Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра автоматического управления

**М.А. Крупская**

# ***ИнформационнОЕ оБЕСПЕЧЕНИЕ***

# ***систем управления (Часть 1)***

Методическое пособие

к лабораторным работам для студентов специальности

1 53 01 07 «Информационные технологии и управление

в технических системах»

всех форм обучения

Минск 2020

УДК 681.5 (075.8)

ББК 32.965 я 73

А 72

Р е ц е н з е н т:

зав. кафедрой теоретических основ электротехники БГУИР,

д-р техн. наук, проф. Л.Ю. Шилин

**Крупская М.А.**

#### А 72 Информационное обеспечение систем управления: Метод. пособие к лаб. работам для студентов спец. 1-53 01 07 «Информационные технологии и управление в технических системах» всех форм обучения / М.А. Крупская. – Мн.: БГУИР, 2020. – 81 с.

#### ISBN 985-444-503-8.

В пособии приведены описание и порядок выполнения лабораторных работ по курсу «Информационное обеспечение систем управления». Содержание работ определено рабочей программой курса в соответствии с учебным планом специальности 1 53 01 07 «Информационные технологии и управление в технических системах».

В каждой лабораторной работе сформулирована цель, изложены краткие теоретические сведения, разработаны задания для самостоятельного выполнения, приведены примеры выполнения заданий. Лабораторные работы выполняются в системе управления базами данных Microsoft Ассеss.

**УДК 681.5 (075.8)**

**ББК 32.965 я 73**

© Крупская М.А., 2014

**ISBN 985-444-503-8** © БГУИР, 2011

# **СОДЕРЖАНИЕ**

[Введение 4](#_Toc378696037)

[Лабораторная работа № 1 Знакомство с СУБД MS Access. Создание диаграммы предметной области 5](#_Toc378696038)

[Лабораторная работа № 2 Представление данных с помощью модели «Сущность-связь». Работа с таблицами MS Access 11](#_Toc378696039)

[Лабораторная работа №3 Создание простых запросов и запросов на изменение 22](#_Toc378696040)

[Лабораторная работа № 4 Создание сложных запросов. Работа с отчетами 32](#_Toc378696041)

[Лабораторная работа № 5 Создание форм. Привязка информационных полей через связи между таблицами 41](#_Toc378696042)

[Лабораторная работа № 6 Написание процедур обработки событий на VBA 50](#_Toc378696043)

[Лабораторная работа № 7 Проектирование базы данных по индивидуальному заданию 64](#_Toc378696044)

[Список использованных источников 69](#_Toc378696045)

[Приложение 1 70](#_Toc378696046)

[Приложение 2 78](#_Toc378696047)

[Приложение 3 80](#_Toc378696048)

# **Введение**

Хранение информации – одна из важнейших функций технических систем. Самым распространенным средством хранения являются базы данных (БД). БД – это организованная в соответствии с определёнными правилами и поддерживаемая в памяти компьютера совокупность данных, характеризующая актуальное состояние некоторой предметной области и используемая для удовлетворения информационных потребностей пользователей.

**Структура реляционных БД (РБД)*.*** Большинство РБД имеют табличную структуру, в которой адрес данных определяется пересечением строк и столбцов. В таблицах РБД столбцы называются полями, а строки − записями. Поля образуют структуру таблиц БД, а записи составляют информацию, которая в ней содержится. Для того чтобы легко усвоить понятие структуры БД, надо представить себе пустую базу, в которой пока еще нет никаких данных. Несмотря на то, что данных в базе нет, информация в ней все-таки есть – это структура базы, т.е. набор связанных таблиц с определенными полями.

**Простейшие БД.** Простейшие базы можно создавать, не прибегая к специальным программным средствам. Чтобы файл считался базой данных, информация в нем должна иметь структуру (поля и записи) и быть отформатирована так, чтобы содержимое соседних полей легко различалось. Простейшие базы можно создавать в текстовом редакторе, т.е. обычный текстовый файл при определенном форматировании тоже может считаться базой данных. Существует, по крайней мере, два формата текстовых БД: с заданным разделителем; с фиксированной длиной поля. Несмотря на «примитивность» таких текстовых БД, мощные системы управления базами данных (СУБД) позволяют импортировать подобные файлы и преобразовывать их в настоящие БД.

**СУБД MS Access**. СУБД – это специализированная программа (чаще комплекс программ), предназначенная для манипулирования базой данных. В мире существует немало различных систем управления базами данных, например Cache, DB2, Firebird, Informix, InterBase, MSSQLServer, MySQL, Oracle, PostgreSQL, Sybase и т.д. Некоторые из БД на самом деле являются не законченными продуктами, а специализированными языками программирования, с помощью которых каждый, освоивший данный язык, может сам создавать такие структуры, какие ему удобны, и вводить в них необходимые элементы управления, например Clipper, Paradox, FoxPro и др.

Необходимость программировать всегда сдерживала широкое внедрение БД в малом бизнесе. Только крупные предприятия могли позволить себе сделать заказ на программирование специализированной системы «под себя». Малым предприятиям зачастую не по силам было не только решить, но даже и правильно сформулировать техническое задание на проектирование. Положение изменилось с появлением в составе пакета Microsoft Office СУБД Access. Обычные пользователи получили удобное средство для создания и эксплуатации достаточно мощных БД без необходимости что-либо программировать. В то же время возможность программирования не исключается, при желании систему можно развивать и настраивать на языке программирования Visual Basic for Application (VBA). Достоинством MS Access также является его интегрированность с другими программами пакета MS Office.

# Лабораторная работа № 1 Знакомство с СУБД MS Access. Создание диаграммы предметной области

## **Цель работы:** познакомиться с интерфейсом СУБД MS Access, получить начальные навыки работы с ее объектами путем создания БД при помощи локального шаблона, научиться создавать диаграммы потоков данных по заданной предметной области.

### **Краткие теоретические сведения**

**Объекты MS Access.** Основными объектами БД являются таблицы, в которых хранятся данные. РБД может иметь много взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного типа. Строка таблицы содержит данные об одном объекте (например, товаре, клиенте), а столбцы таблицы (атрибуты) описывают различные характеристики этих объектов (например, наименование, код товара, сведения о клиенте). Записи, т. е. строки таблицы, имеют одинаковую структуру – они состоят из полей, хранящих атрибуты объекта. Каждое поле, т. е. столбец, описывает только одну характеристику объекта и имеет строго определенный тип данных. Все записи имеют одни и те же поля, только в них отображаются различные информационные свойства объекта.

Запрос – это объект БД, который служит для извлечения данных из таблиц и предоставления их пользователю в удобном виде. Особенность запросов состоит в том, что они черпают данные из базовых таблиц и создают на их основе временную таблицу. Применение запросов позволяет избежать дублирования данных в таблицах и обеспечивает максимальную гибкость при поиске и отображении данных в БД. С помощью запросов данные упорядочивают, фильтруют, отбирают, изменяют, объединяют, удаляют, т.е. обрабатывают.

Форма – это объект БД, предназначенный для ввода и отображения информации. Формы позволяют выполнить проверку корректности данных при вводе, проводить вычисления, обеспечивают доступ к данным в связанных таблицах с помощью подчиненных форм.

Отчет – это объект БД, который предназначен для вывода информации из БД, прежде всего на принтер. Отчеты позволяют выбрать из таблиц нужную пользователю информацию, оформить ее в виде документа, перед выводом на печать просмотреть на экране. Источником данных для отчета может служить таблица или запрос. Кроме данных, полученных из таблиц, в отчете могут отображаться вычисляемые поля, например, итоговые суммы.

Макросы – это макрокоманды. Если какие-то операции с базой производятся особенно часто, имеет смысл сгруппировать несколько команд в один макрос и назначить его выделенной комбинации клавиш.

Модули – это программные процедуры, написанные на языке VBA. Если стандартных средств MS Access не хватает для удовлетворения требований заказчика, программист может расширить возможности системы, написав для этого необходимые модули или используя готовые.

**Режимы работы с MS Access***.*С организационной точки зрения в работе с любой БД есть два разных режима: проектировочный и эксплуатационный (пользовательский). В проектировочном режиме владелец схемы БД имеет право создавать в ней новые объекты (например, таблицы), задавать их структуру, менять свойства полей, устанавливать необходимые связи. Он работает со структурой базы и, как правило, имеет полный доступ к базе. У одной базы может быть один, два или несколько владельцев - разработчиков. Пользователи базы наполняют ее информацией с помощью форм, обрабатывают и отбирают данные с помощью запросов и получает результаты в виде отчетов. У одной БД могут быть миллионы пользователей, и, конечно, доступ к структуре базы для них закрыт.

При работе с объектами MS Access можно выполнять следующие действия:

1. Открыть выбранный объект для просмотра. Если это таблица, то ее можно просмотреть, внести новые записи или изменить те, что были внесены ранее.

2. Открыть выбранный объект в режиме конструктора. Если это таблица, в нее можно вводить новые поля или изменять свойства существующих. Если это форма, в ней можно изменять или создавать элементы управления.

3. Создать новый объект БД. Таблицы, запросы, формы и отчеты можно создавать разными способами:

* автоматически;
* вручную с помощью Конструктора;
* вручную с помощью Мастера.

**Создание диаграммы предметной области.** Прежде чем приступать к созданию БД, разработчик должен сформировать понятия о предметах, фактах и событиях, которыми будет оперировать проектируемая система.

В теории проектирования информационных систем предметную область принято рассматривать в виде трех представлений:

1. как она реально существует;
2. как ее воспринимает человек (проектировщик БД);
3. как она может быть описана с помощью символов.

Эти представления определили и три основных этапа проектирования БД. Первым этапом проектирования любой БД является **концептуальное проектирование**.

Концептуальное проектирование включает сбор, анализ и редактирование требований к данным. Для этого осуществляются следующие мероприятия:

* обследование предметной области, изучение информационной структуры объекта;
* выявление всех фрагментов реальности, каждый из которых характеризуется пользовательским представлением, информационными объектами и связями между ними, происходящими в них или над ними процессами;
* моделирование и интеграция всех представлений.

Под ***предметной областью*** в этом случае понимают часть реального мира, подлежащую изучению с целью организации управления и, в конечном счете, создания информационной системы.

Можно выделить два способа разработки концептуальной модели – это создание диаграммы потоков данных и разработка модели «сущность-связь».

Вторым этапом является **логическое проектирование** – преобразование концептуальных или семантических требований к данным в конкретные логические структуры данных. На этом этапе часто моделируют базы данных применительно к различным СУБД и проводят сравнительный анализ моделей.

Третий этап – **физическое проектирование** – определение особенностей хранения данных, методов доступа и т.д.

**Построение диаграммы потоков данных.** Диаграмма потоков данных (DFD – Data Flow Diagramm) предназначена для представления семантики предметной области на самом высоком уровне абстракции. Это означает, что устранена или минимизирована необходимость использовать понятия «низкого уровня», связанные со спецификой физического представления и хранения данных.

Диаграммы потоков данных строятся из следующих элементов, представленных в табл. 1.

Таблица 1

Условные обозначения нотации Йордана – Де Марко

|  |  |  |
| --- | --- | --- |
| Элемент | Описание | Обозначение |
| Функция | Действие, выполняемое моделируемой системой |  |
| Поток  данных | Объект, над которым выполняется действие. Может быть информационным (логическим) или управляющим. Управляющие потоки обозначаются пунктирной линией со стрелкой. |  |
| Хранилище  данных | Структура для хранения информационных объектов |  |
| Внешняя  сущность | Внешний по отношению к системе объект, обменивающийся с нею потоками данных |  |

Такой тип обозначений элементов DFD получил название «нотации Йордана – Де Марко», по именам разработавших его специалистов.

Функции, хранилища и внешние сущности на диаграмме потоков данных связываются дугами, представляющими потоки данных. Дуги могут разветвляться или сливаться, что означает соответственно разделение потоков данных на части либо их слияние. При интерпретации диаграммы используются следующие правила:

1. Функции преобразуют входящие потоки данных в выходящие.
2. Хранилища данных не изменяют потоки данных, а служат только для хранения информации о рассматриваемых в предметной области объектах.
3. Преобразования потоков данных во внешних сущностях игнорируются, т.е. они используются только как источники или приемники данных.
4. Хранилища данных и внешние сущности не могут быть связаны напрямую, только через функцию. То же касается связей между двумя хранилищами или двумя внешними сущностями, но связывать между собой потоками данных функции допустимо.

Помимо этого, для каждого информационного потока и хранилища определяются связанные с ними элементы данных, продумываются возможные ограничения. Каждому элементу данных присваивается имя, также для него может быть указан тип данных и формат. Эта информация является затем исходной при построении на следующем этапе модели «сущность-связь». При этом, как правило, информационные хранилища преобразуются в сущности новой диаграммы, проектировщику остается только решить вопрос с использованием элементов данных, не связанных с хранилищами.

Помимо нотации Йордана – Де Марко для элементов DFD могут использоваться и другие условные обозначения (OMT, SSADM, нотации Гейна – Сарсона и т.д.). Все они обладают практически одинаковой функциональностью и различаются лишь в деталях.

В качестве примера приведена диаграмма потоков данных, отражающая работу библиотеки (рис. 1). Эта диаграмма представляет один из верхних уровней концептуальной модели, созданной на основе контекстной диаграммы []. Дальнейшее уточнение модели производится путем декомпозиции функций и хранилищ на DFD следующего уровня.

При построении контекстной диаграммы рекомендуется сначала определить хранилища, т.е. решить, информация о каких объектах заданной предметной области будет храниться в будущей БД (на рис. 1 - это Книги, Читатели, Издательства, Темы), затем нужно определить внешние сущности (на рис. 1 - это Посетители, Библиотекарь, Директор, Издательства), которые выполняют роль управляющего воздействия и/или источника данных и связать их через соответствующие функции с хранилищами. Так, например, для создания хранилища Издательства используется функция «заключение договоров с издательствами», которую выполняют внешние сущности Директор и Издательства, при этом предполагается, что от внешней сущности Издательства поступает необходимая информация (название, адрес), которая вносится в хранилище директором, например, из заключенных договоров, которые указываются на DFD как потоки данных. По другому принципу рассуждений на модели появляется хранилище Выдача книг: оно создается в результате выполнения Библиотекарем функции «оформление выдачи книг», которая предполагает использование данных из созданных ранее хранилищ Книги и Читатели. Таким образом, все хранилища появляются на диаграмме в результате выполнения внешними сущностями какого-либо процесса и сами являются источниками информации для других хранилищ или внешних сущностей.

На этапе концептуального проектирования также определяют простые ограничения на данные, например, можно указать, что работа библиотеки может быть организована только с определенными издательствами или потребовать, чтобы стоимость книг находилась в заданных пределах.

Очень важным является также понимание и написание так называемой бизнес-логики, т.е. неких правил, принятых в данной конкретной сфере, отражающих реальные процессы работы с данными, но не укладывающиеся в рамки простых проверок на их значения. Например, в рассматриваемой библиотеке можно отслеживать, чтобы на руках у читателя одновременно не оказалось более 10 книг (при этом надо понимать, что поля с количеством книг на руках в БД нет, это количество изменятся во времени и может быть рассчитано в дальнейшем при помощи написания соответствующего кода), также можно контролировать дату возврата книг и начислять пеню за просрочку.

## **Задание к лабораторной работе**

1. Запустить MS Access, ознакомиться с его функциональными возможностями, внешним видом окна создания компонентов новой БД. Запустить из меню *Справка* учебную БД «Борей» или создать БД «Проекты» при помощи локального шаблона. Ознакомиться со структурой и содержимым полученных примеров БД.
2. Ознакомиться со структурой и содержимым полученных примеров БД. При этом освоить основные пункты меню и иконки для работы с MS Access, приобрести навыки работы с окном БД: научиться просматривать объекты БД в различных режимах работы, познакомиться с окном схемы данных, редактором VBA.
3. Получить номер варианта индивидуального задания (у заочников вариант соответствует номеру зачетной книжки). Варианты заданий приведены в [Прил. 1](../pril1/pril1.htm). Самостоятельно изучить предметную область и в отчете сформулировать цели и задачи проектирования, описать простые ограничения на данные, которые предполагается хранить в БД, также сформулировать основные бизнес-правила необходимые для отображения реальных процессов предметной оласти на модели хранения данных.
4. Выделить основные внешние объекты, хранилища, функции и потоки данных для разрабатываемой модели БД. Создать диаграмму потоков данных в нотациях Йордана – Де Марко для полученного варианта.



Рис. 1. Пример диаграммы потоков данных

## **Контрольные вопросы**

1. Что понимают под предметной областью при разработке БД?
2. Почему анализ предметной области считают самым важным этапом в проектировании БД?
3. Что такое диаграмма потоков данных, для чего она используется?
4. Перечислить и объяснить назначение основных нотаций Йордана - Де Марко.
5. Какие способы создания DFD, кроме рассмотренного в лабораторной работе, существуют?
6. Какие типичные ошибки возможны на диаграммах потоков данных?
7. Что называют контекстной диаграммой?
8. Какие процессы сопровождают переход от контекстной диаграммы к диаграммам более низкого уровня?
9. Какие основные объекты можно создавать в СУБД MS Access?
10. Что называется шаблоном, как он используется для создания БД?
11. Что понимают под простыми ограничениями на данные?
12. Что такое бизнес-логика ?

# Лабораторная работа № 2 Представление данных с помощью модели «Сущность-связь». Работа с таблицами MS Access

## **Цель работы:** изучение принципов построения модели «сущность-связь». Получение навыков работы по созданию и изменению структуры таблиц, их заполнению данными. Создание ключевых индексированных полей, ограничений на свойства полей, установка связей между таблицами.

## **Краткие теоретические сведения**

Разработка **модели «сущность-связь»**. Предметная область, которая была рассмотрена в предыдущей лабораторной работе, представляет собой схематическое изображение процессов, происходящих в реальном мире. Вторым способом представления данных, независимо от реализующего его программного обеспечения, является модель «сущность-связь».

Модель «сущность-связь» (англ. Entity-Relationship model), или ER-модель, предложенная П. Ченом в 1976 г., является наиболее известным представителем класса семантических (концептуальных, инфологических) моделей предметной области.

Основные преимущества ER-моделей это наглядность, возможность проектировать БД с большим количеством объектов и атрибутов, а также возможность реализации во многих системах автоматизированного проектирования БД (например, ERWin).

Основными конструктивными элементами ER-моделей являются сущности, связи между ними и их свойства (атрибуты).

Сущность - это любой различимый объект или процесс (который можно отличить от другого по набору свойств), информацию о котором необходимо хранить в БД. Сущностями могут быть люди, места, детали, оборудование, заказы, контракты и т.д. Необходимо различать такие понятия, как тип сущности и экземпляр сущности. Понятие тип сущности относится к набору однородных личностей, предметов, событий или идей, выступающих как целое. Экземпляр сущности относится к конкретной вещи в наборе. Например, типом сущности может быть ГОРОД, а экземпляром - Москва, Киев и т.д.

Атрибут - это поименованная характеристика (свойство) сущности. Его наименование должно быть уникальным для конкретного типа сущности, но может быть одинаковым для различного типа сущностей (например, размер может быть определен для многих сущностей: деталь, инструмент, изделие и т.д.). Атрибуты используются для определения того, какая информация должна быть собрана о сущности. Примерами атрибутов для сущности деталь являются наименование, размер, материал и т.д. Здесь также существует различие между типом и экземпляром. Тип атрибута материал имеет много экземпляров или значений: чугун, золото, железо, медь и т.д., однако каждому экземпляру сущности присваивается только одно значение атрибута.

Абсолютное различие между типами сущностей и атрибутами отсутствует. Атрибут является таковым только в связи с типом сущности. В другом контексте атрибут может выступать как самостоятельная сущность. Например, для автомобильного завода цвет – это только атрибут продукта производства, а для лакокрасочной фабрики цвет – тип сущности.

Ключ сущности – это минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности. Минимальность означает, что исключение из набора любого атрибута не позволяет однозначно идентифицировать сущность по оставшимся.

Связь – это ассоциирование двух или более сущностей с помощью ключевых атрибутов. Если бы назначением БД было только хранение отдельных, не связанных между собой данных, то ее структура могла быть очень простой. Однако одно из основных требований к организации БД – это обеспечение возможности поиска одних сущностей по значениям других, для чего необходимо установить между ними определенные связи. Так как в реальных БД нередко содержатся сотни или даже тысячи сущностей, то теоретически между ними может быть установлено более миллиона связей. Наличие множества связей и определяет сложность инфологических моделей.

Важным свойством модели «сущность-связь» является то, что она может быть представлена в виде графической схемы. В табл. 2 приведен список обозначений нотаций Чена-Мартина для построения ER-диаграммы.

 Таблица 2

Условные обозначения нотации Чена-Мартина

|  |  |
| --- | --- |
| Обозначение | Значение |
|  | Независимая сущность |
|  | Зависимая сущность |
|  | Атрибут сущности |
|  | Ключевой атрибут |
|  | Связь между сущностями |

На схеме атрибуты с сущностями и сущности со связями соединяются прямыми линиями.

Связь устанавливается путем сопоставления данных в ключевых атрибутах; обычно это столбцы, имеющие в обеих сущностях одинаковые названия.

Существует три вида связей между сущностями.

Связь «один ко многим» – наиболее распространенный вид связи. При такой связи каждому экземпляру сущности А может соответствовать множество экземпляров сущности Б, однако каждому экземпляру сущности Б может соответствовать только один экземпляр сущности А. Например, между сущностями «Издатели» и «Книги» устанавливается связь «один ко многим»: каждый из издателей может опубликовать множество книг, однако каждая книга публикуется лишь одним издателем.

Важной характеристикой связи является класс принадлежности входящих в нее сущностей или кардинальность связи. Так как каждый автор обязательно должен иметь книгу, то каждой сущности «Автор» непременно должна соответствовать сущность «Книга». Однако не каждая книга должна иметь конкретного автора (например, энциклопедии и сборники), следовательно, в данной связи не каждая сущность «Автор» имеет ассоциированную с ней сущность «Книга» (рис. 2).



Рис. 2. Пример связи «один ко многим»

Таким образом, говорят, что сущность «Книга» имеет обязательный класс принадлежности – , а сущность «Автор» имеет необязательный класс принадлежности – .



Связь «многие ко многим». При установлении связи «многие ко многим» каждому экземпляру сущности А может соответствовать множество экземпляров сущности Б и наоборот. Например, между сущностями «Читатель» и «Книга» должна устанавливаться связь вида «многие ко многим» (рис.3), так как один читатель заказывает многие книги и, наоборот, одна книга заказывается многими читателями. На практике (в физической модели данных) такие связи создаются с помощью двух связей вида «один ко многим», которые устанавливаются между каждой из первоначальных сущностей и новой сущностью, например, «Корзина». Ключевым атрибутом сущности «Корзина», как правило, выбирают сочетание ключевых атрибутов двух первоначальных сущностей (рис. 4).



Рис. 3. Пример связи «многие ко многим»



Рис. 4. Пример практической реализации связи «многие ко многим»

 Связи «один к одному». При установлении связи «один к одному» каждому экземпляру сущности А может соответствовать только один экземпляр сущности Б и наоборот. Этот вид связи используется редко, поскольку в такой ситуации связываемые данные обычно можно хранить в одной сущности. Использовать связь вида «один к одному» можно в указанных ниже случаях:

* чтобы разделить сущность, содержащую слишком много атрибутов;
* чтобы изолировать часть атрибутов сущности по соображениям безопасности;
* для хранения данных кратковременного использования, удалить которые проще всего путем удаления сущности из модели;
* для хранения данных, имеющих отношение только к подмножеству основной сущности.



Рис. 5. Пример связи «один к одному»

В качестве примера рассмотрим пример построения ER-модели для БД библиотеки (рис. 6).



Рис. 6. Пример ER-модели

**Создание таблиц в режиме конструктора MS Access.** Чтобы правильно создавать таблицы необходимо знать, что таблица – это объект БД, который служит для ввода и хранения информации. Таблица состоит из записей (строк), которые составляют информацию, хранящуюся в ней, и полей (столбцов), образующих структуру таблицы.

В MS Access выберите в меню создать таблицу, переключитесь ее соответствующей иконкой в режим конструктора, предварительно сохранив в БД под своим именем.

В конструкторе для каждого поля задается имя поля, тип данных, перечень свойств, описание.

Имя поля должно быть уникальным, содержать не более 64 символов, включая пробелы, но не должно начинаться с пробела. Имена полей могут содержать любые символы, за исключением (.); (!); ('); ([); (]).

Тип данных определяет вид и диапазон значений, которые могут содержаться в данном поле. MS Access предлагает следующие типы данных.

Текстовой – тип данных, используемый для хранения символьных или числовых данных, не требующих вычислений. Поле данного типа может содержать до 255 символов.

Числовой – тип данных для хранения числовых значений, используемых в математических расчетах. Данные этого типа могут содержаться в 1, 2, 4, 8 байтах в зависимости от значения поля Размер поля.

Поле Memo – специальный тип данных для хранения больших объемов информации, по объему превышающей 255 символов. Это поле может содержать до 65535 символов.

Дата / время – тип данных для хранения календарных дат и текущего времени.

Денежный тип – тип данных для хранения денежных значений, точность представления которых колеблется от 1 до 4 знаков после запятой. Целая часть может содержать до 15 десятичных знаков. Длина поля 8 байт.

Счетчик – специальный тип данных для уникальных натуральных чисел, обладающий свойством автоматического наращивания при каждой новой записи. Длина поля 4 байта.

Логический тип – специальный тип данных, имеющий только два значения, интерпретируемых как Да/Нет, Истина/Ложь, Вкл/Выкл. Длина поля 1 байт.

Поле объекта OLE – содержит ссылку на OLE-объект, например документы Microcoft Word, Microcoft Excel, звук, изображение и другие виды объектов. Ограничено дисковым пространством.

Гиперссылка – специальное поле, предназначенное для хранения гиперссылки, с помощью которой можно ссылаться на произвольный фрагмент данных внутри файла или страницы на том же компьютере, в локальной сети или Интернете. Длина до 2048 символов.

При выборе типа данных для каждого поля открывается соответствующий набор свойств, с помощью которого определяют вид и функциональные характеристики данного поля.

Наиболее важными являются следующие свойства.

Размер поля – определяет максимальную длину данных, которые могут содержаться в данном поле.

Формат поля – определяет способ формирования данных в данном поле таблицы.

Число десятичных знаков – определяет количество знаков после запятой для полей числового и денежного типа.

Маска ввода – определяет форму ввода данных в поле и облегчает контроль вводимых символов.

Подпись – задает названия полей таблицы, которые выводятся в различных режимах.

Значение по умолчанию – указывает значение, автоматически вводимое в поле при создании новой записи.

Условие на значения – ограничение, накладываемое на вводимые в данное поле данные.

Сообщение об ошибке – это текст сообщения, которое будет выводиться в случае нарушения условия на значение.

Обязательное поле – определяет, является ли ввод данных в это поле обязательным. Это очень важное свойство столбца, которое переносится на физическую модель с логической диаграммы (кардинальность связей). Если поле указано, как необязательное, то можно ошибочно не указать важную информацию о сущности, так, например, непонятно какое значение будет иметь запись о книге без ее названия, кода издательства, объема и стоимости, но в то же время вполне допустимо и даже необходимо оставить поле Автор необязательным для заполнения, иначе ввести книгу без автора будет невозможно.

Индексированное поле – определяет, является ли данное поле индексированным. Индекс ускоряет все операции, связанные с сортировкой, группировкой или поиском записей по значению.

При вводе данных в поле осуществляется автоматическая проверка типа данных и условия на значение, которое позволяет обеспечить ввод корректных данных в соответствии с простым ограничением. Проверка значения осуществляется при переходе к другому полю или другой записи. Если введенное значение не соответствует заданному условию, то на экран выводится сообщение об ошибке.

Например, на рис.6.5 показано как задать число десятичных знаков, значение по умолчанию, допустимый диапазон значений и сообщение об ошибке для денежного типа поля Стоимость.

Часто возникает необходимость задать условие на значение одного поля, используя значения другого в этой же таблице. Предположим, что в библиотеке будут поля Дата заказа и Дата возврата, разница между которыми не должна превышать 30 дней. В этом случае, говорят об ограничении на уровне таблицы и для его реализации необходимо открыть меню Свойства щелчком на правой кнопке мыши, где в графе Условие на значение написать: [Дата возврата] - [Дата заказа]<=30.

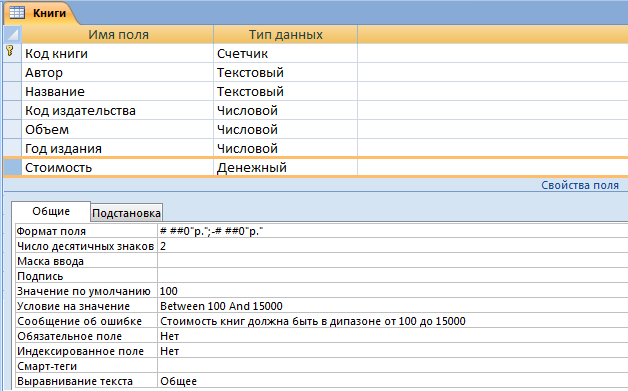


Рис. 6.5 – Указание условия на значение для поля Стоимость

Раздел описаний необязателен для заполнения, но информация, введенная в данный раздел будет отображаться в строке состояния при вводе данных в соответствующие поля, т.е., по сути, может быть использована в качестве контекстной подсказки при вводе данных.

Ключевые поля. Каждая реляционная таблица должна иметь ключевое поле. Ключ однозначно определяет каждую запись в таблице и повторяющиеся значения ключа не допускаются автоматически. Для организации связей между таблицами в MS Access используются ключевые поля и схема данных.

Связи между таблицами часто устанавливаются, чтобы при работе с базой вводить как можно меньше данных. Поэтому, если ввод каких-то данных приходится повторять неоднократно в одной или нескольких таблицах, то эти данные хранят в отдельной таблице, связанной с другими реляционными ключами.

Связываться могут только таблицы, имеющие ключевые поля. Это правило нужно соблюдать для того, чтобы связи между таблицами работали надежно, и по записи из одной таблицы можно было однозначно найти запись в другой таблице.

Например, если из таблицы «Выдача книг» будет известно, что заказ сделал читатель Иванов. Но в таблице «Читатели» может быть несколько разных Ивановых и компьютер не разберется, кто  из них сделал данный заказ, а это означает, что поле «Фамилия» не является уникальным и потому его нельзя использовать в качестве ключевого для связи между таблицами. Поле номера телефона – более удачный кандидат, но одним телефоном могут пользоваться несколько разных читателей. Если ни одно поле таблицы не приемлемо в качестве ключевого, то его можно создать искусственно. Так в рассматриваемом примере в таблице «Читатели» создано поле «Код читателя». В качестве первичного ключа в таблицах часто используют поле, имеющее тип Счетчик. Ввести два одинаковых значения в такое поле нельзя по определению, поскольку приращение значения поля производится автоматически, при этом надо понимать, что если запись была удалена, то новую запись с таким же номером вставить уже не получится.

Кроме того, существуют следующие правила для установления связей между таблицами:

− таблицы должны содержать поля с одинаковыми записями;

− типы данных, длины полей, а главное – значения обязательно должны совпадать.

Для определения ключа записи выделяется, как правило, одно поле, и на панели инструментов нажимается кнопка Ключ с изображением ключа (логотип MS Access) или используют список возможностей, привязанный к нажатию правой кнопки мыши. Когда данные одного поля не могут быть уникальными для каждой записи, можно назначить ключевыми два или более полей – составной ключ. Ключевые поля таблицы индексируются автоматически.

В приведенном ниже примере ключевым полем является поле Код выдачи. Связь между таблицами показана на рис. 7.

Таблица 3

|  |  |  |  |
| --- | --- | --- | --- |
| Код выдачи | Код читателя | Код книги | Дата заказа |
| 1 | 1 (Иванов И.И.) | 1 (Беспалько) | 01.12.09 |
| 2 | 2 (Петров П.П.) | 1 (Беспалько) | 01.03.10 |
| 3 | 2 (Петров П.П.) | 2 (Сканави) | 20.04.10 |
| 4 | 3 (Сидоров С.С.) | 2 (Сканави) | 04.11.10 |

Поля Код читателя, Код книги содержат числовые значения. Эти значения устанавливаются в соответствии с ключевыми полями соответствующих таблиц «Читатели» и «Книги». То есть в поле Код читателя записано значение 2, которое является ссылкой на запись в таблице «Читатели» с кодом 2 – Петрова П.П. В дальнейшем (см. лабораторную работу №5) на формах числовые значения кодов будут заменяться данными других полей, чтобы пользователю было понятно, с чем он работает.

**Схема данных.** Связи между таблицами устанавливаются на Схеме данных. Схема данных строится в соответствии с ER-моделью. В этом случае каждой сущности будет соответствовать таблица. Атрибуты сущностей будут определены соответствующими полями таблиц.

При построении схемы данных MS Access сам автоматически выбирает тип связи по выбранным полям таблиц. Одна из связанных таблиц является главной, вторая – подчиненной. MS Access позволяет устанавливать связи типов один-к-одному и один-ко-многим.

Связь «один к одному» создается в том случае, когда оба связываемых поля являются ключами или на них наложены ограничения уникальности.

Связь «один ко многим» создается в том случае, когда только на одно из полей наложено ограничение уникальности или оно является первичным ключом таблицы.

Сторона связи «один ко многим», которой соответствует первичный ключ, обозначается символом единицы – 1. Сторона связи, которой соответствует внешний ключ, обозначается символом бесконечности –



Чтобы вызвать окно Схема данных необходимо найти соответствующую кнопку на панели инструментов. Добавить таблицу в окно Схема данных можно, вызвав правой кнопкой мыши контекстное меню и выбрав пункт Добавить таблицу.

В окне Схема данных для установки связи между таблицами следует перетащить поле, по которому устанавливается связь, из родительской таблицы в поле подчиненной (дочерней). Полям, на основе которых создаются связи между таблицами, необязательно иметь одинаковые имена, однако они должны быть одного типа и размера. Исключение составляют поля счетчиков, которые можно связывать с числовыми полями (размер - длинное целое).

В появившемся окне Изменение связей обязательно нужно задать параметры для устанавливаемой связи:

* обеспечение целостности связи;
* каскадное обновление полей;
* каскадное удаление связанных полей.

Целостность данных означает систему правил, которые используются для поддержания связей между записями в связанных таблицах, а также обеспечивают защиту связанных данных от случайного удаления или изменения.

Все лабораторные работы в дальнейшем выполняются с использованием учебной БД «Библиотека», которая состоит из пяти таблиц, приведенных в [Прил](../pril2/pril2.htm). 2. Схема данных для БД «Библиотека» приведена на рис. 7.



Рис. 7. Схема данных БД «Библиотека»

После создания структуры таблиц и установления связей между ними, БД заполняют данными. На практике эта задача решается с помощью форм и выполняется другими специалистами (не разработчиками). В целях обучения работе в MS Access можно добавить несколько новых записей (строк) в таблицы БД «Библиотека». Для этого необходимо открыть таблицу в режиме просмотра и заполнить поля новой информацией. Сохранение таблицы не требуется, так как оно производится автоматически при переходе к следующей записи или закрытии таблицы.

## **Задание к лабораторной работе**

1. Открыть учебную [БД «Библиотека», сохранить копию исходного файла для выполнения следующих лабораторных работ.](../БД%20Библиотека.mdb)

а) открыть поочередно таблицы в режиме Конструктора, проанализировать выбор ключевых полей, а также выбор наименований, типов данных и свойства обязательности данных для остальных полей;

б) самостоятельно добавить несколько новых полей в таблицы БД (по собственному усмотрению, но не менее 3), придумать и установить для них или для уже существующих полей свойства Условие на значения, Значение по умолчанию и Сообщение об ошибке, проверить их работоспособность;

в) внести в новые поля данные в режиме таблицы;

г) изменить формат поля «Дата заказа» и установить на поле соответствующую формату маску для ввода в это поле значений (допустимые знаки маски ввода и примеры приведены в [Прил. 3](../pril3/pril3.htm)), также установить в это поле с помощью встроенной функции Date() или Now() значение по умолчанию;

д) самостоятельно придумать шаблон маски и реализовать его для одного произвольного **текстового** поля в любой таблице;

е) открыть окно Схема данных, ознакомиться и объяснить, как и почему установлены связи между таблицами, объяснить для чего они необходимы;

ж) создать и сохранить в БД новую таблицу «Журналы», которая будет содержать поля: код журнала (счетчик, ключевое), название (текстовое), количество страниц (целое), дата выпуска (дата/время), код издательства (целое, для связи с таблицей Издательства);

з) добавить в таблицу «Выдача книг» новое необязательное поле Код журнала, предназначенное для указания журнала, который может быть заказан читателем;

и) добавить таблицу «Журналы» на схему данных, установить связь типа «один ко многим» с таблицами «Выдача книг» и «Издательства», указать в окне изменения связи все параметры обеспечения целостности;

к) внести в таблицу «Выдача книг» новые записи (строки), содержащие заказы каких-либо читателей на журналы и книги, заказах только на журналы и заказы только на книги;

л) создать таблицу «Личные данные о читателях» (самостоятельно сделать выбор полей и типов данных новой таблицы), добавить ее на схему данных и установить «связь один к одному» с таблицей «Читатели». Обосновать назначение связи, установить параметры обеспечения целостности;

м) создать таблицу «Корзина\_выдачи» для связи «многие ко многим» между таблицами «Выдача книг» и «Книги», осуществить выбор полей новой таблицы, установить составной ключ, добавить таблицу на схему данных и реализовать через нее необходимую связь. При этом связь «один ко многим» между таблицами «Выдача книг» и «Книги» должна быть удалена, как и поле Код книги в таблице «Выдача книг»;

2. Спроектировать на основе DFD модель «сущность-связь» (ERD) для индивидуального задания.

3.Реализовать сущности ERD из п.2 в виде таблиц с помощью Конструктора в MS Access, определить типы и основные свойства полей, задать простые ограничения (условия на значения), необходимые значения по умолчанию и маски.

4. Открыть окно схемы данных, добавить таблицы из п. 3 и установить связи между ними в соответствии с моделью «сущность-связь», обеспечить целостность.

5.Заполнить таблицы данными, начиная с родительских.

**Контрольные вопросы:**

1.  Что понимают под логической моделью данных?

2.  Что означает термин ERD?

3.  Какие виды связей между сущностями бывают?

4.  Приведите примеры связей «многие ко многим», «один ко многим»

5.  Для чего используется связь «один к одному»?

6.  Что такое кардинальность связи?

7.  Как реализуется на практике каждый из видов связей?

8.  Что называется первичным ключом? Внешним ключом?

9.  Какие ограничения накладываются на поля, участвующие в организации связи?

10.   Что такое ссылочная целостность?

11.   По каким правилам преобразуется DFD в ERD?

12.   Какие нотации могут быть использованы для построения диаграмм «сущность-связь»?

# **Лабораторная работа №3 Создание простых запросов и запросов на изменение**

## **Цель работы:** работа с фильтрами, создание запросов на выборку, условных запросов, запросов на удаление, обновление, добавление данных, запросов на создание таблиц на основе учебной БД «Библиотека».

**Краткие теоретические сведения**

При работе с БД постоянно требуется организовывать просмотр, сортировку, фильтрацию, выборку, поиск данных и вычисление значений как в пределах одной таблицы, так и по всей БД. Например, нужно просмотреть все заказы, сделанные какой-либо фирмой, либо все заказы, сделанные за определенный месяц и т.п. В MS Access поставленные задачи можно решать при помощи фильтров или при помощи запросов.

**Фильтры.** Фильтры обычно используются при работе с одной таблицей, их преимуществом перед запросами является простота. Кроме того, фильтры не занимают места на диске, в отличие от запросов. Для фильтрации данных в таблицах можно воспользоваться следующими способами:

*Поиск данных*. Чтобы выполнить простой поиск по одному полю, сначала нужно выделить его, а затем использовать команду меню «Найти». В образце поиска можно использовать подстановочные символы. Символ **\*** (звездочка) заменяет строку любой длины, а символ ? (знак вопроса) – любой произвольный символ.

*Фильтр по выделенному.* Предположим, нужно выбрать все записи таблицы «Издательства», в которых указан город Москва. Для этого находят одну запись, в которой указан город Москва, устанавливают в эту ячейку курсор мыши и выбирают «Фильтр по выделенному». В результате будут отображены все записи, у которых поле «Город» имеет значение «Москва». Опция «Исключить выделенное»– наоборот, оставит на экране поля, значения которых не совпадают с выделенной записью.

Чтобы отфильтровать записи по значениям в различных полях, используют опцию «Изменить фильтр», которая позволяет сформировать строку с необходимыми значениями с помощью простейших математических и логических операторов. Просмотреть результат работы такого фильтра можно после нажатия на кнопку «Применить фильтр». Для удаления ненужного фильтра используется опция «Очистить (удалить) все фильтры».

**Запросы.** Одно из преимуществ запросов состоит в том, что они позволяют достаточно быстро отобрать необходимые данные из нескольких связанных таблиц. В общем, запросы полезны и для работы с одной таблицей, и для сложных многотабличных запросов.

В отличие от реальной таблицы набор записей – результат запроса физически не существует в БД. MS Access создает его из данных таблиц только во время выполнения запроса.

Для создания запросов к БД существует специальный язык запросов. Он называется SQL (Structured Query Language – структурированный язык запросов). Но MS Access использует более простое средство, которое называется бланком запроса по образцу. С его помощью можно сформировать запрос простыми приемами, перетаскивая элементы запроса между окнами в Конструкторе запросов.

Все запросы можно разделить на две группы: запросы-выборки, запросы-действия.

Запросы-выборки осуществляют выборку данных из таблиц в соответствии с заданными условиями. К этой группе запросов относятся следующие.

Запрос к связанным таблицам – позволяет производить выборку данных из связанных таблиц.

Условный запрос – позволяет производить отбор записей в соответствии с заданным критерием поиска.

Запрос с вычисляемым полем – позволяет рассчитать данные на основе других полей из той же строки запроса.

Итоговый запрос – производит математические вычисления с помощью встроенных статистических функций и выдает общий результат.

Параметрический запрос – позволяет пользователю задать критерий отбора, введя нужный параметр непосредственно при вызове запроса.

Перекрестный запрос – отображает итоговые данные с группировкой их по горизонтали и вертикали, выводя результаты их обработки в виде таблиц.

Запросы-действия позволяют модифицировать данные в таблицах: удалять, обновлять, добавлять записи. Эти запросы называют еще запросами на изменение и выделяют четыре типа:

Запросы на создание таблицы – создают таблицы на основании данных, содержащихся в результирующем множестве запроса.

Запросы на добавление записей – позволяют добавлять в таблицу записи, создаваемые запросом.

Запросы на обновление – изменяют значения существующих полей в соответствии с заданным критерием.

Запросы на удаление – удаляют записи из одной или нескольких таблиц одновременно.

В MS Access можно создавать запросы при помощи Мастера запросов и с помощью Конструктора.

Для создания запросов при помощи Мастера имеются следующие мастера: Мастер создания простого запроса; Мастер создания перекрестного запроса; Мастер создания запросов на поиск повторяющихся записей; Мастер создания запросов на поиск записей без подчиненных записей.

При создании запроса с помощью Мастера производится пошаговое выполнение действий в диалоговом окне Мастера запросов.

Конструктор запросов позволяет создавать новые и изменять существующие запросы, поэтому он является основным способом.

При создании запросов в режиме Конструктора открывается специальный бланк, называемый бланком запросов по образцу. Этот бланк состоит из двух областей. В верхней части отображаются структуры таблиц, к которым адресован запрос, а нижняя часть представляет собой таблицу описания запроса, которая имеет 5 основных строк. Количество столбцов в запросе определяется количеством выбранных полей, которые будут использованы в запросе.

Назначение строк в бланке Конструктора запросов:

1-я строка – *Поле* предназначена для ввода имени поля запроса, переименования его названия, создания вычисляемого выражения;

2-я строка – *Имя таблицы* предназначена для задания имени таблицы – источника данных поля запроса;

3-я строка – *Сортировка* предназначена для указания порядка сортировки в данном поле. Может принимать следующие значения: по возрастанию, по убыванию, отсутствует;

4-я строка – *Вывод на экран* содержит флажок, указывающий на то, будут ли выводиться значения поля на экран при просмотре результатов запроса;

5-я строка – *Условие отбора* содержит критерий, по которому будет происходить отбор записей в результирующую таблицу.

Рассмотрим пример условного запроса с выбором данных из одной таблицы: выберем читателей с фамилией Бобров.

Для этого необходимо выбрать Создание запроса в режиме Конструктора. В появившемся бланке Конструктора из предложенного списка таблиц выбрать Читатели, нажать кнопку «Добавить» и закрыть диалоговое окно выбора таблиц.

Чтобы заполнить бланк конструктора запроса в строке *Поле:* в первом столбце выбрать поле Фамилия, во втором – Имя, в третьем – Отчество. Строка *Имя таблицы:* при этом будет заполняться автоматически. Тот же результат можно получить, если последовательно выбирать нужные поля в таблице двойным щелчком мыши.

Далее в первом столбце, в строке *Условие отбора:* ввести: Like "Бобров". Запрос будет иметь следующий вид (рис. 8):

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле:** | Фамилия | Имя | Отчество |
| **Имя таблицы:** | Читатели | Читатели | Читатели |
| **Сортировка:** |  |  |  |
| **Вывод на экран:** | ✓ | ✓ | ✓ |
| **Условие отбора:** | Like "Бобров" |  |  |
| **или:** |  |  |  |

Рис. 8. Параметры запроса на выборку

Потом можно закрыть окно Конструктора запроса. Перед закрытием MS Access запросит имя, под которым сохранить запрос. Запросы всегда следует называть в контексте с той информацией, которую они возвращают, поэтому данный запрос можно назвать «Бобров».

Для того чтобы просмотреть результат работы запроса его либо просто открывают двойным щелчком мыши по названию запроса, либо выбирают опцию «Открыть» правой кнопкой мыши. В результате появится окно (рис. 9).

|  |  |  |  |
| --- | --- | --- | --- |
|  | Фамилия | Имя | Отчество |
|  | Бобров | Виктор | Иванович |
| \* |  |  |  |

Рис. 9. Результат выполнения запроса на выборку

Приведем пример выборки данных из нескольких таблиц. Найдем читателей, которые в 1996 г. заказали «Сборник задач» М.И. Сканави. При этом пусть требуется вывести только название книги и фамилию читателя.

Для начала необходимо открыть окно Конструктора запросов и добавить таблицы «Читатели», «Книги», «Выдача книг».

Так как следует организовать выборку по фамилиям читателей, названию книги и дате заказа, то второй шаг - выбрать соответствующие поля в бланк запроса. Далее ввести условие отбора для поля «Название книги» – «Сборник задач», а в условии отбора в поле Дата заказа – оператор Between 01.01.96 Аnd 31.12.96, который организует выборку данных в указанном промежутке значений.

Чтобы скрыть отображение поля Дата заказа, так как оно необходимо только для задания условия отбора, уберем галочку в строке *Вывод на экран*.  Бланк запроса будет иметь следующий вид (рис. 10).

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле:** | Фамилия | Название | Дата заказа |
| **Имя таблицы:** | Читатели | Книги | Выдача книг |
| **Сортировка:** |  |  |  |
| **Вывод на экран:** | ✓ | ✓ |  |
| **Условие отбора:** |  | "Сборник задач" | Between #1.01.1996# And #31.12.1996# |
| **или:** |  |  |  |

Рис. 10. Многотабличный запрос на выборку

Можно сохранить запрос под именем «Сборник задач в 1996 году» и выполнить его. Результат запроса имеет вид, представленный на рис. 11.

|  |  |  |
| --- | --- | --- |
|  | Фамилия | **Название** |
|  | Федосенко | Сборник задач |
|  | Захаров | Сборник задач |
| \* |  |  |

Рис. 11. Результат многотабличного запроса на выборку

В многотабличных запросах очень важно понимать, что данные выводимые в результате без указания ключевых полей могут дублироваться, то есть в рассмотренном выше примере вполне вероятно, что в течении 1996 года одни и те же читатели могли брать одни и те же книги по несколько раз (у них будут разные коды выдачи, но в запросе они отсутствуют). Чтобы в запросе были только уникальные записи необходимо открыть запрос в режиме SQL и непосредственно после слова SELECT написать DISTINCT или использовать для подобных запросов Мастер создания запросов на поиск повторяющихся записей.

**Вычисления в запросах.** Поле, содержимое которого является результатом расчета, основанного на математических операциях с участием значений полей БД, числовых и текстовых литералов, встроенных функций и т.п., называется вычисляемым полем. Вычисляемое поле существует только в результирующей таблице запроса, т.е. в исходных таблицах БД такое поле не создается и данных в существующих полях не изменяет.

Для создания запроса, производящего вычисления в полях, служит тот же бланк запроса Конструктора. Разница только в том, что в одном из столбцов вместо имени поля записывается формула для расчета новых значений поля. В формулу чаще всего входят заключенные в квадратные скобки названия полей БД и знаки математических операций, например, можно создать новое поле:

Сумма продажи: [Цена] \* [Количество], где *Сумма продажи* – это название нового вычисляемого поля запроса, которое указывается вначале строки перед двоеточием, за которым следует непосредственно формула для расчета.

Комбинация клавиш SHIFT+F2 открывает вспомогательное диалоговое окно, которое называется «Область ввода» и предназначено для удобства записи сложных и длинных вычисляемых выражений. Созданное выражение после закрытия окна помещается в бланк запроса.

Вычисляемое поле может выводиться на экран, тогда результаты расчетов будут отображаться в запросе, также вычисляемое поле можно сделать полем сортировки, чтобы не только получать новые результаты, но и анализировать их.

Возможности вычисляемых полей распространяются и на операцию конкатенации (сцепки) нескольких текстовых полей. Поэтому, чтобы вывести, например, Фамилию, Имя и Отчество в одном поле, создают выражение в поле запроса по формуле: ФИО: [Фамилия]& " "&[Имя]& " "&[Отчество]. При этом в строке *Имя таблицы* необходимо выбрать таблицу «Читатели».

**Условные запросы.** Для построения условий отбора могут применяться пять групп (критериев) условий отбора:

1.  Сравнение, используются элементарные математические функции (=, <, >, <=, >=, <>);

2.  Принадлежность диапазону, используется оператор Between … and …;

3.  Соответствие шаблону, используется оператор Like "шаблон";

4.  Принадлежность множеству значений, используется оператор  In (список значений через запятые).

5.  Проверка на пустые значения, операторы Not Null, Null.

Кроме этого все перечисленные условия могут быть скомбинированы логическими функциями and(и), or(или), NOT (не).

Примеры формирования некоторых условий отбора:

"Москва" OR "Минск" – издательства Москвы или Минска. Такой запрос можно реализовать, используя бланк запроса Конструктора, если ввести в строку *Условие отбора:*– "Москва", а ниже в строку *или:* "Минск". Если требуется выбирать из большего количества условий или, то их можно располагать и ниже строки *или:* столько раз, сколько необходимо.

Not Like "Москва" – все города, кроме Москвы.

Like "С\*" – все записи, которые начинаются с буквы С.

>#01.03.98# – начиная с указанной даты;

In ("Бобров "Захаров"; "Гуляев") – совпадает хотя бы с одним значением из списка;

Для работы со значениями  полей типа дата/время, используют встроенные функции MS Access: Date (), Day(), Month(), Year(), DatePart().

Функция Date () или Now() используется без аргументов и возвращает значение текущей даты. Если требуется связать запрос с текущим днем, месяцем, годом, то можно находить значения типа Month(Now()) – месяц от текущей даты.

Функции Day ([имя\_поля]), Month ([имя\_поля]), Year ([имя\_поля]) возвращают для дат, указанных в столбце [имя\_поля] целочисленное значение дня, месяца и года.

Функция DatePart ("интервал"; [имя\_поля]) – возвращает для указанного поля с типом дата/время целочисленное значение, заданное параметром "интервал". Этот параметр – строковая переменная – может принимать значения: "yyyy" – год, "Q" – квартал, "m" – месяц, "Y" – день года, "D" – день месяца, "w" – день недели, "ww" – неделя года, "h" – часы, "n" – минуты, "s" – секунды.

Например, функция DatePart ("yyyy"; [Дата Заказа]) возвращает год от значений поля *Дата заказа*, а DatePart ("Q"; Date ())будет определять и возвращать номер квартала для текущей даты.

Все вышеприведенные функции в бланке Конструктора запроса помещаются либо в строку *Поле*:, чтобы выделить нужную часть от значений дат, хранящихся в таблицах БД; либо в *Условие отбора:*, чтобы сравнить часть даты с каким-либо конкретным значением. С помощью функций, рассмотреных выше, многотабличный запрос на выборку можно создать в следующем виде (рис. 12):

|  |  |  |  |
| --- | --- | --- | --- |
| **Поле:** | Фамилия | Название | Выражение1: Year ([Дата заказа]) |
| **Имя таблицы:** | Читатели | Книги | Выдача книг |
| **Сортировка:** |  |  |  |
| **Вывод на экран:** | ✓ | ✓ |  |
| **Условие отбора:** |  | "Сборник задач" | 1996 |
| **или:** |  |  |  |

Рис. 12. Многотабличный запрос на выборку с функцией Year

**Запросы на изменение.** Позволяют изменять таблицы БД – обновлять, дополнять новыми записями, удалять некоторые записи, а также создавать новые таблицы в БД.

Для создания запросов на изменение в MS Access необходимо открыть Конструктор запросов, как при создании обычного запроса, а затем на панели инструментов Конструктора в разделе Запросы  выбрать иконку с нужным типом запроса на изменение (рис. 13).



Рис. 13

Запрос на обновление изменяет значения заданных полей в определенной группе записей одной или нескольких таблиц. Например, Конструктор запроса на обновление: увеличить стоимость московских издательств на 10%, будет выглядеть, как показано на рис. 14.

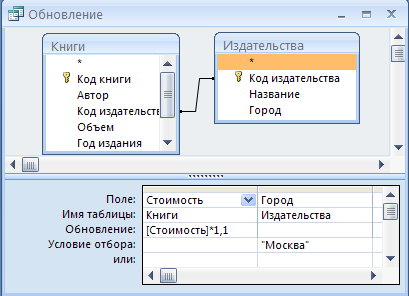


Рис.14. Запрос на обновление

Запрос на создание таблицы позволяет создать новую таблицу на основании всех либо какой-то части данных имеющихся таблиц. Например, запрос на создание таблицы: создать архивную таблицу, содержащую все записи о выдаче книг за два года, например, 1996 и 1997 год, будет выглядеть, как показано на рис. 15.

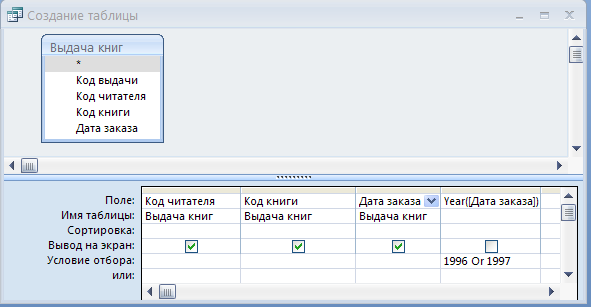


Рис. 15. Запрос на создание таблицы

Запрос на удаление удаляет определенную группу записей из одной или нескольких таблиц. С помощью такого запроса можно удалить только всю запись, а не отдельные поля внутри ее. Например, запрос на удаление: удалить все заархивированные записи из таблицы «Выдача книг», будет выглядеть, как показано на рис. 16.

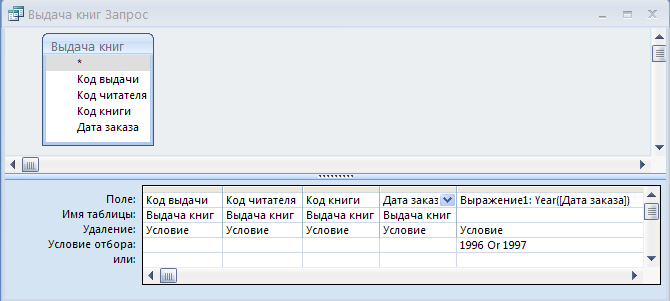


Рис. 16. Запрос на удаление

Запрос на добавление добавляет группу записей из одной таблицы в конец другой таблицы. Запрос полезен для объединения таблиц, хранящихся в разных файлах. Запрос на добавление: восстановить в таблицу «Выдача книг» записи из архива, будет выглядеть, как показано на рис. 17.

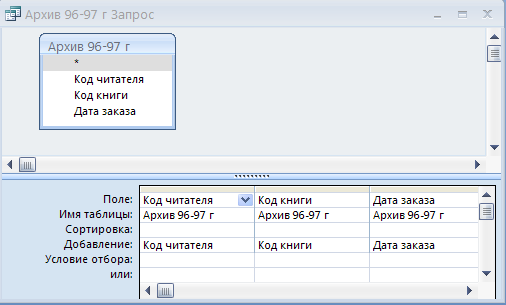


Рис. 17. Запрос на добавление

## **Задание к лабораторной работе**

1.  Открыть исходный файл БД Библиотека (можно использовать и файл с изменённой в ходе второй лабораторной работы БД, но тогда обязательно самостоятельно учитывать сделанные изменения в формулировках и результатах запросов).

2.  Выполнить условные запросы и запросы на выборку, сохранив их в БД в следующем порядке:

вывести в алфавитном порядке фамилии читателей и их домашние телефоны;

вывести список тем с указанием названия издательства;

вывести названия и авторов книг издательства «Мир» в обратном алфавитном порядке;

 вывести список книг, имеющих сочетание букв «упр» или «бин» в названии темы;

 вывести **в одном поле** фамилию, имя и адрес читателей, заказывавших книги Сканави или Перминова;

 вывести фамилии читателей, не имеющих телефона.

3.  Создать и сохранить запросы с использованием встроенных функций Day ([имя\_поля]), Month ([имя\_поля]), Year ([имя\_поля]), Now() и DatePart ("интервал"; [имя\_поля]), которые позволят произвести следующие выборки:

Информация о заказах на книги за последние 20 дней.

 Все книги, заказанные после 15 числа любого месяца.

 Читатели, сделавшие заказы на книги с 1996 по 2006 год.

 Книги, заказанные в январе 1997 года.

 Читатели, сделавшие заказы на книги в текущем месяце текущего года.

 Читатели, сделавшие заказы на книги в первом квартале текущего года.

4.  Выполнить следующие запросы на изменение:

изменить фамилию читательницы Семеновой Г.И. на Кормухина Г.И. (запрос на обновление);

создать таблицу-архив со всеми заказами Гуляева (запрос на создание таблицы);

удалить все заказы Гуляева из «Выдачи книг» (запрос на удаление);

восстановить заказы Гуляева за последний год (запрос на добавление);

5.  Работа с индивидуальным заданием:

 выполнить заданный в варианте условный запрос;

самостоятельно придумать задание и создать условные запросы, с использованием встроенных функций обработки данных в поле с типом дата/время (аналогично запросам п.3);

самостоятельно придумать задание и создать запросы на обновление данных, на создание таблицы, на удаление и добавление данных (аналогично запросам п.4).

## **Контрольные вопросы**

1. Что такое запрос?
2. Как создать запрос с помощью конструктора MS Access?
3. Чем отличаются запрос на выборку и условный запрос?
4. Какие запросы относятся к запросам на изменение?
5. Какие функции используют для полей с типом «дата-время»?
6. В чем особенность запросов к данным, содержащим пустые значения?
7. Как создать вычисляемое поле?
8. Как можно реализовать в конструкторе логические функции OR и AND по отношению к одному полю? К нескольким полям?
9. Для чего и как используется функция LIKE?
10. Какие параметры интервалов используются в функции DatePart()? Приведите примеры.

# **Лабораторная работа № 4 Создание сложных запросов. Работа с отчетами**

## **Цель работы:** создание итоговых, параметрических, перекрестных запросов и запросов на объединение; создание и настройка отчетов по запросам на основе учебной БД «Библиотека».

**Краткие теоретические сведения**

**Создание итоговых запросов.** С помощью итогового запроса получают не отдельные записи таблицы, а итоговые значения по группам данных. Например, общее количество взятых книг для всех читателей за последний месяц или средний объем заказов по каждому месяцу прошлого года. Для задания вычислений итоговых значений в обычный бланк Конструктора запросов добавляют возможность выбора групповой операции. Это делается с помощью иконки (рис. 18) на  панели инструментов Конструктора запросов, в результате  в бланке запроса появляется строка *Групповая операция:*.



Рис. 18

Если выполнить запрос с этой строкой, то получится набор записей, содержащий по одной строке для каждого уникального значения поля запроса, но без итогов. Для вычисления итогов операция *Группировка* в строке *Групповая операция:* должна быть заменена на конкретную итоговую функцию.

MS Access предоставляет девять встроенных функций, обеспечивающих выполнение групповых операций, которые можно выбрать в выпадающем списке столбцов в строке *Групповая операция:*. К итоговым функциям относятся:

*Sum*()– возвращает сумму всех значений данного поля в каждой группе. Используется только для числовых или денежных полей.

*Avg*() – возвращает среднее арифметическое всех значений данного поля в каждой группе. Используется только для числовых или денежных полей. При вычислении функции исключаются значения Null. С функцией *Avg*  практически всегда необходимо использовать встроенную функцию округления *Round*, чтобы исключить возможность переполнения поля в случае возникновения в результате вычисления среднего значения бесконечной дроби.

*Min*() – возвращает наименьшее значение, найденное в данном поле внутри каждой группы, для числовых полей возвращает наименьшее значение, для текстовых полей – наименьшее из символьных значений независимо от регистра, игнорирует значения Null.

*Мах*() – возвращает наибольшее значение, найденное в данном поле внутри каждой группы, для числовых полей возвращает наибольшее значение, для текстовых полей – наибольшее из символьных значений независимо от регистра, игнорирует значения Null.

*Count*() – возвращает число записей, в которых значения данного поля отличны от Null. Чтобы подсчитать число записей в каждой группе с учетом значений Null в строку *Поле:* вводится выражение *Count (\*).*

*StDev*() – возвращает стандартное отклонение всех значений данного поля в каждой группе. Эта функция применяется только к числовым или денежным полям. Если в группе меньше двух строк, возвращает значение Null.

*Var*()– возвращает дисперсию значений данного поля в каждой группе. Эта функция применима только к числовым или денежным полям. Если в группе менее двух строк, возвращает значение Null.

*First*()– возвращает первое значение данного поля в группе.

*Last* ()– возвращает последнее значение данного поля в группе.

**Создание параметрических запросов.** Запрос с параметрами – это запрос, при выполнении которого в диалоговом окне пользователю выдается приглашение ввести какие-либо данные, необходимые для формирования результирующего множества запроса. Количество приглашений для ввода данных в одном запросе практически не ограничено.

Запросы с параметрами удобно использовать в качестве основы для формирования форм и отчетов. Например, можно создать месячный отчет о заказах. При выполнении данного отчета на экране будет появляться приглашение ввести месяц года, заказы за который интересуют пользователя. После ввода месяца пользователю представляется требуемый отчет.

Для создания параметрического запроса необходимо сначала создать любой другой запрос или использовать уже имеющийся в БД. Для каждого поля, в которое предполагается вводить параметры, в строку *Условие отбора:* добавляют текст приглашения, заключенный в квадратные скобки [ ]. Это приглашение будет выводиться при запуске запроса. Текст подсказки должен отличаться от имени поля, но может включать его.

При создании параметрических запросов необходимо определять тип данных для вводимых параметров. Это делается с помощью пункта меню *Параметры запроса*, который открывает дополнительное окно. В предложенном окне в колонке *Параметр* указывается содержимое квадратных скобок в точном соответствии с бланком запроса, а затем выбирается нужный тип данных.

**Создание перекрестных запросов.** Перекрестные запросы используют для расчетов и представления данных в табличной структуре, облегчающей их анализ. Перекрестный запрос подсчитывает сумму, среднее число значений или выполняет другие итоговые расчеты, после чего результаты группируются в виде таблицы по двум полям данных, один из которых определяет заголовки столбцов, а другой – заголовки строк новой таблицы.

Перекрестный запрос создается с помощью соответствующего Мастера или самостоятельно в режиме Конструктора запроса. В бланке Конструктора запроса нужно указать поля, значения которых будут заголовками столбцов и строк, а также поле, значения которого следует использовать в вычислениях.

В перекрестном запросе, в котором заголовками столбцов являются, например, названия месяцев, можно выводить их в хронологическом, а не в алфавитном порядке, как это делается по умолчанию, или выводить только конкретно указанные месяцы.

Для этого используют окно свойств перекрестного запроса, открытого в режиме Конструктора. Окно открывается выбором опции *Свойства*, при нажатии правой кнопки мыши в фоновой области окна Конструктора запроса (вне бланка запроса и списка полей). В появившемся окне свойств в строке *Заголовки столбцов* вводят заголовки столбцов перекрестной таблицы, в том порядке, в котором их следует выводить в запросе. Заголовки столбцов разделяют точкой с запятой.

Введенные заголовки столбцов должны точно соответствовать заголовкам столбцов в запросе в режиме таблицы.

Если перекрестный запрос выполняется часто или если он используется как базовый для формы или отчета, то задание фиксированных заголовков столбцов позволяет существенно уменьшить время выполнения запроса.

При частом использовании одних и тех же заголовков в разных запросах можно создать таблицу с одним текстовым полем, в котором будут храниться заголовки столбцов.

**Создание запросов на объединение.**

Запрос на объединение объединяет наборы результатов из нескольких похожих запросов на выборку. Он используется для одновременного просмотра всех данных, возвращаемых несколькими сходными запросами на выборку, в виде объединенного набора.

Когда два (или более) запроса подвергаются объединению, их столбцы вывода должны быть совместимы для объединения. Это означает, что в каждом объединяемом  запросе должно указываться одинаковое число столбцов и в том же порядке что и первом, втором, третьем запросе, и так далее, и каждый столбец должен иметь тип данных, совместимый с каждым другим  объединяемым столбцом. Другое ограничение на совместимость - это когда пустые значения (Null) запрещены в любом столбце объединения, причем эти значения необходимо запретить и для всех соответствующих столбцов в других запросах объединения.

В запросах на объединение числовой и текстовый типы данных являются совместимыми.

При выполнении объединения данные из каждого набора соответствующих полей объединяются в одно выходное поле, поэтому выходные данные запроса имеют столько же полей, сколько и каждый запрос по отдельности.

Запрос на объединение является SQL-запросом. SQL-запросы не отображаются в режиме Конструктора и пишутся непосредственно на языке SQL. В MS Access для написания SQL-запросов, включая запросы на объединение, используется режим SQL (выбирается в меню *Режимы*).

При создании запроса с помощью конструктора всегда можно переключиться в режим SQL, чтобы просмотреть инструкцию SQL, обрабатываемую при выполнении запроса. Рекомендуется просматривать свои запросы в режиме SQL, чтобы ознакомиться с языком SQL и лучше понять принципы работы запросов. В некоторых случаях режим SQL может оказаться полезным для отладки запроса, который возвращает не те данные, которые ожидаются.

Ниже представлен базовый синтаксис SQL для запроса на объединение двух запросов на выборку.

SELECT поле\_1 [, поле\_2,…]

FROM таблица\_1 [, таблица\_2,…]

WHERE (условие)

UNION [ALL]

SELECT поле\_a [, поле\_b,...]

FROM таблица\_a [, таблица\_b,…]

WHERE (условие);

Ключевое слово UNION и необязательное ключевое слово ALL обозначают, что все результаты инструкции SELECT, предшествующей слову UNION, будут объединены со всеми результатами инструкции SELECT, следующей после UNION.

Существует два основных способа создания запроса на объединение:

1.  в режиме Конструктора с последующим объединением запросов в режиме SQL;

2.  непосредственно в режиме SQL (не рассматривается в лабораторной работе).

В первом способе каждый запрос на выборку создается как обычно в режиме Конструктора, а затем эти запросы объединяются в режиме SQL.

**Отчеты**. Отчет является эффективным средством представления данных в печатном формате. Имея возможность управлять размером и внешним видом всех элементов отчета, пользователь может отобразить сведения желаемым образом. Большинство отчетов являются присоединенными к одной или нескольким таблицам и запросам из БД, так как источником записей отчета являются поля в базовых таблицах и запросах. Отчет не должен включать все поля из каждой таблицы или запроса, на основе которых он создается.

Отчет получает данные из базового источника записей, но другие данные, такие, как заголовок, подзаголовки, визы исполнителей, дата и номера страниц, сохраняются в Конструкторе отчета.

В MS Access используется несколько видов отчетов:

Одноколонный отчет (в столбец) – это столбец текста, содержащий названия полей и их значения из всех записей таблицы или запроса.

Многоколонный отчет – создается из отчета в одну колонку и позволяет вывести данные отчета в несколько колонок.

Табличный отчет – отчет, имеющий табличную форму.

Отчет с группировкой данных и подведением итогов – создается из табличного отчета объединением данных в группы с подсчетом итогов.

Перекрестный отчет – строится на основе перекрестных запросов и содержит итоговые данные.

Составной отчет – отчет, имеющий сложную структуру, включающий один или несколько отчетов.

Отчет, полученный слиянием документов с MS Word (составной документ).

Почтовые наклейки – специальный тип многоколонного отчета, предназначенный для печати имен и адресов групп.

Отчет по связанным таблицам – отчет, позволяющий выводить данные из нескольких таблиц, имеющих связь типа «один-ко-многим».

Отчет может состоять из следующих разделов:

Заголовок отчета – отображается только на первой странице и определяет высоту области заголовка отчета, который может содержать текст, графику и другие элементы управления.

Верхний колонтитул – выводится на верху каждой страницы; как правило, содержит заголовки столбцов.

Заголовок группы – печатается перед первой записью, входящей в группу; обычно содержит итоговые данные по записям, входящим в группу.

Область данных – предназначена для отображения записей источника данных отчета. Может содержать вычисляемые поля, предназначенные для отображения в отчетах значений выражений на основе исходных данных.

Область примечания группы – отображается после обработки последней записи группы; обычно содержит итоговые данные по записям, входящим в группу.

Нижний колонтитул – выводится внизу каждой страницы; содержит, например, номер страницы отчета, дату печати отчета и т. д.

Раздел примечаний – отображается только внизу последней страницы; сюда помещают поля с итоговыми значениями по всем записям, включенным в отчет.

Отчет может быть создан при помощи Мастера отчетов (автоотчет),  или в режиме Конструктора. Обычно используют оба способа. Мастер отчетов позволяет ускорить процесс создания отчета, работа в нем производится в пошаговом режиме в диалоге с пользователем. Доработать созданный мастером отчет можно в режиме Конструктора.

В режиме Конструктора доступны такие свойства отчета, как *Сортировка* *и группировка* данных, которые расположены на панели инструментов Конструктора отчетов. В диалоговом окне *Сортировка и группировка* можно определить поле или выражение, по которому будут сгруппированы данные, а также сгруппированные данные можно отсортировать по возрастанию или убыванию. Например, можно группировать данные двумя способами – по категориям и по диапазону значений, который может быть как числовым, так и алфавитным.

**Задание к лабораторной работе**

1.  Чтобы получить практические навыки, необходимо создать следующие запросы:

а) итоговый запрос, который выводит список читателей и общее количество заказанных ими книг. Для этого:

 создать новый запрос в режиме Конструктора;

 добавить таблицы «Читатели», «Выдача книг» и закрыть окно;

 в бланк запроса добавить из таблицы «Читатели» поле Фамилия, а из таблицы «Выдача книг» – Код книги;

 выбрать из меню Групповые операции. По умолчанию в новой строке бланка для всех полей устанавливается Группировка и при выполнении запроса итоги не подводятся;

 для получения итогового значения для поля Код книги в строке Групповая операция: выбрать функцию Count;

 в строке Поле: этого столбца ввести его новое имя, для чего перед именем столбца – Код книги ввести Общее кол-во книг:;

 установить сортировку по убыванию и сохранить запрос.

б) Создать параметрический запрос, который выводит в алфавитном порядке список читателей, заказавших книги в определенный интервал времени, который будет устанавливаться пользователем. Для этого:

  создать обычный запрос с помощью Конструктора;

 добавить таблицы «Читатели» и «Выдача книг», закрыть окно;

 в бланк запроса добавить из таблицы Читатели» поля Фамилия, Имя, Отчество, а из таблицы «Выдача книг» – Дата заказа;

 для поля Фамилия установить сортировку по возрастанию;

 в строке Условия отбора: для поля Дата заказа ввести выражение: Between [Введите начальную дату:] And [Введите конечную дату:];

 указать тип каждого параметра: выбрать в меню Параметры и в открывшемся окне в столбец Параметры ввести две строки с именем каждого параметра (Введите начальную дату: и Введите конечную дату:), а затем во втором столбце выбрать тип каждого параметра (Дата/Время) и закрыть окно;

сохранить и выполнить запрос.

Создать параметрический запрос, который выводит книги по первым буквам фамилии автора. Для этого:

 создать новый запрос в режиме Конструктора;

 добавить таблицы «Книги » и «Издательства»;

 выбрать поля Автор, Название, Год издания из таблицы «Книги » и Город, Название из таблицы «Издательства»;

 для поля Автор установить сортировку по возрастанию;

в строке Условия отбора: для поля Автор ввести выражение: Like [Введите первые буквы фамилии:] &"\*";

 указать тип данных параметра запроса – текстовый;

 сохранить и выполнить запрос.

в) Создать перекрестный запрос, который содержит список издательств и для каждого издательства подсчитывает и выводит количество книг, сгруппированных по годам издания. Для этого:

 создать новый запрос в режиме Конструктора;

 добавить таблицы «Книги» и «Издательства»;

 выбрать поля Год издания и Код книги из таблицы «Книги» и поле Название из таблицы «Издательства»;

 в меню выбрать вид запроса Перекрестный (в бланке запроса появятся две новые строки Групповая операция: и  Перекрестная таблица:).

 в строке Перекрестная таблица: для поля Название установить из выпадающего списка значение Заголовки строк, а для поля Год издания – Заголовки столбцов. В строке Групповая операция: в этих полях по умолчанию остается значение Группировка;

 для поля Код книги в строке Перекрестная таблица: установить Значение, а в строке Групповая операция: выбрать функцию Count;

 добавить итоговый столбец, для чего повторно добавить в бланк запроса поле Код книги. В строке Перекрестная таблица: для нового столбца установить Заголовок строк, а в строке Групповая операция: выбрать Count. Переименовать столбец в «Итого:»;

 сохранить и выполнить запрос.

г) Создать запрос на объединение названий книг и авторов в одном поле. Для этого:

1.  создать в Конструкторе, с учетом требований к полям запросов на объединение, обычный запрос на выборку или условный запрос, проверить его выходные данные; т.е. в данном случае создать запрос на выборку поля  *Название*  из таблицы «Книги»;

2.   сохранить запрос и не закрывать его;

3.  повторить шаги 1 и 2 для всех запросов на выборку, которые необходимо будет объединить; т.е. в данном случае создать второй запрос на выборку для поля *Автор* из таблицы «Книги»;

4.   создать новый запрос в Конструкторе запросов, сразу закрывая диалоговое окно *Добавление таблицы*. Найти на панели управления  и  нажать кнопку *Объединение* (рис. 19). При этом привычное окно Конструктора запросов будет скрыто, а отобразится пустое окно  режима SQL, в котором можно объединить запросы на выборку;



рис. 19

5.   переключить первый из сохраненных запросов на выборку в режим SQL, скопировать находящуюся там SQL инструкцию и вставить ее в пустое окно нового запроса. Удалить точку с запятой (;) в конце инструкции SQL;

6.   переместить курсор на строку ниже и ввести слово UNION в новой строке, снова переместить курсор;

7.   не удалять точку с запятой и не вводить ничего после инструкции SQL в последнем объединяемом запросе на выборку.

Запрос на объединение, формирующий общий список книг и авторов,  будет иметь вид:

*SELECT Книги.Название*

*FROM Книги*

*UNION*

*SELECT Книги.Автор*

*FROM Книги;*

2.  Самостоятельно создать и сохранить следующие запросы в базе данных «Библиотека».

а) Итоговые запросы:

1.  список городов и общая сумма стоимости книг по городам (вычисляемому полю присвоить имя *Общая стоимость книг*);

2.  список фамилий читателей и для каждого читателя – количество книг, заказанных им в 1997 г. (вычисляемому полю присвоить имя *Общее количество кн*иг);

3.  список наименований издательств, общее количество книг, изданных каждым издательством и средний объем страниц по издательству (вычисляемым полям присвоить имена *Общее количество книг* и *Среднее количество страниц*).

б) Параметрические запросы:

1. полная информация о книгах определенного года издания;
2. поиск книг по указанному ключевому слову (части слова) в теме. Запрос создать на основании таблиц «Книги», «Издательства», «Темы»;
3. список читателей, заказавших указанную книгу в указанный месяц или в текущем году.

в) Перекрестные запросы:

1. сводная таблица должна содержать фамилии читателей и для каждого читателя должно быть указано общее количество заказанных им книг, а также количество книг по каждому году;
2. сводная таблица должна содержать информацию о количестве выдач каждой книги за определенный год, а также суммарное количество выдач каждой книги по месяцам данного года;
3. сводная таблица должна содержать список всех издательств, общее количество изданных им книг, а также количество книг, изданных за каждый отдельный год. В качестве заголовков столбцов использовать все года по порядку от 1987 до 1995.

г) Запросы на объединение:

1. вывести общий список названий издательств и городов для книг, изданных после 1990 года;
2. вывести общий список фамилий читателей, которые брали книги в текущем году и авторов взятых ими книг;
3. вывести общий список всех годов выдачи заказов и месяцев, в которые брали книги читатели, фамилии которых начинаются в диапазоне букв от А до К.

3. По каждому типу запроса создать отчет с помощью Мастера, провести его более точную настройку в режиме Конструктора (добавить заголовок, дату, нумерацию страниц, подпись и т.п., подкорректировать размеры полей, недостаточных для отображения содержимого).

4.  Выполнить итоговый, параметрический, перекрестный запрос и запрос на объединение в индивидуальном задании, создать и настроить для них отчеты.

## **Контрольные вопросы**

1. Как создать итоговой запрос в конструкторе MS Access?
2. Какие функции используются для подведения итогов, как их называют?
3. Поясните логику работы итоговых запросов на примере функции *Sum*(), *Count*()?
4. Чем отличаются функции *Count*() и *Count*(\*)?
5. Как правильно создать запрос с параметром?
6. Что понимают под перекрестным запросом? Из каких основных компонентов он состоит?
7. Почему заголовки столбцов не могут быть использованы несколько раз при создании перекрестного запроса?
8. Какие особенности создания имеет запрос на объединение в MS Access?
9. Чем отличаются операторы UNION и UNION ALL в запросе на объединение?
10. Для чего используются отчеты? Какие виды отчетов используются в MS Access?

# **Лабораторная работа № 5 Создание форм. Привязка информационных полей через связи между таблицами**

## **Цель работы:** создание форм с помощью Мастера и элементов управления Конструктора. Подстановка данных из родительских таблиц в поля внешних ключей дочерних.

**Краткие теоретические сведения**

Обычно разработчик БД создает структуру таблиц и запросов, но заполнением таблиц информацией не занимается. Для этого есть специальные кадры, выполняющие функции наборщиков. Для упрощения их труда разработчик базы может подготовить специальные объекты – формы. Форма представляет собой электронный бланк, в котором есть поля для ввода данных. Наборщик вводит данные в  поля формы, и они автоматически заносятся в таблицы.

Данные в таблицу можно вносить и без помощи форм, но существуют, по крайней мере, четыре причины, которые делают формы незаменимым средством ввода данных в базу. Во-первых, малоквалифицированному персоналу нельзя предоставлять доступ к таблицам (самому ценному из того, что есть в базе). Представьте, что будет, если новичок «наведет порядок» в таблице банка, хранящей расчетные счета клиентов. Во-вторых, разные люди могут иметь разные права доступа к информации, содержащейся в таблицах. Например, один имеет право вводить только имена и адреса клиентов, другой – только номера расчетных счетов, а третий – только денежные суммы, хранящиеся на счетах. Для ввода данных им предоставляют разные формы, хотя данные из форм будут поступать в одну таблицу. В-третьих, ввод данных в таблицу – чрезвычайно утомительное занятие. Уже после нескольких часов работы люди делают ошибки. Ввод данных в форму можно автоматизировать таким образом, чтобы при вводе данных выполнялась их первичная проверка. В-четвертых, информация для ввода в БД, как правило, берется из бумажных бланков (анкеты, заявления, накладные, счета, описи, ведомости, справки и т. п.). Экранные формы можно сделать точной копией бумажных бланков, с которых происходит ввод данных. Благодаря этому во много раз уменьшается количество ошибок при вводе и значительно снижается утомляемость персонала.

**Режимы работы.** Работа с формами может происходить в трех режимах: в режиме Формы, в режиме Таблицы, в режиме Конструктора.

В режимах Формы и Таблицы можно осуществлять добавление, удаление и редактирование записей в таблице или в запросе, являющемся источником данных для формы.

В режиме Конструктора можно производить изменение внешнего вида формы, добавление и удаление элементов управления.

**Виды форм.** В MS Access можно создать формы следующих видов:

1.  Форма в столбец или полноэкранная форма, которая представляет собой совокупность определенным образом расположенных полей ввода с соответствующими им метками и элементами управления. Чаще всего эта форма используется для ввода и редактирования данных.

2.  Ленточная форма, которая служит для отображения полей группы записей. Поля не обязательно располагаются в виде таблицы, однако для одного поля отводится столбец, а метки поля располагаются как заголовки столбцов.

3.  Табличная форма, которая отображает данные в режиме таблицы.

4.  Форма главная/подчиненная, которая представляет собой совокупность формы в столбец и табличной. Ее имеет смысл создавать при работе со связанными таблицами, в которых установлена связь типа «один-ко-многим».

5.  Форма Сводная таблица выполняется Мастером создания сводных таблиц  Excel на основе таблиц и запросов MS Access (Мастер сводных таблиц является объектом, внедренным в MS Access, чтобы использовать его, необходимо установить MS Excel). Сводная таблица представляет собой перекрестную таблицу данных, в которой итоговые данные располагаются на пересечении строк и столбцов с текущими значениями параметров.

6.  Форма с диаграммой. В MS Access в форму можно вставить диаграмму, созданную в приложении MS Graph, которое является внедряемым OLE-приложением и может быть запущено из MS Access. С внедренной диаграммой можно работать так же, как и с любым объектом OLE.

**Структура формы.** Любой вид формы может включать следующие разделы:

* заголовок формы – определяет верхнюю часть формы и может содержать текст, графику и другие элементы управления;
* верхний колонтитул – раздел отображается только в режиме предварительного просмотра и обычно содержит заголовки столбцов;
* область данных – определяет основную часть формы, содержащую поля, полученные из источника данных;
* нижний колонтитул – раздел отображается только в режиме предварительного просмотра в нижней части экранной страницы и обычно содержит номер страницы, дату и т. д.;
* примечание формы – отображается внизу последней экранной страницы формы.

Форма может содержать все перечисленные разделы или некоторые из них.

Как и любой объект БД, форма имеет свойства. Значения этих свойств для всей формы, ее разделов или элементов управления задаются в окнах свойств соответствующих объектов. Для отображения на экране окна свойств нужно нажать кнопку *Свойства* на панели инструментов Конструктора форм или выбрать в меню, появляющемся при нажатии правой кнопки мыши на объекте.

Окно свойств выделенного объекта содержит следующие вкладки: *Макет*, с помощью которого задается макет формы; *Данные*, которые указывают источник данных и некоторые его параметры; *События*, которые отображают перечень свойств, связанных с программным управлением объектом; *Другие*, которые задают перечень остальных свойств и *Все* – перечень всех свойств.

Основные свойства формы это:

* подпись – позволяет задать название формы, которое будет выводиться в области заголовка;
* режим по умолчанию – определяет режим открытия формы (простая, ленточная, табличная формы);
* допустимые режимы – свойство, которое определяет, можно ли с помощью команд меню Вид переходить из режима формы в режим конструктора;
* свойства полосы прокрутки, область выделения, кнопки перехода, разделительные линии, кнопка оконного меню, размеров окна, кнопка закрытия, кнопка контекстной справки, тип границы – определяют, будут ли выводиться эти элементы в окно формы;
* свойства разрешить добавления, разрешить удаления, разрешить изменения – определяют, можно ли пользователю редактировать данные через форму. Эти свойства могут принимать значения Да/Нет;
* ввод данных – определяет режим открытия формы и принимает значения Да/Нет. Режим Да – открытие формы только для добавления новых записей. Режим Нет – просмотр существующих записей и добавление новых;
* блокировка записей – определяет способы блокировки записей в режиме многопользовательской работы с базой данных.

**Режимы создания.** Для создания форм используются следующие возможности:

1.  Автоформа – автоматизированное средство для создания форм трех стандартных типов: в столбец, ленточная, табличная. При этом в форму вставляются все поля источника данных.

2.  Мастер форм – программное средство, которое позволяет создавать структуру одного из трех стандартных типов формы в режиме диалога с разработчиком формы. При этом в форму вставляются выбранные пользователем поля из источника данных.

3.  Конструктор форм – позволяет конструировать форму пользователем в окне Конструктора форм.

Самым удобным способом создания новой формы является следующая технология: форма создается с использованием автоформы или Мастера форм, а затем дорабатывается в режиме Конструктора.

Источником данных формы являются одна или несколько связанных таблиц и/или запросов.

**Работа с элементами управления.** Элементом управления называют любой объект формы или отчета, который служит для вывода данных на экран, оформления или выполнения макрокоманд. Элементы управления могут быть связанными, вычисляемыми или свободными.

Связанный (присоединенный) элемент управления присоединен к полю базовой таблицы или запроса. При вводе значения в связанный элемент управления поле таблицы текущей записи автоматически обновляется. Поле таблицы является источником данных связанного элемента управления.

Вычисляемый элемент управления создается на основе выражений. В выражениях могут использоваться данные полей таблицы или запроса, данные другого элемента управления формы или отчета и функции.

Свободные элементы управления предназначены для вывода на экран данных, линий, прямоугольников и рисунков. Свободные элементы управления называют также переменными или переменными памяти.

Все элементы управления могут быть добавлены в форму или отчет с помощью панели инструментов, которая автоматически появляется при работе с формой или отчетом в режиме Конструктора. Размеры разделов и размеры рабочего поля формы можно изменять с помощью мыши.

В автоформах обычно присутствуют элементы управления двух типов: связанное поле и присоединенная надпись. Содержание присоединенной надписи совпадает с названием связанного поля, но это можно и изменить.

Редактирование форм состоит в создании новых или изменении имеющихся элементов управления, а также в изменении их взаимного расположения.

Например, создадим с помощью Мастера форму на основе какой-нибудь таблицы БД «Библиотека». Мастер не заполняет  раздел заголовка. Поэтому, перетащив вниз разделительную границу между заголовком и областью данных, можно освободить место для создания надписи. На панели элементов управления для создания заголовков существует элемент *Надпись*. Выбрав его и установив в нужном месте раздела заголовка, получим текстовую рамку, в которую можно вводить произвольный текст.

Для форматирования элемента управления его надо сначала выделить. При выделении элемента управления вокруг него образуется рамка, которая содержит восемь маркеров (по углам и по центрам сторон рамки). Рамку можно растягивать или сжимать методом перетаскивания границ. При наведении на маркер указатель мыши меняет форму, принимая вид открытой ладони. В этот момент рамку можно перемещать. Когда объект выделен, можно изменять параметры шрифта, метод выравнивания текста и другие элементы форматирования, доступные через соответствующую панель инструментов MS Access. Если щелкнуть на выделенном элементе правой кнопкой мыши, откроется его контекстное меню, в котором имеются дополнительные возможности изменения оформления.

**Создание и редактирование связанных полей**.Для создания связанных полей используется элемент *Поле* на панели инструментов. При создании связанного поля вместе с ним одновременно образуется еще один элемент управления – *Присоединенная надпись*. Она перемещается вместе со связанным полем и образует с ним единое целое. Оторвать поле от присоединенной надписи позволяет левый верхний маркер рамки, при наведении на который указатель мыши принимает форму указательного пальца. В этот момент связанное поле можно оторвать от присоединенной надписи и перемещать отдельно.

Если перемещать элементы управления и изменять их размеры с помощью мыши не очень удобно, то используют для этой цели клавиши перемещения курсора в комбинации с SHIFT или CTRL. В первом случае (с SHIFT) происходит изменение размеров элемента управления, а во втором (с CTRL) – изменение его расположения. Чтобы элементы управления располагались в форме ровными рядами, существуют специальные команды выравнивания. Сначала надо выделить группу элементов управления (группа выбирается при нажатой клавише SHIFT), а потом применить команду *Выровнять* и выбрать метод выравнивания.

**Элементы управления формы**.При создании формы вручную элементы управления размещают на ней так, как удобно проектировщику. Кроме рассмотренных выше элементов управления, существуют следующие:

1)  *Переключатель* (группа переключателей) – с ними можно связать команды, например, выполняющие фильтрацию;

2)  *Флажок* – действуют аналогично переключателям, но, в отличие от них, допускают множественный выбор, удобны для управления режимами сортировки данных;

3)  *Список* – может содержать фиксированный набор значений или значения из заданного поля одной из таблиц, позволяет не вводить данные, а выбирать их из списка;

4)  *Поле* *со* *списком* – применяется так же, как и список, но занимает меньше места в форме, поскольку список открывается только после щелчка по кнопке выбора;

5)   *Кнопка* – с каждой можно связать команду, например, команду поиска записи, перехода между записями и др.

6)   *Вкладки* – позволяют разместить много информации на ограниченной площади, на них размещают другие элементы управления;

7)  *Поле объекта OLE* – служит для размещения внешнего объекта, соответствующего принятой в Windows концепции связывания и внедрения объектов. Объектом, как правило, является иллюстрация, например фотография, но это может быть и видеозапись, и музыкальный фрагмент, и голосовое сообщение.

Существуют два типа полей для размещения объектов OLE: свободная рамка объекта и присоединенная рамка объекта. В первом случае рамка не связана ни с каким полем таблиц БД. Объект, находящийся в ней, выполняет роль иллюстрации и служит для оформления формы. С присоединенной рамкой связано одно из полей таблицы. В ней отображается содержимое этого поля. Это содержимое может меняться при переходе от одной записи к другой.

Как правило, выбор любого элемента управления и перемещение его на форму сопровождается появлением соответствующего Мастера.

Например, при выборе элемента управления *Кнопка* открывается диалоговое окно Мастера кнопок, показанное на рис. 20.

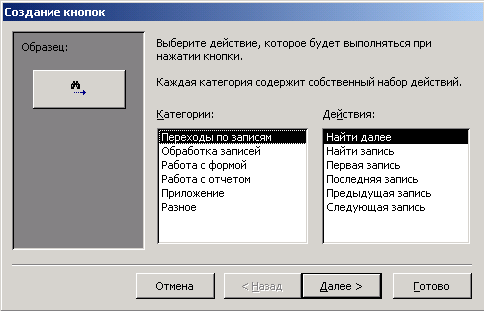


 Рис. 20. Окно мастера создания кнопок

**Подстановка данных из родительских таблиц в поля внешних ключей дочерних*.*** Таблица «Выдача книг» формируется на основе ключевых полей других таблиц и имеет следующий вид (табл. 4):

Таблица 4

Таблица «Выдача книг»

|  |  |  |  |
| --- | --- | --- | --- |
| Код выдачи | Код читателя | Код книги | Дата заказа |
| 1 | 1 | 1 | 01.09.96 |
| 2 | 1 | 2 | 01.03.97 |
| 3 | 1 | 5 | 20.04.97 |
| 4 | 2 | 1 | 04.11.95 |

Очевидно, что такой вид таблицы неудобен для заполнения. В MS Access предусмотрены элементы управления, позволяющие привести таблицы к виду, представленному в табл. 5:

 Таблица 5

Таблица «Выдача книг», форма для заполнения

|  |  |  |  |
| --- | --- | --- | --- |
| Код выдачи | Код читателя | Код книги | Дата заказа |
| 1 | Минкевич | Беспалько | 01.09.96 |
| 2 | Минкевич | Сканави | 01.03.97 |
| 3 | Минкевич | Грибанов | 20.04.97 |
| 4 | Гуляев | Беспалько | 04.11.95 |

Рассмотрим привязку информации на примере таблиц «Читатели» и «Выдача книг». Нужно, чтобы значения поля *Код читателя* (числовой тип  данных) таблицы «Выдача книг» заменялись соответствующими значениями поля *Код читателя* таблицы «Читатели» (тип данных «Счетчик»).

Для этого нужно открыть таблицу «Выдача книг» в режиме Конструктора, выбрать поле *Код читателя* и перейти в нижней части окна на вкладку *Подстановка*. Выбрать тип элемента управления *Поле со списком*. В списке свойств в качестве *Источника строк* выбрать таблицу «Читатели». Таблица «Читатели» содержит поля *Код читателя*, *Фамилия*, *Имя*, *Отчество*, *Адрес*, *Домашний телефон*. При оформлении выдачи книги достаточно видеть, например, поля  *Фамилия* и *Домашний телефон.* Для этого в строке *Ширина столбцов* вводятся нулевые значения размеров тех столбцов, которые отображать не планируется. Окно параметров подстановки в данном случае будет иметь вид (рис. 21).

|  |  |
| --- | --- |
| Тип элемента управления | Поле со списком |
| Тип источника строк | Таблица или запрос |
| Источник строк | Читатели |
| Присоединенный столбец | 1 |
| Число столбцов | 6 |
| Заглавия столбцов | Да |
| Ширина столбцов | 0см; 3см;0см; 0см; 0см; 4см |
| Число строк списка | 8 |
| Ширина списка | 5см |
| Ограничиться списком | Да |
| Разрешение нескольких значений | Нет |
| Разрешить изменение списка значений | Нет |
| Форма изменения элементов списка |  |
| Только значения источника строк | Нет |

Рис. 21. Параметры подстановки для поля «Код читателя»

Аналогично настраиваются свойства других связанных полей.

На практике структуры таблиц изменяют редко, так как работают с данными через формы. Поэтому можно в Конструкторе форм заменить элемент управления *Поле* на *Поле со списком* и установить в окне свойств последнего параметры подстановки аналогично рассмотренному примеру.

## **Задание к лабораторной работе**

1.  Подставить реальные данные о читателях (*Фамилия*, *Имя*, *Отчество* и *Домашний телефон*), о книгах (*Название*, *Автор*) в таблице «Выдача книг»; об издательствах (*Название*, *Город*) в таблице «Книги», о книгах (*Название*, *Автор*) в таблице «Темы».

2.  Создать автоформы по всем таблицам БД «Библиотека», доработать их в режиме Конструктора, изменяя макет формы в соответствии с содержимым полей и добавляя новые элементы управления (по своему усмотрению, но так, чтобы использовались *Флажки*, *Переключатели*, *Списки*, *Объекты* *OLE*).

3.  Создать форму для просмотра и печати отчетов по ранее выполненным видам запросов. Для этого выбрать из лабораторных работ №3 и №4 пять запросов, по одному каждого вида (условный, итоговый, параметрический, перекрестный, запрос на объединение).

4.  Создать в Конструкторе две новые формы на основе необходимых полей таблиц БД «Библиотека», которые позволят:

а) осуществлять в наиболее удобной форме оформление нового заказа на книги (предусмотреть выбор существующего читателя из списка или добавление нового в таблицу «Читатели» с возвратом к оформлению заказа; в поле *Дата заказа* использовать календарь или текущую дату по умолчанию);

б) осуществлять выбор из *Поля со списком* интересующей темы и вывод с помощью элемента управления *Подчиненная форма* информации о соответствующей книге, как показано на рис. 22.

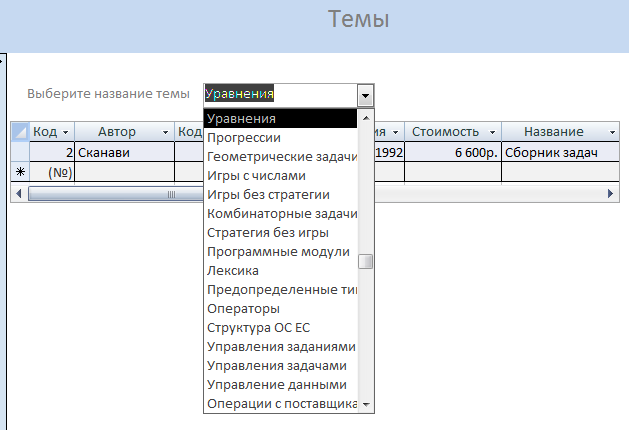


Рисунок 22. Работа с подчиненной формой

5.  Создать главную форму, с которой будет осуществляться переход на все другие объекты БД. Предусмотреть возврат из второстепенных форм на главную. На главной форме должны размещаться:

а) кнопка перехода на форму, с которой возможно просмотреть автоформы всех таблиц БД «Библиотека» (переход между автоформами должен быть осуществлен с учетом связей между таблицами БД);

б) кнопка перехода на форму просмотра отчетов по запросам;

в) кнопки перехода на формы из пункта 4;

г) кнопка выхода из приложения;

д) кнопка просмотра информации о БД и ее разработчике.

По желанию, главная форма может быть построена с помощью элемента управления *Вкладка*.

6. Создать аналогичные формы к БД индивидуального задания, которые позволят автоматизировать работу с данными: добавлять, удалять и модифицировать записи в любой таблице без применения стандартных панелей MS Access, запускать созданные запросы, просматривать и печатать отчеты, осуществлять поиск и просмотр информации в подчиненных формах. Все коды связанных полей на стороне отношения «многие» должны быть заменены при помощи подстановки на данные исходных родительских таблиц.

## **Контрольные вопросы**

1. Что представляет собой форма?
2. В чем заключается смысл использования форм?
3. Поясните алгоритм замены внешних ключей дочерних таблиц данными из родительских таблиц?
4. В каких режимах можно работать с формами?
5. Какие виды форм можно создать в MS Access?
6. Какие разделы могут входить в форму?
7. Перечислить основные свойства форм
8. Какие элементы управления называют связанными?
9. Чем отличаются и для чего используются элементы управления «поле» и «поле со списком»?
10. Что называется подчиненной формой, для чего ее используют на практике?

# **Лабораторная работа № 6 Написание процедур обработки событий на VBA**

## **Цель работы:** изучить основные принципы написания программ на VBA, создать форму для удаления существующих и добавления новых полей в таблицы БД «Библиотека».

### **Краткие теоретические сведения**

Структурированный полуфункциональный язык программирования высокого уровня VBA является неотъемлемой составной частью MS Access. Он используется для разработки приложений, которые управляют данными в базе, а также для создания и настройки пользовательского интерфейса. В нем, как и в других языках, есть операторы проверки условий, циклического выполнения операций, обмена данными с памятью и дисками. В языке VBA реализованы общие принципы объектно-ориентированного программирования. Это означает, что пользовательская среда, управляемая приложением, не подвергается изменениям путем выполнения последовательности процедур и операторов, но реагирует на события, связанные с различными объектами: полями ввода, кнопками, разделами форм и отчетов. В языке VBA программный код привязан непосредственно к объектам и срабатывает тогда, когда случается определенное событие.

**Создание кода VBA с помощью Мастеров элементов управления***.* Одним из самых распространенных способов использования VBA является привязка кода к элементам управления, добавленным на форму при ее создании, например к кнопке. Чтобы просмотреть код, необходимо выполнить следующее:

1.  открыть любую форму, в которой есть кнопки, в режиме Конструктора (например, пусть на форме будет размещено две кнопки – кнопка\_13 и кнопка\_14, одна из которых осуществляет переходы по записям, а вторая закрытие формы);

2.   выбрать команду *Программа* из меню Вид или найти на панели инструментов иконку редактора VBA (в рассматриваемом примере окно редактора VBA будет иметь вид, представленный на рис. 23).

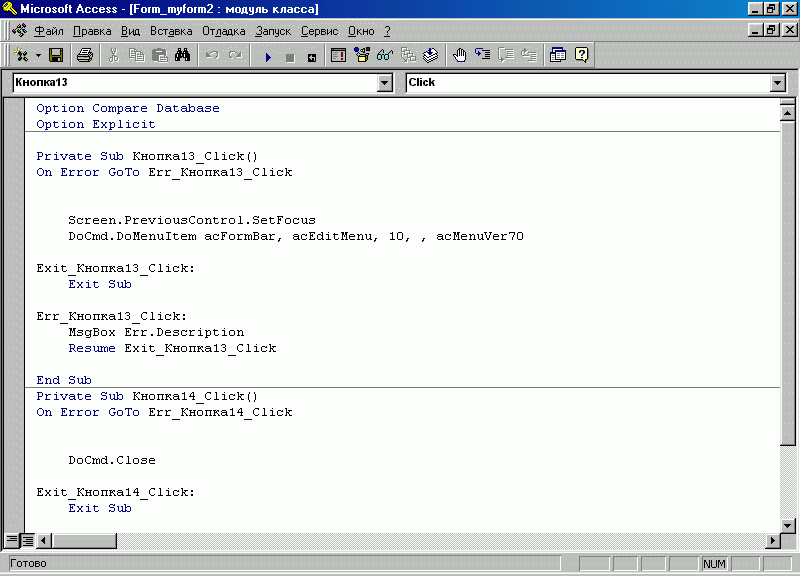


Рис. 23. Окно редактирования программных модулей

В этом окне отображается три раздела программы на языке VBA: первый раздел – это раздел объявлений, который может содержать код, относящийся ко всей форме в целом (в данном случае пуст), второй раздел относится к кнопке\_13, третий – к кнопке\_14.

Последние два раздела начинаются с заголовка процедуры (Private Sub Кнопка13\_Click () ) и заканчиваются оператором: End Sub.

Операторы, заключенные между ними, выполняются после щелчка по этой кнопке при работе с формой. Основная часть работы выполняется оператором:

DoCmd.DoMenuItem acFormBar, AcEditMenu,10, AcMenuVer70

Этот оператор выполняет те же действия, что и команда меню *Найти*, т.е. выводит на экран диалоговое окно *Поиск в поле*.

Открыв, таким образом, программу в редакторе VBA, ее можно редактировать как текст в любом текстовом редакторе. В окне редактора в код можно вставить текст из другого файла, можно вывести текст программы на печать и т.д.

**Модули.** Модуль является структурным элементом программы, написанной на языке VBA. Это совокупность объявлений и процедур, объединенных в единое целое. Существуют модули трех типов: стандартные модули, модули формы и модули отчетов. В отличие от стандартного модуля, который создается таким же образом, как и любой другой объект БД, и может выполнять практически любые вычисления, модули форм и отчетов разрабатываются для обработки событий, связанных с элементами формы или отчета.

Каждый модуль состоит из области описания и одной или нескольких процедур. Входящие в модуль процедуры объединены общей областью описания. В ней описываются данные и объекты, которые являются общепринятыми для процедур модуля.

Процедуры делятся на две категории: процедуры-подпрограммы (подпрограммы) и процедуры-функции (функции).

Процедура-подпрограмма активизируется при обращении к ней по имени, вследствие чего выполняется определенная последовательность операторов (инструкций). Подпрограмму используют, например, для задания свойства формы или заполнения списка значениями, полученными в результате вычислений.

Процедура-функция после выполнения возвращает некоторое значение, которое можно применять в операторах и выражениях в качестве переменной. Например, функции могут возвращать значение, используемое по умолчанию для некоторого поля, или вычислять сложный критерий в рамках запроса. В модуль можно включать любое количество функций и подпрограмм.

Каждая процедура состоит из операторов, в которых применяются встроенные в MS Access функции, методы и свойства, предназначенные для выполнения каких-либо операций над данными. Например, к свойству Click кнопки \_13 приведенного ранее примера привязана следующая процедура:

Private Sub Command6\_Click ( )

On  Error GoTo Err\_Command6\_Click

        Screen.PreviosControl.SetFocus

DoCmd.DoMenuItem acFormBar, acEditMenu, 10, ,  acMenuVer70

Exit Command6\_Click:

        Exit Sub

Err\_Command6\_Click:

        MsgBox Err.Descrition

        Resume Exit\_Command6\_Click

End Sub

Для обращения к объектам, а также для обозначения операторов используются ключевые слова, которые записываются с прописной буквы (например, Function).

Теоретически каждая процедура может быть вызвана из любого модуля, а функция – из таких объектов, как форма, запрос или отчет. Наряду с общедоступными процедурами (Public), которыми являются все процедуры по умолчанию, существуют локальные процедуры (Private), доступные только в том модуле, в котором они описаны.

Для передачи значений из вызывающих операторов в вызываемые процедуры служат аргументы. С помощью аргументов ведется контроль выполнения процедуры, устанавливается способ получения результата, определяются параметры вычислений и т.д.

**Создание процедур обработки событий***.* Начинающим разработчикам приложений MS Access понадобится набор процедур для отклика на различные события, например, внесение изменений в поле или щелчок на кнопке. Код процедуры обработки события привязывается к событию, которое может произойти при работе с формой, отчетом или элементом управления. Результат выполнения процедуры-функции обычно применяется:

в качестве значения по умолчанию для поля таблицы;

в качестве значения критерия для запросов или фильтров;

 в качестве содержимого поля.

Конкретный вид процедуры зависит от события и желаемого отклика, однако можно предложить общую последовательность действий по созданию процедуры обработки события:

1.  Открыть нужную форму или отчет в режиме Конструктора. Если необходимо привязать процедуру к элементу управления, выбрать этот элемент.

2.  Выбрать правой кнопкой мыши команду *Свойства*, чтобы открыть окно свойств формы, отчета или элемента управления.

3.  Перейти на вкладку *События*.

4.  Выбрать событие, которое должно запускать процедуру. Например, если процедура должна запускаться в ответ на изменение пользователем данных, выбрать строку *После обновления*.

5.  Выполнить щелчок по кнопке с тремя точками, чтобы открыть окно «Построитель».

6.  В этом окне выбрать *Программы*. Откроется окно редактора VBA, в котором автоматически появятся начальная и конечная строки новой процедуры.

7.  Написать программный код процедуры.

Все операторы процедуры записываются с отступом от начала строки. Для создания отступа можно использовать клавишу Tab. По умолчанию позиции табуляции установлены через четыре символа. Посредством отступов выделяют вложенные циклы и условные операторы внутри процедуры.

Тексты программ лучше снабжать комментариями. В начале каждой строки комментария ставится апостроф, и такие строки, как известно, не влияют на выполнение программы, а при синтаксическом анализе и компилировании – пропускаются.

Можно использовать различные шрифты и цвета для выделения фрагментов текста модуля.

**Редактирование процедур***.* Код модуля и тексты подпрограмм/функций редактируются так же, как документы в обычном текстовом редакторе. Для поиска процедуры в тексте модуля используется список процедур, находящийся в верхней части окна модуля.

При перемещении курсора на новую строку программа автоматически проверяет синтаксис введенной строки и в случае обнаружения ошибки выводит на экран соответствующее сообщение.

В MS Access при редактировании кроме общепринятых используются дополнительные комбинации клавиш:

Ctrl+Y – позволяет вставить в буфер строку, в которой находится курсор, не выполняя его маркировку;

F3 или  Shift + F3 – позволяют просмотреть все фрагменты модуля, в которых встречается искомая последовательность символов (дублируют команду *Найти далее*).

**Использование аргументов***.* Благодаря аргументам пользователь имеет возможность управлять выполнением программы. Например, ввод коэффициента при пересчете рублей в доллары в функции с именем *Рубли\_в\_Доллары*:

Function Рубли\_в\_Доллары (Коэффициент)

При вызове такой функции значение аргумента указывается в скобках после имени функции. Например:

=Рубли\_в\_Доллары (4500)

Вызов процедуры, имеющей аргументы, должен сопровождаться заданием значений для всех объявленных аргументов. Эти значения перечисляются в том же порядке, что и аргументы при объявлении, и разделяются запятыми. В пользовательских функциях или модулях в качестве аргументов могут применяться выражения.

**Основные операторы и переменные VBA***.* Переменные используются для сохранения значений величин, изменяющихся в процессе выполнения программы. Каждая переменная имеет имя, по которому к ней обращаются. Правила присвоения имен переменных аналогичны правилам присвоения имен для полей. Присвоение значения переменной осуществляется посредством оператора присваивания. В левой части оператора  указывается имя переменной, а в правой – значение или выражение.

Переменные могут создаваться автоматически, по мере появления в процедуре (неявное объявление). По умолчанию неявно объявленные переменные имеют тип Variant. Внутри такой переменной кроме значения хранится индикатор типа значения.

Неявное объявление типа допустимо только в случае отсутствия оператора Option Explicit в области описания модуля.

Рекомендуется описывать все переменные явно, это позволит избежать ошибок, связанных с преобразованием типов данных. Явное описание типа производится оператором Dim, после которого указывается имя переменной и ее тип. Если при явном объявлении переменной не указан тип данных, а задано только имя переменной, то будет создана переменная типа Variant, которой можно присваивать цепочки символов, числа с плавающей точкой, значения даты и времени. MS Access однозначно идентифицирует присваиваемые значения, поскольку наряду с ними он хранит в переменной и признак типа.

Типы данных и размер хранимых значений, которые используются в MS Access, можно найти в справочной информации приложения.

Для объявления нескольких переменных можно пользоваться одним оператором Dim, перечисляя их через запятую.

При объявлении переменной следует помнить об области действия. Переменные, объявленные в процедуре, можно использовать только внутри этой процедуры. Если переменная должна быть доступна во всех процедурах одного модуля, ее необходимо объявить в области описания модуля. Здесь также можно применять оператор Dim. Наибольшую область действия имеет переменная, объявленная глобальной (с префиксом Global) в области описания модуля. Синтаксис оператора Global не отличается от синтаксиса оператора Dim.

В отличие от переменной константа содержит фиксированное значение, которое не может быть изменено в процессе выполнения программы. Константы, как и переменные, следует объявлять явно, указывая их имена и значения. Для объявления константы используется оператор Const. Объявленную константу можно использовать в программе, обращаясь к ней по имени. Например:

Const Число\_Пи = 3.1415926

*Условные операторы.*В языке VBA, как и в других языках программирования, основными элементами, управляющими ходом выполнения процедуры, являются условные операторы. Наиболее простой из них – оператор If … Then:

If  Полина = "Ж" Then

  Поздравление = "С 8 марта!"

End if

Если условие, заданное выражением между ключевыми словами If и Then, выполняется, соблюдаются инструкции внутри блока, ограниченного ключевыми словами Then и End if. Иначе операторы между ключевыми словами не выполняются, а программа переходит к обработке оператора, который следует за ключевыми слова End if. В случае необходимости произвести два различных действия (одно при соблюдении условия, а второе – нет), надлежит воспользоваться полной формой оператора If:

 If  выражение Then

   Оператор1

 Else

   Оператор2

 End if

Если условие соблюдается, выполняется Оператор1 (или группа операторов), расположенный между ключевыми словами Then и Else, а если не соблюдается – Оператор2 (или группа операторов), расположенный между ключевыми словами Else и End if.

Для выбора одного из множеств вариантов решений используется оператор Select Case:

 Select Case Город

   Case "Киев"

     Код\_МГТС = "044"

   Case "Москва"

     Код\_МГТС = "095"

   Case "Запорожье"

     Код\_МГТС = "061-2"

   Case Else

     Print "Я не знаю такого города!"

 End Select

При выполнении этого оператора проверяется значение переменной *Город*. В зависимости от результата проверки переменной *Код\_МГТС* присваивается телефонной код некоторого города. Если значение переменной *Город* не совпадает ни с одним из значений, перечисленных в строках с ключевым словом Case, производится действие, указанное между ключевыми словами Case Else и End Select.

*Операторы цикла.* Цикл используется для многократного повторения одной или нескольких инструкций. Количество повторений цикла связано с некоторым условием. В VBA предусмотрено несколько видов циклов. Простейшим примером циклической конструкции является цикл по счетчику:

For Счетчик = 1 To 10

    Print Счетчик

Next Счетчик

Цикл по счетчику ограничивается ключевыми словами For и Next. После ключевого слова For указывается имя переменной, которая будет выполнять роль счетчика, после знака равенства – начальное значение счетчика, а после ключевого слова To – конечное значение счетчика.

Еще один вид цикла – Do While. Условие выполнения команд внутри такого цикла определяется некоторым условным оператором:

Do While Счетчик <> 10

  Print

Loop

Разновидностью цикла Do является Until, который выполняется, пока условное выражение ложно:

Do Until Счетчик=10

    Print

  Счетчик=Счетчик+1

Loop

**Макросы.** Макросом называется набор из одной или более макрокоманд, выполняющих определенные операции. Выполнение макроса возможно:

из окна базы данных: на вкладке Макросы следует нажать кнопку Запуск;

при возникновении определенного события некоторого объекта, например, нажатие кнопки. В этом случае макрос выполняется автоматически.

Макрос является удобным средством выполнения простых задач, таких как открытие или закрытие форм, вывод на экран и скрытие панелей инструментов, запуск отчетов и т. д.

Макросу присваивается заданное пользователем имя. Если макросу присвоить стандартное имя, например, Autoexec, то такой макрос будет выполняться автоматически сразу же после загрузки MS Acces.

Макрос может состоять из одной или нескольких макрокоманд. Процедура создания макроса в основном состоит в выборе макрокоманд и определении их параметров. Для создания макроса следует:

 в окне БД на вкладке Макросы нажать кнопку *Создать*;

на панели макрокоманд в ячейке столбца *Макрокоманда* выбрать из раскрывающегося списка имя макрокоманды;

в ячейку столбца *Примечание* ввести текст комментария к макрокоманде (необязательно);

в нижней половине окна указать аргументы макрокоманды.

Имеющиеся в MS Access макрокоманды можно классифицировать по назначению следующим образом:

макрокоманды перемещения по данным;

макрокоманды выполнения макроса, процедуры или запроса;

макрокоманды прерывания выполнения;

макрокоманды экспорта-импорта;

макрокоманды работы с объектами;

прочие макрокоманды.

Макрокоманды выполняются в порядке их расположения в бланке. Однако MS Access позволяет задать условие выполнения определенных макрокоманд. Условие задается в столбце *Условие* (для вывода столбца следует выполнить команду *Условие* меню *Вид*) и представляет собой условное выражение. Условное выражение можно ввести с клавиатуры или задать с помощью построителя выражений. В ячейках столбца *Условие* нельзя использовать выражения SQL.

При запуске макроса проверяется значение каждого условного выражения. Если это выражение истинно, выполняется макрокоманда, находящаяся в той же строке, и все следующие макрокоманды, у которых в ячейках столбца Условие содержится многоточие (...). Если условное выражение ложно, MS Access проигнорирует эти макрокоманды и перейдет к ближайшей строке, в которой содержится новое условие или ячейка условий является пустой.

MS Access позволяет объединять группу родственных макросов в один макрос. В этом случае макет макроса (окно макроса в режиме конструктора) должен содержать столбец Имя макроса. Чтобы его вывести, следует выполнить команду *Имена макросов* меню *Вид*. Для запуска макроса из группы макросов следует указать: *Имя Группы макросов. Имя Макроса*.

## **Задание к лабораторной работе**

1. Написать программный код для добавления полей в таблицу, следуя приведенному ниже примеру (поля будут добавляться только в таблицу «Книги»):

а) Создать новую форму в режиме Конструктора, в которой будут размещаться элементы управления, предназначенные для добавления поля.

б) Выбрать элемент управления *Поле* на панели элементов управления, предназначенный для ввода имени добавляемого поля и поместить его в области данных формы.

в) Изменить свойства созданного элемента управления, для чего правой кнопкой мыши щелкнуть по нему и выбрать в появившемся списке - *Свойства*. В качестве имени поля указать *Field\_Name*.

г) Отредактировать надпись слева от созданного поля ввода. В свойствах изменить подпись на *Название поля:*. Для того, чтобы подпись полностью вмещалась ширину поля указать 2,6 см.

д) Создать элемент управления, в котором будет содержаться перечень возможных типов данных для добавляемого поля. Для этого выбрать на панели элементов *Поле со списком* и поместить его на форму. Следуя за работой Мастера в предложенном списке способов заполнения элементов списка выбрать *Будет введен фиксированный набор значений* и нажать кнопку *Далее*. Число столбцов оставить равным единице. В качестве элементов списка указать «Текстовый», «Числовой», «Дата/время». Нажать кнопку *Далее*, в качестве подписи указать *Тип поля:* и нажать кнопку *Готово*.

е) В свойствах созданного поля со списком изменить *Название поля* на *Field\_Type*.

ж) Создать кнопку, с которой будет связана программа добавления поля. Выбрать элемент управления *Кнопка* и поместить ее на форме. При работе с Мастером будет предложен список действий, которые должны выполняться при нажатии на данную кнопку. Поскольку  готовой программы для добавления полей нет, то нажать кнопку *Отмена*.

з) Открыть свойства созданной кнопки. *Подпись* изменить на *Добавление поля*, *Имя* на *Add*. На вкладке *События* выбрать строку *Нажатие кнопки*. Слева от поля нажать кнопку с тремя точками и в появившемся диалоговом окне выбрать *Программы*, нажать кнопку *ОК*. Открывшееся окно редактирования программ VBA содержит следующие строки:

Option Compare Database

Option Explicit

Private Sub Add\_Click ()

End Sub

Установить курсор в пустую строку после строки Private Sub Add\_Click ().

и) Далее необходимо получить доступ ко всей БД в целом и к нужной таблице в частности, предусмотреть, где будет содержаться информацию о новом поле. Для этого объявить следующие переменные:

Dim dbs As Database, tdf As TableDef, fld As Field

Переменная dbs имеет тип Database и предназначена для получения указателя на базу данных, переменная tdf имеет тип TableDef – для получения указателя на таблицу, переменная fld имеет тип Field – для получения указателя на поля таблицы.

к) Установить указатель на текущую базу данных:

Set dbs = CurrentDb

Ключевое слово CurrentDb означает, что будет использоваться информация из текущей БД.

л) Установить указатель на таблицу:

Set tdf = dbs.TableDefs ("Книги")

Обратите внимание, что ввод точки после dbs приводит к выводу перечня переменных и функций текущей БД. Выбранная функция TableDefs предназначена для получения указателя на указанную таблицу.

м) В зависимости от выбранного типа данных организовать создание нового поля в таблице. Для организации условного перехода внутри программы воспользуемся оператором Select Case:

Select Case Field\_Type.Value

 Case "Текстовый"

  Set fld=tdf.CreateField (Me.Field\_Name, dbText)

 Case "Числовой"

  Set fld=tdf.CreateField (Me.Field\_Name, dbLong)

 Case "Дата/время"

  Set fld=tdf.CreateField (Me.Field\_Name, dbDate)

End Select

В зависимости от значения поля со списком Field\_Type происходит создание поля с выбранным типом. Для создания поля используется метод  CreateField. Созданное поле размещается в переменной fld. Ключевое слово Ме означает, что следующее за ним поле принадлежат текущей форме.

н)  добавить новое поле в таблицу с помощью команд:

tdf.Fields.Append fld

tdf.Fields.Refresh

Первая команда предназначена для добавления информации о новом поле в таблицу, вторая – для обновления произведенных изменений.

Полностью программа имеет следующий вид:

Private Sub Add\_Click ()

Dim dbs As Database, tdf As TableDef, fld As Field

Set dbs = CurrentDb

Set tdf = dbs.TableDefs ("Книги")

Select Case Field\_Type.Value

 Case "Текстовый"

  Set fld=tdf.CreateField (Me.Field\_Name, dbText)

 Case "Числовой"

  Set fld=tdf.CreateField (Me.Field\_Name, dbLong)

 Case "Дата/время"

  Set fld=tdf.CreateField (Me.Field\_Name, dbDate)

End Select

tdf.Fields.Append fld

tdf.Fields.Refresh

End Sub

о) после написания программы закрыть текущее окно, сохранить форму и открыть ее в режиме просмотра.

р) проверить работу программы: в качестве названия поля ввести «Test», в качестве типа указать «Текстовый». Нажать кнопку «Добавление поля», а затем открыть таблицу «Книги» и убедиться в присутствии нового поля.

2. Аналогично написать программу для удаления полей из таблицы «Книги»:

а) Открыть форму в режиме Конструктора.

б) Выбрать элемент управления *Список* на панели элементов управления для отображения названий полей таблицы и поместить его в области данных формы. При работе с Мастером из предлагаемых способов заполнения элементов списка выбрать *Будет введен фиксированный набор значений* и нажать кнопку *Далее*. Число столбцов оставить равным единице. В качестве элементов списка указать поля *Автор*, *Название*, *Код издательства*, *Объем*, *Год издания*, *Стоимость*, *Test*. Поле *Код книги* лучше не добавлять, так как оно является ключевым и не может быть удалено из таблицы. Нажать кнопку *Далее*, в качестве подписи указать *Список полей* и нажать *Готово*. При необходимости выровнять размер подписи.

в) В свойствах созданного *Списка* изменить его *Имя* на *Fields*.

г) Создать кнопку, которая будет связана с  программой удаления поля. На панели элементов выбрать элемент управления *Кнопка* и поместить ее на форме. При работе с Мастером нажать *Отмена,* когда будет предложено выбрать выполняемое действие при нажатии на кнопку.

д) В свойствах созданной кнопки *Подпись* изменить на *Удаление поля*, *Имя* на *Del* и открыть программу обработки события в строке *Нажатие кнопки*. Установить курсор после строки Private Sub Del\_Click (). Написать программу обработки события:

Private Sub Del\_Click ()

Dim dbs As Database, tdf As TableDef

Set dbs = CurrentDb

Set tdf = dbs.TableDefs ("Книги")

tdf.Fields.Delete (Me.Fields.Value)  
tdf.Fields.Refresh  
End Sub

е) Сохранить форму и открыть ее в режиме просмотра.

ж) выбрать созданное ранее поле *Test* и нажать кнопку *Удаление поля*. В  таблице «Книги» убедиться, что поле удалено.

3. Создать форму (рис. 24) , в которой:

а) Реализовать добавление и удаление любого поля (кроме ключевых, ) из любой таблицы. При попытке удалить ключевое поля перелавливать сообщение MSAccess о невозможности этого действия и заменять его своим.

б) В качестве возможных типов данных добавляемых полей должны быть: «Текстовый», «Числовой», «Дата/время», «Денежный», «Логический», «Поле MEMO».

в) Перед удалением поля (при нажатии соответствующей кнопки *Удалить*) программно реализовать запрос на подтверждение удаления выбранного поля в виде пользовательского сообщения в диалоговом окне (функция *MsgBox*). Поле удалять только в случае согласия.

г) Обеспечить обновление *Списка* полей таблицы и поля ввода имени нового поля на форме после добавления/удаления полей в выбранной таблице.

д)  Предусмотреть программную защиту и вывод пользовательских сообщений (функция *MsgBox*) при попытке удаления ключевых полей; при  добавлении уже существующих полей; при попытке добавления полей с неуказанным именем или типом данных.

е) запрограммировать возможность открытия любой таблицы из созданной формы для просмотра внесенных изменений.

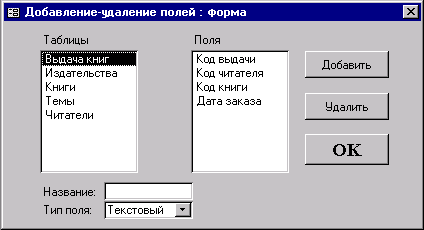


Рис. 24 Форма для добавления и удаления полей

4. Создать аналогично п.3 форму-мастер и под нее на VBA программу, позволяющую добавлять и удалять любые не ключевые поля в любой из таблиц БД индивидуального задания. При этом должна изменяться только структура таблиц, отображать изменение состава полей на формах не следует. Просмотр изменений в таблицах должен осуществляться с формы-мастера. Все производимые действия на форме должны сопровождаться контекстными сообщениями, направляющими работу пользователя с мастером.

*Примечание.* Иногда в программе могут возникать ошибки, связанные с отсутствием подключения какой-либо библиотеки VBA. На рис. 25 приведен пример правильного и достаточного для нормальной работы перечня компонентов, который выбирается в меню редактора VBA *Tools - References*

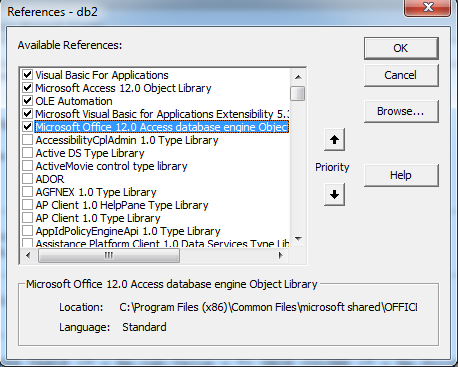


Рис. 25

## **Контрольные вопросы**

1. Что такое VBA?
2. Назовите основные принципы ООП
3. Как происходит привязка программы к элементам управления на форме?
4. Какими возможностями обладает редактор VBA, как он открывается?
5. Что такое модуль, какие типы модулей существуют?
6. Чем отличаются процедуры-функции от процедур-подпрограмм?
7. Привести пример использования аргументов функции
8. Назвать основные условные и циклические операторы VBA
9. Что такое макрос, как его создать?
10. Какие виды макросов по их назначению можно выделить?

# **Лабораторная работа № 7 Проектирование базы данных по индивидуальному заданию**

## Цель работы:создать собственную БД по индивидуальному заданию, проходя самостоятельно все этапы проектирования от анализа предметной области до собственно БД и пользовательских приложений, работающих в СУБД Microsoft Access.

### **Краткие теоретические сведения**

Целями разработки любой БД являются, в первую очередь, надежное хранение и удобный доступ к информации в рамках интересующей заказчика предметной области. Для реализации этой цели разработчики используют следующие основные инструменты: реляционную модель данных и язык SQL.

При разработке любой БД происходит постепенный переход от анализа предметной области к конкретной реализации БД средствами выбранной СУБД. Выделяют следующие этапы этого процесса перехода:

1. анализ предметной области;
2. модель предметной области;
3. логическая модель данных;
4. физическая модель данных;
5. собственно БД и приложения.

**Анализ предметной области**. Первый этап – анализ предметной области – в первую очередь предполагает достаточно подробное изучение той части реального мира, данные о которой необходимо будет отразить в БД. В качестве предметной области может выступать любой технологический процесс, система управления каким-либо оборудованием, система сбора данных, бухгалтерия какого-либо предприятия, отдел кадров, банк, магазин и т.д. Предметная область содержит как важные понятия, данные и ограничения, так и малозначащие или вообще ничего не значащие данные. Например, если в качестве предметной области выбрать учет товаров на складе, то понятия «накладная» и «счет-фактура» являются важными, а то, что сотрудница, принимающая накладные, имеет двоих детей для учета товаров неважно. Однако с точки зрения разработки БД для отдела кадров данные о наличии детей будут являться важными. Таким образом, необходимо на первом этапе четко сформулировать цели и задачи проектирования БД, собрать сведения о необходимых для хранения данных, изучить их структуру, определить возможные зависимости, ограничения и другие особенности.

**Модель предметной области**. На втором этапе строится модель предметной области, цель которой упорядочить и отразить знания о предметной области заказчика БД с точки зрения разработчика. Знания могут быть как в виде неформальных представлений в голове эксперта, так и выражены формально при помощи каких-либо средств. В качестве таких средств могут выступать текстовые описания предметной области, техническая документация, наборы должностных инструкций, правила проведения экспериментов и т.п. Опыт показывает, что текстовый способ представления модели предметной области крайне неэффективен. Гораздо более информативными и полезными при разработке БД являются описания предметной области, выполненные при помощи специализированных графических нотаций. Из наиболее известных методик описания предметной области можно назвать метод структурного анализа (SADT) и основанные на нем IDEF0, IDEF1 диаграммы, диаграммы потоков данных Гейна-Сарсона, метод объектно-ориентированного анализа (UML), и др. Пример построения модели предметной области описывается в лабораторной работе №1. От того, насколько правильно проведен анализ и смоделирована предметная область, зависит успех всей дальнейшей разработки БД и приложений.

**Логическая модель данных**. На следующем, более низком уровне находится логическая модель данных предметной области. Логическая модель описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью.

Логическая модель данных является начальным прототипом будущей БД. Логическая модель строится в терминах информационных единиц, но без привязки к конкретной СУБД. Более того, логическая модель данных необязательно должна быть выражена средствами именно реляционной модели данных. Основным средством разработки логической модели данных в настоящий момент являются различные варианты ER-диаграмм. Одну и ту же ER-модель можно преобразовать как в реляционную модель данных, так и в модель данных для иерархических и сетевых СУБД, или в объектную модель данных.

Решения, принятые на предыдущем уровне, при разработке модели предметной области, определяют некоторые границы, в пределах которых можно развивать логическую модель данных и принимать различные решения. Например, модель предметной области складского учета содержит понятия "склад", "накладная", "товар". При разработке соответствующей реляционной модели эти термины обязательно должны быть использованы, но различных способов реализации много – можно создать одно отношение, в котором будут присутствовать в качестве атрибутов «склад», «накладная», «товар», а можно создать три отдельных отношения, по одному на каждое понятие.

Пример и методика построения логической реляционной модели данных на основе нотаций Чена-Мартина приведены в лабораторной работе №2.

**Физическая модель данных**. На еще более низком уровне находится физическая модель данных. Физическая модель данных описывает данные средствами конкретной СУБД. Отношения, разработанные на стадии формирования логической модели данных, преобразуются в реляционные таблицы, атрибуты становятся столбцами таблиц, для ключевых атрибутов создаются уникальные индексы, домены преображаются в типы данных, принятые в конкретной реляционной СУБД.

Ограничения, имеющиеся в логической модели данных, реализуются различными средствами СУБД, например, при помощи индексов, декларативных ограничений целостности, триггеров, хранимых процедур. При этом снова решения, принятые на уровне логического моделирования определяют некоторые границы, в пределах которых можно развивать физическую модель данных и принимать различные решения. Например, отношения, содержащиеся в логической модели данных, должны быть преобразованы в таблицы, но для каждой таблицы можно дополнительно объявить различные индексы, повышающие скорость обращения к данным.

Пример построения физической модели в СУБД Microsoft Access, которая в итоге представляется схемой данных, рассматривается в лабораторной работе №2.

**Собственно база данных и приложения.** Результатом предыдущих этапов является собственно сама БД на конкретной программно-аппаратной платформе, выбор которой также будет влиять на качество работы с БД. Например, можно выбирать различные типы компьютеров, менять количество процессоров, объем оперативной памяти, дисковые подсистемы и т.п. Большое значение имеет также настройка СУБД в пределах выбранной программно-аппаратной платформы. Примерами настройки СУБД под нужды конкретной БД можно считать разработку и хранение различных SQL запросов, отчетов и форм, а также приложений на VBA, которые были рассмотрены в лабораторных работах №3,4,5,6.

## **Задание к лабораторной работе**

1. Получить вариант индивидуального задания у преподавателя (или по номеру зачетной книжки заочников). Варианты задания приведены в [приложении](../pril1/pril1.htm) №1.
2. Изучить и проанализировать предметную область, указанную в варианте, на основе собранного материала сформулировать основные цели и задачи создания БД.
3. Разработать диаграмму потоков данных и оформить ее в электронном виде, например, используя приложение Microsoft Visio или графический редактор Microsoft Word.
4. Создать логическую модель, используя диаграммы «сущность-связь», также оформить ее в электронном виде.
5. В Microsoft Аccess на основе логической модели данных создать таблицы, задать необходимые ограничения на данные в полях, установить реляционные связи между таблицами на схеме данных, после чего заполнить таблицы данными в соответствии с контекстом предметной области.
6. Выполнить указанные в варианте задания запросы, создать по ним отчеты, а также самостоятельно придумать и выполнить запросы с условной выборкой по полям с типом дата/время; запросы на удаление, на добавление, на обновление и на создание таблиц (теория и примеры изложены в лабораторных работах № 3, 4).
7. Создать для пользователя БД экранные формы, которые позволят автоматизировать работу с данными: просматривать, добавлять, удалять и модифицировать записи в любой таблице без применения стандартных панелей Microsoft Access, запускать созданные запросы, просматривать и печатать отчеты, осуществлять поиск и просмотр информации связанных таблиц в подчиненных формах. Все коды полей связанных таблиц на стороне отношения «многие» должны быть заменены при помощи подстановки на данные исходных родительских таблиц. Все необходимые элементы управления должны быть размещены на формах так, чтобы пользователю было интуитивно понятно и удобно работать с БД (теория и примеры изложены в лабораторной работе № 5).
8. Создать форму для добавления и удаления полей и написать программу на VBA, позволяющую с помощью элементов этой формы добавлять и удалять любые не ключевые поля в любой из таблиц БД. При этом должна изменяться только структура таблиц, отображать изменение состава полей на формах не следует. Просмотр изменений в таблицах должен осуществляться с данной формы. Все производимые на форме действия (выбор таблицы, выбор поля, указание нового имени поля, выбор типа данных) в случае какой-либо ошибки (ключевое поле, поле уже существует, не указан тип поля, не выбрана таблица и т.д.) должны сопровождаться контекстными сообщениями, направляющими работу пользователя (теория и примеры изложены в лабораторной работе № 6).
9. Подготовить отчет о проделанной работе. Примерное содержание отчета:

* вариант задания, цели и задачи проектирования БД,
* диаграмма предметной области,
* модель сущность-связь, описание ограничений на значения полей,
* таблицы в режиме конструктора,
* схема данных в MS Access,
* конструкторы запросов (всего не менее 10 запросов разных типов),
* отчеты по запросам (как подготовленные к печати документы),
* экранные формы: главная и дочерние, описание назначения элементов управления на формах, навигации и возможностей работы с данными через формы.
* форма мастера добавления и удаления полей, код программы на VBA с комментариями
* заключение.

### **Контрольные вопросы:**

1. Перечислить этапы проектирования БД
2. Для чего используются БД?
3. Что понимают под предметной областью?
4. Какие основные задачи решаются на этапе проектирования?
5. Назовите основные способы создания модели предметной области
6. Что понимают под логической моделью БД?
7. Какие средства разработки логических моделей БД существуют?
8. Чем отличается физическая модель от логической модели БД?
9. Как происходит преобразование логической модели в физическую?
10. Что значит понятие последнего этапа проектирования «собственно БД и приложения».

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Access 2000. Базы данных и приложения: Лекции и упражнения. – Киев: ДиаСофт, 2000. – 512 с.
2. Боровиков В.В. Аccess 2002. Программирование и разработка баз данных и приложений., –М.: Солон - Р, 2002. – 560 с.
3. Веремеенко Е.Г. Лабораторные работы по курсу «Базы данных и знаний». Ч. 1. URL:http://khpi-iip.mipk.kharkiv.edu.
4. Кириллов, В.В. Основы проектирования реляционных баз данных: учеб. пособие / В.В. Кириллов. −  СПб. : ИТМО, 1994. – 90 с. [URL: http://khpi-iip.mipk.kharkiv.edu](URL:http://khpi-iip.mipk.kharkiv.edu).
5. Коннолли, Т. Базы данных: проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг. − 3 изд., перераб. − M.: Вильямс, 2003 −1440 с.
6. Кузьменко, В.Г. Программирование на VBA 2002 / В.Г. Кузьменко. – М. : ООО «Бином-Пресс», 2003. – 880 с.
7. Новалис C. Access 2000: Руководство по VBA: Пер. А. Киселева. – М.: ЛОРИ, 2001. – 506 с.
8. Профессиональное программирование в MS Access 2002. – М: Вильямс, 2002. – 992 с.
9. Харитонова С.А. Access 2000. Разработка приложений. – СПб.: BHV–Санкт-Петербург, 2000. – 832 с.

# Приложение 1

ВАРИАНТЫ ИНДИВИДУАЛЬНЫХ ЗАДАНИЙ

**Вариант 1.** Создайте БД отдела кадров университета. Ориентировочные таблицы-составляющие: «Сотрудники», **«Контракты»**, «Кафедры», «Должности». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Доценты» (условная выборка); «Сводка количества работающих на каждой должности» (итоговый запрос); «Сотрудники, нуждающиеся в продлении контракта» (параметрический запрос); «Динамика текучести кадров» (перекрестный запрос), вывести общий список сотрудников и кафедр (запрос на объединение).

**Вариант 2.** Создайте БД галантерейного магазина. Ориентировочные таблицы-составляющие: «Партии товара», «Поставщики», **«Продажи**», “Продавцы”. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Залежавшийся товар» (условная выборка); «Рейтинг спроса по фирмам» (итоговый запрос); «Поставщики партий заданного объема» (параметрический запрос); «Динамика продаж по базовым видам продукции» (перекрестный запрос), вывести общий список продавцов и поставщиков (запрос на объединение).

**Вариант 3.** Создайте БД отдела доставки почтового отделения. Ориентировочные таблицы-составляющие: «Подписчики», «Периодика», **«Подписки»**, «Почтальоны». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Подписчики газеты "Вечерний Минск"» (условная выборка); «Количество подписок на каждое издание» (итоговый запрос); «Подписки дешевле заданной стоимости» (параметрический запрос); «Динамика цен на подписку на заданные издания» (перекрестный запрос), вывести общий список подписчиков и почтальонов (запрос на объединение).

**Вариант 4.** Создайте БД универмага. Ориентировочные таблицы-составляющие: «Товары», «Отделы», **«Продажа»**, «Продавцы». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список отделов, реализующих парфюмерию» (условная выборка); «Сводка продаж по отделам» (итоговый запрос); «Список продукции в заданном отделе» (параметрический запрос); «Динамика продаж по отделам» (перекрестный запрос), вывести общий список отделов и продавцов (запрос на объединение).

**Вариант 5.** Создайте БД штатного расписания предприятия. Ориентировочные таблицы-составляющие: «Отделы», «Должности», «Тип производства», «Сотрудники», **«Штатное расписание»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список сотрудников, не занятых в основном производстве» (условная выборка); «Сводка по заработной плате каждого отдела» (итоговый запрос); «Список сотрудников заданной должности» (параметрический запрос); «Динамика освобождения рабочих мест по мере выхода сотрудников на пенсию» (перекрестный запрос), вывести общий список отделов и должностей (запрос на объединение).

**Вариант 6.** Создайте БД фирмы по производству пиломатериалов. Ориентировочные таблицы-составляющие: «Изделия», «Сырье», **«Продажи»**, «Поставщики». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Реализованные изделия за последнюю неделю» (условная выборка); «Сводка расхода сырья» (итоговый запрос); «Продажи изделий заданной породы древесины» (параметрический запрос); «Динамика продаж изделий по группам» (перекрестный запрос), вывести общий список изделий и сырья (запрос на объединение).

**Вариант 7.** Создайте БД программы выпуска деталей литейного цеха. Ориентировочные таблицы-составляющие: «Детали», «Материал», **«Технологические процессы»**, «Рабочие». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Чугунные отливки большого объема» (условная выборка); «Расход материала в плановом периоде» (итоговый запрос); «Процессы выплавки деталей заданных габаритов» (параметрический запрос); «Динамика выпуска деталей (по материалу)» (перекрестный запрос), вывести общий список деталей и материалов (запрос на объединение).

**Вариант 8.** Создайте БД инфекционного отделения городской больницы. Ориентировочные таблицы-составляющие: «Койко-место», «Больные», «Диагнозы», **«Госпитализация»**, «Врачи». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Больные-пенсионеры» (условная выборка); «Количество больных в каждой палате» (итоговый запрос); «Палаты больных с заданной температурой» (параметрический запрос); «Динамика заболеваемости» (перекрестный запрос), вывести общий список врачей и больных (запрос на объединение).

**Вариант 9.** Создайте БД процессов обработки партий деталей. Ориентировочные таблицы-составляющие: «Детали», «Оборудование», **«Технологические карты»**, «Процессы», «Бригады». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список оборудования для высококачественной обработки деталей» (условная выборка); «Загруженность оборудования» (итоговый запрос); «Партии, проходящие заданную операцию» (параметрический запрос); «Количество партий на оборудование по заданным интервалам времени» (перекрестный запрос), вывести общий список деталей и оборудования (запрос на объединение).

**Вариант 10.** Создайте БД фирмы по оптовой реализации бытовой техники. Ориентировочные таблицы-составляющие: «Продукция», «Клиенты», **«Заказы»**, «Производители». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Крупнейшие партии» (условная выборка), «Сводка по технике, закупленной каждым клиентом» (итоговый запрос); «Продукция, реализованная в заданный период времени» (параметрический запрос); «Динамика популярности фирм-изготовителей» (перекрестный запрос), вывести общий список клиентов и производителей (запрос на объединение).

**Вариант 11.** Создайте БД оборудования НИИ. Ориентировочные таблицы-составляющие: «Оборудование», «Исследовательские работы», **«Акты проведения работ»**, «Руководители работ». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Наиболее загруженное оборудование» (условная выборка); «Количество часов наработки» (итоговый запрос); «Проведение работ на заданной единице оборудования» (параметрический запрос); «Распределение общего количества единиц оборудования по темам исследований во времени» (перекрестный запрос), вывести общий список оборудования и работ (запрос на объединение).

**Вариант 12.** Создайте БД фирмы по производству столярных работ. Ориентировочные таблицы-составляющие: «Продукция», «Клиенты», **«Договора»**, «Услуги», «Материалы». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Постоянные клиенты фирмы» (условная выборка); «Прибыль по каждому виду продукции/услуги» (итоговый запрос); «Продукция/услуги, реализованные на заданную сумму» (параметрический запрос); «Динамика пользовательского спроса продукции» (перекрестный запрос), вывести общий список услуг и продукции (запрос на объединение).

**Вариант 13.** Создайте БД агентства недвижимости. Ориентировочные таблицы-составляющие: «Объекты недвижимости», «Типы объектов», «Типы сделок», «Покупатели», «Продавцы», **«Сделки»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список объектов, предлагаемых к продаже» (условная выборка); «Сальдо по видам объектов» (итоговый запрос); «Объекты заданной стоимости» (параметрический запрос); «Динамика продаж по районированию объектов» (перекрестный запрос), вывести общий список покупателей и продавцов (запрос на объединение).

**Вариант 14.** Создайте БД фирмы-распространителя программного обеспечения. Ориентировочные таблицы-составляющие: «Программное обеспечение», «Клиенты», **«Производимые работы»**, «Производители ПО», «Гарантийные сроки». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список клиентов, программному обеспечению которых предстоит обновление» (условная выборка); «Сводка реализованного программного обеспечения» (итоговый запрос); «Список клиентов, купивших продукцию заданного вида» (параметрический запрос); «Динамика обновления версий» (перекрестный запрос), вывести общий список ПО и его производителей (запрос на объединение).

**Вариант 15.** Создайте БД инструментального склада. Ориентировочные таблицы-составляющие: «Обслуживаемое оборудование», «Инструмент», **«Технологические карты»**, «Рабочие». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Инструмент с высокой степенью используемости» (условная выборка); «Количество единиц инструмента для каждого оборудования» (итоговый запрос); «Список инструмента, подверженного повышенному износу» (параметрический запрос); «Количество единиц инструмента для токарного оборудования по технологическим картам» (перекрестный запрос), вывести общий список оборудования и инструментов (запрос на объединение).

**Вариант 16.** Создайте БД страховой фирмы. Ориентировочные таблицы-составляющие: «Виды страховок», «Клиенты», «Объекты страховки», **«Страховая деятельность»**, «Агенты». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Клиенты, застраховавшие свою жизнь за последний месяц» (условная выборка); «Сводка полученных/выплаченных сумм страховок по клиентам» (итоговый запрос); «Объекты, застрахованные на заданную сумму» (параметрический запрос); «Динамика заключения страховых договоров (по основным видам)» (перекрестный запрос), вывести общий список клиентов и агентов (запрос на объединение).

**Вариант 17.** Создайте БД подъемно-транспортного оборудования машиностроительного предприятия. Ориентировочные таблицы-составляющие: «Техника», «Производимые работы», «Виды ремонта», «Рабочие», **«Учет работ»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Работа техники высокой грузоподъемности» (условная выборка); «Сводка часов простоя единиц оборудования во внеплановом ремонте» (итоговый запрос); «Список техники с заданным коэффициентом загрузки» (параметрический запрос); «Динамика поломок» (перекрестный запрос), вывести общий список рабочих и техники (запрос на объединение).

**Вариант 18.** Создайте БД музея. Ориентировочные таблицы-составляющие: «Экспонаты», «Авторы», «Виды экспозиции», «Жанры», **«Проведение выставки»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Малоизвестные экспонаты» (условная выборка); «Выставлено работ по авторам» (итоговый запрос); «Экспозиции работ повышенной ценности» (параметрический запрос); «Динамика экспозиции работ по видам» (перекрестный запрос), вывести общий список авторов и экспонатов (запрос на объединение).

**Вариант 19.** Создайте БД НИИ. Ориентировочные таблицы-составляющие: «Сотрудники», «Научно-исследовательские разработки», **«Составление штатного расписания»**, «Отделы», «Должности». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Сотрудники, задействованные в научно-исследовательских разработках» (условная выборка); «Выплаченная заработная плата (по отделам)» (итоговый запрос); «Список ответственных по научно-исследовательским разработкам» (параметрический запрос); «Динамика ведения научных работ по месяцам» (перекрестный запрос), вывести общий список разработок и сотрудников (запрос на объединение).

**Вариант 20.** Создайте БД студии видеозаписи. Ориентировочные таблицы-составляющие: «Режиссеры», «Актеры», «Фильмы», «Жанры», **«Продажи»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Оскар» (условная выборка); «Рейтинг продаж по актерам» (итоговый запрос); «Фильмы с заданным актерским дуэтом» (параметрический запрос); «Динамика продаж по жанрам» (перекрестный запрос), вывести общий список режиссеров и фильмов (запрос на объединение).

**Вариант 21.** Создайте БД деканата. Ориентировочные таблицы-составляющие: «Список курсов и групп», «Список студентов», **«Сдача сессии»**, «Предметы», «Преподаватели». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Неуспевающие студенты» (условная выборка); «Средний балл по предметам» (итоговый запрос); «Студенты, имеющие хорошие оценки по заданному предмету» (параметрический запрос); «Динамика сдачи сессий по группам» (перекрестный запрос), вывести общий список студентов и преподавателей (запрос на объединение).

**Вариант 22.** Создайте БД автозаправочной станции. Ориентировочные таблицы-составляющие: «Горюче-смазочные материалы (ГСМ)», «Поставщики», **«Накладные»**, «Рабочие АЗС». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Закупки дизельного топлива» (условная выборка); «Затраты по видам продукции» (итоговый запрос); «Поставщики дешевой смазки» (параметрический запрос); «Динамика закупочных цен по основным видам продукции» (перекрестный запрос), вывести общий список поставщиков и ГСМ (запрос на объединение).

**Вариант 23.** Создайте БД выставки продукции. Ориентировочные таблицы-составляющие: «Продукция», «Предприятие», «Выставочное место», **«Организация выставки»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Предприятия, арендовавшие наибольшие выставочные площади» (условная выборка); «Сводка по продукции/предприятиям» (итоговый запрос); «Продукция заданного предприятия» (параметрический запрос); «Количество выставляемой продукции по предприятиям» (перекрестный запрос), вывести общий список продукции и предприятий (запрос на объединение).

**Вариант 24.** Создайте БД полиграфической фирмы. Ориентировочные таблицы-составляющие: «Материалы», «Техника», **«Контракты»**, «Заказчики», «Перечень работ». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Материалы заказов малых объемов» (условная выборка); «Сводка расходов материалов» (итоговый запрос); «Заказы на заданную сумму» (параметрический запрос); «Динамика загруженности оборудования» (перекрестный запрос), вывести общий список материалов и техники (запрос на объединение).

**Вариант 25.** Создайте БД фирмы-поставщика медицинской техники. Ориентировочные таблицы-составляющие: «Ассортимент», «Заказчики», **«Контракты»**, «Производители». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Последние поступления» (условная выборка); «Рейтинг заказчиков по общим суммам контрактов» (итоговый запрос); «Клиенты, заказавшие заданный вид изделия» (параметрический запрос); «Динамика реализации заданных единиц ассортимента» (перекрестный запрос), вывести общий список производителей и заказчиков (запрос на объединение).

**Вариант 26.** Создайте БД планово-финансового отдела. Ориентировочные таблицы-составляющие: «Участки», «Работники», **«Наряды»**, «Мероприятия», «Должности». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Опасные работы» (условная выборка); «Начисление заработной платы по участкам» (итоговый запрос); «Списки работников по заданным датам/участкам» (параметрический запрос); «Количество работников на участках по месяцам» (перекрестный запрос), вывести общий список работников и участков (запрос на объединение).

**Вариант 27.** Создайте БД строительной фирмы. Ориентировочные таблицы-составляющие: «Объекты», «Этапы выполнения», «Стройматериалы», «Бригады», **«Строительство»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Текущие этапы работы» (условная выборка); «Сроки строительства объектов» (итоговый запрос); «Объекты заданного процента завершенности» (параметрический запрос); «Распределение материалов по объектам» (перекрестный запрос), вывести общий список (запрос на объединение).

**Вариант 28.** Создайте БД опытного цеха. Ориентировочные таблицы-составляющие: «Производимые изделия», «Оборудование», **«Технологические карты»,** «Рабочие», «Инвентарь». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Изделия/карты фрезерной обработки» (условная выборка); «Количество изделий по каждому оборудованию» (итоговый запрос); «Карты изделий заданных габаритов» (параметрический запрос); «Распределение количества единиц оборудования по заданным изделиям во времени» (перекрестный запрос), вывести общий список изделий и оборудования (запрос на объединение).

**Вариант 29.** Создайте БД механообрабатывающего цеха. Ориентировочные таблицы-составляющие: «Детали», «Материалы», «Оснастка», **«Работа цеха»**, «Рабочие». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Оснастка для обработки деталей высокой твердости» (условная выборка); «Расход материалов по видам деталей» (итоговый запрос); «Детали заданного материала» (параметрический запрос); «Распределение количества деталей по материалу и требуемым видам оснастки» (перекрестный запрос), вывести общий список материалов и оснастки (запрос на объединение).

**Вариант 30.** Создайте БД фирмы по установке и обслуживанию ЛВС. Ориентировочные таблицы-составляющие: «Сетевые конфигурации», «Клиенты», **«Договора»**, «Оборудование», «Исполнители». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список клиентов, с которыми необходимо перезаключать договора» (условная выборка); «Протяженность кабеля у каждого клиента» (итоговый запрос); «Список ЛВС с заданным количеством станций» (параметрический запрос); «Динамика спроса базовых сетевых конфигураций» (перекрестный запрос), вывести общий список клиентов и исполнителей (запрос на объединение).

**Вариант 31.** Создайте БД фирмы по продаже стройматериалов. Ориентировочные таблицы-составляющие: «Сырье», «Продукция», «Заказчики», **«Контракты»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Продажа кирпича» (условная выборка); «Сводка сумм контрактов по каждому заказчику» (итоговый запрос); «Список продукции, заказанной в заданный период» (параметрический запрос); «Динамика заключения контрактов по месяцам» (перекрестный запрос), вывести общий список продукции и сырья (запрос на объединение).

**Вариант 32.** Создайте БД ветеринарной станции. Ориентировочные таблицы-составляющие: «Ветеринарные работы», «Обслуживаемые хозяйства», «Медикаменты», «Ветеринары», **«Проведенные работы»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Список хозяйств, запланированных на ближайшую неделю» (условная выборка); «Объемы проведенных работ в каждом хозяйстве» (итоговый запрос); «Выполнение плана по хозяйствам на заданное число» (параметрический запрос); «Распределение количества осмотренных животных по дням недели» (перекрестный запрос), вывести общий список обслуживаемых хозяйств и ветеринаров (запрос на объединение).

**Вариант** **33.** Создайте БД фирмы по проведению буровых работ. Ориентировочные таблицы-составляющие: «Скважины», «Клиенты», «Бригады», «Оборудование», **«Контракты»**.Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Контракты на бурение глубинных скважин» (условная выборка); «Количество контрактов на каждый вид работ» (итоговый запрос); «Контракты с заданным клиентом» (параметрический запрос); «Метраж пробуренных скважин за последние 10 недель» (перекрестный запрос), вывести общий список клиентов и бригадиров (запрос на объединение).

**Вариант 34.** Создайте БД загрузки аудиторий. Ориентировочные таблицы-составляющие: «Аудитории», «Учебные дисциплины», «Преподаватели», «Кафедры», «Группы», **«Загрузка аудиторий»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Поточные (лекционные) аудитории» (условная выборка); «Рейтинг дисциплин по загрузке аудиторий» (итоговый запрос); «Аудитории заданной дисциплины» (параметрический запрос); «Динамика занятости заданной группой аудиторий по дням недели» (перекрестный запрос), вывести общий список кафедр и групп (запрос на объединение).

**Вариант 35.** Создайте БД трикотажной фабрики. Ориентировочные таблицы-составляющие: «Сырье», «Изделия», **«Сбыт изделий»**, «Покупатели», «Размеры». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Сбыт изделий, содержащих искусственные волокна» (условная выборка); «Рейтинг себестоимости изделий» (итоговый запрос); «Картина сбыта изделий заданного размера» (параметрический запрос); «Динамика сбыта шерстяных изделий по месяцам» (перекрестный запрос), вывести общий список сырья и изделий (запрос на объединение).

**Вариант** **36.** Создайте БД ателье головных уборов. Ориентировочные таблицы-составляющие: «Изделия», «Клиенты», «Материалы», «Размеры», **«Квитанции»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Заказчики зимнего ассортимента» (запрос – условная выборка); «Популярность моделей» (итоговый запрос); «Модели шляп заданной стоимости» (параметрический запрос); «Количество заказов заданных моделей по месяцам» (перекрестный запрос), вывести общий список изделий и материалов (запрос на объединение).

**Вариант 37.** Создайте БД гостиницы. Ориентировочные таблицы-составляющие: «Номера», «Клиенты», «Обслуживающий персонал», **«Счета»**, «Услуги». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Богатые клиенты» (условная выборка); «Рейтинг загрузки номеров» (итоговый запрос); «Счета указанного клиента» (параметрический запрос); «Динамика наплыва посетителей по месяцам» (перекрестный запрос), вывести общий список клиентов и персонала (запрос на объединение).

**Вариант** **38.** Создайте БД жилищного коммунального хозяйства. Ориентировочные таблицы-составляющие: «Специалисты», «Жилищный фонд», «Мероприятия», **«Заказы»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Деятельность, запланированная на ближайшую неделю» (условная выборка); «Рейтинг работников по количеству мероприятий» (итоговый запрос); «Мероприятия, проведенные заданным работником» (параметрический запрос); «Количество проведенных мероприятий по месяцам» (перекрестный запрос), вывести общий список специалистов и мероприятий (запрос на объединение).

**Вариант 39.** Создайте БД стоматологической поликлиники. Ориентировочные таблицы-составляющие: «Врачи», «Пациенты», **«Обслуживание»**, «Услуги». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Протезирование» (условная выборка); «Рейтинг услуг» (итоговый запрос); «Пациенты заданного врача» (параметрический запрос); «Динамика обращений за помощью по месяцам года» (перекрестный запрос), вывести общий список врачей и пациентов (запрос на объединение).

**Вариант 40.** Создайте БД сборочного процесса. Ориентировочные таблицы-составляющие: «Комплектующие», «Изделия», **«Сборка»**, «Технологические процессы». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Изделия, требующие микросборки» (условная выборка); «Количество комплектующих в каждом изделии» (итоговый запрос); «Информация по изделиям заданного технологического процесса» (параметрический запрос); «Динамика сборки изделий за заданный интервал времени планового периода» (перекрестный запрос), вывести общий список комплектующих и изделий (запрос на объединение).

**Вариант 41.** Создайте БД мебельной фабрики. Ориентировочные таблицы-составляющие: «Изделия», «Материалы», «Заказчики», «Поставщики», **«Контракты»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Контракты по корпусной мебели» (условная выборка); «Рейтинг продукции» (итоговый запрос); «Последние контракты» (параметрический запрос); «Динамика популярности заданных видов продукции» (перекрестный запрос), вывести общий список заказчиков и поставщиков (запрос на объединение).

**Вариант 42.** Создайте БД кабельного завода. Ориентировочные таблицы-составляющие: «Сырье», «Продукция», «Технологические процессы», **«Производство»**, «Рабочие». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Монтажные провода» (условная выборка); «Затраты сырья по видам продукции» (итоговый запрос); «Исходные материалы для производства проводников заданного типа» (параметрический запрос); «Расход меди на погонный метр в зависимости от толщины кабеля по видам продукции» (перекрестный запрос), вывести общий список сырья и продукции (запрос на объединение).

**Вариант 43.** Создайте БД механизированной колонны. Ориентировочные таблицы-составляющие: «Техника», «Выполняемые работы», **«Путевки»**, «Водители». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Экскаваторщики» (условная выборка); «Отработано персоналом часов за неделю» (итоговый запрос); «Наряды в заданный промежуток времени» (параметрический запрос); «Динамика загруженности техники» (перекрестный запрос), вывести общий список водителей и техники (запрос на объединение).

**Вариант 44.** Создайте БД санатория. Ориентировочные таблицы-составляющие: «Оздоровительные программы», «Отдыхающие», «Врачи», **«Обслуживание»**, «Заболевания». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Программы по желудочно-кишечным заболеваниям» (условная выборка); «Сводка отдыхающих за плановый период» (итоговый запрос); «Отдыхающие, завершающие курс оздоровления» (параметрический запрос); «Динамика заболеваний по месяцам» (перекрестный запрос), вывести общий список врачей и отдыхающих (запрос на объединение).

**Вариант 45.** Создайте БД геологоразведочной экспедиции. Ориентировочные таблицы-составляющие: «Регионы», «Карты», **«Экспедиции»**, «Геологи». Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Экспедиции по Беларуси» (условная выборка); «Протяженность маршрутов по регионам» (итоговый запрос); «Перечень карт заданного региона» (параметрический запрос); «Количество листов карт по каждой экспедиции» (перекрестный запрос), вывести общий список экспедиций и геологов (запрос на объединение).

**Вариант 46.** Создайте БД фирмы по автоматизации производства. Ориентировочные таблицы-составляющие: «Каталог устройств и программного обеспечения», «Клиенты», «Персонал», «Услуги»,**«Договора»**. Общее количество полей в базе – не менее 20, общее количество записей – не менее 30.

Создайте отчеты по запросам: «Услуги, оказанные в текущем месяце» (условная выборка); «Сводка доходов по виду услуг за отчетный период» (итоговый запрос); «Договора по установке программного обеспечения» (параметрический запрос); «Динамика спроса по месяцам» (перекрестный запрос), вывести общий список клиентов и персонала (запрос на объединение).

# Приложение 2

ПЕРЕЧЕНЬ ТАБЛИЦ УЧЕБНОЙ БАЗЫ ДАННЫХ «БИБЛИОТЕКА»

Таблица 1

Таблица «Издательства»

|  |  |  |
| --- | --- | --- |
| **Код издательства** | **Название** | **Город** |
| 1 | Наука | Москва |
| 2 | Мир | Москва |
| 3 | Радио и связь | Москва |
| 4 | Машиностроение | Киев |

Таблица 2

Таблица «Книги»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Код книги** | **Автор** | **Название** | **Код издательства** | **Объем** | **Год издания** | **Стоимость** |
| 1 | Беспалько | Педагогика | 2 | 340 | 1994 | 9 900 р. |
| 2 | Сканави | Сборник задач | 2 | 634 | 1992 | 6 600 р. |
| 3 | Арсак | Программирование | 1 | 273 | 1989 | 19 800 р. |
| 4 | Перминов | Язык АДА | 3 | 278 | 1987 | 4 400 р. |
| 5 | Грибанов | Операционные машины | 3 | 446 | 1991 | 13 200 р. |
| 6 | Ульман | БД на Паскале | 4 | 563 | 1992 | 3 200 р. |

Таблица 3

Таблица «Выдача книг»

|  |  |  |  |
| --- | --- | --- | --- |
| **Код выдачи** | **Код читателя** | **Код книги** | **Дата заказа** |
| 1 | 1 | 1 | 01.09.96 |
| 2 | 1 | 2 | 01.03.97 |
| 3 | 1 | 5 | 20.04.97 |
| 4 | 2 | 1 | 04.11.95 |
| 5 | 2 | 3 | 20.12.96 |
| 6 | 3 | 1 | 03.08.95 |
| 7 | 3 | 2 | 20.01.96 |
| 8 | 4 | 2 | 03.02.96 |
| 9 | 5 | 1 | 04.10.96 |
| 10 | 6 | 3 | 18.03.97 |
| 11 | 7 | 4 | 23.04.97 |
| 12 | 7 | 6 | 06.05.97 |
| 13 | 8 | 5 | 01.03.97 |
| 14 | 8 | 6 | 04.05.97 |
| 15 | 9 | 6 | 27.04.97 |
| 16 | 10 | 5 | 20.10.97 |

Таблица 4

Таблица «Темы»

|  |  |  |
| --- | --- | --- |
| **Код темы** | **Код книги** | **Название темы** |
| 1 | 1 | Личность человека |
| 2 | 1 | Проектирование ППС |
| 3 | 1 | Технология обучения |
| 4 | 1 | Анализ учебного процесса |
| 5 | 2 | Уравнения |
| 6 | 2 | Прогрессии |
| 7 | 2 | Геометрические задачи |
| 8 | 3 | Игры с числами |
| 9 | 3 | Игры без стратегии |
| 10 | 3 | Комбинаторные задачи |
| 11 | 3 | Стратегия без игры |
| 12 | 4 | Программные модули |
| 13 | 4 | Лексика |
| 14 | 4 | Предопределенные типы |
| 15 | 4 | Операторы |
| 16 | 5 | Структура ОС ЕС |
| 17 | 5 | Управление заданиями |
| 18 | 5 | Управление задачами |
| 19 | 5 | Управление данными |
| 20 | 6 | Операции с поставщиками |
| 21 | 6 | Бухгалтерская книга |
| 22 | 6 | Платежная ведомость |
| 23 | 6 | Реляционная алгебра |
| 24 | 6 | Правила нормализации |

Таблица 5

Таблица «Читатели»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Код читателя** | **Фамилия** | **Имя** | **Отчество** | **Адрес** | **Домашний телефон** |
| 1 | Минкевич | Андрей | Яковлевич | Ул. Московская, 17‑19 | 233-33-33 |
| 2 | Гуляев | Сергей | Дмитриевич | Ул. Козлова, 7‑38 | 236-93-79 |
| 3 | Федосенко | Олег | Георгиевич | Ул. Богдановича, 102‑34 | 232-23-16 |
| 4 | Захаров | Михаил | Степанович | Ул. Плеханова, 34‑67 | 266-19-75 |
| 5 | Бобров | Виктор | Иванович | Ул. Казинца, 23‑47 | 245-42-26 |
| 6 | Гусакович | Елена | Анатольевна | Пр. Скорины, 5‑8 |  |
| 7 | Семенова | Галина | Ивановна | Ул. Денисовская, 20‑47 | 251-49-67 |
| 8 | Савченко | Ольга | Павловна | Пр. Пушкина, 30‑41 | 231-99-06 |
| 9 | Мицкевич | Мария | Степановна | Ул. Чкалова, 12‑56 | 220-12-35 |
| 10 | Карпович | Татьяна | Дмитриевна | Ул. Малинина, 118‑56 | 234-80-34 |

# Приложение 3

Допустимые знаки для создания масок ввода в текстовых полях и полях с типом дата/время приведены в таблице:

Таблица 1

|  |  |
| --- | --- |
| **Знак** | **Описание** |
| 0 | Цифра (от 0 до 9, ввод обязателен; знаки плюс [+] и минус [-] не допускаются). |
| 9 | Цифра или пробел (ввод не обязателен; знаки плюс и минус не допускаются). |
| # | Цифра или пробел (ввод не обязателен; пустые знаки преобразуются в пробелы, допускаются знаки плюс и минус). |
| L | Буква (от A до Z или от А до Я, ввод обязателен). |
| ? | Буква (от A до Z или от А до Я, ввод не обязателен). |
| A | Буква или цифра (ввод обязателен). |
| a | Буква или цифра (ввод необязателен). |
| & | Любой знак или пробел (ввод обязателен). |
| C | Любой знак или пробел (ввод необязателен). |
| . , : ; - / | Десятичный разделитель и разделители тысяч, значений дат и времени. (Отображаемый знак зависит от настроек языка и стандартов на панели управления Microsoft Windows.) |
| < | Указывает перевод всех следующих знаков на нижний регистр. |
| > | Указывает перевод всех следующих знаков на верхний регистр. |
| ! | Указывает заполнение маски ввода справа налево, а не слева направо. Заполнение маски знаками всегда происходит слева направо. Восклицательный знак в маске ввода можно помещать в любую позицию |
| \ | Указывает ввод любого следующего знака в качестве текстовой константы. Используется для отображения всех перечисленных в данной таблице знаков как текстовых констант (например, \A выводится как знак «A»). |
| Пароль | Значение Пароль, заданное для свойства Маска ввода (InputMask), создает поле для ввода пароля. Любой знак, введенный в поле, сохраняется как знак, но отображается как звездочка (\*). |

Чтобы включить в маску текстовые константы, отличные от представленных в таблице, в том числе знаки и пробелы, следует просто ввести их в нужную позицию. Чтобы включить один из перечисленных, необходимо перед ним ввести знак обратной косой черты (\).

В приведенной ниже таблице указаны описания некоторых масок ввода и примеры значений, которые в них могут быть введены.

Таблица 2

|  |  |
| --- | --- |
| Описание маски ввода | Примеры значений |
| (000) 000-0000 | (206) 555-0248 |
| (999) 999-9999! | (206) 555-0248, ( ) 555-0248 |
| (000) AAA-AAAA | (206) 555-TELE |
| #999 | -20, 2000 |
| >L????L?000L0 | GREENGR339М3, МАЙ Р 452Ю7 |
| >L0L 0L0 | Т2Ф 8М4, |
| 00000-9999 | 98115- ,98115-3007 |
| >L<?????????????? | Мария, Иван |
| ISBN 0-&&&&&&&&&-0 | ISBN 1-55615-507-7, ISBN 0-13-964262-5 |
| >LL00000- | DB51392-0493 |

Учебное издание

**Крупская** Марина Александровна

***ИНФОРМАЦИОННЫЕ ОСНОВЫ СИСТЕМ УПРАВЛЕНИЯ***

Методическое пособие

к лабораторным работам для студентов специальности

1 53 01 07 «Информационные технологии и управление

в технических системах»

всех форм обучения

.

Редактор Т.А. Лейко

Корректор Е.Н. Батурчик

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подписано в печать 05.2011. Формат 60x84 1/16. Бумага офсетная.

Печать ризографическая. Гарнитура «Таймс». Усл. печ. л. .

Уч.-изд. л. 3,5. Тираж 200 экз. Заказ 99.

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Издатель и полиграфическое исполнение:

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Лицензия ЛП № 156 от 30.12.2002.

Лицензия ЛВ № 509 от 03.08.2001.

220013, Минск, П. Бровки, 6.