку **ОК**.

5. Если выбранное для экспортируемого объекта имя уже используется в конечной базе данных, MSAccess предупредит об этом и спросит, хотите ли вы заместить существующий объект новым. НажмитеДа, чтобы продолжить выполнение операции, или **Нет**, чтобы отменить экспорт. Если экспорт объекта завершился успешно, вы обнаружите новый объект в конечной базе данных. Так как внутри базы данных MSAccess объекты могут ссылаться друг на друга по имени, тщательно проверьте ссылки в конечной базе данных.

Задание

1. Выполните экспорт таблицы **Читатели** из базы данных **Резерв** в базу данных **Штат**. Для этого сделайте следующее:

– Откройте базу данных **Резерв**. В окне базы данных **Резерв** выделите имя таблицы **Читатели**.

– Выполните команду **Экспорт** в меню **Файл**. В появившемся окне диалога **Экспорт объекта** выберите папку и имя базы данных, в которую вы хотите экспортировать объект (это показано рис. 1), и нажмите кнопку **Экспорт**.

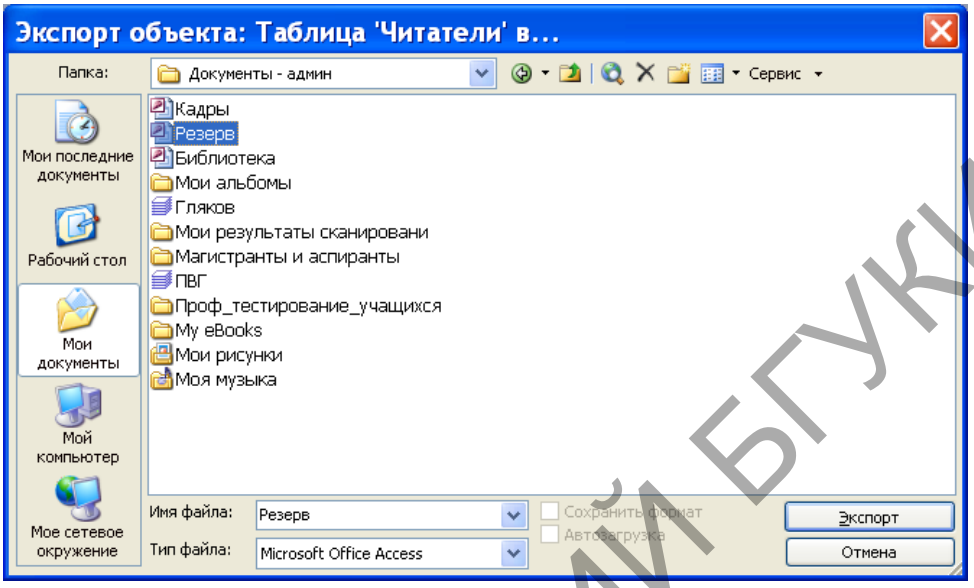


Рис. 1. Диалоговое окно **Экспорт объекта**.

– В появившемся диалоговом окне **Экспорт** (см. рис. 2) MSAccess предлагает сохранить экспортируемую таблицу под старым именем – это то, что нам и надо, а поскольку для экспорта таблицы включен переключатель **Структура и данные**, то нам осталось только нажать кнопку **ОК**. Нажмите ее.

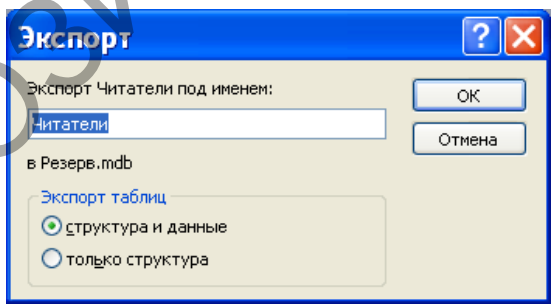


Рис. 2. Диалоговое окно **Экспорт**.

Если бы в базе данных **Библиотека** была таблица **Читатели**, то MSAccess выдал бы сообщение, приведенное на рис. 3.

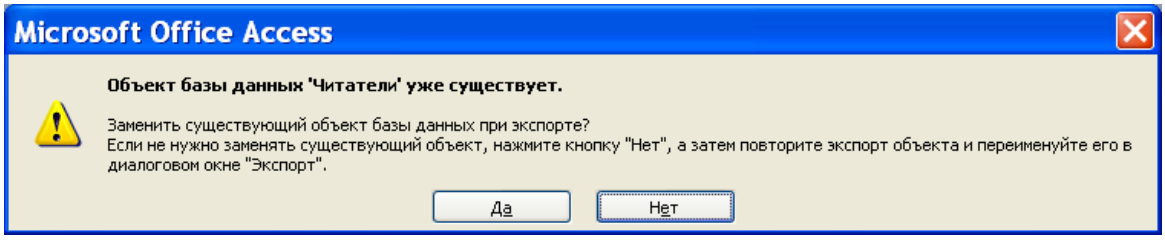


Рис. 3. Сообщение о существовании объекта при экспорте.

В нашем случае это сообщение не появится, поскольку в предыдущей лабораторной работе мы переместили таблицу **Читатели** из базы данных **Библиотека** в базу данных **Резерв** (вследствие этого таблица **Читатели** будет отсутствовать в базе данных **Библиотека**).

2. После выполнения этих действий откройте базу данных **Библиотека** и убедитесь в том, что в ней появилась экспортируемая нами таблица **Читатели**.

3. Удалите из базы данных **Кадры** связанные таблицы **Читатели**, **Сотрудники** и **Должности сотрудников** и сделайте в нее экспорт этих таблиц из базы данных **Библиотека**.Выясните, что сделать быстрее: импорт или экспорт нескольких таблиц.

4. Выполните экспорт всех запросов из базы данных **Библиотека** в базу данных **Кадры**.

Экспорт в электронную таблицу

Для экспорта таблицы, набора записей запроса на выборку или перекрестного запроса в электронную таблицу используйте следующую процедуру:

1. Откройте базу данных MSAccess, из которой вы хотите экспортировать объект. Если она уже открыта, переключитесь в окно базы данных.

2. В окне базы данных выделите имя копируемого объекта и выберите команду **Экспорт** в меню **Файл**. MSAccess откроет диалоговое окно **Экспорт объекта**. В этом окне выберите тип файла, папку и укажите имя файла, в который вы хотите экспортировать выбранный объект, и нажмите кнопку**Сохранить**.

3. Если процедура экспорта завершится успешно, MSAccess создаст новый файл, с которым вы сможете работать с помощью программы электронной таблицы или соответствующей СУБД.

MSAccess предоставляет дополнительную возможность быстрого экспорта данных любых таблицы, набора записей на выборку или перекрестного запроса в электронную таблицу MicrosoftExcel. Выделите в окне базы данных имя таблицы или запроса, данные из которых вы хотите экспортировать, и выполните команду **Связи с Office**| **Анализ в MSExcel** в меню **Сервис** или выберите команду **Анализ в MSExcel** в раскрывающемся списке кнопки **Связи с Office** на панели инструментов. Access скопирует данные в файл электронной таблицы и откроет его в Excel. Если файл с таким именем уже существует, MSAccess спросит, хотите ли вы заменить существующий файл. Если вы нажмете кнопку**Нет**, MSAccess предложит ввести другое имя файла.

Задание

Выполните экспорт таблицы **Сотрудники** из базы данных **Кадры**, используя общий алгоритм экспорта в электронную таблицу. Это можно сделать так:

1. Откройте базу данных **Кадры**. В окне базы данных выделите имя таблицы **Сотрудники**, а затем выберите команду **Экспорт** в меню **Файл**. MSAccess откроет диалоговое окно **Экспорт объекта (**см. рис. 1).

2. В этом окне выберите тип файла **MicrosoftExcel 97-2003** и папку для экспорта таблицы. MSAccess предлагает назвать файл электронной таблицы, в который будет выполняться экспорт таблицы, назвать именем исходной таблицы **Сотрудники**, - с этим мы согласимся. После этого нажмите кнопку **Экспорт**.

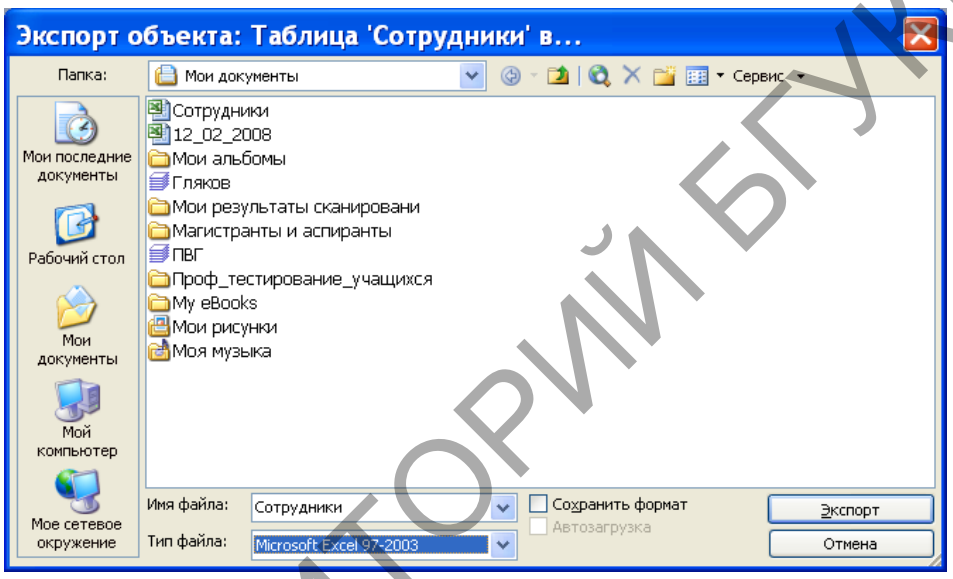
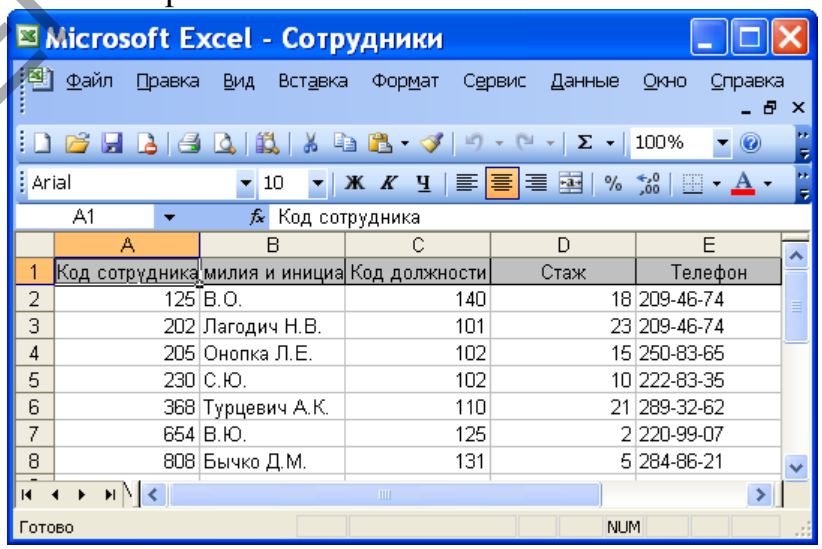


Рис. 1. Диалоговое окно **Экспорт объекта.**

После выполнения этих действий в папке, имя которой вы выбрали, будет находиться файл электронной таблицы с именем **Сотрудники**. Откройте этот файл и просмотрите его. Он должен иметь вид, показанный на рис. 2.



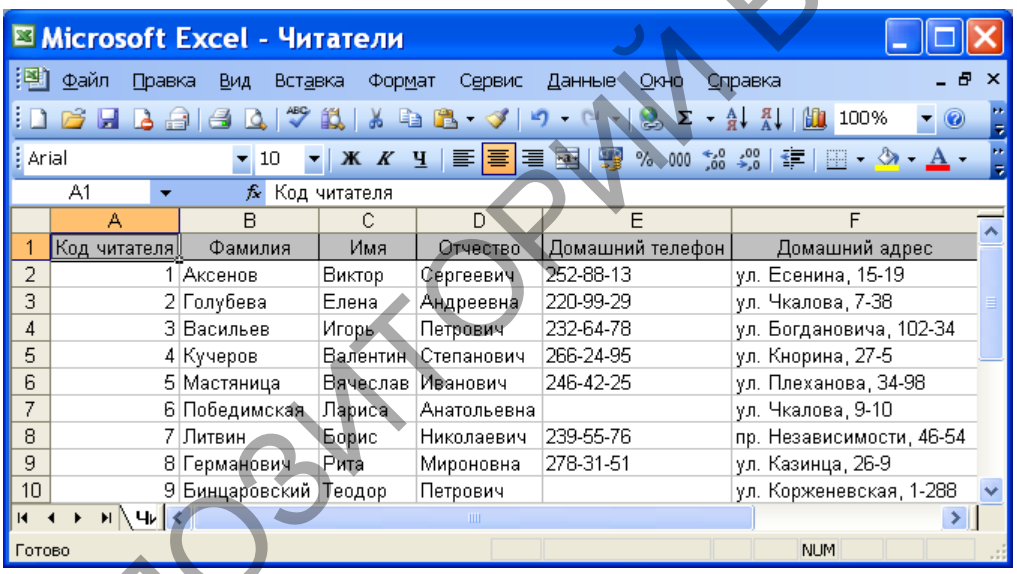
	A	B	C	D	E
1	Код сотрудника	милия и инициа	Код должности	Стаж	Телефон
2	125	В.О.	140	18	209-46-74
3	202	Лагодич Н.В.	101	23	209-46-74
4	205	Онопка Л.Е.	102	15	250-83-65
5	230	С.Ю.	102	10	222-83-35
6	368	Турцевич А.К.	110	21	289-32-62
7	654	В.Ю.	125	2	220-99-07
8	808	Бычко Д.М.	131	5	284-86-21

Рис. 2. Импортированная в MSExcel таблица **Сотрудники**.

Рассмотрим процедуру быстрого экспорта в MicrosoftExcel на следующем примере. Пусть нам требуется экспортировать таблицу **Читатели** из базы данных **Библиотека** в электронную таблицу. Быстрое решение этой задачи можно получить следующим образом.

В окне базы данных базы данных **Библиотека** выделите имя **Читатели** и выберите команду **Анализ в MSExcel** в раскрывающемся списке кнопки **Связи с Office** на панели инструментов.

MSAccess сразу выполнит экспорт таблицы **Читатели** из базы данных **Библиотека** и разместит файл электронной таблицы MicrosoftExcel с импортируемой таблицей в папке **Мои документы**. Более того, сама импортированная электронная таблица будет открыта. Это показано на рис. 3.Просмотрите эту таблицу, сохраните ее в той папке, в которой вы пожелаете, а затем закройте ее.



	A	B	C	D	E	F
1	Код читателя	Фамилия	Имя	Отчество	Домашний телефон	Домашний адрес
2	1	Аксенов	Виктор	Сергеевич	252-88-13	ул. Есенина, 15-19
3	2	Голубева	Елена	Андреевна	220-99-29	ул. Чкалова, 7-38
4	3	Васильев	Игорь	Петрович	232-64-78	ул. Богдановича, 102-34
5	4	Кучеров	Валентин	Степанович	266-24-95	ул. Кнорина, 27-5
6	5	Мастяница	Вячеслав	Иванович	246-42-25	ул. Плеханова, 34-98
7	6	Победимская	Лариса	Анатольевна		ул. Чкалова, 9-10
8	7	Литвин	Борис	Николаевич	239-55-76	пр. Независимости, 46-54
9	8	Германович	Рита	Мироновна	278-31-51	ул. Казинца, 26-9
10	9	Бинцаровский	Теодор	Петрович		ул. Корженевская, 1-288

Рис. 3. Таблица Читатели в MSExcel.

Найдите отличие, которое имеет место при экспорте таблицы из базы данных в электронную таблицу первым и вторым способом (имеется ввидупредставление конечного результата).

Экспорт в текстовый файл

Можно экспортировать данные таблицы, набора записей запроса на выборку или перекрестного запроса в текстовый файл одного из двух форматов: с разделителями или с фиксированной длиной записи.

Для экспорта выполните следующие действия:

1. Откройте базу данных MSAccess, из которой вы хотите экспортировать данные. Если она уже открыта, переключитесь в окно базы данных.

2. В окне базы данных выделите имя экспортируемого объекта и выберите команду Э**кспорт** в меню **Файл**. MSAccess откроет окно диалога **Экспорт объекта**. В раскрывающемся списке **Тип файлов** выберите тип **Текстовые файлы**, укажите папку и имя файла, в который вы хотите экспортировать данные, а затем нажмите кнопку**Сохранить**.

3. MSAccess запустит мастера экспорта текста и откроет его первое окно, позволяющее выбрать формат текстового файла: с разделителями или с фиксированной длиной записей. Выбрав подходящий вариант, нажмите кнопку**Далее**.

4. При экспорте данных в текстовый файл с разделителями мастер откроет окно, в котором можно выбрать символы для разделителя полей и ограничителя текста. Кроме того, можно попросить MSAccess включить в первую строку имена полей. Если вы импортируете данные в текстовый файл с фиксированной длиной записей, мастер откроет окно, позволяющее подобрать ширину полей. Нажав кнопку **Дополнительно**, вы можете выбрать или изменить спецификацию импорта/экспорта. В последнем окне мастера экспорта текста можно изменить предлагаемое имя текстового файла. После нажатия кнопки **Готово** в случае успешного завершения экспорта данных создаст текстовый файл выбранного вами формата.

Задание

1. Выполните экспорт таблицы **Сотрудники** из базы данных **Кадры** в текстовый файл с таким же именем. Для текстового файла выберите формат с фиксированной длиной записи. Откройте полученный текстовый файл **Учреждения** и посмотрите, как он выглядит.

Для решения данной задачи сделайте следующее:

– Откройте базу данных **Кадры** и во вкладке **Таблицы** окна базы данных выделите имя **Сотрудники**. Выберите команду Э**кспорт** в меню **Файл**. В появившемся диалоговом окне **Экспорт объекта** из списка **Тип файла** выберитетип **Текстовые файлы**, а в поле **Папка** укажите адрес, где будет размещаться экспортируемый текстовый файл **(**см. рис. 1), и нажмите на кнопку **Экспорт**.

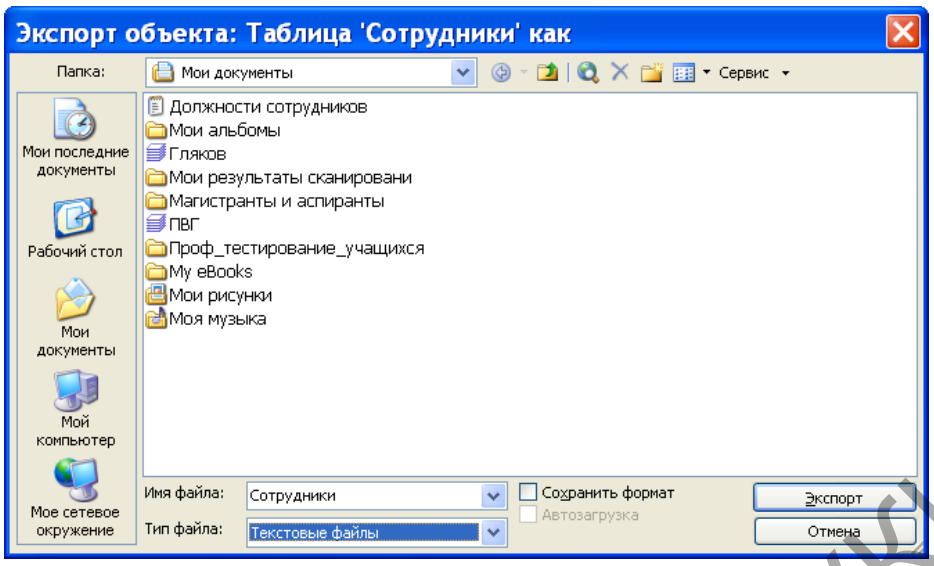


Рис. 1. Диалоговое окно **Экспорт объекта**.

– MSAccess запустит мастера экспорта текста и откроет его первое окно (см. рис. 2). В этом окне установите формат **фиксированная ширина полей – интервалы заполняются пробелами** и нажмите кнопку**Далее**.

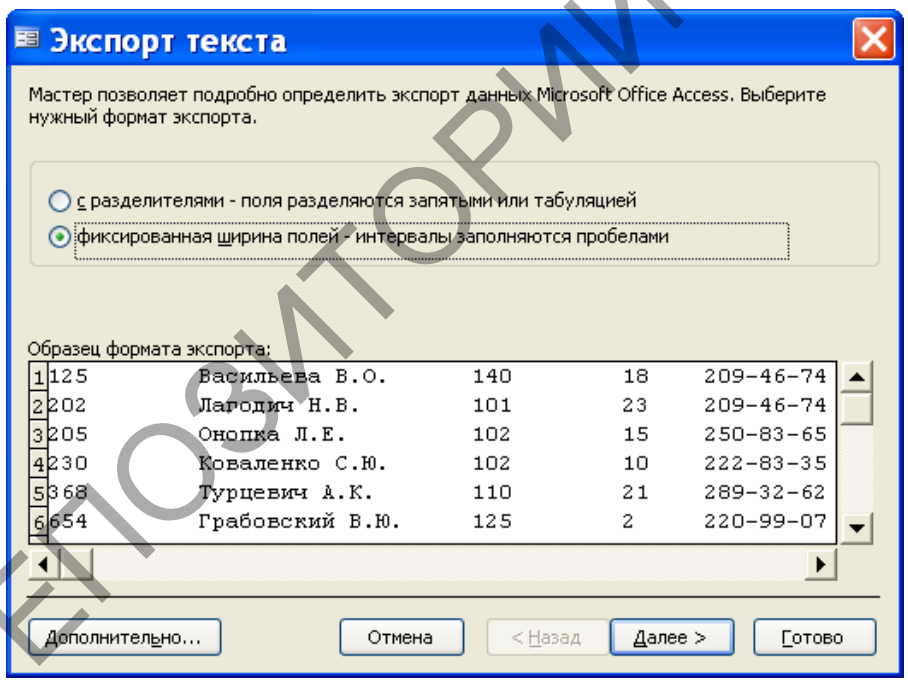


Рис. 2. Первое окно мастера экспорта.

– В следующем окне мастера экспорта нажмите на кнопку **Дополнительно**. При этом появится окно, в котором можно изменить спецификацию экспорта. В этом окне измените описание полей так, как это показано на рис. 3, и нажмите кнопку **ОК** для закрытия окна спецификации экспорта.

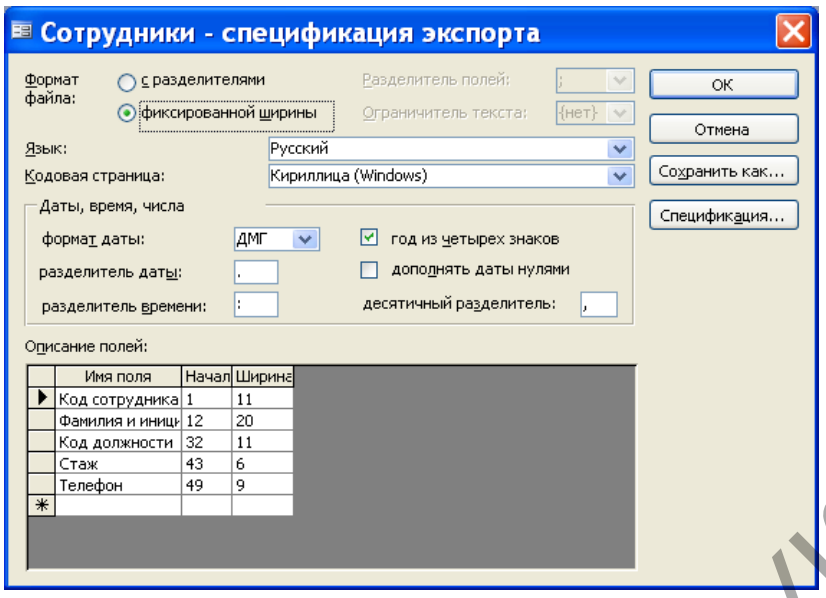


Рис. 3. Окно для спецификации экспорта.

– В последнем окне мастера экспорта текста нажмите на кнопку **Готово**, в результате чего MSAccess выдаст сообщение, показанное на рис. 4, и создаст текстовый файл выбранного нами формата.

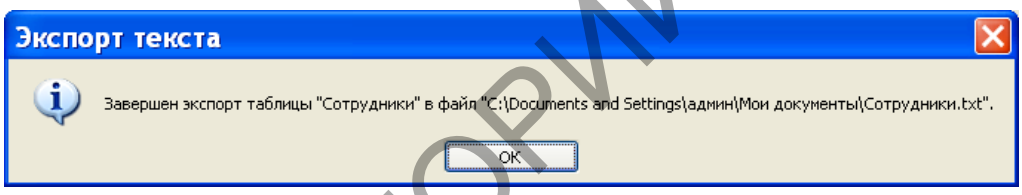


Рис. 4. Сообщение о завершении экспорта таблицы.

2. Откройте созданный вами текстовый файл и убедитесь в том, что он будет иметь вид, показанный на рис. 5.

Рис. 5. Текстовый файл с фиксированной длиной записей.

3. Выполните экспорт таблицы **Читатели** из базы данных **Библиотека** в текстовый файл с разделителями. В качестве разделителей полей выберите символ табуляции, ограничителя текста – одинарную кавычку. В первую строку включите имена полей. Полученный файл сохраните под старым именем. Откройте созданный текстовый файл и убедитесь в том, что вы правильно выполнили экспорт

таблицы.

Лабораторная работа 15

Подготовка серийных писем

Цель работы: Сформировать умения связывать данные таблицы и запроса с документом MicrosoftWord.

Возможно, одной из самых интересных черт MSAccess является его способность связывать данные таблицы или набора записей запроса с документом MicrosoftWord. После установления связи документ можно открыть в MSWord и использовать, например, для печати серийных писем, в которые автоматически вставляются текущие данные из MSAccess.

Чтобы связать данные из базы MSAccess с документом MSWord, можно выполнить следующие действия:

1. Откройте базу данных и выберите таблицу или запрос.

2. Выполните команду **Сервис| Связи с Office| Слияние с MSWord** или выберите команду **Слияние с MSWord** в раскрывающемся списке кнопки **Связи cOffice** на панели инструментов. MSAccess запустит мастера слияния с документами MicrosoftWord.

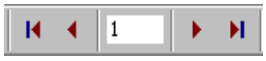
3. У вас есть две возможности: связать данные с существующим документом или создать новый документ MSWord. Если вы хотите связать данные с существующим документом, мастер попросит указать его местонахождение. Сделав это, нажмите кнопку **ОК**.

4. Мастер запустит MicrosoftWord и установит связь между документом и таблицей или запросом. При создании составного документа с внедренными полями из таблицы или запроса MSAccess можно использовать средства, предоставляемые панелью инструментов **Слияние** в Word (см. рис. 1).

Рис. 1. Панель инструментов **Слияние**.

Назначение кнопок панели инструментов **Слияние** приведено в следующей таблице:

Список или группа кнопок	Назначение списка или группы кнопок
Добавить поле слияния	Добавляет в документ MSWord поле из таблицы или запроса базы данных MSAccess.
Добавить поле Word	Добавляетвдокумент Word одноизследующихполей: ASK, FILLIN, IF…THEN…ELSE, MERGEREC,

	MERGESEQ, NEXT, NEXTIF, SET, SKIPIF.
	Устанавливает для полей режим данных либо отменяет его.
	Выполняет переход к первой, предыдущей, с указанным номером, следующей или последней записи соответственно.
	Выводит на экран диалоговое окно **Слияние**.
	Выполняет соответственно одно из следующих действий: поиск ошибок, слияние в новый документ, слияние при печати или объединение.
	Осуществляет поиск записи или правку источника данных соответственно.

Рассмотрим, как можно подготовить серийные письма на примере. Пусть требуется подготовить письма для вызова слушателей курсов переподготовки кадров профессионального образования на вторую сессию.

Подготовьте в текстовом редакторе MSWord документ, показанный на рис. 2. Сохраните этот документ под именем **Вызов** и закройте его.

Откройте базу данных "**Кадры**". В окне базы данных "Кадры" во вкладке **Запросы** выделите имя запроса **Список слушателей курсов переподготовки**. Выполните команду **Сервис**| **Связи cOffice**| **Слияние cMSWord**. В появившемся окне **Слияние с документами MicrosoftWord**активизируйте параметр**Установить связь с текстовым документом MicrosoftWord** и нажмите кнопку **ОК**. Появится окно **Выбор документаMicrosoftWord**. В нем укажите местонахождение документа **Вызов**, выделите его имя и нажмите кнопку**Открыть**. После этого перейдите в окно MicrosoftWord с документом **Вызов**.

Уважаемый(ая) !

Приглашаем Вас в УО РИПО на занятия. Вторая сессия состоится с 20 октября 2016 года по 18 ноября 2016 года.

Декан факультета повышения
квалификации и переподготовки кадров
профессионального образования Н.Я.Кушнир

Рис. 2. Вид документа **Вызов** для подготовки серийных писем.

Установите текстовый курсор в том месте документа **Вызов,** где должно быть вставлено поле **Фамилия и инициалы,** сформированное в запросе **Список слушателей курсов переподготовки** базы данных **"Кадры"**. В данном случае курсор надо расположить в первой строке документа перед восклицательным знаком. Раскройте список**Добавить поле слияние** на панели инструментов **Слияние** и выберите из него поле **Фамилия и инициалы**. Имя этого поля появится в угловых скобках там, где был расположен текстовый курсор (см. рис. 3).

Нажмите на кнопку**Объединить** на панели инструментов **Слияние**. При этом появится окно **Слияние,** в котором в списке **Назначение** можно выбрать команду **Новый документ**, если требуется сохранить полученный после слияния документ, либо выбрать команду **Принтер**, если полученный документ не предполагается сохранять, а сразу выводить его на печать, либо команду **Электронная почта**, если документ надо отправить по электронной почте. Во всех этих случаях

Уважаемый(ая) «Фамилия и инициалы»!

Приглашаем Вас в УО РИПО на занятия. Вторая сессия состоится с 20 октября 2016 года по 18 ноября 2016 года.

Декан факультета повышения
квалификации и переподготовки кадров
профессионального образования Н.Я.Кушнир

можно задать диапазон записей.

Рис. 3. Вид документа **Вызов** после вставки поля.

Для определенности выберите команду **Новый документ**, задайте диапазон записей с 1 по 5 и нажмите кнопку**Объединить**. После этого в окне MicrosoftWord откроется документ с именем по умолчанию **Формы1**, содержащий серийные письма. Причем каждое письмо будет расположено в отдельном разделе документа **Формы1**.

Задание

В базе данных **Библиотека** создайте запрос **Список читателей, не сдавших книги в срок**, в котором должны быть поля: **Фамилия и инициалы**, **Количество книг**, **Домашний адрес**, **Домашний телефон**. На

основании этого запроса подготовьте серийные письма, которые должны иметь вид, показанный на рис. 4.

Уважаемый(ая) «Фамилия и инициалы»!

Напоминаем Вам, что Вы должны вернуть «Количество книг» книг(и, у) в библиотеку до 20 декабря 2016 года.

Зав. библиотекой В.Н.Василевская

Рис. 4. Вид серийного письма.

Документ с подготовленными серийными письмами сохраните под именем **Напоминание**.

Лабораторная работа 16

Автоматизация работы с помощью макросов

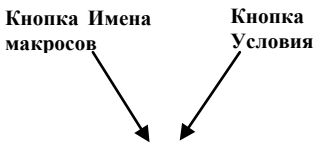
Цель работы: Сформировать умения для автоматизировать работу приложения с помощью макросов.

В MSAccess можно определить макрос, выполняющий практически те же действия, которые реализуются с помощью клавиш на клавиатуре или мыши. Макрос может содержать одну или несколько макрокоманд, позволяющих автоматизировать работу приложения. Основным преимуществом макросов является то, что они могут выполняться в ответ на многие события, например изменение данных, открытие или закрытие формы или отчета, а также передачу фокуса от одного элемента управления к другому. В макросе можно задать проверку условий таким образом, что в зависимости от значений данных в формах или отчетах будут выполняться различные действия.

Макросы особенно полезны для построения небольших персональных приложений или создания прототипов больших приложений. Изучение макросов – прекрасное введение в программирование MSAccess в целом.

Создание макроса

Откройте базу данных **Библиотека**. В окне базы данных перейдите на вкладку **Макросы** и нажмите кнопку**Создать**. В результате появится окно для создания макроса, подобное тому, что приведено на рис. 1. Верхняя часть окна используется для определения макроса, а нижняя предназначена для ввода значений аргументов макрокоманд, включенных в него.



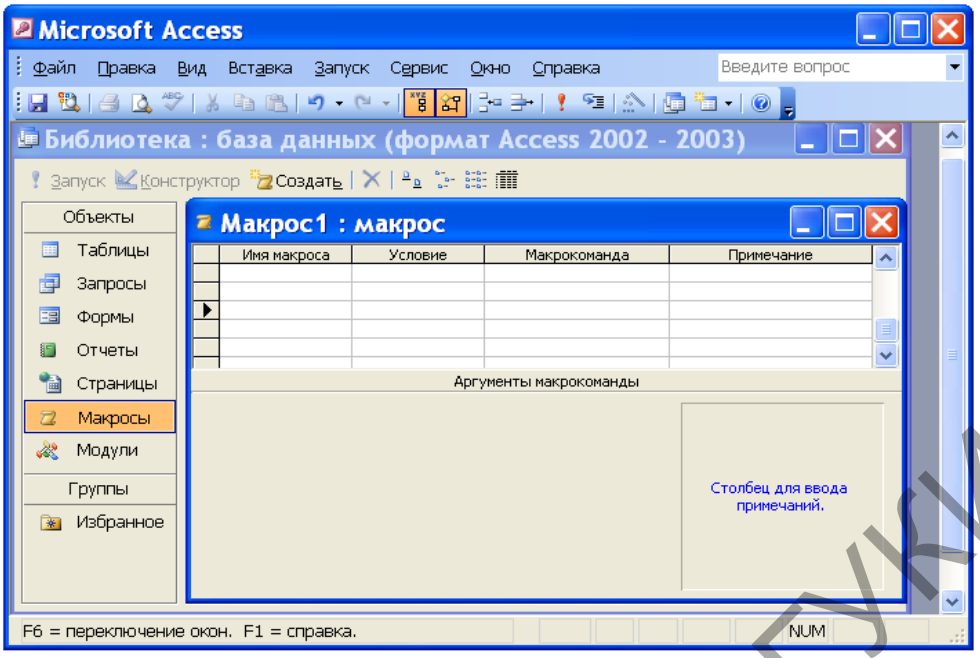


Рис. 1. Окно нового макроса.

В верхней части окна присутствуют, по крайней мере, два столбца с заголовками **Макрокоманда** и **Примечание**. Чтобы увидеть все четыре столбца, показанные на рис. 1, нажмите кнопки **Имена макросов** и **Условия** на панели инструментов.

В правой нижней части окна макроса выводится краткая справка. Ее содержание меняется в зависимости от положения курсора в верхней части окна макроса. Напомним, что для получения полной контекстной справки всегда можно нажать клавишу **F1**.

В столбце **Макрокоманда** задается одна из 49 макрокоманд, предоставляемых MSAccess. Если вы щелкните в любой ячейке этого столбца, то в правом конце ячейки появится кнопка со стрелкой вниз. Нажатие этой кнопки открывает список макрокоманд, показанный на рис. 2. Напомним, что процедуру раскрытия списка в ячейке можно выполнить быстрее. Для этого достаточно сразу выполнить щелчок мышью на месте, где должна появиться кнопка со стрелкой.

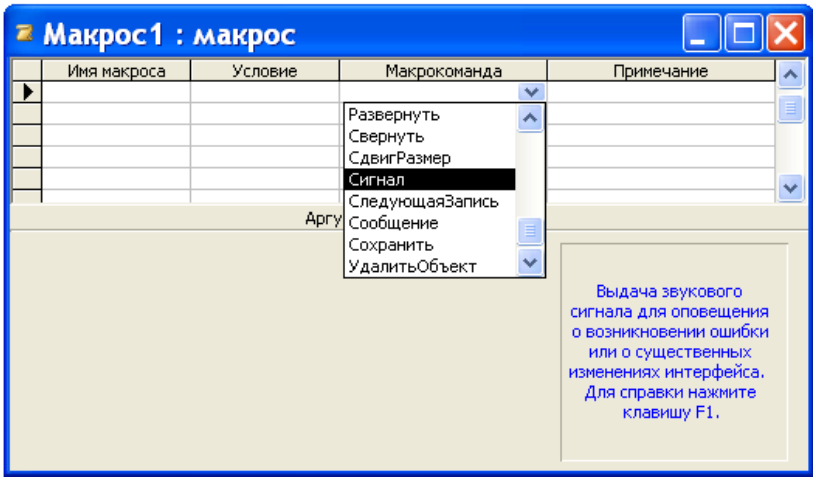


Рис. 2. Раскрывающийся список макрокоманд.

Знакомство с работой окна макроса начнем с макрокоманды **Сообщение**. Выберите ее в списке макрокоманд. Эта макрокоманда используется для открытия модального окна с сообщением. Такое окно может использоваться для вывода в приложении разного рода предупреждающих или информационных сообщений без создания для этого специальной формы.

Предположим, что наше сообщение представляет собой приветствие. В соответствующую ячейку столбца **Примечание** введите текст «Приветствие». Этот столбец особенно полезен для документирования сложных макросов, содержащих множество макрокоманд.

В нижней части окна макроса задайте аргументы макрокоманды **Сообщение** так, как это указано на рис. 3. Значение аргумента Сообщение представляет собой текст, который будет выводить MSAccess в окне диалога. Введите для этого аргумента следующий текст: Добро пожаловать в Автоматизированную информационную систему «Управление качеством». Аргумент **Сигнал** служит для воспроизведения звукового сигнала при появлении окна диалога. Если мы желаем, чтобы сигнал воспроизводился, то значение этого аргумента должно быть «да», в противном случае – «нет».

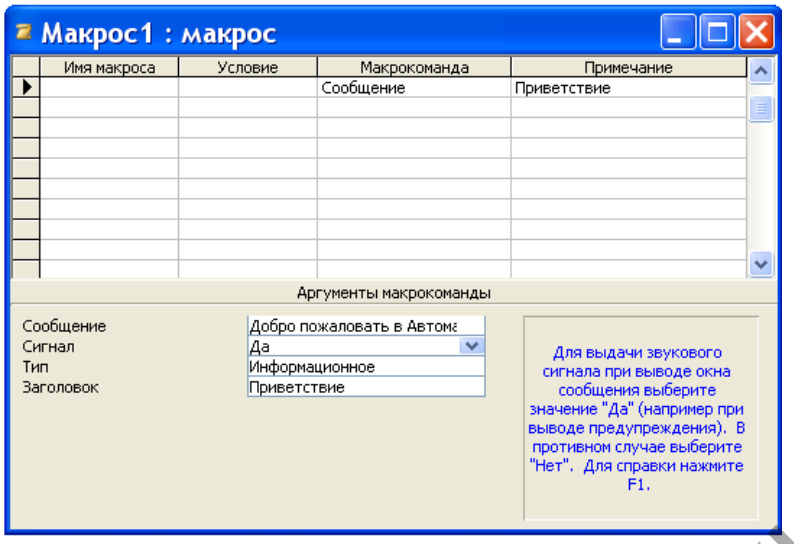


Рис. 3. Аргументы макрокоманды **Сообщение.**

Аргумент **Тип** позволяет поместить в окно сообщения значок. Этот аргумент может принимать одно из следующих пяти значений: «Отсутствует», «Критическое», «Предупреждающее?», «Предупреждающее!», «Информационное». Установите для этого аргумента значение «Информационное». В ячейку аргумента **Заголовок** вводится текст, отображаемый в заголовке окна диалога. Введите для этого аргумента текст «Приветствие».

Сохранение макроса

После того, как заданы аргументы для макрокоманд макроса, макрос надо сохранить. Для сохранения макроса используется в меню **Файл** команда**Сохранить** или **Сохранить как**. При использовании команды**Сохранить** откроется окно диалога, приведенное на рис. 4. Введите в это окно имя **Приветствие системы** и нажмите кнопку **ОК**, чтобы сохранить макрос.

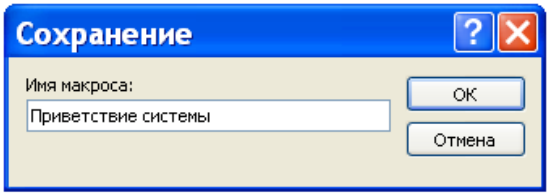


Рис.4. Окно диалога для задания имени макроса.

Проверка работы макроса

Макросы, не зависящие от элементов управления открытой формы или отчета (как в нашем случае), могут быть запущены из окна базы данных или окна макроса. Если макрос зависит от формы или отчета, его надо связать с соответствующим событием и запускать при его

возникновении. Перед запуском макроса полезно проверить его работу, выполнив макрос в пошаговом режиме.

Для пошаговой проверки созданного макроса перейдите в окно базы данных, на вкладке **Макросы** выделите его имя и нажмите копку **Конструктор**. После открытия окна макроса нажмите кнопку**По шагам** на панели инструментов либо выберите команду **По шагам** в меню**Запуск**. Запустите макрос на выполнение, нажав на панели инструментов кнопку **Запуск**. У вас появится на экране окно диалога **Пошаговое исполнение макроса**, показанное на рис. 5.

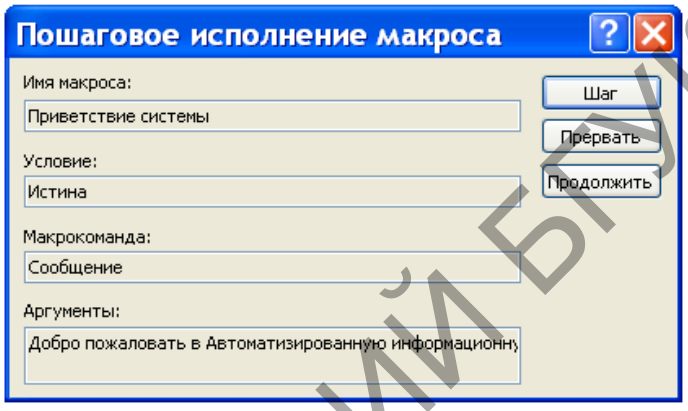


Рис. 5. Окно диалога **Пошаговое исполнение макроса.**

Нажмите в этом окне кнопку **Шаг**. При этом запустится макрокоманда, представленная в окне диалога, и MSAccess выведет на экран модальное окно диалога с созданным вами сообщением (см. рис. 6).

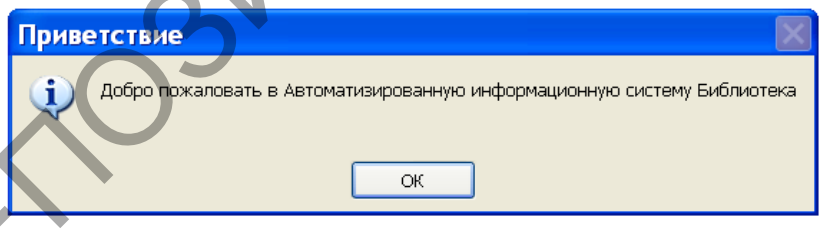


Рис. 6. Окно диалога, созданное макросом Приветствие системы.

Нажмите кнопку **ОК**, чтобы закрыть это окно. Если в макросе определено несколько макрокоманд, после первого шага вы вернетесь в окно диалога **Пошаговое исполнение макроса**, которое покажет следующую готовую к исполнению макрокоманду. Так будет происходить до тех пор, пока не будут исполнены все макрокоманды макроса.

Если во время выполнения приложения в каком-нибудь макросе встретится ошибка, MSAccess сначала выведет окно диалога, объясняющее ее. Затем появится окно диалога **Ошибка выполнения макрокоманды** с

информацией об ошибке. В этот момент следует нажать кнопку**Прервать,** а затем исправить в макросе причину ошибки.

После завершения отладки макроса надо вернуться в окно макроса и нажать на кнопку**По шагам**, чтобы отменить пошаговый режим. В противном случае все остальные макросы будут выполняться в пошаговом режиме.

Внутри одного макроса можно определить несколько макрокоманд и порядок их выполнения. Такой макрос называют сложным. Примером такого макроса может послужить макрос **Autoexec**, приведенный на рис. 7.

Обратим ваше внимание на одно уникальное свойство, которым обладает макрос с именем **Autoexec**. Такой макрос всегда запускается автоматически при открытии базы данных, в которой он имеется. Если мы желаем, чтобы этот макрос автоматически не запустился надо при открытии базы данных удерживать клавишу **Shift**.

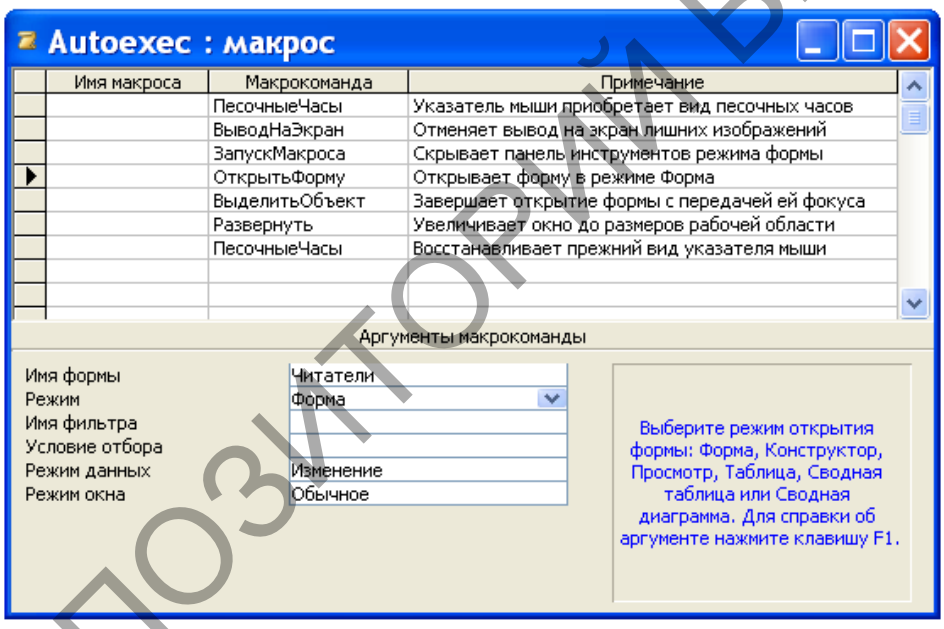


Рис. 7. Автоматически запускаемый макрос **Autoexec.**

Задание

1. Создайте макрос, который будет удалять встроенную панель инструментов режима формы. Для этой цели воспользуйтесь макрокомандой **ПанелиИнструментов,** у которой аргументу **Панель инструментов** задайте значение «режим формы», а аргументу**Показать** – значение «нет».Назовите созданный макрос **Удаление панели.**

2. Создайте макрос, приведенный на рис. 7. Обратите внимание, что этот макрос содержит семь макрокоманд, которые MSAccess будет автоматически выполнять при открытии базы данных. Первая макрокоманда **ПесочныеЧасы** выводит указатель мыши в виде песочных

часов, чтобы показать пользователю, что выполнение макроса может потребовать нескольких секунд. Следующая макрокоманда **ВыводНаЭкран** с аргументом**Включить вывод**, установленным в значение **Нет**, предназначена для того, чтобы на экран не выводились лишние изображения при выполнении макроса.

Макрокоманда **ЗапускМакроса** в данном случае должна запустить макрос **Удаление панели**, который удалит встроенную панель инструментов режима формы.

Макрокоманда **ОткрытьФорму** открывает форму **Читатели**. На рис. 7 показано, что работу этой макрокоманды определяют четыре аргумента. Аргумент **Имя формы**предназначен для заданияимя открываемой формы, а аргумент **Режим** устанавливает режим, в котором форма будет открыта. В нашем примере используется режим **Форма**.

По умолчанию для аргумента **Режим данных** устанавливается значение **Изменение**, позволяющее пользователю добавлять, удалять и изменять записи во время работы с формой. Аргумент **Режим окна** по умолчанию принимает значение **Обычное**. В этом случае форма открывается в режиме, заданном ее свойствами.

MSAccess не всегда ждет завершения работы макрокоманды перед выполнением следующей. Можно завершить открытие формы, передав ей так называемый фокус. Это делается с помощью макрокоманды **ВыделитьОбъект**, в которой указывается объект, получивший фокус (в примере это форма **Читатели**). Затем макрокоманда**Развернуть** увеличивает активное окно (то есть окно, обладающее фокусом) до размеров рабочей области MSAccess. Последняя макрокоманда **ПесочныеЧасы** в макросе **Autoexec** восстанавливает прежний вид указателя мыши, свидетельствуя об окончании работы макроса.

3. Поэкспериментируйте с уникальным свойством, которым обладает макрос **Autoexec**. Для этого один раз откройте базу данных **Библиотека** с нажатой клавишей **Shift**, а второй раз откройте базу данных **Библиотека**, не используя клавишу **Shift.**

Лабораторная работа 17

Использование кнопок в формах

Цель работы: Сформировать умения разрабатывать кнопочную форму для автоматизации работы с приложением.

На заключительной стадии разработки базы данных для автоматизации работы с приложением разрабатывают так называемую кнопочную форму, которая позволяет запускать все процессы, выполняемые в базе данных. Как правило, такая форма является заставкой приложения и появляется на экране при открытии базы данных.

Кнопочную форму можно создать следующим образом.

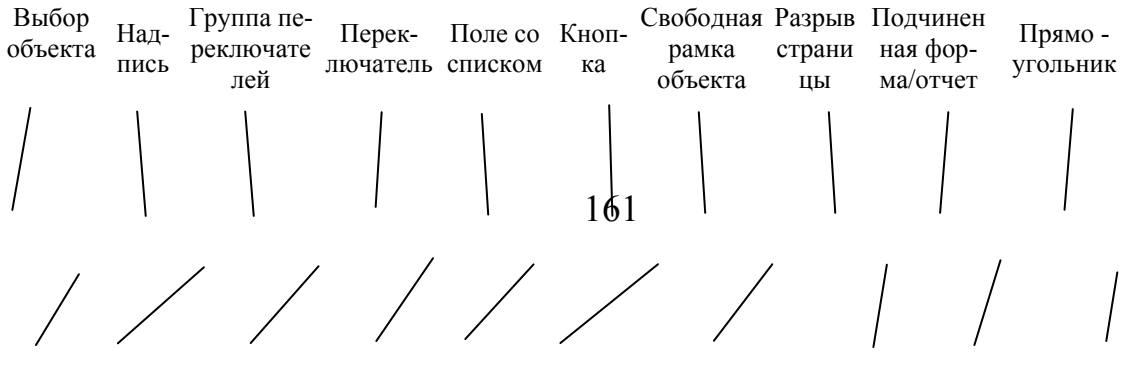
1. В окне базы данных выполняют щелчок мышью на кнопке **Формы** на панели **Объекты** окна базы данных.

2. Создают форму без источника записей.Для создания формы выполняют двойной щелчок мышью на команде **Создание формы в режиме конструктора.**

3. Настраивают в режиме конструктора макет формы. Для задания высоты раздела **Область данных** щелчком правой клавиши мыши в области данных формы вызывают контекстное меню, из него выбирают команду **Свойства** и для свойства Высота устанавливают требуемое значение.

Настройка непосредственно свойств макета формы осуществляется в окне **Форма**. Напомним, что для вызова этого окна надо выполнить двойной щелчок мышью на маркере выделения формы, расположенный на пересечении горизонтальной и вертикальной линеек в окне формы в режиме конструктора. В окне **Форма** можно настраивать такие свойства макета, как **Подпись**, **Ширина**, **Кнопка контекстной справки**, **Полосы прокрутки**, **Область выделения** и другие (всего 31 свойства).

4. Размещают на макете формы кнопки. Для создания кнопок в кнопочной форме используют **Панель элементов** (см. рис. 1). Эта панель появляется автоматически при открытии формы в режиме конструктора.



Мастера | Поле | Выключатель | Флажок | Списокк | Рисунок | Присоединенная рамка рисунка | Наборвкладок | Линия | Другиеэлементы

Рис. 1. **Панель элементов** с кнопками.

Самый простой способ создать кнопку в кнопочной форме – воспользоваться мастером создания кнопок. При использовании мастера создания кнопок выполняют следующее. Вначале включают щелчком мыши кнопку **Мастера** на **Панели элементов** (если она отключена, то есть не выделена цветом). После этого выполняют щелчок мышью на элементе **Кнопка** на **Панели элементов**, перемещают указатель мыши в требуемое место раздела **Область данных** и выполняют щелчок.

В появившемся диалоговом окне мастера **Создание кнопок** определяют, напримеркатегорию **Работа с формой** идействие**Открыть форму** инажимают кнопку**Далее**. Новое диалоговое окно требует выбора формы, которую надо будет открывать нажатием данной кнопки. После нажатия кнопки**Далее** надо указать, требуется ли отбор сведений для отображения в форме, и нажать кнопку **Далее**. В следующем диалоговом окне надо указать, что необходимо разместить на кнопке: текст или рисунок и нажать кнопку**Далее**. В последнем диалоговом окне задают имя кнопке, которое будет упрощать дальнейшие ссылки на нее, и нажимают кнопку **Готово**.

5. Размещают на кнопочной форме другие элементы, позволяющие улучшить ее дизайн.

Для размещения текста в форме можно воспользоваться кнопкой **Надпись** на **Панели элементов**. Выполняют это следующим образом. Выбирают элемент **Надпись** на **Панели элементов** и в разделе **Область данных** формы этим элементом прорисовывают прямоугольник, в котором следует поместить текст. Далее требуемый текст вводят с клавиатуры**.** Когда элемент **Надпись** является активным (по его контуру размещены маленькие прямоугольники) его можно форматировать, используя кнопки панели **Формат**.

Элемент **Рисунок** на **Панели элементов** позволяет размещать изображения в разделе **Область данных** формы. После того как вы выбрали этот элемент и начертили прямоугольную область, в которую надо поместить рисунок, появляется диалоговое окно **Выбор рисунка**. В

этом окне надо найти графический файл с требуемым рисунком и нажать кнопку **ОК**.

Задание

Создайте главную кнопочную форму, похожую на ту, которая показана рис. 2. На этой форме имеются восемь кнопок: **Поступление книг**, **Месячная загрузка**, **Рейтинг книг**, **Поиск книг**, **Просмотр содержания**, **Выдача книг**, **Возврат книг**, **Письменное уведомление**, **Выход из Access**, а также текст и рисунок.

1. В верхней части окна формы в режиме конструктора наберите текст, показанный на рис 2. Для набора текста в форме используется кнопка **Надпись** на **Панели элементов**.Нажмите кнопку **Надпись** на **Панели элементов**. Установите указатель мыши в области данных окна формы в режиме конструктора в том месте, где должен размещаться текст, и при нажатой левой клавише создайте рамку надписи, а затем в ней наберите текст. Отформатируйте набранный текст подходящим образом.

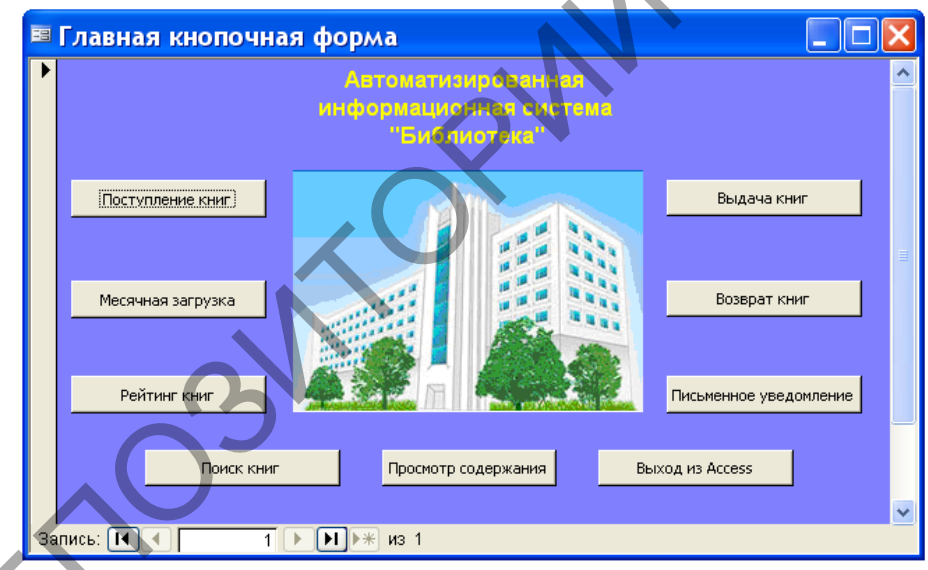


Рис. 2. Пример главной кнопочной формы.

2. Вместо фотографии с изображением БГУКИ, показанной в примере кнопочной формы, вставьте рисунок, который, на ваш взгляд, будет подходить для разрабатываемой информационной системы. Выберите элемент **Рисунок** на **Панели элементов** и начертите прямоугольную область, в которую надо поместить рисунок. В диалоговом окне **Выбор рисунка** найдите графический файл с требуемым рисунком и нажмите кнопку **ОК**.

3. Создайте запрос, который будет содержать следующие поля: **Код книги**, **Автор**, **Название**, **Кодиздательства**, **Наименование**, **Город**,

Объем, **Годиздания**, **Стоимость**. Запрос назовите **Поступлениекниг**. По данному запросу создайте автоформу в столбец, которую также назовите **Поступление книг** (см. рис. 3).

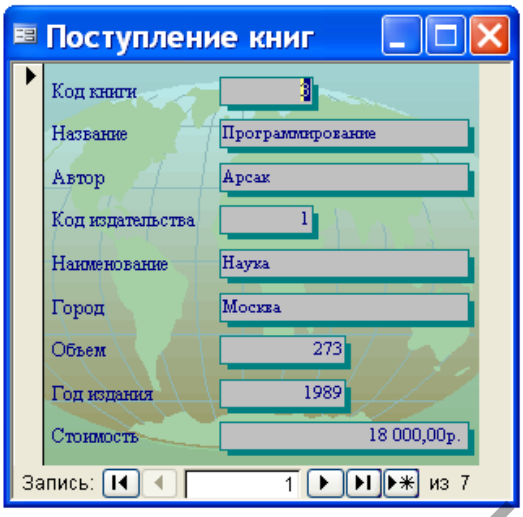


Рис. 3. Форма **Поступление книг**.

Выполните щелчок мышью на элементе **Кнопка** на **Панели элементов** и изобразите прямоугольник в разделе Область данных главной кнопочной формы. В появившемся диалоговом окне мастера **Созданиекнопок** выберите категорию **Работа с формой** и действие**Открытьформу**. В следующем диалоговом окне мастера кнопок укажите в качестве источника данных форму **Поступлениекниг**. В соответствующем диалоговом окне мастера кнопок укажите, что на кнопке надо поместить текст **Поступлениекниг**, и нажмите на кнопку **Готово**.

4. Для расчета количества выданных книг по месяцам ранее мы составили перекрестный запрос **Выдача книг по месяцам**. Сейчас мы свяжем этот запрос с кнопкой **Месячная загрузка**. Для этого в диалоговом окне мастера кнопок укажите категорию **Разное**, а действие – **Выполнить запрос**. При указании имени источника данных в следующем диалоговом окне укажите имя запроса **Выдача книг по месяцам**.

6. Для создания кнопки **Поиск книг** воспользуйтесь ранее созданным запросом с параметром **Поиск книг по фамилии автора**. Параметр позволяет набирать не все буквы фамилии автора, а только несколько первых букв. Напомним, что диалоговое окно для ввода значения параметра в этом запросе имело вид, показанный на рис. 4.

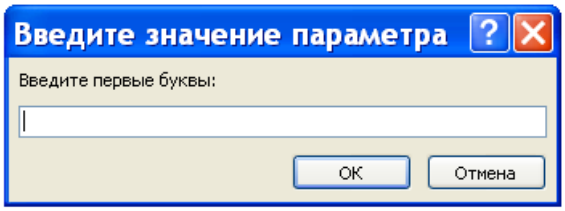


Рис. 4. Диалоговое окно для ввода значения параметра.

7. Чтобы можно было просматривать содержание книг, воспользуемся ранее созданной формой **Содержание книг**. Для связывания кнопки **Просмотр содержания** с источником данных, являющимся формой **Содержание книг**, в диалоговом окне мастера кнопок выбирают категорию **Работа с формой** и действие**Открыть форму**, а в следующем диалоговом окне выбирают форму **Содержание книг**, которая будет открываться нажатием данной кнопки.

5. Кнопка **Рейтинг книг** создается подобно кнопке **Просмотр содержания**, но источником данных для нее будет не форма, а страница, которую мы ранее назвали **Books**. Поэтому в диалоговом окне мастера кнопок укажите категорию **Работа с формой**, а действие – **Открыть страницу**. При указании имени страницы выберите имя **Books**.

8. Создайте кнопку **Выдача книг,** котораявкачествеисточникаданныхбудетиспользоватьформу **Выдача книг,** содержащуюполя**: Код читателя**, **Код книги**, **Дату возврата и Датузаказа**.

9. При создании кнопки **Возврат книг** вначале создайте запрос. Этот запрос должен для конкретного читателя выводить список книг, которые он не вернул в библиотеку. Для ввода информации о читателе создайте параметр **Код читателя**. В список книг включите следующие поля: **Фамилия**, **Имя**, **Отчество**, **Автор**, **Название**, **Год издания**, **Стоимость, Дата заказа, Дата возврата**. Запрос назовите **Возврат книг**.

На языке SQLэтот запрос будет выглядеть следующим образом:

SELECT Читатели.Фамилия, Читатели.Имя, Читатели.Отчество, Книги.Автор, Книги.Название, Книги.[Год издания], Книги.Стоимость, [Выдача книг].[Дата заказа], [Выдача книг].[Дата возврата]

FROM Читатели INNER JOIN (Книги INNER JOIN [Выдачакниг]

ON Книги.[Код книги] = [Выдача книг].[Код книги]) ON Читатели.[Код читателя] = [Выдача книг].[Код читателя]

WHERE ((([Выдача книг].[Код читателя])=[Введите Код читателя:]) AND (([Выдача книг].[Дата возврата]) IsNull));

После этого при создании кнопки **Возврат книги** в диалоговом окне мастера кнопок выберите категорию **Разное**, а в ней выберите действие**Выполнить запрос**. В следующем диалоговом окне в качестве источника данных укажите созданный запрос **Возврат книг**.

10. Кнопку **Письменное уведомление** создайте точно таким же образом, как и предыдущую, но в качестве источника данных для нее

используйте запрос с параметром **Список читателей для вызова**. В качестве параметра в этом запросе задается количество дней, которые читатель может на руках держать книгу.

11. Кнопка**Выход из Access**предназначена для завершения работы с приложением MSAccess. Для ее создания выполните следующее. В режиме конструктора форм с помощью мастера кнопок выберите категорию **Приложение**, а в ней – действие**Выйти из приложения**.

12. Создайте форму, которая приведена на рис.5.

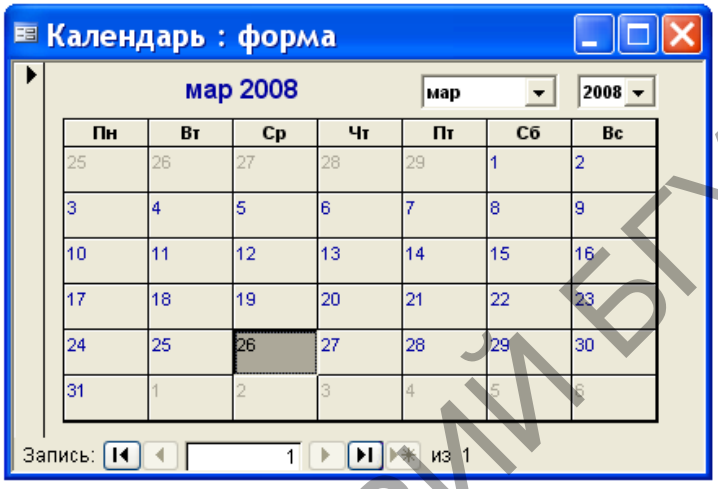


Рис. 5. Форма с элементом управления Календарь 11.0.

Для создания этой формы используйте на **Панели элементов** элемент **Другие элементы**(см. рис. 1). В списке элементов, открываемых этим элементом, выберите элемент управления **Календарь 11.0**. Форму назовите **Календарь**. Выясните с помощью этой формы, на какой день недели попадает 1 января 2010 года.

Лабораторная работа 18

Поддержка баз данных

Цель работы: Сформировать умения выполнять операции, обеспечивающие поддержку баз данных в MS Access.

Создание резервной копии

Чтобы обезопасить себя от случайных ошибок или непредвиденных последствий, перед внесением серьезных изменение сделайте резервную копию файла базы данных. Это можно выполнить, например, с помощью проводника Windows.

Если вы хотите внести изменения в одну таблицу, ее копию можно создать прямо в MSAccess. Для копирования структуры любой таблицы (содержимого окна таблицы в режиме конструктора), данных таблицы (содержимого окна таблицы в режиме таблицы) либо структуры и данных можно воспользоваться следующей процедурой:

– Откройте базу данных, содержащую таблицу, которую вы хотите скопировать. Если она уже открыта, в окне базы данных перейдите на вкладку **Таблицы**.

– Выделите нужную таблицу, щелкнув на ее имени в окне базы данных.

– Выберите команду**Копировать** в меню **Правка** илинажмитекнопку **Копировать** на панели инструментов. С помощью этой операции вся таблица (структура и данные) копируется в буфер обмена.

– Выберите команду**Вставить** в меню **Правка** или нажмите кнопку **Вставить** на панели инструментов. MSAccess откроет диалоговое окно **Вставка таблицы**. Введите новое имя таблицы и дату создания копии. Выберите режим копирования, установив соответствующий переключатель: **Структура и данные**, **Только структура** или **Добавление данных в таблицу**. Нажмите после этого кнопку **ОК**.

Шифрование базы данных

Шифрование (кодирование)– защита базы данных от несанкционированного доступа с помощью текстового редактора или средств работами с файлами, например входящих в состав Windows или NortonUtilites. Информация в зашифрованной базе данных недоступна для чтения. Шифрование несколько замедляет работу MSAccess: время расходуется на шифрование и дешифрование файлов.

Шифрование и дешифрование базы данных могут производить только члены группы **Admins**. Эти операции выполняются следующим образом.

1. Запустите MSAccess с выбранной рабочей группой.
2. В подменю **Защита** меню **Сервис** активизируйте команду **Закодироваить или раскодировать базу данных**, вследствие чего на экране появится окно выбора базы данных для шифрования.
3. Выберите базу данных, которую необходимо зашифровать или дешифровать, и нажмите кнопку **ОК**. Если выбранная база данных не зашифрована, откроется диалоговое окно **Кодирование базы данных под именем**, в котором программа предложит новое имя для зашифрованной базы данных. Если же выбранная база данных зашифрована, появится окно **Дешифрование базы данных под именем**.
4. Выберите имя файла для новой, закодированной или декодированной базы данных, и нажмите кнопку**Сохранить**.

Шифрование базы данных с помощью средств безопасности объектов базы данных SQLServer. Чтобы защитить данные и объекты данных, сохраненные в базе данных SQLServer (таблицы, представления, сохраненные процедуры и схемы) нужно зашифровать приложение базы данных с помощью инструкций SQL, приписав в конце код шифровки. После шифрования представления невозможно изменить его макет.

Повышение быстродействия приложения

Полезным инструментом в MSAccess является надстройка Анализатор быстродействия, которая анализирует базу данных и дает рекомендации по оптимизации приложения. Запуск этой надстройки осуществляется в результате активизации команды **Быстродействие** из подменю **Анализ** меню **Сервис**. После вызова данной команды открывается диалоговое окно для указания подлежащих анализу объектов (см. рис.1).

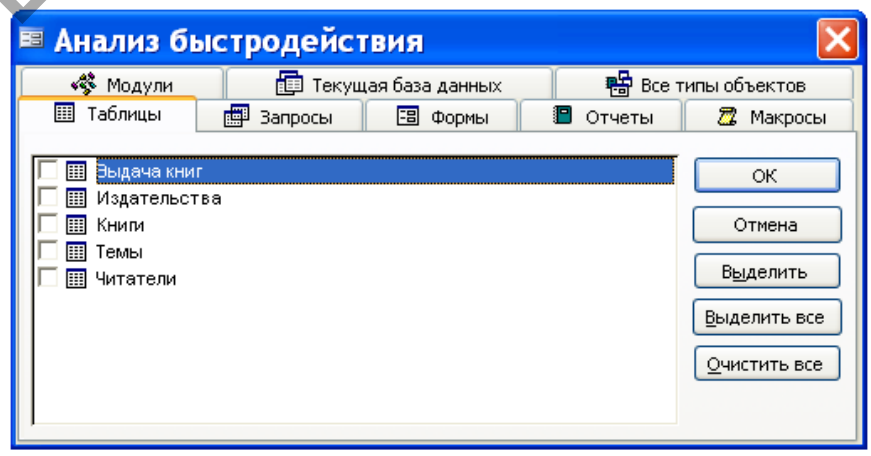


Рис. 1. Окно для выбора объектов.

В окне **Анализ быстродействия** объекты рассортированы по типам и по этому признаку объединены в списки, расположенные на отдельных вкладках. Таким образом, выбор типа объекта происходит при открытии той или иной вкладки, а выбор самого объекта – путем маркировки его имени в списке. После нажатия кнопки **ОК** производится анализ выбранных объектов. Результаты анализа выводятся на экран в поле **Результаты анализа** окна **Анализ быстродействия**.

Результаты анализа представляются в одной из следующих форм: совет, предложение, мысль или исправлено, как это показано на рис.2.

После выбора рекомендации и нажатия кнопки**Оптимизировать** выполняется оптимизация, после чего оптимизированный объект обозначается словом **исправлено**.

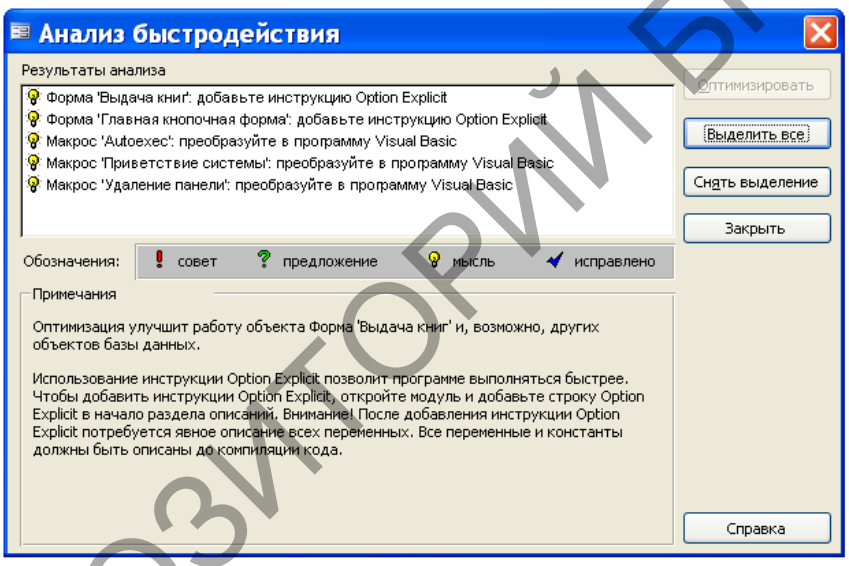


Рис. 2. Результаты анализа быстродействия.

Рекомендации по оптимизации, предложенные программой в списке **мысль**, разработчик приложения должен выполнять самостоятельно.

Сжатие базы данных

В результате удаления одних и создания новых объектов файл базы данных может стать фрагментированным. С течением времени он разрастается и занимает гораздо больше места, чем необходимо для хранения всех определений и данных.

Для удаления неиспользуемых разрозненных участков памяти базу данных следует периодически сжимать. Перед сжатием все базы данных должны быть закрыты. Кроме того, надо убедиться, что в сети никто не работает с вашей базой данных. Выберите в подменю **Служебные**

программы меню **Сервис** команду**Сжать базу данных**. MSAccess откроет диалоговое окно **База данных для сжатия**. Выберите базу данных и нажмите кнопку**Сжать**. MSAccess запросит имя для сжатой базы данных. Вы можете создать сжатую копию базы данных под другим именем или ввести имя, уже принадлежащее исходной базе данных. В последнем случае MSAccess попросит у вас подтверждение на ее замену. Получив подтверждение, MSAccess производит сжатие базы данных во временном файле. Если сжатие заканчивается успешно, MSAccess удаляет исходную базу данных и автоматически присваивает ее имя сжатой копии.

Задание

1. Зашифруйте файл базы данных **Библиотека**. Зашифрованной базе данных дайте имя **Зашифрованная база**.

2. Выполните сжатие базы данных **Библиотека**. Сжатой базе данных дайте имя **Сжатая база**.

3. В Проводнике Windows, используя вид **Таблица**, просмотрите объемы файлов **Библиотека,Зашифрованная база** и **Сжатая база**. Что можно сказать об их объемах?

4. Расшифруйте базу данных **Зашифрованная база**. Полученную базу данных назовите **Расшифрованная база**.

5. Запустите текстовый редактор MSWordи откройте в нем файл **Библиотека**. Пролистайте его. Эти же операции выполните для файлов **Зашифрованная база** и **Расшифрованная база**. Какой можно сделать вывод о том, что вы увидели?

6. Оптимизируйте быстродействие базы данных **Библиотека**, воспользовавшись командой**Быстродействие** из подменю **Анализ** меню **Сервис**.

Лабораторная работа № 19

Средства защиты базы данных

Цель работы: Сформировать умения устанавливать пароль в базе данных.

Средства защиты баз данных, реализованные в MSAccess, позволяют предотвратить умышленные или случайные просмотр, изменение и удаление информации лицами, которые не имеют соответствующих прав доступа. Эти средства особенно важны при функционировании базы данных в сети.

В MSAccess предусмотрены различные уровни защиты данных и администрирования доступа к ним. Возможности MSAccess позволяют обеспечить безопасность как самого приложения, так и файла базы данных. Простейшим средством защиты базы данных от несанкционированного доступа является пароль.

Стандартными средствами MSAccess пароль может быть установлен следующим образом:

1. Откройте базу данных с монопольным доступом.

2. В меню **Сервис** выберите команду **Защита**| **Задать пароль базы данных**.

3. В открывшемся окне **Задание пароля базы данных** введите в строку **Пароль** и в строку **Подтверждение** пароль и нажмите на кнопку **ОК**.

Чтобы открыть базу данных с монопольным доступом надо выполнить следующее:

1. Откройте базу данных с помощью команды**Открыть** из меню **Файл**.

2. В диалоговом окне **Открытие файла базы данных** выделите имя открываемой базы данных, щелкните стрелку справа от кнопки**Открыть** и из списка выберите команду **Монопольно**.

3. В появившемся окне **Предупреждение системы безопасности** нажмите на кнопку**Открыть**.

После того как пароль установлен, при каждом открытии базы данных будет появляться диалоговое окно, в которое требуется ввести пароль. Пользователи смогут открыть базу данных только после ввода правильного пароля. Этот способ достаточно надежен, поскольку Access шифрует пароль, так что к нему нет прямого доступа при чтении файла базы данных. Недостаток такого способа защиты в том, что он применяется только при открытии базы данных. После открытия базы

данных все объекты становятся доступными для пользователя (если не определена защита на уровне пользователей). Для базы данных, которая совместно используется небольшой группой пользователей или на автономном компьютере, установки пароля обычно достаточно.

Дляудаление пароля можно выполнить следующие действия:

1. Выполните команду**Открыть** в меню **Файл**.

2. В диалоговом окне **Открытие файла базы данных** выделите имя открываемой базы данных, щелкните стрелку справа от кнопки**Открыть** и из списка выберите команду **Монопольно**.

3. В появившемся окне введите пароль базы данных, который надо удалить, и нажмите на кнопку **ОК**.

4. После этого появится окно **Предупреждение системыбезопасности**. Нажмите в нем кнопку**Открыть**. При этом откроется окно базы данных.

5. Выполните в меню **Сервис** команду **Защита**|**Удалить парольбазы данных**.

6. В окно **Удаление пароля базы данных** введите пароль и нажмите кнопку **ОК**. В результате этих действий пароль будет удален из базы данных.

Усовершенствовать защиту позволяют средства поддержки рабочих групп, ведения учетных (регистрационных) записей, задания прав владения и прав доступа. С помощью средств защиты можно указать, какие операции по обработке объектов базы данных разрешается выполнять пользователю или группе пользователей. О каждом пользователе или группе ведутся учетные записи с указанием прав доступа к тем или иным объектам.

Задание

1. Создайте для базы данных **Библиотека** пароль и запомните его (В противном случае вы не сможете ни открыть базу данных **Библиотека**, ни удалить из нее пароль).

2. Попытайтесь несколько раз открыть базу данных **Библиотека**, указав неверный пароль. Какое при этом сообщение выдаст MS Access?

3. Удалите пароль из базы данных **Библиотека.**

4. Создайте собственную форму для проверки пароля входа в базу данных **Библиотека**. Для этого выполните следующее:

– На панели **Объекты** окна базы данных выполните щелчок мышью на кнопке **Формы**, а затем двойной щелчок мышью на команде **Создание формы в режиме конструктора**;

– Выполните щелчок правой клавишей мыши в разделе данных формы и в появившемся контекстном меню выберите команду **Свойства**. После этого в диалоговом окне **Раздел: ОбластьДанных** задайте высоту раздела данных 5 см. Закройте это диалоговое окно;

– Выполните двойной щелчок мышью на маркере выделения формы (он расположен в левом верхнем угле окна формы на пересечении вертикальной и горизонтальной линеек). В появившемся окне свойств формы отключите вывод полос прокрутки, области выделения, кнопок перехода, разделительных линий, кнопки оконного меню, кнопок размеров окна, кнопки закрытия окна, задайте ширину формы 8 см. Закройте окно свойств формы.

– Выберите элемент **Поле** на **Панели элементов** и с его помощью определите в центральной части области данных формы место для поля, в которое будет вводиться пароль;

– Выполните щелчок правой клавишей мыши на данном поле. В контекстном меню выберите команду **Свойства**. В появившемся окне в строку **Имя** введите имя **Вход**. Закройте окно свойств;

– Выберите элемент **Надпись**на **Панели элементов**и сверху над полем **Вход** создайте рамку для надписи. Введите в нее текст «Введите пароль».

– Сохраните созданную форму под именем **Начало работы**. Просмотрите эту форму в режиме формы. Она должна быть похожа на форму, приведенную на рис. 1. Исключение составляет наличие в этой форме кнопки **ОК**, которую мы будем создавать позже. Закройте форму **Начало работы**.

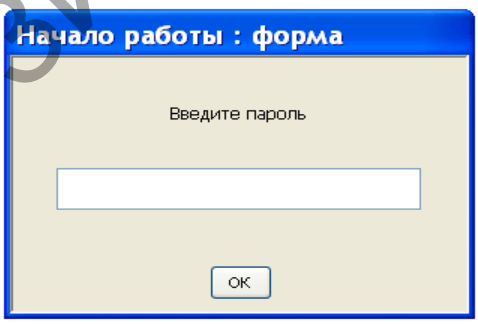


Рис. 1. Форма **Начало работы** в режиме формы.

5. Создайте макрос для проверки пароля входа в базу данных **Библиотека**(см. рис. 2):

– Выполните щелчок мышью на кнопке **Макросы** на панели **Объекты** окна базы данных. После этого нажмите кнопку**Создать** на панели инструментов окна базы данных.

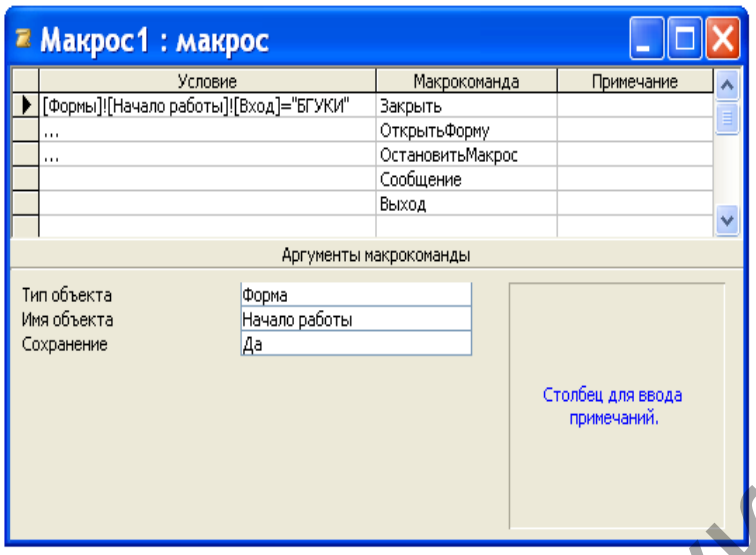


Рис. 2. Макрос **Защита** в режиме конструктора.

– В окне конструктора макроса отобразите столбец условий, нажав мышью на кнопку **Условия**, расположенную на панели **Конструктор макросов**.

– В первую строку столбца **Условие** введите следующее условие:

[Формы]![Начало работы]![Вход]="БГУКИ".

– Для первой строки столбца **Макрокоманда** выберите из списка макрокомандуЗакрыть. Задайте для нее следующие аргументы: **Тип объекта** – Форма, **Имя объекта** – Начало работы;

– Во вторую строку столбца **Условие** введите многоточие (...), для этой же строки в столбец **Макрокоманда** из списка макрокоманд выберите макрокоманду **ОткрытьФорму**. В нижней части окна в качестве аргумента **Имя формы** введите название формы **Библиотека**.

– В третьей строке в столбец **Условие** введите многоточие, а в столбец **Макрокоманда** – макрокоманду **ОстановитьМакрос**.

– В четвертую и пятую строки введите макрокоманды, которые надо выполнить в том случае, если пароль будет введен неверно. Для этого из списка макрокоманд для четвертой строки выберите макрокоманду **Сообщение** и задайте для нее следующие аргументы: **Сообщение** – Постарайтесь вспомнить его., **Сигнал** – Да, **Тип** – Отсутствует, **Заголовок** – Пароль неверный! В пятую строку в столбец **Макрокоманда** выберите из списка макрокоманду **Выход**.

– Сохраните созданный макрос под именем **Защита** и закройте его.

6. Создайте кнопку **ОК** в форме **Начало работы**, которая будет запускать макрос **Защита** для проверки пароля. Вид этой кнопки показан на рис. 1. Выполните следующие действия:

– Откройте форму **Начало работы**в режиме конструктора.

– Выполните щелчок мышью на элементе **Кнопка** на **Панели элементов** и укажите место в форме, где будет находиться кнопка, запускающая макрос для проверки пароля;

– Выполните щелчок правой клавишей мышина созданной кнопке и из контекстного меню выберите команду **Свойства**;

– В окне свойств кнопки выполните следующее: для свойства **Имя** укажите значение **ОК**, для свойства **Подпись** укажите также значение **ОК**, для свойства **Нажатие кнопки** из списка выберите имя макроса **Защита**;

– Закройте окно свойств кнопки и форму **Начало работы**.

7. Создайте макрос **Autoexec**, показанный на рис. 3. Макрос **Autoexec** обладает уникальным свойством – при запуске приложения он сразу же начинает выполняться. В этот макрос включите в указанной последовательности следующие макрокоманды: **ОткрытьФорму**, **ОстановитьМакрос**. Для макрокоманды **ОткрытьФорму** укажите аргумент **Имя формы** – Начало работы.

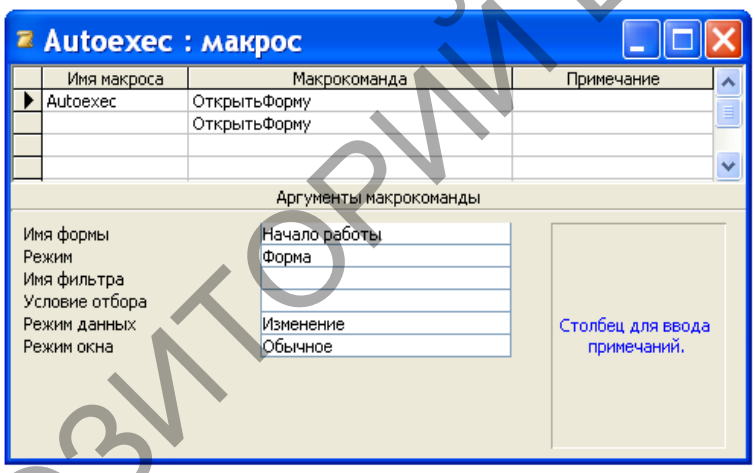


Рис. 3. Макрос в режиме конструктора.

8. Закройте базу данных **Библиотека**. Выполните запуск базы данных **Библиотека** при верном и неверном пароле.

4 РАЗДЕЛ КОНТРОЛЯ ЗНАНИЙ

4.1 Задание для контролируемой самостоятельной работы студентов

Самостоятельная работа студентов направлена на совершенствование их умений и навыков по дисциплине «Базы данных».

Цель самостоятельной работы студентов - способствование усвоению в полном объеме учебного материала дисциплины через систематизацию, планирование и контроль собственной деятельности. Преподаватель дает заданиедля самостоятельной работы и регулярно проверяет его исполнение.

Содержание и формы контролируемой самостоятельной работы студентов рекомендуется непосредственно связывать с использованием метода проектов, что позволяет реализовывать индивидуальный подход к обучению. В ходе работы над проектом студенты лучше углубляются в предметную область. В результате каждый студент создает свой собственный проект. Такая организация работы способствует развитию как информационной, так и профессиональной компетенции.

Творческоекомплексное задание

Некоторый научно-консультативный центр приглашает экспертов для чтения лекций и проведения консультаций. Менеджер центра должен иметь следующую информацию о каждом эксперте – фамилию, имя, отчество, сферу компетенции, научную степень, место работы, контактный телефон. Необходимо также иметь сведения об организациях, где работают эксперты – название организации, город, адрес, телефон. Кроме того, необходимо накапливать сведения о работе, выполненной экспертами по заявкам центра, а также об оплате их услуг.

При желании всю эту информацию можно разместить в одной большой таблице. При этом в столбцах таблицы (полях) будет собрана информация определенного типа (фамилия, сфера компетенции, научная степень, код организации...), а строки будут содержать информацию об эксперте. Однако такая таблица будет иметь существенный недостаток – многочисленные сведения будут повторяться. Кроме того, в строках, в которые помещают информацию об экспертах, работающих в одной организации, будут повторяться сведения об этой организации. Очевидно, такой способ построения таблиц ведет не только к нерациональному использованию памяти компьютера, но и к ошибкам, которые неизбежны при вводе повторявшихся данных, а это будет источником ошибок при обработке информации. Потому целесообразно разбить таблицу на несколько таблиц, которые будут содержать сведения об отдельных объектах.

В нашем проекте сведения об экспертах будем сохранять в такой таблице:

Таблица 1

Эксперты

Кодэксперта	Фамилия	Имя	Отчество	Сфераком-петенции	Научнаясстепень	Кодорга-низации	Контактный телефон

Сведения об организациях разместим в таблице следующего вида:

Таблица 2

Организации

Кодорганизации	Названиеорганизации	Город	Адрес	Телефон

Таблица для хранения данных о работах, выполненных экспертами, будет иметь вид:

Таблица 3

Работы

Кодработы	Дата	Кодэксперта	Количествочасов	Тема

Для того чтобы получать необходимые сведения из набора таблиц, необходимо создать связи, которые будут соединять эти таблицы. В нашем примере таблица 3 Работы может быть связана с таблицей 1 Эксперты через соответствующие поля этих таблиц – Код эксперта; таблица 1 Эксперты – с таблицей 2 Организации через поля, которые имеют в обеих таблицах одинаковое наименование – Код организации.

В проекте информационной системы, который вы разрабатываете, выполните следующее:

1. Создайте, используя систему управления базами данных MS Access 2003, файл базы данных с названием Менеджер.
2. В этой базе данных создайте макеты приведенных выше трех таблиц.
3. Установите связи между таблицами; связи между таблицами должны быть одного типа – один ко многим. Задайте режим обеспечения целостности данных.
4. Введите самостоятельно в каждую таблицу примерно по 20 записей, которые отображают деятельность центра.
5. Сформулируйте шесть примеров на создание запросов для автоматизации деятельности менеджера и создайте эти запросы, причем два с их должны содержать поля, которые вычисляются, два содержать параметры, а один запрос должен быть перекрестным. Имена запросов должны отображать назначение автоматизированных функций в информационной системе.

6. Разработайте дизайн главной кнопочной формы информационной системы, которая будет появляться на экране компьютера при открытии файла базы данных Менеджер. Используйте для этих целей возможности панели элементов в режиме конструктора системы управления базами данных MS Access 2007.

7. Разместите на этой форме кнопки, которые позволяют осуществлять вызов запросов, которые вы разработали для автоматизации деятельности менеджера. Предусмотрите на этой форме кнопку для завершения работы с базой данных.

8. Улучшите пользовательский интерфейс информационной системы, создав необходимые для работы с ней формы, отчеты и страницы доступа.

9. Используя возможности макрокоманд в системе управления базами данных MS Access 2007, создайте проверку пароля при входе в информационную систему «Менеджер».

10. Запустите надстройку «Анализатор быстродействия» в MS Access 2007 и выполните все ее рекомендации по оптимизации быстродействия приложения. Ликвидируйте фрагментацию файла базы данных с помощью команды «Сжать базу данных».

4.2 Контрольные вопросы

Тема 1. Основные понятия баз данных

1. Связь данных и информации. Понятие информации.
2. Оперативная, тактическая и стратегическая информация.
3. Понятие информационной системы. Информационная технология.
4. Автоматизированные информационные системы.
5. Операции с базой данных.
6. Области применения АИС.
7. Понятие базы данных. Операции с базой данных.
8. Понятие системы управления базами данных.

Тема 2. Жизненный цикл базы данных

1. Этапы жизненного цикла базы данных.
2. Предварительное планирование.
3. Проверка осуществимости. Определение требований.
4. Концептуальное проектирование.
5. Реализация. Оценка и усовершенствование.
6. Комбинированная модель жизненного цикла АИС.
7. Разработка стратегии автоматизации.
8. Оценивание реализуемости. Анализ требований.
9. Разработка технического задания.
10. Логическое проектирование.
11. Физическое проектирование.
12. Программирование.
13. Отладка и испытание.
14. Внедрение исопровождение.
15. Анализ опыта эксплуатации.
16. Технология проектирования SSADM.
17. Методическое обеспечение технологии SSADM.

Тема 3. Классификация и функции СУБД

1. Иерархическая модель данных.
2. Сетевая модель данных.
3. Реляционная модель данных.
4. Понятия: отношение, атрибут, схема отношения, кортеж.
5. Связь между элементами файла базы данных, таблицы, отношения и сущности.
6. Основные функции СУБД.
7. Определение структуры таблицы.
8. Типы связей между таблицами.

9. Простейшие операции с данными.

10. Возможности реляционных систем.

11. Представление информации в виде таблиц.

12. Использование языков высокого уровня.

13. Основные реляционные операции (выбор, проектирование и объединение).

14. Поддержка целостности, авторизации, транзакций и восстановления данных.

Тема 4. Нормализация отношений

1. Объектные и связные отношения. Понятие ключа.

2. Ссылочная целостность данных.

3. Первая нормальная форма отношения (1НФ).

4. Требования реляционной модели к отношениям.

5. Функциональная зависимость атрибутов. Вторая нормальная форма (2НФ). Приведение отношения к 2НФ.

6. Транзитивная зависимость атрибутов. Третья нормальная форма (3НФ). Приведение отношения к 3НФ.

Тема 5. Развитие технологий разработки баз данных

1. Язык структурированных запросов SQL.

2. Система управления базами данных MicrosoftAccess. Технология «Клиент-сервер». Модели технологии «Клиент-сервер».

3. Требования к современному серверу базы данных.

4. Стандарт интеграции прикладных программ OLE-2.0. Технология ODBC.

5. Объектно-ориентированный язык VBA.

Тема 6. Основные характеристики и возможности MSAccess

1. Обработка данных с помощью VBA. Построитель меню.

2. Средства отладки. Процедура обработки ошибок.

3. Программная поддержка механизма OLE. Программы-надстройки.

4. Мастера в MSAccess. Мастер по анализу таблиц.

5. Мастера по созданию форм и отчетов. Мастер автоформата.

6. Мастер подстановок. Мастера по импорту/экспорту.

7. Мастер защиты. Мастер по разделению базы данных.

8. Объекты базы данных. Таблицы. Запросы. Формы. Отчеты. Страницы доступа. Макросы. Модули.

Тема 7. Концептуальное моделирование баз данных

1. Понятие концептуальной модели.

2. Анализ предметной области.

3. Основные компоненты концептуальной модели.

4. Требования к концептуальной модели.

5. Преимущества использования ER-моделирования.

6. Использование CASE-средств для моделирования.

Тема 8. Этапы проектирования базы данных MSAccess

1. Определение цели создания базы данных.

2. Определение таблиц, которые должна содержать база данных.

3. Определение необходимых в таблице полей.

4. Определение связей между таблицами.

5. Обновление структуры базы данных.

6. Добавление данных и создание других объектов базы данных.

7. Использование средств анализа в MS Access.

8. Защита информации в базах данных.

9. Использование параметров запуска. Использование пароля. Использование защиты на уровне пользователя.

Тема 9. Создание таблиц и связей между ними

1. Создание файла пустой базы данных. Способы создания структуры таблицы.

2. Создание структуры таблицы в режиме конструктора.

3. Имена полей таблиц типы данных, свойства полей. Установка ключей.

4. Создание структуры таблицы с помощью мастера.

5. Создание структуры таблицы путем ввода данных.

6. Установка связей между таблицами. Типы связей. Целостность данных.

7. Режим обеспечения целостности данных. Режим каскадное обновление связанных полей. Режим каскадное удаление связанных записей.

8. Требования к типам данных и свойствам при связывании таблиц.

Тема 10. Работа с базой данных

1. Режимы работы с таблицей. Перевод таблицы из режима конструктора в режим таблицы и наоборот.

2. Порядок ввода данных в таблицы. Переход от одного поля к другому при вводе данных в таблицу.

3. Расширение базы данных. Добавление в базу данных новых таблиц. Добавление в схему данных новых таблиц.

Тема 11. Создание простых запросов

1. Понятие запроса. Динамический набор записей.

2. Типы запросов. Запрос на выборку.

3. Групповой запрос. Групповая операция. Запрос на изменение.

4. Перекрестный запрос.

5. Запрос SQL. Запрос с ограничением.

6. Создание запроса с помощью мастера.

7. Создание запроса в режиме конструктора.

8. Окно запроса в режиме конструктора. Механизм запросов по образцу QBE. Структура бланка QBE.

9. Работа с бланком QBE. Размещение полей. Работа со строкой вывода на экран.

10. Задание направления сортировки. Изменение порядка полей.

11. Установка оптимальной ширины столбца списка. Сохранение запроса.

12. Задание условий отбора. Использование построителя выражений для отбора данных.

13. Элементы выражения: операторы, константы, литералы, значения, функции, названия свойств, имен полей и элементов управления.

14. Классы операторов: арифметические, сравнения, логические.

15. Символы шаблона. Использование оператора Like с символами шаблона.

16. Примеры использования функций DatePart, Format, Date.

Тема 12. Просмотр и изменение динамического набора

1. Свойства запроса, поля и списка полей.

2. Просмотр, определение и изменение свойств запроса и его элементов.

3. Изменение формата поля. Изменение порядка следования полей. Вставка полей и их удаление. Удаление всех полей из бланка QBE.

4. Изменение ширины столбцов. Установка оптимальной ширины столбцов.

5. Изменение названия поля. Справочные сведения о свойстве поля. Редактирование значения свойства поля.

Тема 13. Запросы с параметрами

1. Понятие параметра. Использование диалогового окна для ввода значения параметра. Способ записи параметра в бланке QBE.

2. Создание запроса с параметрами. Использование диалогового окна с параметрами запроса.

3. Установка типа данных для параметров. Ввод значения параметра.

4. Использование ключевого слова для поиска информации.

5. Поиск по двум ключевым словам. Поиск по первым буквам искомого значения.

Тема 14. Запросы с вычисляемыми полями

1. Понятие вычисляемого поля. Создание вычисляемого поля.

2. Формат вычисляемого поля. Использование выражений в вычисляемом поле.

3. Операторы для работы со строками. Функции Left, Right, Mid.

4. Форматирование вычисляемого поля. Свойства вычисляемого поля.

5. Использование групповых операций для вычислений.

Тема 15. Перекрестный запрос

1. Понятие перекрестного запроса.

2. Создание перекрестного запроса с помощью мастера по разработке перекрестных запросов.

3. Создание перекрестного запроса в режиме конструктора.

4. Использование бланка QBE. Строка Групповая операция.

5. Назначение групповых операций: Группировка, Sum, Avg, Min, Max, Count, StDev, Var, First, Last, Выражение, Условие.

6. Назначение строки Перекрестная таблица. Требования к определению заголовков строк, заголовков столбцов и значений перекрестной таблицы.

7. Постоянные заголовки столбцов. Определение постоянных заголовков столбцов.

Тема 16. Язык конструирования запросов SQL

1. Простейший вид оператора SELECT. Предложения SELECT и FROM. Условие WHERE. Использование логических операторов And и Or.

2. Использование вычисляемых значений в предложении SELECT. Использование конструкции AS.

3. Получение итогов и других обобщающих величин (среднее, минимум, максимум и др.). Набор агрегатных функций.

4. Общая характеристика оператора SELECT. Синтаксис оператора SELECT.

5. Предложения SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY.

6. Использование функций агрегирования в предложении GROUP BY.

7. Использование логических операторов в сложных условиях. Использование операторов сравнения в выражениях. Значения выражений: TRUE, FALSE, UNKNOWN.

8. Предикаты, используемые в предложении WHERE. Интервальный предикат BETWEEN ... AND.

9. Предикат IN. Предикат проверки на неопределенное значение. Предикат подобия LIKE.

10. Агрегатные функции в предложениях SELECT, HAVING.

11. Корректирующие операторы: INSERT, UPDATE, DELETE.

Тема 17. Представление данных в виде форм

1. Понятие формы. Использование мастера для создания форм.

2. Три вида представления форм: одиночная форма, подчиненная форма и связанная форма.

3. Внешний вид формы. Форма в один столбец. Ленточная форма, Табличная или выровненная форма.

4. Подчиненная форма. Задание стиля формы.

5. Четыре вида работы с формой: основной режим работы, табличный режим, режим конструирования и режим предварительного просмотра.

6. Графическая форма. Диаграмма. Обработка диаграммы с помощью приложения MS Graph. Вызов режима конструирования формы.

Тема 18. Обработка данных с помощью отчетов

1. Назначение отчетов. Использование конструктора отчетов.

2. Разделы бланка отчета: Верхний колонтитул, Нижний колонтитул. Область данных. Заголовок и примечание отчета.

3. Создание заголовков группы и примечания группы. Размещение подписей в заголовке группы.

4. Использование в заголовке группы функций: Sum, Count, Avg.

5. Выравнивание подписей. Использование инструментов для оформления групп.Вставка номеров страниц и текущей даты.

Тема 19. Использование мастера отчетов

1. Создание отчета с помощью мастера. Автоматическое создание отчета в один столбец. Создание ленточного автоотчета.

2. Создание почтовых наклеек на конверты. Выбор размера наклейки. Создание прототипа наклейки. Сортировка наклеек.

3. Вычисление промежуточных итогов по числовым полям и общего итога для всех групп. Учет типа данных в группировке. Интервалы группирования. Стили отчета.

4. Комбинированный способ создания отчета.

Тема 20. Импорт данных

1. Импорт данных из электронных таблиц. Учет заголовков столбцов при импорте. Изменение определения полей таблицы.
2. Учет типов данных. Добавление данных в существующую таблицу.

3. Импорт текстовых файлов. Требования к подготовке импортируемого текстового файла.
4. Стандартные разделители полей. Учет формата файла. Создание разделителя полей.
5. Перемещение разделителя полей. Требования к данным при добавлении данных в существующую таблицу.
6. Импорт объектов MS Access. Типы импортируемых объектов. Импорт объектов нескольких типов.
7. Алгоритм импорта объект из другой базы данных MS Access.
8. Импорт связей между таблицами. Проверка установленных ссылок. Импорт специальных меню и панели инструментов.
9. Импорт наборов записей. Проверка ранее установленных ссылок на переименованные объекты.

Тема 21. Связывание файлов и таблиц

1. Отличие связывания от импорта. Объекты связывания.

2. Способы обработки связанных данных. Использование команды Связь с таблицами.

3. Операции, применяемые над связанными файлами и таблицами.

4. Изменение свойств в связанной таблице. Диспетчер связанных таблиц. Проверка местонахождения связанных таблиц.

Тема 22. Экспорт данных

1. Типы объектов, в которые осуществляется экспорт. Использование команды Экспорт. Экспорт данных из одной базы данных в другую.

2. Экспорт в электронную таблицу. Алгоритм экспорта таблицы, набора записей запроса на выборку или перекрестного запроса в электронную таблицу. Диалоговое окно для экспорта объекта.

3. Быстрый экспорт данных таблицы, набора записей на выборку или перекрестного запроса в электронную таблицу Microsoft Excel. Использование команды Анализ в MS Excel.

4. Экспорт в текстовый файл. Типы форматов текстовых файлов, в которые можно экспортировать данные из базы данных.

5. Алгоритм экспорта в текстовый файл. Диалоговые окна мастера экспорта текста.

Тема 23. Подготовка серийных писем

1. Установка связи данных таблицы или набора записей запроса с документом Microsoft Word.

2. Использование команды Слияние с MS Word.Назначение кнопок панели инструментов Слияние.

3. Алгоритм подготовки серийных писем. Параметр Установить связь с текстовым документом Microsoft Word.

Тема 24. Автоматизация работы с помощью макросов

1. Назначение макроса. Структура макроса.

2. Создание макроса. Окно для создания макроса. Аргументы макрокоманды Сообщение.

3. Сохранение макроса. Проверка работы макроса. Выполнение макроса в пошаговом режиме.Диалоговое окно Ошибка выполнения макрокоманды.

4. Назначение макроса Autoexec.

Тема 25. Использование кнопок в формах

1. Кнопочная форма. Создание формы в режиме конструктора.

2. Создание кнопочной формы с помощью мастера создания кнопок. Использование Панели элементов. Размещение изображения.

Тема 26. Поддержка баз данных

1. Создание резервной копии. Назначение шифрования базы данных.

2. Операции шифрования и дешифрования базы данных.

3. Повышение быстродействия приложения. Сжатие базы данных.

Тема 27. Средства защиты базы данных

1. Уровни защиты данных и администрирования доступа к ним. Безопасность приложения и файла базы данных.

2. Установка пароля для защиты базы данных от несанкционированного доступа.

3. Алгоритм удаления пароля из базы данных.

4. Защита с помощью средств поддержки рабочих групп, ведения учетных (регистрационных) записей. Задание прав владения и прав доступа.

4.3 Перечень вопросов к экзамену

1. Понятие информации. Связь данных и информации. Оперативная, тактическая и стратегическая информация.

2. Понятие информационной системы. Информационная технология. Автоматизированные информационные системы (АИС).

3. Требования к организации баз данных.

4. Понятие базы данных. Понятие системы управления базами данных (СУБД). Понятие банка данных.

5. Информационная модель базы данных. Иерархическая модель данных. Сетевая модель данных. Реляционная модель данных.

6. Понятия: отношение, атрибут, схема отношения, кортеж.

7. Связь между элементами файла базы данных, таблицы, отношения и сущности.

8. Объектные и связные отношения. Понятие ключа. Ссылочная целостность данных.

9. Первая нормальная форма отношения (1НФ). Требования реляционной модели к отношениям.

10. Функциональная зависимость атрибутов. Вторая нормальная форма (2НФ). Приведение отношения к 2НФ.

11. Транзитивная зависимость атрибутов. Третья нормальная форма (3НФ). Приведение отношения к 3НФ.

12. Каскадная и спиральные моделижизненного цикла АИС.

13. Стадии жизненного цикла АИС.

14. Технология проектирования SSADM.

15. Понятие концептуальной модели. Основные компоненты концептуальной модели.

16. Требования к концептуальной модели.

17. Инфологическое моделирование. Простые и сложные объекты. Составные, обобщенные и агрегированные объекты.

18. Изображение объектов при инфологическом моделировании.

19. Создание структуры таблицы в режиме конструктора.

20. Способы создания структуры таблицы.

21. Установка связей между таблицами. Типы связей. Требования к типам данных и свойствам при связывании таблиц.

22. Режим обеспечения целостности данных. Режим каскадное обновление связанных полей. Режим каскадное удаление связанных записей.

23. Режимы работы с таблицей. Переход из одного режима в другой.

24. Добавление в БД новых таблиц. Добавление в схему данных новых таблиц.

25. Понятие запроса. Динамический набор записей. Типы запросов.

26. Создание запроса с помощью мастера.

27. Создание запроса в режиме конструктора.

28. Задание условий отбора. Элементы выражения в условии отбора.

29. Символы шаблона. Использование оператора Like с символами шаблона.

30. Примеры использования функций DatePart, Format, Date.

31. Просмотр и изменение свойств запроса и его элементов.

32. Создание запроса с параметрами.

33. Использование ключевого слова для поиска информации. Поиск по двум ключевым словам. Поиск по первым буквам искомого значения.

34. Создание запроса с вычисляемым полем. Использование выражений в вычисляемом поле.

35. Операторы для работы со строками. Функции Left, Right, Mid.

36. Использование групповых операций для вычислений.

37. Создание перекрестного запроса с помощью мастера.

38. Создание перекрестного запроса в режиме конструктора.

39. Требования к определению заголовков строк, заголовков столбцов и значений перекрестной таблицы.

40. Назначение оператора SELECT.Простейший видоператора SELECT.

41. Понятие формы. Использование мастера для создания форм.

42. Назначение отчетов. Использование конструктора отчетов.

43. Создание заголовков группы и примечания группы. Размещение подписей в заголовке группы.

44. Создание отчета с помощью мастера. Автоматическое создание отчета в один столбец. Создание ленточного автоотчета.

45. Создание почтовых наклеек на конверты. Выбор размера наклейки. Создание прототипа наклейки. Сортировка наклеек.

46. Вычисление промежуточных итогов по числовым полям и общего итога для всех групп.

4.4 Примерные темы курсовых работ

1. Автоматизация работы ГЭК.
2. Автоматизация работы консультативного центра.
3. Автоматизация учета материальных ценностей на кафедре.
4. Автоматизация экскурсионной деятельности.
5. Автоматизированный учет распределения выпускников.
6. Автоматизированный учет успеваемости.
7. База данных «Замки Беларуси».
8. База данных вокальных коллективов Беларуси.
9. База данных вокальных коллективов Беларуси.
10. База данных выпускников кафедры.
11. База данных городского методического центра народного творчества.
12. База данных для хранения нумизматической коллекции.
13. База данных известных людей Беларуси.
14. База данных исторических достопримечательностей Минска.
15. База данных кинотеатров Минска.
16. База данных культурно-спортивных организаций города.
17. База данных культурных достопримечательностей Беларуси.
18. База данных куратора студенческой группы.
19. База данных музея музыкальных инструментов университета.
20. База данных музыкальных альбомов.
21. База данных музыкальных коллективов Беларуси.
22. База данных произведений современных писателей Беларуси.
23. База данных производственной практики студентов кафедры.
24. База данных профсоюзной организации университета.
25. База данных социальных паспортов студентов кафедры.
26. База данных театральной деятельности.
27. База данных театров Минска.
28. База данных туристического агентства.
29. База данных художественной галереи университета.
30. База данных художественных коллективов университета.
31. Информационная система "Куратор".
32. Информационная система "Общежитие".
33. Информационная система менеджера-культуролога.
34. Информационная система спортивно-культурного центра.
35. Информационная система учета посещаемости занятий.
36. Информационная система дворца внешкольной работы.

4.5 Методические рекомендации по выполнению курсовой работы

Выполнение курсовой работы является составной частью образовательного процесса по дисциплине "Базы данных". Курсовая работа – вид самостоятельной учебной работы и контроля качества обучения студента, которая носит творческий исследовательский характер и направлена на приобретение и развитие практических умений и навыков по данной учебной дисциплине и компетенций по избранной специальности.

Основными целями курсовой работы являются:

– углубление теоретических знаний и приобретение практических навыков по учебной дисциплине;

– формирование умения логически формулировать и выражать свои мысли;

– приобретение навыков проведения самостоятельного научного исследования.

Выбор темы курсовой работы производится из списка приведенной выше тематики курсовых работ. Следует обратить внимание на то, что приведенная тематика курсовых работ примерная. После выбора темы студент должен уточнить ее название у преподавателя, составить план работы и список литературы.

Структура курсовой работы должна включать введение, основную часть, заключениеи список литературы.

Во введении должна быть обоснована актуальность выбранной темы курсовой работы, показана роль автоматизированных информационных систем в управлении учреждениями культуры или образования, сформулированы объект и предмет исследования, цель и задачи курсовой работы.

Основная часть работы должна включать следующие пункты: "Предметная область", "Цель автоматизации", "Структура базы данных", "Описание полей таблиц", "Схема базы данных", "Содержание и вид запросов", "Главная кнопочная форма".

В пункте "Цель автоматизации" надо показать, какой процесс (или его функции) управления учреждением культуры или образования предполагается автоматизировать, и объяснить целесообразность автоматизации.

В пункте "Структура базы данных" следует выяснить, из каких таблиц будет состоять разрабатываемая для информационной системы база данных. При этом предполагается, что для разработки информационной системы будет использоваться система управления базами данных

реляционного типа MicrosoftAccess. Количество таблиц в базе данных должно быть не менее четырех.

Пункт "Описание полей таблиц" предназначен для указания типов и свойств полей, используемых в базе данных таблиц в терминах СУБД MicrosoftAccess. В этом пункте можно включить для каждой из таблиц копию экрана компьютера, содержащего в режиме конструктора окно таблицы.

В пункте "Схема базы данных" должно быть указано, какие поля используются в качестве ключей в таблицах, и каких типов отношения установлены между таблицами. В качестве иллюстрации целесообразно включить в работу копию экрана с окном "Схема данных". На схеме данных должны быть видны типы связей. В таблицах не должно быть скрытых имен полей данных. Между таблицами базы данных должно быть не менее трех связей типа "один-ко-многим".

Пункт "Содержание и вид запросов" должен содержать описание запросов на выдачу информации из базы данных. Иллюстрациями в этом разделе могут служить экранные копии окон запросов в режиме конструктора и в режиме таблицы. Среди запросов должно быть не менее одного перекрестного, не менее двух – с параметрами, не менее двух – с вычисляемыми полями.

В пункте "Главная кнопочная форма" должен быть приведен скриншот этой формы и показана работоспособность базы данных.

В заключении даются основные выводы по проделанной работе, рассматриваются организационные и технические вопросы внедрения работы в учреждениях культуры или образования, возможные направления доработки информационной системы или расширения ее функциональных возможностей. Выводы должны соответствовать предметной области.

Оформление курсовой работы

Курсовая работа должна быть аккуратно оформлена. Структурно она должна содержать оглавление, описанные выше пункты, список использованной литературы и, возможно, приложения. Список использованной литературы должен быть оформлен в соответствии с инструкцией ВАК Республики Беларусь.

Объем курсовой работы должен быть не менее 25 страниц машинописного текста (размер шрифта – 14 пунктов, тип шрифта – TimesNewRoman, интервал – полуторный, поля: левое – 3 см, остальные – 2 см).

Курсовая работа должна быть защищена до сдачи экзамена.

4.6 Примеры схем данных для разработки информационных систем

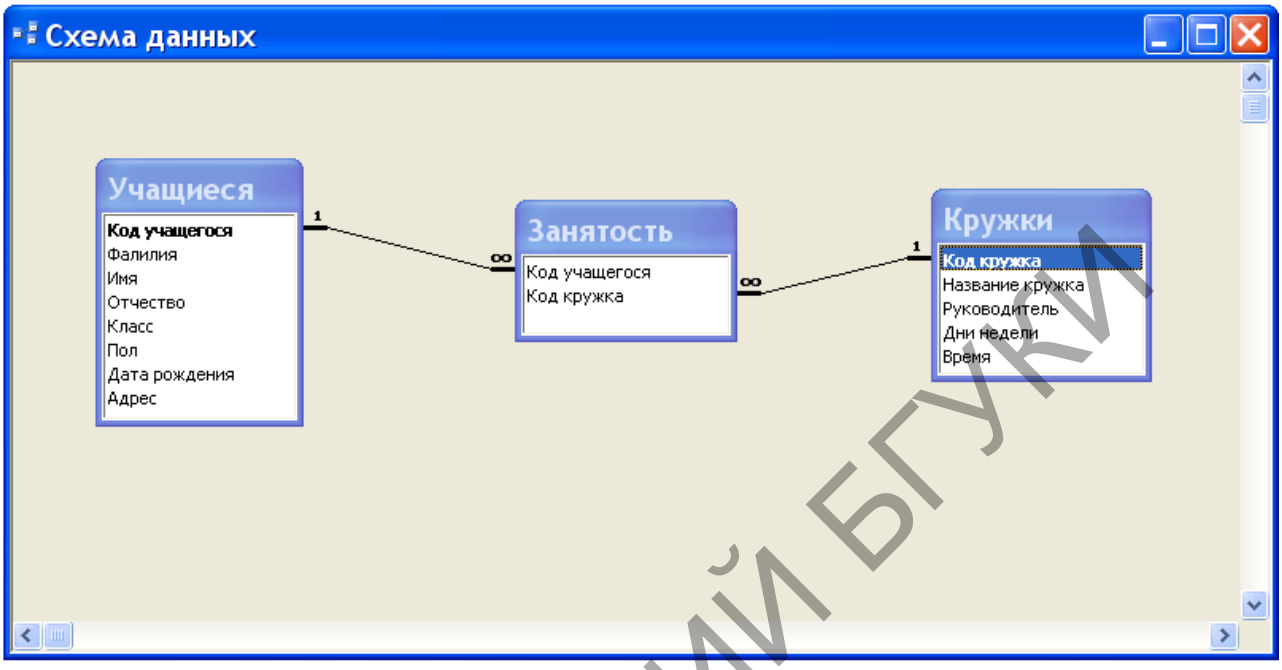


Схема данных **Занятость учащихся**.

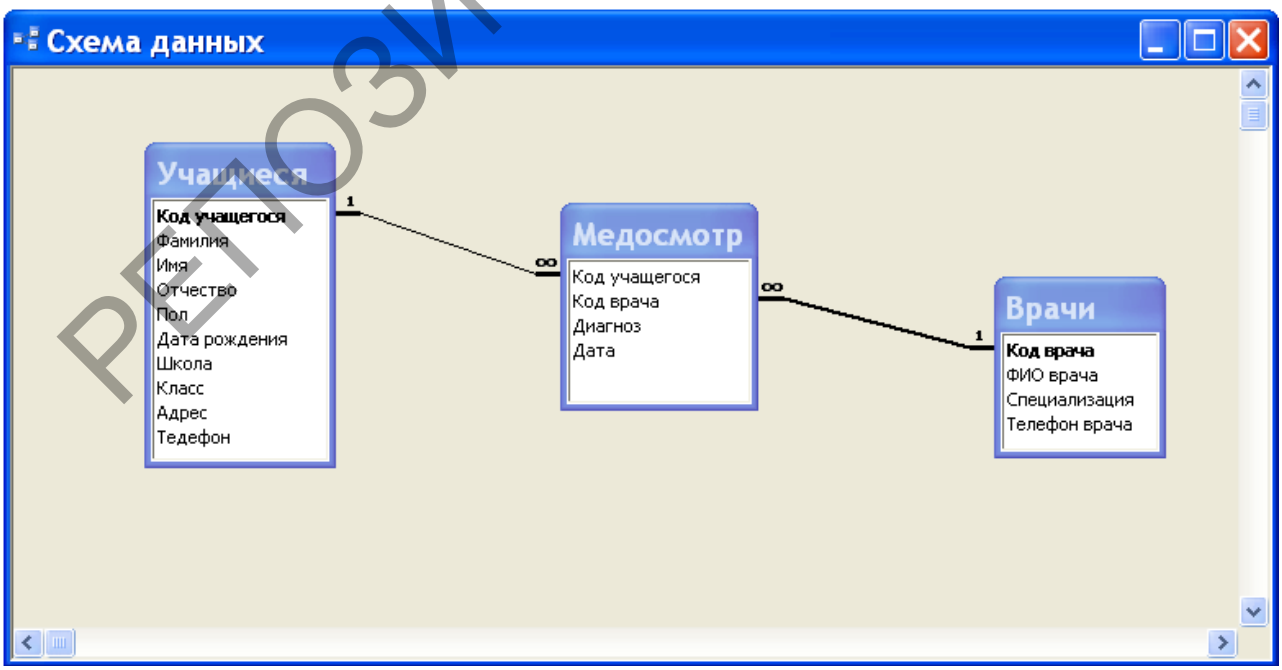


Схема данных **Медосмотр учащихся.**

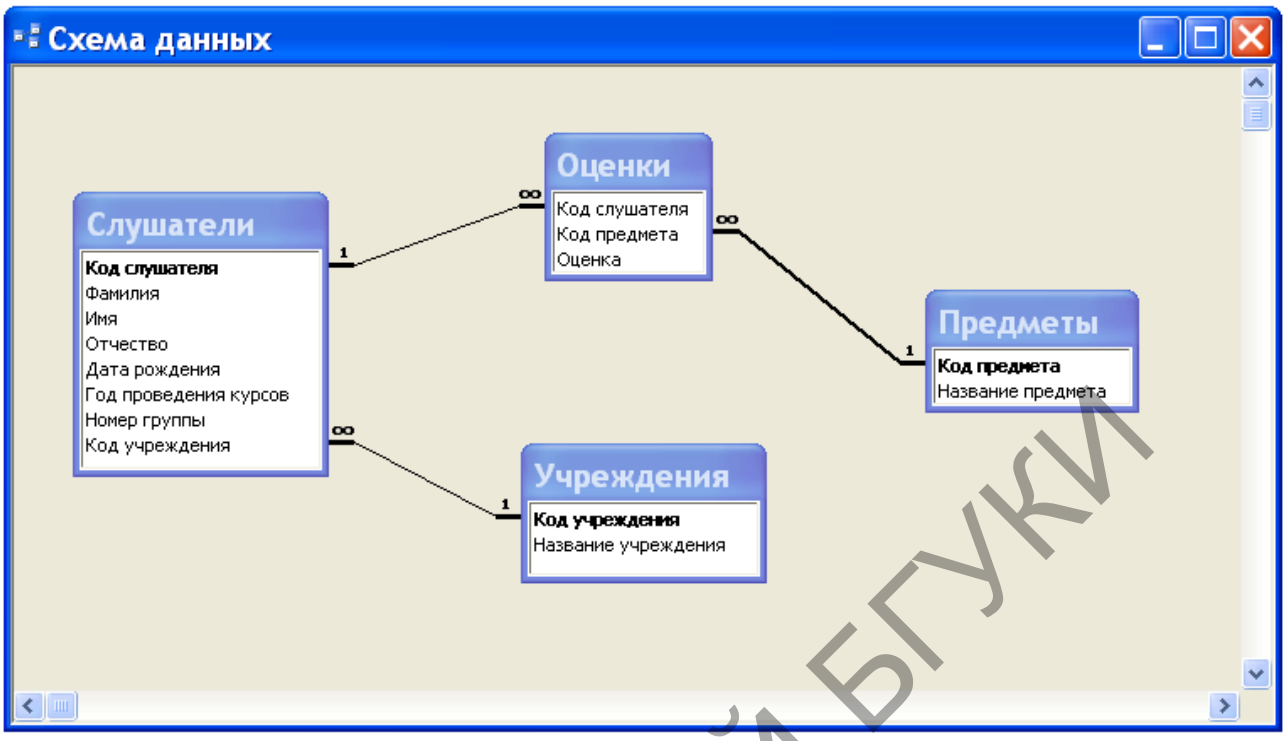


Рис. 4. Схема данных **Результаты обучения слушателей**.

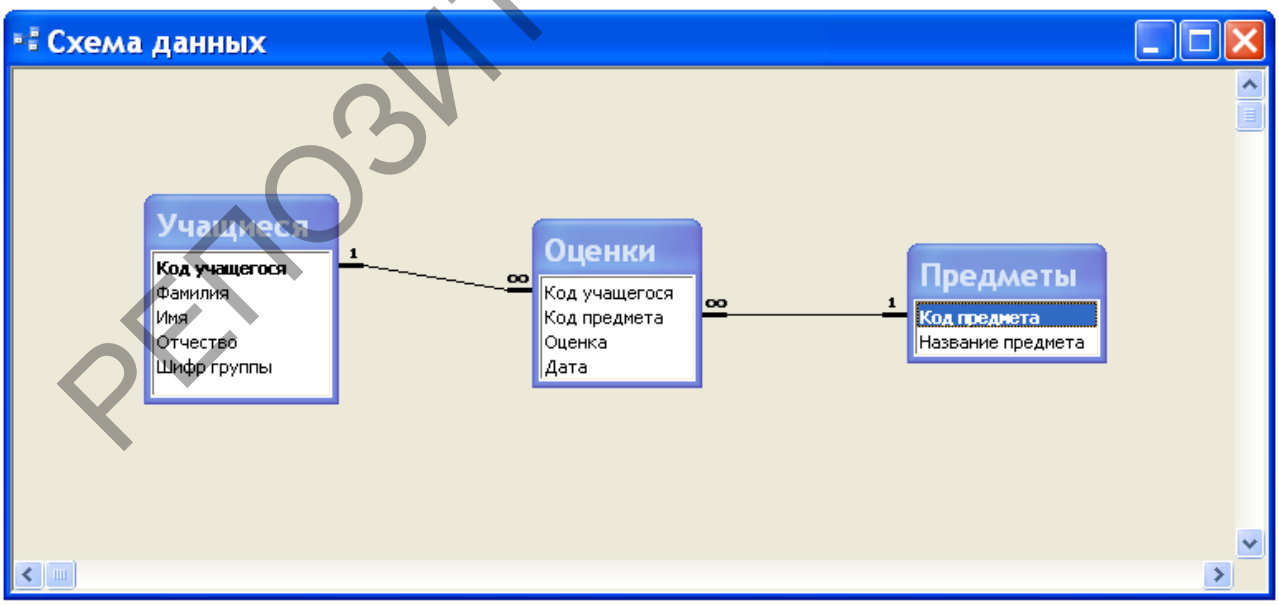


Схема данных **Текущая успеваемость.**

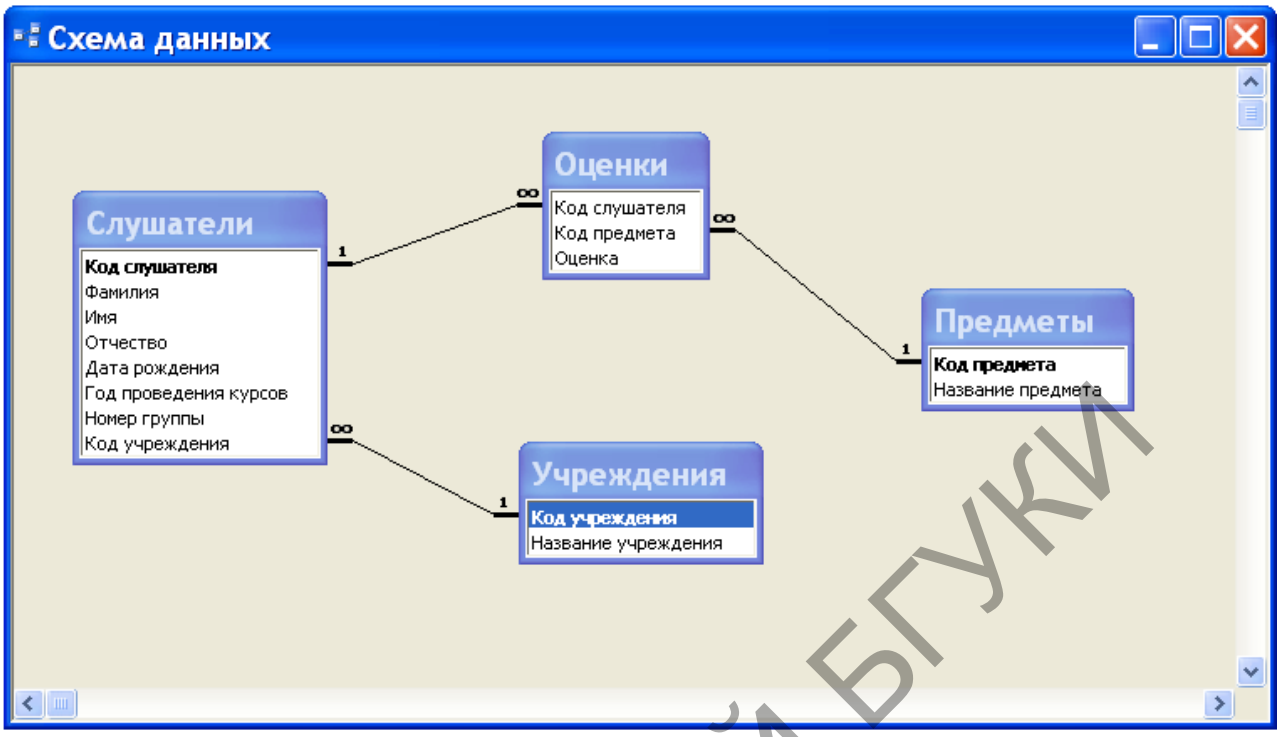


Схема данных **Успеваемость слушателей курсов**.

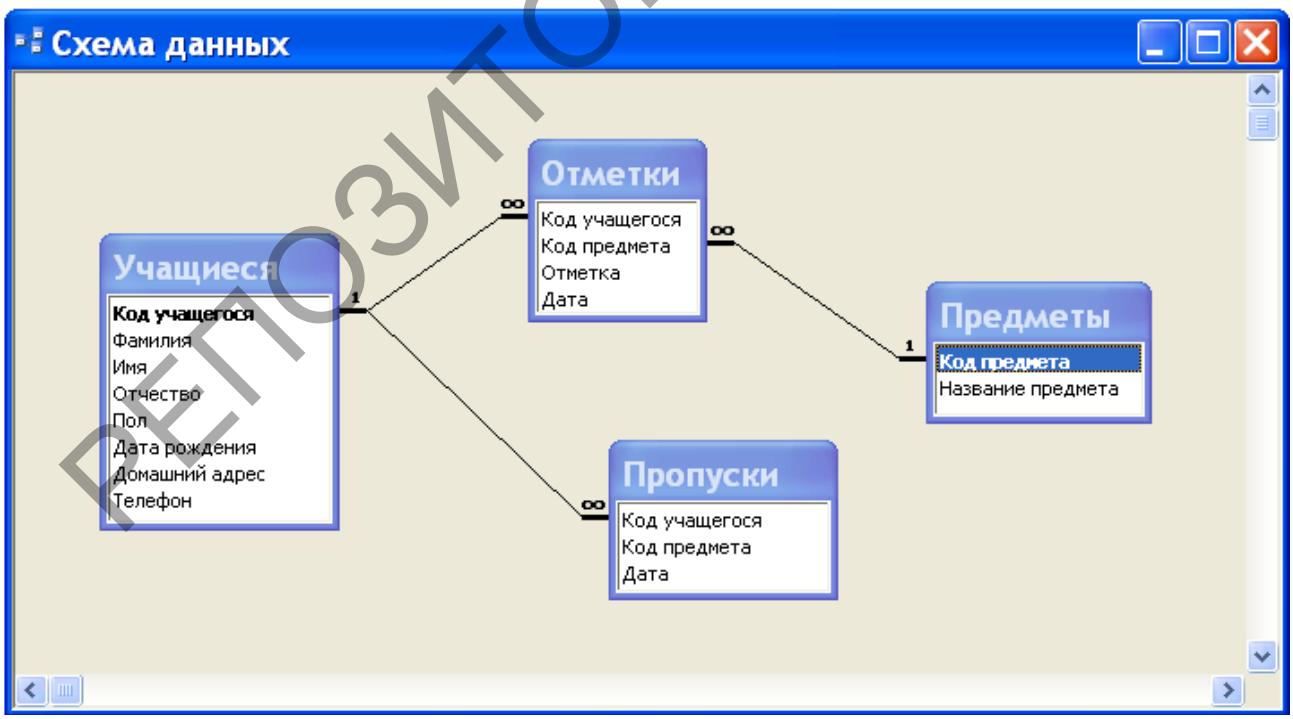


Схема данных **Успеваемость и посещаемость**.

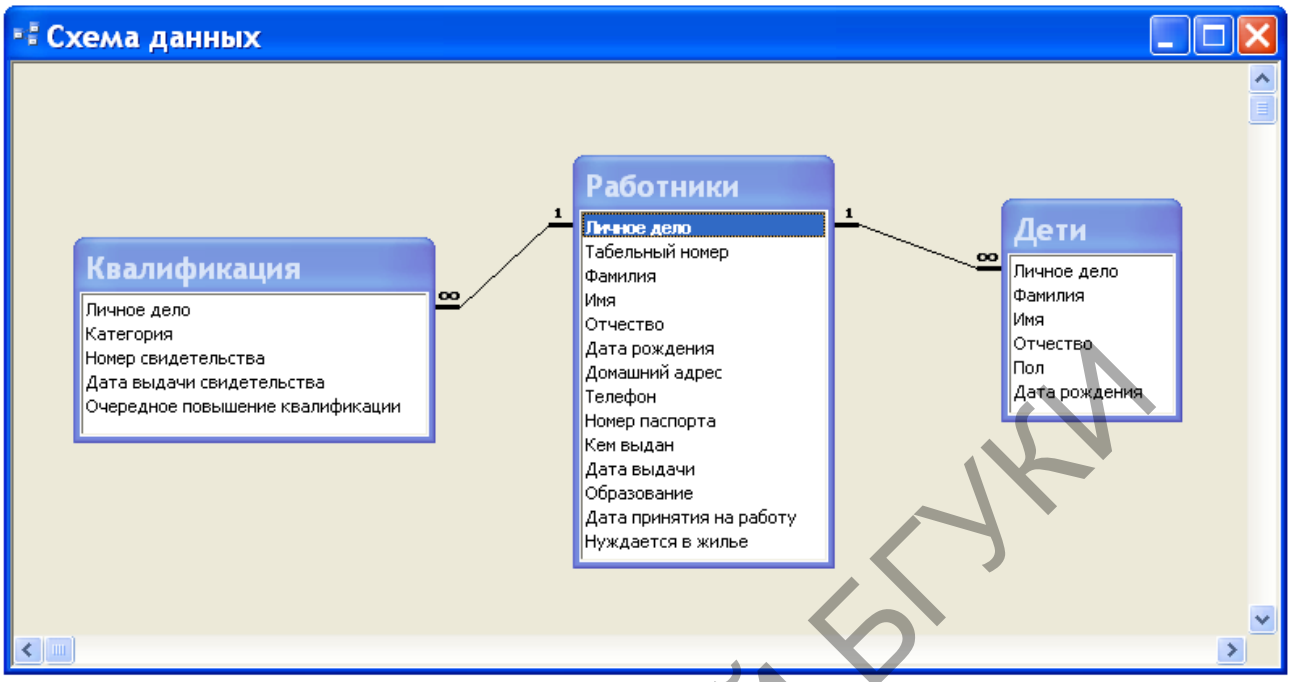


Схема данных **Учет кадров**.

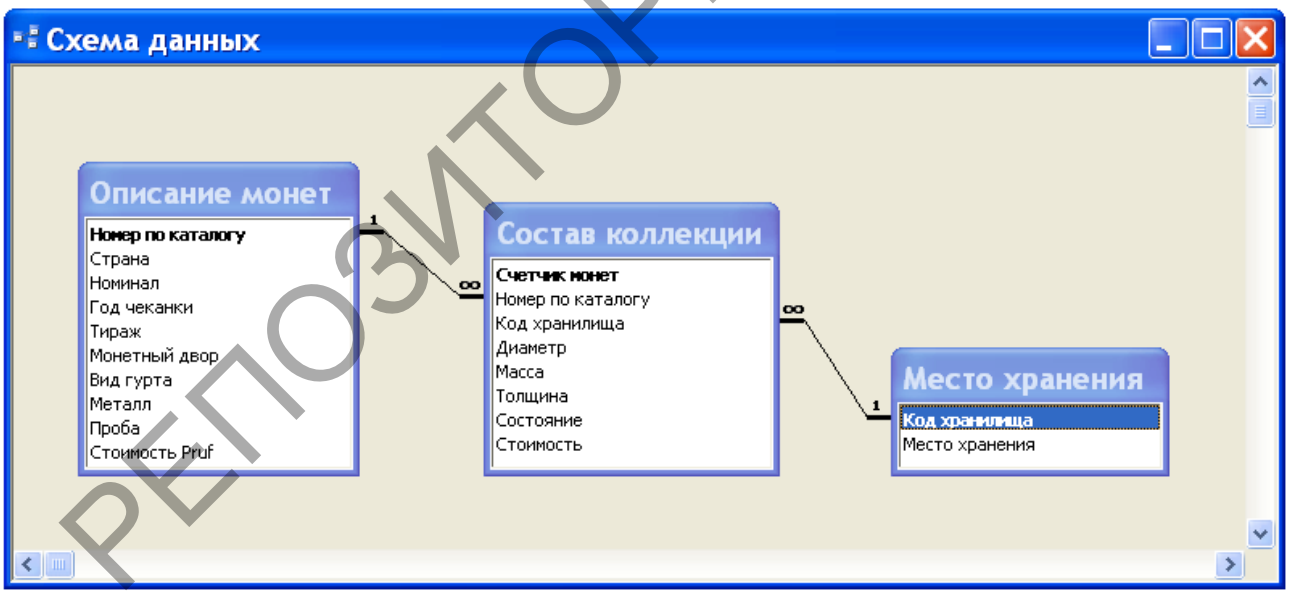


Схема данных **Коллекция монет**.

4.7 Критерии оценки результатов учебной деятельности студентов

Контроль знаний, умений и навыков учащихся является важной составной частью образовательного процесса. Целью контроля является определение качества усвоения обучаемыми программного материала, диагностирование и корректирование их знаний и умений, воспитание ответственности к учебной работе. Для выяснения роли контроля в образовательном процессе рассматривают его наиболее значимые функции: обучающую, диагностическую, прогностическую, развивающую, ориентирующую и воспитывающую.

Обучающая функция контроля заключается в совершенствовании знаний и умений, их систематизация.

Сущность *диагностической функции* контроля состоит в получении информации об ошибках, недочетах и пробелах в знаниях и умениях обучаемых и порождающих их причинах.

Прогностическая функция контроля служит опережающей информацией об образовательном процессе. В результате такого контроля получают основания для прогноза о ходе определенного отрезка учебного процесса: достаточно ли сформированы конкретные знания, умения и навыки для усвоения последующей порции учебного материала.

Развивающая функция контроля состоит в стимулировании познавательной активности студентов, в развитии их творческих сил и способностей.

Сущность *ориентирующей функции* контроля – в получении информации о степени достижения цели обучения отдельным студентом и группой в целом – как усвоен и как глубоко изучен учебный материал.

Сущность *воспитывающей функции* заключается в воспитании у студентов ответственного отношения к учению, дисциплины, аккуратности, честности.

Контроль должен быть целенаправленным, объективным, всесторонним, регулярным.

В соответствии с формами обучения на практике выделяются три формы контроля: *индивидуальная, групповая и фронтальная.*

При индивидуальном контроле каждый студент получает свое задание, которое он должен выполнять без посторонней помощи. Эта форма целесообразна в том случае, если требуется выяснять индивидуальные знания, способности и возможности отдельных студентов.

При групповом контроле группа временно делится на несколько подгрупп (от 2 до 5студентов) и каждой подгруппе дается проверочное задание. В зависимости от цели контроля подгруппам предлагают одинаковые задания или дифференцированные (проверяют результаты письменно-графического задания, которое студенты выполняют по двое, или практического, выполняемого каждой четверкой студентов, или проверяют точность, скорость и качество выполнения конкретного задания по подгруппам. Групповую форму организации контроля применяют при повторении с целью обобщения и систематизации учебного материала, при выделении приемов и методов решения задач, при акцентировании внимания студентов на наиболее рациональных способах выполнения заданий.

При фронтальном контроле задания предлагаются всейгруппе. В процессе этой проверки изучается правильность восприятия и понимания учебного материала, качество словесного, графического предметного оформления, степень закрепления в памяти.

Текущий контроль проводится в течение всего обучения, на каждом занятии, причем почти на каждом его этапе. Оценивание при текущем контроле оказывает огромное воспитательное воздействие.

Для выявления и исключения пробелов в знаниях студентов рекомендуется использовать следующие средства:

1) фронтальный опрос на лекциях, лабораторных и семинарских занятиях;

2) критериально-ориентированные тесты для контроля теоретических знаний современных методов разработки баз данных в сфере культуры и образования;

3) выполнение тестовых заданий с произвольной формой ответа для контроля умения анализировать и грамотно излагать и формулировать свои соображения и выводы в данной предметной области;

4) выполнение творческих заданий, которые предполагают эвристическую деятельность и поиск неформальных решений.

5 ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ

5.1 Учебнаяпрограмма

1. Гляков, П.В. Информационные процессы и системы. Часть 3. Базы данных:учебная программа для специальности 1-21 04 01 Культурология (по направлениям) направления специальности 1-21 04 01-02 Культурология (прикладная)специализации 1-21 04 01-02 04 Информационные системы в культуре / П.В. Гляков. – Минск: Бел.гос. ун-т культуры и искусств, 2013. –24 с.

1.2 Учебно-методическая карта учебной дисциплины для дневной формы получения высшего образования

Номер раздела, темы	Наименование тем, разделов	Лекции	Лабораторные занятия	Количество часов УСР	Форма контроля знаний
Раздел 1	**Введение в базы данных**				
Тема 1	Основные понятия баз данных	2	–	–	
Тема 2	Жизненный цикл базы данных	2	–	–	
Тема 3	Классификация и функции СУБД	2	–	–	
Тема 4	Нормализация отношений	2	–	–	
Тема 5	Развитие технологий разработки баз данных	–	–	2	Отчет
Тема 6	Основные характеристики и возможности MSAccess	2	–	–	
Раздел 2	**Проектирование баз данных**				
Тема 7	Инфологическое моделирование баз данных	–	–	2	Отчет
Тема 8	Этапы проектирования базы данных MSAccess	2	–	–	
Раздел 3	**Создание базы данных**				
Тема 9	Создание таблиц и связей между ними	–	2	–	
Тема 10	Работа с базой данных	–	2	–	
Раздел 4	**Манипулирование данными**				
Тема 11	Создание простых запросов	–	2	–	
Тема 12	Просмотр и изменение динамического набора	–	2	–	
Тема 13	Запросы с параметрами	–	2	–	
Тема 14	Запросы с вычисляемыми полями	–	2	–	
Тема 15	Перекрестный запрос	–	2	–	
Тема 16	Язык конструирования запросов SQL	–	2	–	
Раздел 5	**Представление данных**				
Тема 17	Представление данных в виде форм	–	2	–	
Тема 18	Обработка данных с помощью отчетов	–	2	–	
Тема 19	Использование мастера отчетов	–	2	–	
Раздел 6	**Импорт, экспорт и связывание данных**				

Тема 20	Импорт данных	–	2	–	
Тема 21	Связывание файлов и таблиц	–	2	–	
Тема 22	Экспорт данных	–	2	–	
Тема 23	Подготовка серийных писем	–	–	2	Отчет
Раздел 7	**Пользовательский интерфейс**				
Тема 24	Автоматизация работы с помощью макросов	–	–	2	Отчет
Тема 25	Использование кнопок в формах	–	–	2	Отчет
Раздел 8	**Управление базами данных**				
Тема 26	Поддержка баз данных	–	–	2	Отчет
Тема 27	Средства защиты базы данных	–	–	2	Отчет
	Всего	12	28	14	

5.3 Учебно-методическая карта учебной дисциплины для заочной формы получения высшего образования

Номер раздела, темы	Наименование тем, разделов	Лекции	Лабораторные занятия	Количество часов УСР	Форма контроля знаний
Раздел 1	**Введение в базы данных**				
Тема 1	Основные понятия баз данных	2	–	–	
Тема 2	Жизненный цикл базы данных	2	–	–	
Тема 3	Классификация и функции СУБД	2	–	–	
Тема 4	Нормализация отношений	2	–	–	
Тема 5	Развитие технологий разработки баз данных	–	–	2	Отчет
Тема 6	Основные характеристики и возможности MSAccess	2	–	–	
Раздел 2	**Проектирование баз данных**				
Тема 7	Инфологическое моделирование баз данных	–	–	2	Отчет
Тема 8	Этапы проектирования базы данных MSAccess	–	–	2	Отчет
Раздел 3	**Создание базы данных**				
Тема 9	Создание таблиц и связей между ними	–	2	–	
Тема 10	Работа с базой данных	–	2	–	
Раздел 4	**Манипулирование данными**				
Тема 11	Создание простых запросов	–	2	–	
Тема 12	Просмотр и изменение динамического набора	–	2	–	
Тема 13	Запросы с параметрами	–	2	–	
Тема 14	Запросы с вычисляемыми полями	–	–	2	Отчет
Тема 15	Перекрестный запрос	–	–	2	Отчет
Тема 16	Язык конструирования запросов SQL	–	–	2	Отчет
Раздел 5	**Представление данных**				
Тема 17	Представление данных в виде форм	–	2	–	
Тема 18	Обработка данных с помощью отчетов	–	2	–	
Тема 19	Использование мастера отчетов	–	–	2	Отчет
Раздел 6	**Импорт, экспорт и связывание данных**				

Тема 20	Импорт данных	–	–	2	Отчет
Тема 21	Связывание файлов и таблиц	–	–	2	Отчет
Тема 22	Экспорт данных	–	–	2	Отчет
Тема 23	Подготовка серийных писем	–	–	2	Отчет
Раздел 7	**Пользовательский интерфейс**				
Тема 24	Автоматизация работы с помощью макросов	–	–	2	Отчет
Тема 25	Использование кнопок в формах	–	–	2	Отчет
Раздел 8	**Управление базами данных**				
Тема 26	Поддержка баз данных	–	–	2	Отчет
Тема 27	Средства защиты базы данных	–	–	2	Отчет
	Всего	10	14	30	

5.4 Список основной литературы

1. Бекаревич, Ю. Microsoft Access за 21 занятие для студента / Ю. Бондаренко. – СПб. : БХВ-Петербург, 2005. – 544 с.

2. Бондаренко, М. Microsoft Office 2003 в теории и практике / М. Бондаренко, С.Бондаренко. – Минск : Новое знание, 2004. – 560 с.

3. Вейскас, Дж. Эффективная работа: Microsoft Office Access 2003 / Дж. Вейскас. – СПб. : Питер, 2005. – 1168 с.

4. Гляков, П.В. Система управления базами данных Access 2.0 : учеб.пособие / П.В. Гляков, С.Н. Карачун. – Минск : РИПО, 1998. – 100 с.

5. Гляков, П.В. Импорт, экспорт и связывание данных в Microsoft Access : метод.рекомендации / П.В. Гляков. – Минск : РИПО, 2005. – 34 с.

6. Гляков, П.В. Базы данных: компьютерный практикум : учеб.пособие / П.В. Гляков. – Минск : БГУКИ, 2008. – 130 с.

7. Гляков П.В. Формирование профессиональных компетенций на лабораторных занятиях по базам данных / П.В. Гляков // Компетентностный подход в высшем образовании: проблемы и перспективы : материалы науч.-метод. конф., Минск, 4 февр. 2016 г. / М-во культуры Респ. Беларусь, Белорус. гос. ун-т куль- туры и искусств ; редкол.: Ю. П. Бондарь (пред.) [и др.]. – Минск : БГУКИ, 2016. – 322 с. – С. 153-158.

8. Гринчук, С.Н. Система управления базами данных Microsoft Access / С.Н. Гринчук, И.А. Дюба. – Минск : АПО, 2006. – 187 с.

9. Диго, С.М. Базы данных: проектирование и использование: учебник / С.М. Диго. – М. : Финансы и статистика, 2005. – 592 с.

10. Троян, Г.М. Основы компьютерных технологий в образовании. В 4 ч. Ч.3. Технологии обработки данных : учеб.пособие / Г.М. Троян, Е.М. Зайцева, С.Н. Гринчук [и др.]. Минск : РИВШ БГУ, 2002. – 212 с.

5.5 Список дополнительной литературы

1. Гончаров, А.Ю. Access 2007. Самоучитель с примерами / А.Ю. Гончаров. – М. :Кудиц-образ, 2008. – 296 с.

2. Кошелев, В.Е. Access 2007. Эффективное использование / В.Е. Кошелев. – М. : БИНОМ, 2008. – 592 с.

3. Кузнецов, С.Д. Базы данных. Языки и модели / С.Д. Кузнецов. – М. : БИНОМ, 2008. – 720 с.

4. Кузнецов, С.Д. Основы баз данных : Курс лекций : учеб.пособие для вузов / С.Д. Кузнецов. – М. : Интернет-Университет, 2007. – 484 с.

5. Мак-Федрис, П. Формы, отчеты и запросы в MicrosoftAccess 2003 / П. Мак-Федрис. – М. : Вильямс, 2005. – 416 с.

6. Microsoft Office Access 2003. Русскаяверсия. Шаг за шагом :практ. пособ. / М. : СП ЭКОМ, 2004. – 432 с.

7. О'Хара, Ш. Абсолютно ясно о MicrosoftAccess 2003 / Ш. О’Хара. – М. : Триумф, 2005. – 240 с.

8. Сенов, А. Access 2003. Практическая разработка баз данных : учеб.курс / А. Сенов. – СПб. : Питер, 2005. – 256 с.

9. Туманов, В.Е. Основы проектирования реляционных баз данных / В.Е. Туманов. – М. : Интернет-Университет, 2007. – 420 с.

10. Харитонова, И. Самоучитель OfficeAccess 2003 / И. Харитонов. – СПб. : Питер, 2004. – 464 с.

11. Шевченко, Н.А. Access 2003. Искусство создания базы данных / Н.А. Шевченко. – М. : НТ Пресс, 2005. – 160 с.