

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Логистика и организация производства»

# ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЛОГИСТИКЕ. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ

*Методические рекомендации к лабораторным работам  
для студентов специальностей*

*1-27 02 01 «Транспортная логистика (по направлениям)»  
и 1-27 01 01 «Экономика и организация производства  
(по направлениям)» дневной и заочной форм обучения*

**Часть 2**



Могилев 2019

УДК 004.91  
ББК 32.972  
И 75

Рекомендовано к изданию  
учебно-методическим отделом  
Белорусско-Российского университета

Одобрено кафедрой «Логистика и организация производства»  
«12» сентября 2019 г., протокол № 2

Составитель ст. преподаватель Т. М. Лобанова

Рецензент канд. экон. наук, доц. А. В. Александров

Методические рекомендации к лабораторным работам предназначены для студентов специальностей 1-27 02 01 «Транспортная логистика (по направлениям)» и 1-27 01 01 «Экономика и организация производства (по направлениям)» дневной и заочной форм обучения.

Учебно-методическое издание

## ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЛОГИСТИКЕ

### Часть 2

Ответственный за выпуск	М. Н. Гриневич
Технический редактор	С. Н. Красовская
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.  
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 56 экз. Заказ №

Издатель и полиграфическое исполнение:  
Межгосударственное образовательное учреждение высшего образования  
«Белорусско-Российский университет».  
Свидетельство о государственной регистрации издателя,  
изготовителя, распространителя печатных изданий  
№ 1/156 от 07.03.2019.  
Пр-т Мира, 43, 212022, Могилев.

© Белорусско-Российский  
университет, 2019



## Содержание

Введение.....	4
1 Организация хранения логистической (экономической) информации в MS Access. Реляционная модель данных.....	5
2 Организация хранения логистической (экономической) информации в MS Access. Целостность реляционных данных.....	8
3 Обработка логистической (экономической) информации в MS Access. Запросы на выборку.....	10
4 Изменение логистической (экономической) информации в MS Access. Запросы на изменение данных.....	14
5 Язык SQL.....	17
6 Методы проектирования БД.....	19
7 Создание интерфейса. Формы и элементы управления.....	22
8 Создание и использование макросов.....	26
9 Обработка событий на VisualBasic.....	31
10 Создание отчетов в MS Access.....	32
11 Публикация базы данных в Интернет.....	33
12 Импорт и экспорт данных, сжатие и восстановление. Связи с Office.....	35
Список литературы.....	36



## Введение

Целью проведения лабораторных работ по дисциплинам «Информационные технологии в логистике» и «Информационные технологии в экономике» является приобретение студентами практических навыков использования современных информационных технологий и систем, средств вычислительной техники в качестве инструмента для решения задач в предметной области.

В результате освоения учебной дисциплины студент:

а) познает:

- принципы организации корпоративных информационных систем (КИС) в предметной области; стандарты в области КИС; основные методы и средства защиты информации в КИС;

б) научится:

- использовать сервисы сети Интернет при решении профессиональных задач;

- формулировать задание на проектирование информационной системы;

- решать экономические задачи средствами КИС;

в) овладеет:

- навыками создания текстовых, табличных, графических документов и динамических презентаций;

- технологиями создания БД и их приложений.

Отчёт по лабораторным работам представляет собой файл с выполненным заданием. В процессе защиты работы студент поясняет отдельные этапы выполнения задания, при необходимости выполняет в присутствии преподавателя аналогичные задания.



# 1 Организация хранения логистической (экономической) информации в MS Access. Реляционная модель данных

**Цель работы:** изучить общие понятия реляционного подхода к организации СУБД, научиться создавать таблицы в MS Access.

Задача выполнения лабораторной работы заключается в приобретении навыков создания таблиц в MS Access.

Существует множество видов допустимых баз данных, но на практике только два вида занимают заметную долю рынка:

- базы данных с двумерными файлами;
- реляционные СУБД.

Базы данных с двумерными файлами состоят из одного файла. Классическим примером может быть адресная книга, содержащая одну таблицу с шестью полями: имя, адрес, город, штат, почтовый индекс, телефон. Если это вся база данных, то это и есть двумерный файл. В такой базе слова «таблица» и «база данных» являются синонимами.

Реляционные базы данных состоят из серии таблиц, связанных между собой по одному или нескольким полям.

Создают базы данных и обрабатывают запросы к ним системы управления базами данных – СУБД.

Основными понятиями реляционных баз данных являются *тип данных, домен, атрибут, кортеж, первичный ключ и отношение*.

Понятие **тип данных** в реляционной модели данных полностью адекватно понятию типа данных в языках программирования: числовой тип, денежный, символьный, логический и т. п.

Понятие **домена** имеет некоторые аналогии с подтипами в языках программирования. В общем виде домен определяется заданием базового типа данных и логического выражения (условия), применяемого к элементу типа данных. Наиболее правильной интуитивной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа. Например, домен «ФИО» определен на базовом типе строк символов, но в число его значений могут входить только те строки, которые могут изображать имя (в частности, такие строки не могут начинаться с мягкого знака).

Следует отметить также смысловую нагрузку понятия домена: данные считаются сравнимыми только в том случае, когда они относятся к одному домену. Например, значения доменов «Номера пропусков» и «Номера групп» относятся к типу целых чисел, но не являются сравнимыми.

На практике пользователь представляет себе **отношение** как *таблицу, заголовком* которой является **схема отношения**, а *строками* – **кортежи** отношения; в этом случае **имена атрибутов** именуют *столбцы* этой таблицы. Поэтому иногда говорят «столбец таблицы», имея в виду «атрибут отношения». Этой терминологии придерживаются в большинстве коммерческих



реляционных СУБД.

**Первичный ключ** – это один или несколько атрибутов, значения которых однозначно определяют кортеж отношения. Для каждого отношения по крайней мере полный набор его атрибутов обладает этим свойством.

### Задание 1

Создайте таблицы для организации хранения информации о поставщиках и товарах. Набор полей и их характеристики приведены в таблицах 1.1–1.3. Заполните их.

Таблица 1.1 – Поля таблицы «Поставщик»

Имя поля	Тип данных	Описание
КодП	Текстовый	Уникальный код фирмы-поставщика
ИмяП	Текстовый	Наименование поставщика
Город	Текстовый	
Адрес	Текстовый	
Телефон	Числовой	
Email	Гиперссылка	
Банк	Текстовый	Банк, в котором открыт счёт поставщика
РСчет	Числовой	Номер расчётного счёта поставщика

Таблица 1.2 – Поля таблицы «Товар»

Имя поля	Тип данных	Описание
КодТ	Текстовый	Уникальный код товара
НазваниеТ	Текстовый	Наименование товара
Сред_Цена	Денежный	Средняя цена товара за всё время его поставки
ВНаличии	Числовой	Количество товара на складе (сумма по всем поставкам)

Таблица 1.3 – Поля таблицы «Поставка»

Имя поля	Тип данных	Описание
НомерЗаписи	Счётчик	Уникальный код поставки
Дата	Дата/Время	Дата поставки товара
КодТ	Текстовый	Код товара (внешний ключ)
КодП	Текстовый	Код поставщика (внешний ключ)
Количество	Числовой	Количество привезённого товара
Цена	Денежный	Цена за единицу
Стоимость	Денежный	Стоимость всей партии

## Ход работы

Загрузите MS Access, создайте новую базу данных, указав нужный путь для сохранения файла и его имя.

Новую таблицу для ввода данных можно создать в режиме таблицы или в режиме конструктора.

Создание в режиме таблицы выполняется путем ввода данных в пустую таблицу. При сохранении такой таблицы Access анализирует введенные данные и автоматически присваивает каждому полю соответствующий тип данных и формат.

Режим конструктора используется для самостоятельного определения всех характеристик атрибутов и предоставляет намного больше возможности, чем первый вариант создания таблицы.

Инструменты для создания всех объектов баз данных находятся на закладке «Создание» (рисунок 1.1).

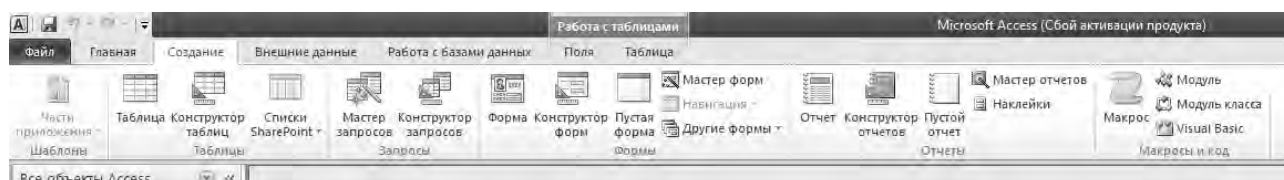


Рисунок 1.1 – Панель инструментов для создания объектов MSAccess

Создайте таблицы в режиме конструктора (рисунок 1.2). В разделе **Свойства поля** на закладке **Общие** установите для некоторых полей размер поля, формат, условие на значение, сообщение об ошибке, маску ввода. На закладке Подстановка для некоторых полей создайте поля со списком.

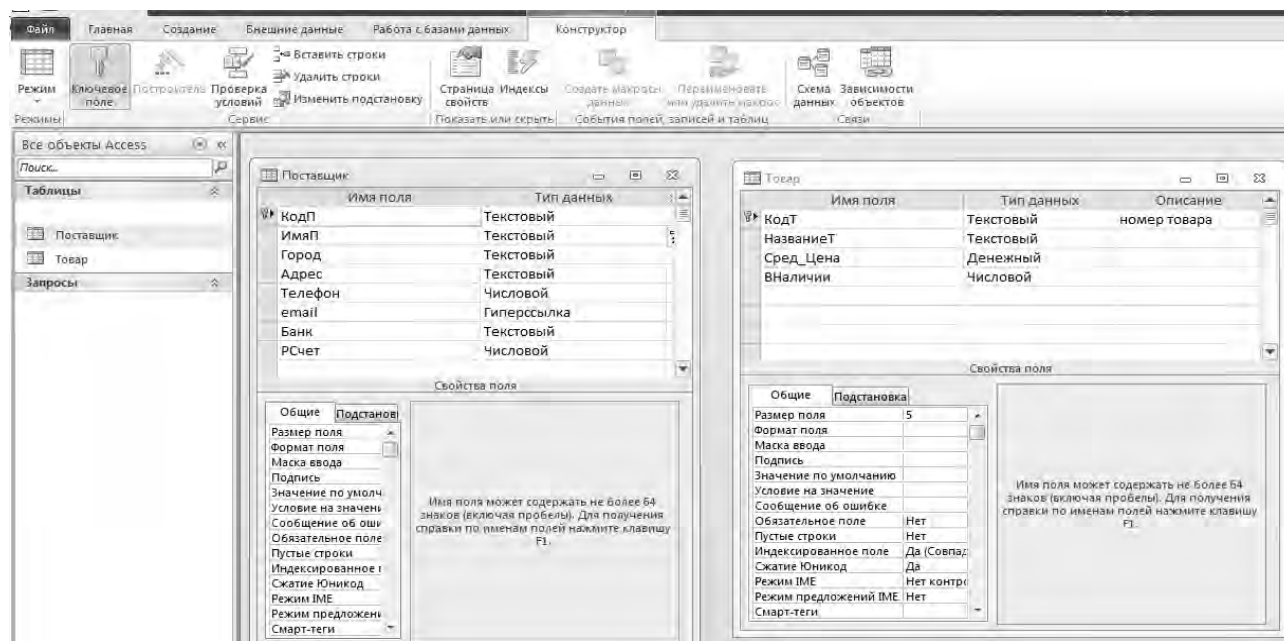


Рисунок 1.2 – Создание таблиц в режиме конструктора

Перейдите из режима конструктора в режим таблицы и внесите в таблицы по несколько записей (рисунок 1.3). Обращайте внимание, как работают дополнительные характеристики полей, которые задавались в разделе **Свойства поля**.

КодП	ИмяП	Город	Адрес	Телефон	email	Банк	РСчет
П1	Иванов	СПБ	Зимний дворец	222333	p1@tut.by	Славнефтебанк	255005450
П2	Петров	Москва	Арбат 12/2	444555	p2@tut.by	Росбанк	405000647
П3	Васильев	Москва	Красная Площадь 1	333444	p3@tut.by	Росбанк	405000632
П4	Сергеев	СПБ	Невский пр-т 5б	555666	p4@tut.by	Газпромбанк	255044783
П5	Сидоров	Минск	Партизанский пр-т 2	222111	p5@tut.by	Белинвестбанк	300984501
П6	Алексеев	Могилев	пр-т Мира 43	333888	p6@tut.by	Беларусбанк	300987503
*				0			0

Рисунок 1.3 – Фрагмент таблицы с данными

## Задание 2

В соответствии с предложенным преподавателем вариантом создать и заполнить таблицы в MS Access.

### Контрольные вопросы и задания

- 1 Дайте определение основным понятиям реляционных баз данных.
- 2 Продемонстрируйте различные способы создания таблиц.
- 3 Установите для ряда атрибутов дополнительные свойства, такие как размер поля, формат, условие на значение, сообщение об ошибке, маску ввода, поле со списком.

## 2 Организация хранения логистической (экономической) информации в MS Access. Целостность реляционных данных

**Цель работы:** изучить структуру реляционной базы данных и способы обеспечения целостности данных, научиться создавать связанные таблицы в Access.

Задача выполнения лабораторной работы заключается в приобретении навыков обеспечения хранения целостной и непротиворечивой информации.

Реляционная модель состоит из трех частей, соответствующих разным аспектам реляционного подхода:

- структурной;
- манипуляционной;
- целостной.

**Структурная часть** отвечает за структуру модели данных (таблицы и связи между ними).

В **манипуляционной части** модели обосновываются два механизма





манипулирования данными: реляционная алгебра и реляционное исчисление.

Реляционная алгебра базируется на классической теории множеств, реляционное исчисление на логическом аппарате исчисления предикатов первого порядка.

**Целостная часть** реляционной модели поддерживает два базовых требования к целостности, которые должны поддерживаться в любой реляционной СУБД.

**Первое:** требование целостности сущностей состоит в том, что любой кортеж отношения отличим от любого другого кортежа этого же отношения. Другими словами любое отношение должно обладать первичным ключом.

**Второе:** требование целостности по ссылкам. Требование целостности по ссылкам (требование внешнего ключа) состоит в том, что для каждого значения внешнего ключа в отношении, на которое ведет ссылка (т. е. в главной таблице), должен найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным (т. е. быть пустым, незаполненным).

На рисунке 2.1 атрибут «Код отдела» сотрудника появляется в отношении **СОТРУДНИКИ** не потому, что он является свойством отношения **СОТРУДНИКИ**, а для того, чтобы установить связь с отношением **ОТДЕЛ**. Атрибут такого рода называется внешним ключом.

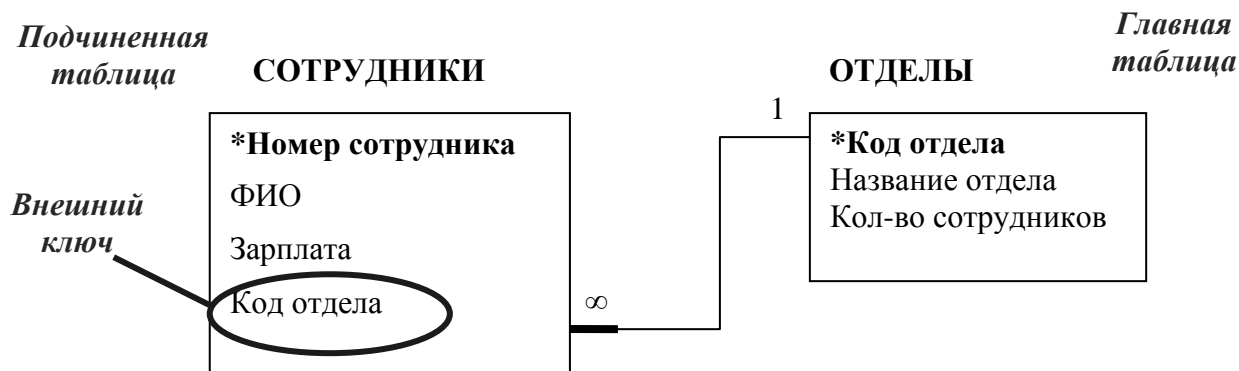


Рисунок 2.1 – Установление связи между главной и подчиненной таблицей

В данном примере это означает, что если для сотрудника указан номер отдела, то этот отдел должен существовать.

### Задание

1 Установите связи, обеспечивающие целостность, между таблицами из предыдущей работы (рисунок 2.2).

2 В подчинённой таблице для атрибутов внешних ключей создайте поля со списками. Элементы списков должны браться из соответствующих главных таблиц.

3 После выполнения учебного примера установите связи между таблицами в индивидуальном варианте из лабораторной работы 1.

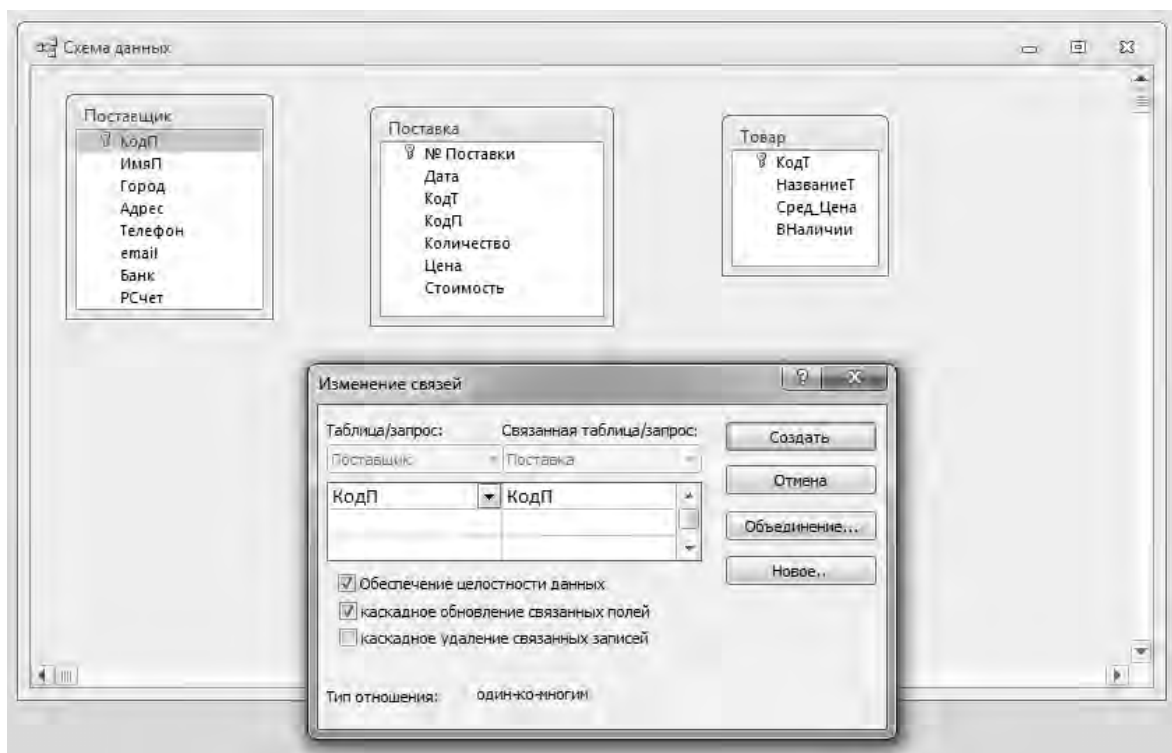


Рисунок 2.2 – Создание связей между таблицами

### **Контрольные вопросы и задания**

- 1 Какие виды связей между таблицами бывают?
- 2 Какие виды целостности поддерживает реляционная модель данных?
- 3 Пр продемонстрируйте создание и изменение связей между таблицами.
- 4 Пр продемонстрируйте на примере, как работает каскадное удаление и обновление связанных полей.

## **3 Обработка логистической (экономической) информации в MS Access. Запросы на выборку**

**Цель работы:** научиться создавать запросы на выборку средствами СУБД MS Access.

**Запросы** используются для просмотра, изменения и анализа данных различными способами. Запросы также можно использовать в качестве источников записей для форм, отчетов.

В Access запросы можно создавать двумя способами: QBE-запросы (Query By Example – запрос по образцу) и SQL-запросы (Structured Query Language – структурированный язык запросов), при создании которых применяются операторы и функции языка SQL.

## Ход работы

Скопировать файл lab3\_sp.mdb в свой каталог и выполнить все задания по методическим указаниям.

### Запросы на выборку

**Запрос 1.** Выбрать сведения о поставщиках из Москвы, которые привозили мониторы в октябре 2019. В результат запроса вывести поля: *ИмяП, Телефон, Город, Дата, Товар*. Использовать таблицы **Товар, Поставка, Поставщик**.

Выбираем создание запроса в режиме конструктора. Добавляем все три таблицы.

Добавить нужные поля в бланк запроса можно или путем перетаскивания их имен из списка, находящегося в верхней части окна конструктора, в строку **Поле**, или двойным щелчком на имени поля. Из таблицы **Поставщик** выберите одним из предложенных способов поля *ИмяП, Город, Телефон*, из таблицы **Поставка** – поле *Дата*, из таблицы **Товар** – поле *НазваниеТ*. Теперь необходимо ввести критерии отбора. Формируемый в данном примере запрос должен отбирать данные о поставщиках из *Москвы*, которые привозили *мониторы* в *октябре 2019*. Поэтому для перечисленных ниже полей установим в строке **Условие отбора** следующие критерии:

**Город** Москва

**НазваниеТ** монитор

**Дата** >=#01.10.19# And <#01.11.19#

После ввода каждого из критериев следует нажимать клавишу <ENTER>, вследствие чего ACCESS проверит его синтаксис и нормализует запись в соответствии с правилами записи критериев. Записи могут быть отсортированы по возрастанию или по убыванию. Отсортируем поставщиков в алфавитном порядке. Для этого в строке **Сортировка** поля *ИмяП* выполните щелчок и в появившемся списке выберите способ сортировки *по возрастанию*. Окно конструктора запроса представлено на рисунке 3.1.

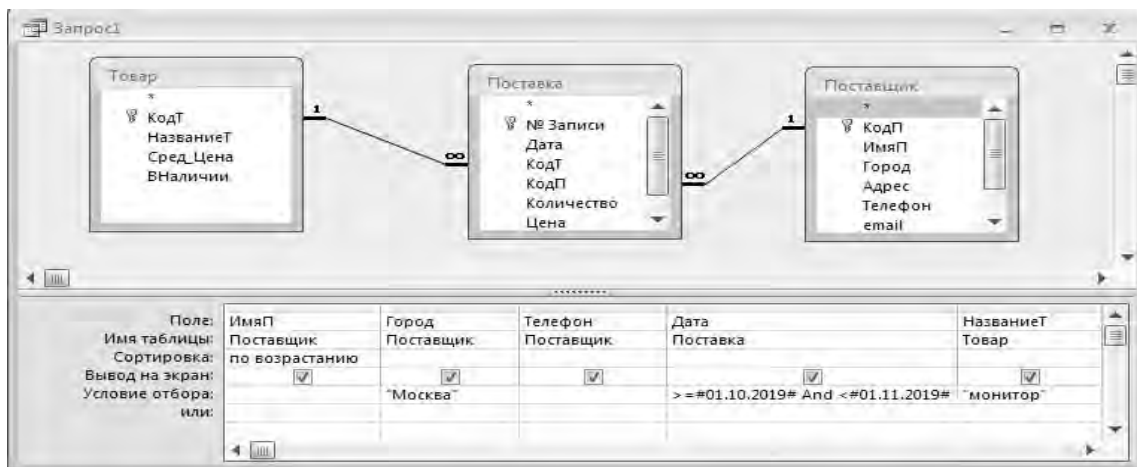


Рисунок 3.1 – Запрос 1 в режиме конструктора

Готовый запрос выполняется после щелчка по кнопке с изображением восклицательного знака на панели инструментов *Конструктора запросов* или командой ЗАПУСК меню ЗАПРОС. Access отобразит на экране результирующий набор записей, которые отобраны из таблицы **Поставщик** в соответствии с заданными критериями (рисунок 3.2).

ИмяП	Город	Телефон	Дата	НазваниеТ
Васильев	Москва	333444	Чт 10.10.19	МОНИТОР
Петров	Москва	444555	Пт 25.10.19	МОНИТОР

Рисунок 3.2 – Результат запроса 1

*Примечание* – Если имя поля содержит знак пробела, то его следует писать в [ ] квадратных скобках

Запрос можно использовать для выполнения расчетов и подведения итогов, обобщив данные из исходных таблиц. Для этих целей в ACCESS предусмотрены статистические функции SQL: Sum, Avg, Min, Max, Count и некоторые другие.

**Запрос 2.** Определить, сколько было поставок каждого товара, максимальную, минимальную и среднюю цену поставки каждого товара. В результат запроса вывести поля: *НазваниеТ* и четыре вычисляемых поля, которым присвоить соответствующие имена: *Число поставок*, *Макс цена*, *Мин цена*, *Средняя цена*. Использовать таблицы **Товар** и **Поставка**.

Это так называемый запрос с группировкой. После вызова окна конструктора запросов и выполнения всех описанных выше действий необходимо в соответствующих полях задать статистические функции **Count**, **Max**, **Min** и **Avg**. Статистические функции задают в строке **Групповая операция** окна **Конструктора запросов**, которая появляется после нажатия кнопки с греческой литерой сигма, расположенной на панели инструментов, или после вызова команды ГРУППОВЫЕ ОПЕРАЦИИ меню ВИД. Новые имена полей { *Число поставок*, *Макс цена*, *Мин цена*, *Средняя цена* } нужно ввести в строку **Поле:** конструктора запросов (рисунок 3.3).

Результат запроса представляет собой следующую таблицу (рисунок 3.4).

**Запрос 3.** Определить количество поставок из Санкт-Петербурга. В результат запроса вывести поля: *Город*, *Число поставок*. Использовать таблицы **Поставка** и **Поставщик**.

Действия по конструированию запроса аналогичны описанным выше, только в строку **Условие отбора** Конструктора запросов необходимо ввести критерий отбора по полю *Город* «СПБ» (рисунок 3.5).

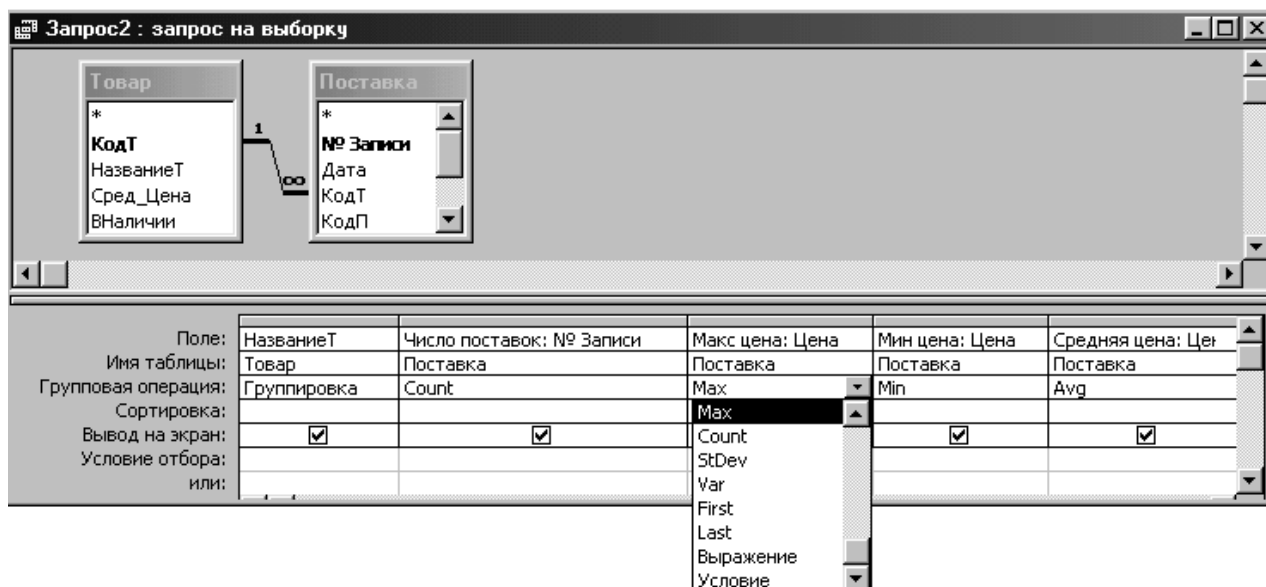


Рисунок 3.3 – Запрос с групповыми операциями в режиме конструктора

	НазваниеТ	Число поставок	Макс цена	Мин цена	Средняя цена
▶	DVD	3	80	65	71,7
	МОДЕМ	2	90	50	70,0
	МОНИТОР	3	265	240	251,7
	ПРИНТЕР	1	190	190	190,0
	СКАНЕР	4	250	215	230,8

Запись: 1 из 5

Рисунок 3.4 – Результат выполнения запроса с групповыми операциями

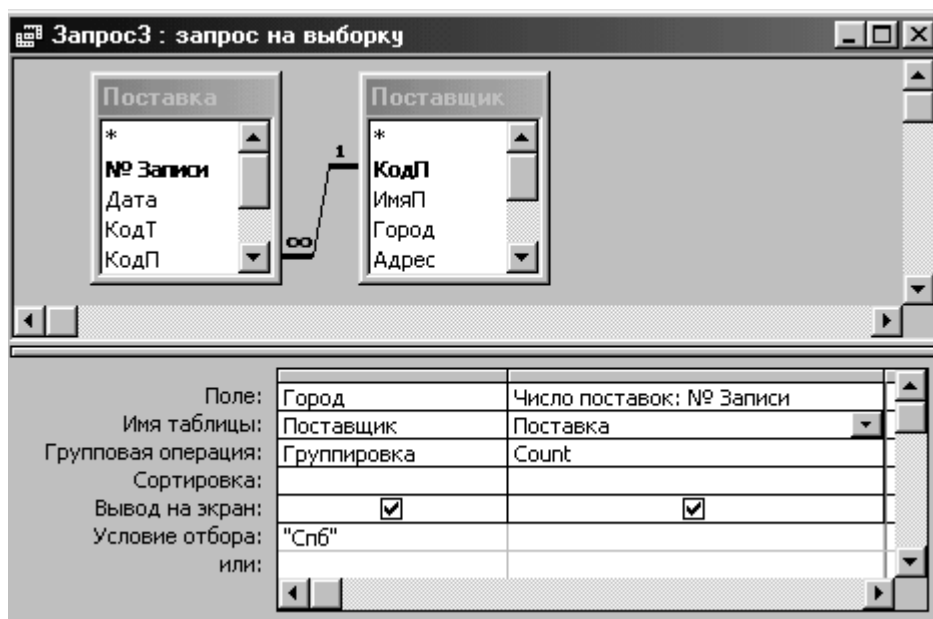
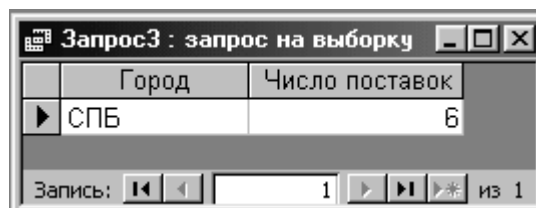


Рисунок 3.5 – Запрос с групповыми операциями и условием отбора в режиме конструктора

Результат выполнения запроса представлен в таблице на рисунке 3.6.



Город	Число поставок
СПБ	6

Рисунок 3.6 – Результат выполнения запроса

### Задание

Составить следующие запросы.

- 1 Выбрать сведения о поставщиках, фамилии которых начинаются на букву «А» или «С» и проживающих не в Минске.
- 2 Выбрать сведения о поставщиках, привозивших сканеры в августе. Выполнить сортировку в хронологическом порядке.
- 3 Получить список товаров, поставок которых было три и более.
- 4 Определить среднюю стоимость поставки по каждому поставщику.
- 5 Определить количество поставок и их суммарную стоимость за каждый месяц.
- 6 Получить список товаров, у которых цена выше средней.

## 4 Изменение логистической (экономической) информации в MS Access. Запросы на изменение данных

**Цель работы:** научиться создавать запросы на модификацию информации средствами СУБД MS Access.

*Запросы на обновление.* Часто возникает необходимость изменить значения какого-либо поля в группе записей таблицы, отобранных на основании определенного критерия или во всех записях таблицы (например, в связи с инфляцией повысить заводскую цену каждой модели автомобиля на определенный процент или рассчитать значение какого-либо поля по уже внесенным в таблицу данным). Для того, чтобы не вводить заново новые значения во все записи, используется так называемый ЗАПРОС НА ОБНОВЛЕНИЕ.

**Запрос 1.** Заполнить поле *Стоимость* таблицы **Поставка**, рассчитав его значения по следующей формуле:

$$\text{Стоимость} = \text{Цена} \cdot \text{Количество}.$$

Для реализации этого запроса загружаем **Конструктор запросов**, затем выбираем тип запроса **ОБНОВЛЕНИЕ**, добавляем таблицу **Поставка**,



выбираем поле **Стоимость** и в появившуюся строку ОБНОВИТЬ Конструктора запросов вводим следующее выражение: [Цена]·[Количество].

Чтобы расчет выполнялся только для тех записей, для которых стоимость ещё не вычислена, в строке **Условие отбора** пишем выражение **Is Null** (рисунок 4.1). Иначе пересчет выполнялся бы для всех записей в таблице, и при большом их количестве это занимало бы лишнее время.

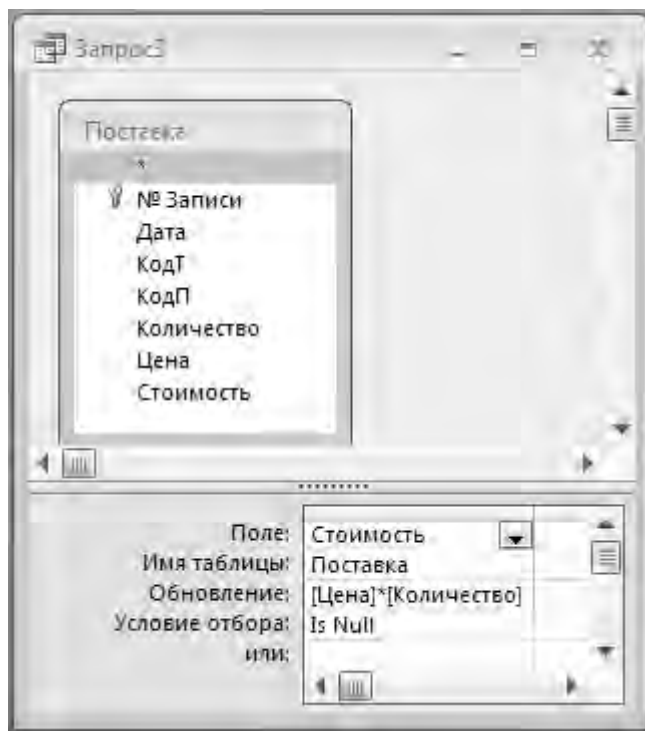


Рисунок 4.1 – Запрос на обновление в режиме конструктора

После запуска запроса на выполнение появится диалоговое окно (рисунок 4.2), в котором нужно нажать кнопку «Да».

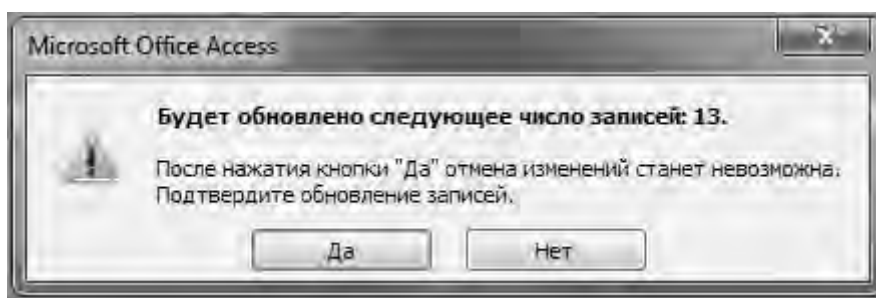


Рисунок 4.2 – Окно предупреждения об обновлении записей

После этого откройте таблицу **Поставка** и убедитесь, что поле **Стоимость** оказалось заполнено рассчитанными значениями.

*Запрос на создание таблицы* создает новую таблицу на основе всех или части данных из одной или нескольких таблиц. В отличие от запроса на выборку в результате выполнения данного вида запроса создается таблица,

в которой будут сохранены отобранные данные.

**Запрос 2.** Выбрать сведения обо все крупных поставках (например, стоимость поставки более 50 000 р.) В результат запроса вывести поля: *№ Записи, Дата, ИмяП, НазваниеТ, Стоимость*.

Сначала нужно создать обычный запрос на выборку. Для поля *Стоимость* в строку **Условие отбора** написать «> 50000».

Далее нужно сохранить результат запроса в таблице **Крупные поставки**.

В меню ЗАПРОС выберите **Создание таблицы**. Откроется диалоговое окно **Создание таблицы** (рисунок 4.3). В поле **имя таблицы** введите имя новой таблицы «**Крупные поставки**» и выберите параметр **в текущей базе данных**. Нажмите кнопку **ОК**.

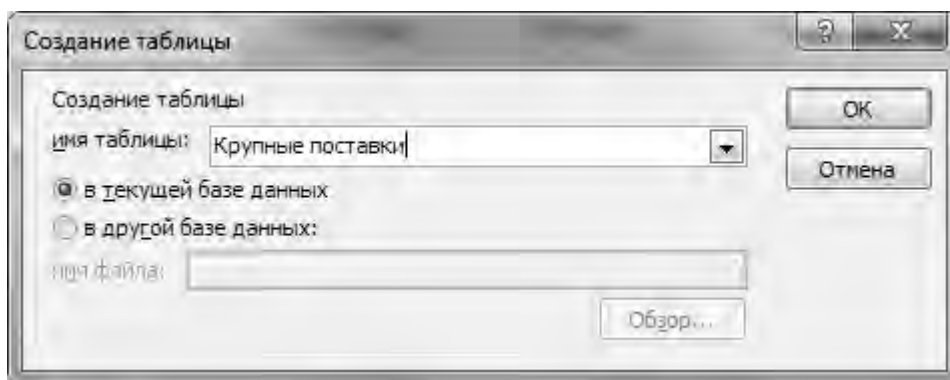


Рисунок 4.3 – Диалоговое окно создания таблицы

Для просмотра новой таблицы до ее создания нажмите кнопку **Вид** на панели инструментов. Чтобы вернуться в режим конструктора запроса и изменить или запустить запрос, снова нажмите кнопку **Вид** на панели инструментов.

Для создания новой таблицы нажмите кнопку **Запуск** на панели инструментов. На экране появится сообщение о том, что в новую таблицу будет помещено четыре записи. Нажмите кнопку «Да». После выполнения запроса перейдите на закладку **Таблицы** в окне базы данных и убедитесь, что появилась новая таблица.

*Запрос на удаление* удаляет группу записей из одной или нескольких таблиц. Например, запрос на удаление позволяет удалить записи о товарах, поставки которых прекращены или на которые нет заказов.

**С помощью запроса на удаление можно удалять только всю запись, а не отдельные поля внутри нее.**

**Запрос:** удалить из таблицы **Крупные поставки** записи со стоимостью поставки менее 80 000 р.

Создайте новый запрос в режиме конструктора и добавьте таблицу **Крупные поставки**. В меню ЗАПРОС выберите **Удаление**. В бланке запроса укажите поле **Стоимость** и Условие «< 80 000». Запустите запрос на выполнение. Подтвердите удаление двух записей и посмотрите результат выполнения запроса в таблице **Крупные поставщики**.



При удалении записей с помощью запроса на удаление отменить операцию невозможно. Поэтому, прежде чем выполнить такой запрос, необходимо просмотреть выбранные для удаления данные. Для этого на панели инструментов нажмите кнопку **Вид** и просмотрите запрос в режиме таблицы.

Если бланк запроса оставить пустым, то удаляться все записи из таблицы.

### Задание 1

Составить следующие запросы.

– обновить поле *ВНаличии* таблицы **Товар**, рассчитав его, как общее количество каждого товара по таблице **Поставка**.

– увеличить цену товаров на 15 %.

### Задание 2

Самостоятельно разработать и выполнить все изученные виды запросов на основе индивидуального варианта задания из предыдущих лабораторных работ.

## 5 Язык SQL

**Цель работы:** научиться создавать запросы на структурированном языке запросов SQL.

Основной конструкцией SQL является конструкция

**Select**  
**From**  
**Where**

### Инструкция SELECT

*Синтаксис*

**SELECT** [ALL / DISTINCT / [TOP *n* [PERCENT]]]

**FROM** *таблица*

**ALL**

Если инструкция SQL не содержит ни одного предиката, то подразумевается предикат ALL. Ядро базы данных MicrosoftJet отбирает все записи, соответствующие условиям, заданным в инструкции SQL. Приведенные ниже инструкции SQL эквивалентны. Они возвращают все записи из таблицы «Поставщик»:



```
SELECT ALL *
FROM Поставщик
ORDER BY НомерПост ASC;
```

```
SELECT *
FROM Поставщик
ORDER BY НомерПост;
```

### ORDER BY

Сортировка по возрастанию (ASC) или убыванию (DESC). Если предикат опущен, то по умолчанию выполняется сортировка по возрастанию.

### DISTINCT

Исключает записи, которые содержат повторяющиеся значения в выбранных полях. Чтобы запись была включена в результат выполнения запроса, значения в каждом поле, включенном в инструкцию SELECT, должны быть уникальными. Например, в таблице **Поставщик** есть однофамильцы. Если две записи содержат значение «Иванов» в поле *Фамилия*, то следующая инструкция SQL возвратит только одну из них:

```
SELECT DISTINCT Фамилия
FROM Поставщик;
```

Если опустить предикат DISTINCT, этот запрос возвратит обе записи для фамилии Иванов. Если предложение SELECT содержит более одного поля, то для включения записи в результат выполнения запроса необходимо, чтобы совокупность значений во всех этих полях была уникальной.

### TOP n [PERCENT]

Возвращает определенное число записей, находящихся в начале или в конце диапазона, отсортированного с помощью предложения ORDER BY. Следующая инструкция SQL позволяет получить список 25 лучших студентов выпуска 2018 г.:

```
SELECT TOP 25 Имя, Фамилия
FROM Студенты
WHERE ГодВыпуска = 2018
ORDER BY СреднийБалл DESC;
```

Если предложение ORDER BY будет опущено, запрос возвратит произвольный набор 25 записей из таблицы **Студенты**, удовлетворяющих предложению WHERE. Предикат TOP не осуществляет выбор между равными значениями. Если в предыдущем примере средние балы двадцать пятого и двадцать шестого студента будут равны, то запрос возвратит 26 записей. Кроме того, можно использовать зарезервированное слово PERCENT для возврата определенного процента записей, находящихся в начале диапазона. Предположим, что вместо 25 лучших студентов следует отобрать студентов, попавших в последние 10 %:



```
SELECT TOP 10 PERCENT Имя, Фамилия
FROM Студенты
WHERE ГодВыпуска = 2017
ORDER BY СреднийБалл ASC;
```

### **Инструкция SELECT...INTO...IN**

Создает запрос на создание таблицы.

В следующем примере из таблицы «ПОСТАВЩИК» отбираются все записи, а затем копируются в новую таблицу с именем «КопияПоставщики»:

```
SELECT * INTO [КопияПоставщики]
FROM Поставщик;
```

В следующем примере создается копия таблицы «ПОСТАВЩИК» и затем помещается **в базу данных** Копия.mdb:

```
SELECT Поставщик.* INTO ПОСТАВЩИК IN Копия.mdb
FROM Поставщик;
```

В следующем примере показывается, какова была бы цена товаров, если бы она возросла на 10 %. Пример не изменяет существующие в базе данных цены.

```
SELECT TN, Цена AS ТекущаяЦена,
           Цена * 1.1 AS НоваяЦена
FROM P;
```

### **Задание**

Напишите запросы из тем 3 и 4 на языке SQL.

### **Контрольное задание**

По заданию преподавателя написать на языке SQL запросы и пояснить назначение основных операторов.

## **6 Методы проектирования БД**

**Цель работы:** научиться создавать логическую и физическую модели базы данных.

Проектирование базы данных начинается с описания предметной области и постановки задачи (технического задания).

**Предметная область** – это часть реального мира, данные о которой требуется отразить в базе данных. Например, в качестве предметной области можно выбрать бухгалтерию какого-либо предприятия, отдел кадров,



банк, магазин и т. д.

От грамотного и детального описания зависит состав и функциональные возможности будущей базы данных.

База данных должна удовлетворять комплексу требований. Эти требования следующие:

- 1) целостность базы данных – требование полноты и непротиворечивости данных;
- 2) многократное использование данных;
- 3) быстрый поиск и получение информации по запросам пользователей;
- 4) простота обновления данных;
- 5) минимизация избыточности данных;
- 6) защита данных от несанкционированного доступа, искажения и уничтожения.

Реляционная база данных представляет собой совокупность связанных между собой таблиц, предназначенных для хранения данных. Таблица реляционной базы данных состоит из множества строк и столбцов. Каждая строка таблицы содержит данные об одном объекте и называется записью. Все записи имеют одинаковую структуру – они состоят из полей, в которых хранятся атрибуты (свойства) объекта. Каждое поле записи содержит некоторое свойство представляемого объекта.

Каждая таблица должна иметь один или несколько столбцов (атрибутов), которые однозначно идентифицируют каждый объект в таблице, т. е. позволяют однозначно отличить один объект от другого. Такие столбцы образуют первичный ключ, и если столбцов несколько, то говорят, что первичный ключ является составным. Поле, представляющее первичный ключ или являющееся частью первичного ключа, называется ключевым. В реляционной базе данных очень важным является понятие связи между таблицами. Связь (relationship) – это логическое отношение между объектами, представленными таблицами. Связь между записями двух таблиц основана на совпадении атрибутов, по которым эта связь устанавливается. Связи бывают следующих видов: один-к-одному, один-ко-многим, многие-ко-многим.

Исходной точкой является представление предметной области в виде одной или нескольких таблиц, и на каждом шаге проектирования путём их декомпозиции формируется новый набор взаимосвязанных таблиц.

Такой процесс проектирования получил название **нормализации**, а состояние набора взаимосвязанных таблиц называется **нормальной формой** (НФ). Каждой нормальной форме соответствуют свои требования, и считается, что модель находится в конкретной нормальной форме, если она удовлетворяет её требованиям. Каждая следующая нормальная форма обладает свойствами лучшими, чем предыдущая.

Основная цель перехода от одной НФ к другой – исключение дублирования информации.

Каждой нормальной форме соответствует некоторый определенный набор ограничений, и отношение находится в некоторой нормальной форме,



если удовлетворяет свойственному ей набору ограничений.

Для *первой* нормальной формы (1НФ) – значения всех атрибутов должны быть атомарны (неделимы). Поскольку требование первой нормальной формы является базовым требованием классической реляционной модели данных, считается, что исходный набор отношений уже соответствует этому требованию.

Требование *второй* нормальной формы (2НФ) – отсутствие частичной зависимости. Другими словами все атрибуты отношения должны полностью зависеть от составного первичного ключа.

Ситуация с частичной зависимостью может иметь место тогда, когда первичный ключ составной и часть атрибутов зависит только от одной его составляющей.

Требование *третьей* нормальной формы (3НФ) – отсутствие транзитивной зависимости.

Транзитивная зависимость – это зависимость какого-либо неключевого атрибута не напрямую от ключа, а от другого неключевого атрибута.

Чтобы избавиться от частичной и транзитивной зависимости, атрибуты, которые её образуют, выделяются в отдельные таблицы. Новые таблицы связываются с исходной по внешнему ключу.

В большинстве случаев достижения 3НФ считается достаточным.

### **Задание**

Из предложенного списка выбрать предметную область или согласовать с преподавателем свой вариант. Сделать её описание, сформулировать требования к составу информации и её возможностям. Разработать модель реляционной базы данных, привести её в 3НФ и реализовать в MSAccess.

Примеры предметных областей:

- 1) грузоперевозки;
- 2) прокат (видео, вечерние платья и т. п.);
- 3) гостиница (учет постояльцев и свободных номеров);
- 4) экзаменационная ведомость;
- 5) учет нарушений ПДД водителями;
- 6) регистрация автотранспортных средств;
- 7) учет выполнения договорных обязательств (контракт, оплата, поставка);
- 8) отдел кадров: учет отпусков сотрудников, перемещение сотрудников;
- 9) товары (номенклатура, ассортимент).

### **Контрольные вопросы**

- 1 Что называется предметной областью?
- 2 Что представляет собой реляционная модель данных?
- 3 Каким требованиям должна отвечать реляционная модель данных?
- 4 Дайте понятие составного первичного ключа.
- 5 В чем заключается процесс нормализации отношений в реляционной модели данных?



- 6 Какие зависимости называются частичными и транзитивными?  
7 Дайте определение первой, второй и третьей нормальной формы.

## 7 Создание интерфейса. Формы и элементы управления

**Цель работы:** научиться создавать простые и составные формы для просмотра и ведения базы данных.

Формы служат удобным средством для ввода, просмотра и редактирования информации БД. Формы для ввода представляют собой бланк, подлежащий заполнению, и дают возможность осуществлять контроль вводимых данных и исключать ввод неверных. Форма-бланк упрощает процесс заполнения БД, благодаря чему в базу могут вводить информацию работники с невысокой квалификацией. При просмотре и редактировании пользователь имеет дело с маской, накладываемой на набор данных. Форма-маска позволяет ограничить доступ пользователя к информации, заблокировав отдельные поля или записи. Формы есть простые и составные. Они могут содержать различные элементы: поля БД и подписи к ним, списки, флажки, переключатели. В них возможны вычисления для отдельных записей и их групп, а также наглядное графическое представление данных в виде диаграмм.

Любая форма строится на основе Access-таблицы или запроса. На основе одной таблицы пользователь может построить несколько форм: одну для руководителя, другую – для бухгалтера, третью – для работника склада и т. д. Поля таблицы, помещаемые в форму, выбираются из таблицы в соответствии с назначением формы. Форму можно создать тремя способами:

- 1) с помощью *конструктора* форм;
- 2) с помощью *мастера* по созданию форм;
- 3) используя *автоформу*.

Форма может быть представлена на экране в одном из трех режимов:

- 1) в режиме конструктора;
- 2) в режиме формы (рабочем режиме);
- 3) в режиме таблицы.

Переключение между перечисленными режимами осуществляется либо с помощью команд меню ВИД, либо с помощью кнопки ВИД на панели инструментов (первая кнопка слева), открывающей список с названиями режимов.

### Создание форм с помощью мастера.

Выберите элемент **Мастер форм** для создания формы. Название таблицы на этом этапе можно не указывать. После нажатия кнопки ОК появится первое диалоговое окно мастера, в котором следует выбрать таблицы и поля для проектируемой формы.

В списке **Таблицы/Запросы** выберите таблицу **Товар**, а в списке **Доступные поля** – поля, которые предполагается использовать в форме,



и перенесите их в список **Выбранные поля** кнопкой с двойной стрелкой. В результате в списке отобразятся названия всех полей. Затем из списка **Таблицы/Запросы** выберите таблицу **Поставка**, на основании которой строится подчиненная форма. Выберите также все поля. Затем из таблицы **Поставщик** добавьте поле *ИмяП*.

Окончив эту процедуру, нажмите кнопку **Далее**. В результате на экране появится окно с установленной опцией **Подчиненные формы**, указывающей на то, что будет сформирована форма с подчиненной формой (рисунок 7.1).

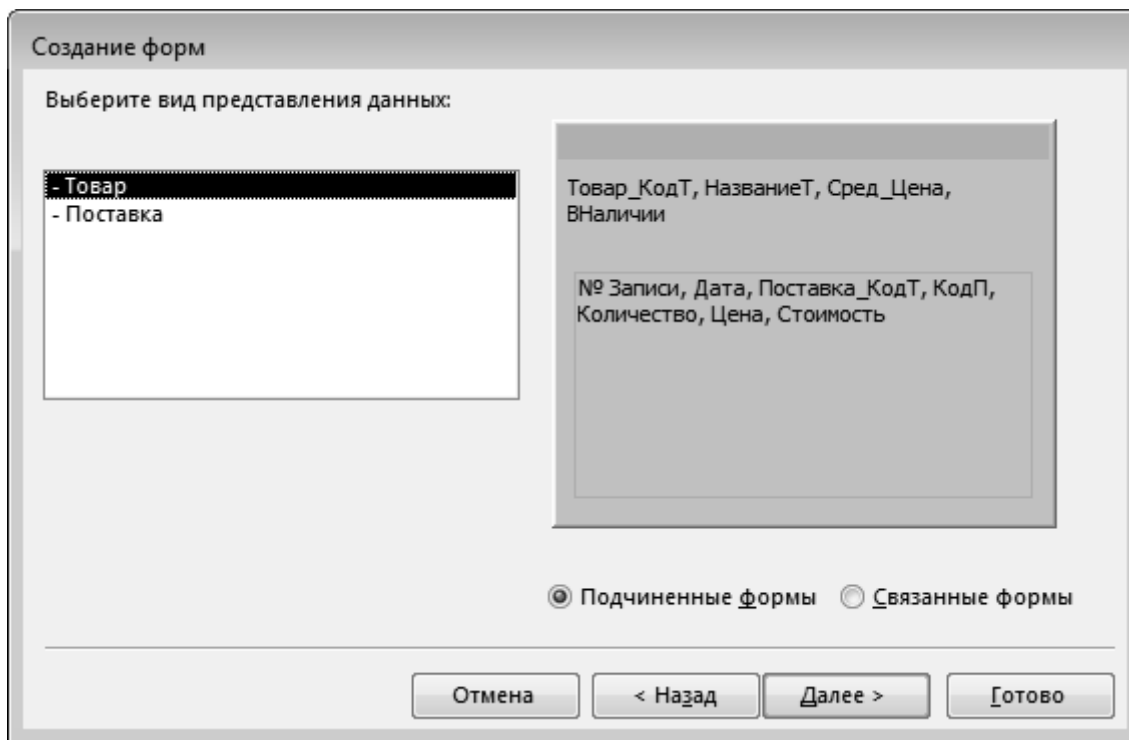


Рисунок 7.1 – Создание составной формы с помощью мастера форм

Проанализировав связи, Access создаст главную форму на основе таблицы **Товар**. Чтобы перейти к следующему окну, нажмите кнопку **Далее**, хотя на этом можно было бы и завершить проектирование формы, нажав кнопку **Готово**. В следующих окнах **Мастера форм** остается только подтвердить установки по умолчанию. В результате Access создаст проект формы, который в дальнейшем может быть доработан пользователем по своему усмотрению. Чтобы сделать форму более привлекательной, в нее можно добавить рисунки, надписи, изменить расположения отдельных полей. Для редактирования формы следует пользоваться режимом конструктора. Пример готовой формы приведен на рисунке 7.2.

Поскольку и главная, и подчиненная формы в процессе создания сохранялись отдельно и имеют уникальное имя, каждая из них может использоваться самостоятельно.

**Товар**

КодТ: T1      Сред\_Цена:

НазваниеТ: СКАНЕР      ВНаличи:

**Поставка**

№ Записи	Дата	КодТ	КодП	ИмяП	КОЛИЧЕСТВО	Цена	Стоимос
1	12.08.2004	T1	П1	Иванов	50	238,00	
2	15.08.2004	T1	П2	Петров	30	250,00	
3	16.08.2004	T1	П3	Васильев	12	220,00	
4	21.08.2004	T1	П4	Сергеев	40	215,00	

\* [Четчик]

Запись: 1 из 4

Запись: 1 из 5

Рисунок 7.2 – Составная форма

### Редактирование формы в режиме конструктора

Откройте форму в режиме конструктора.

В меню **Вид** или с помощью правой клавиши мыши выберите **Свойства**. Окно свойств откроется для того элемента формы, который был активизирован. Активизировать всю форму, а не её отдельные элементы можно нажав на квадратик в левом верхнем углу или на темно-серый фон на макете формы. Также нужный элемент можно выбрать из списка непосредственно в окне свойств.

Обычно, часто используемые пункты меню вынесены на панели в виде пиктограмм (рисунок 7.3).

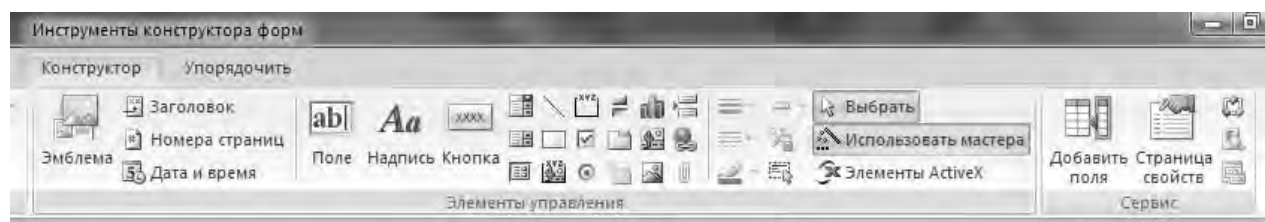


Рисунок 7.3 – Панель инструментов конструктора форм

### Задание 1

Создайте с помощью мастера:

- простую форму **ТОВАР** на основе таблицы **Товар**;
- составную форму **ПОСТАВЩИКИ\_ПОСТАВКИ**, в которой в качестве объекта для главной формы используйте таблицу **Поставщик**, для подчиненной формы – таблицу **Поставка**.

Разместите на форме **ПОСТАВЩИКИ\_ПОСТАВКИ** кнопки для перемещения по записям, закрытия формы, открытия формы **ТОВАР** для выбранного наименования товара.



## Создание формы в режиме конструктора

- 1 Выберите **Создание формы в режиме конструктора**.
- 2 На экране появится макет формы. Сохраните её под именем **ФПоставщики**.
- 3 Нажмите на правую кнопку мыши и отметьте пункт **Заголовок/Примечание формы**. Это можно сделать и через меню **Вид**.
- 4 Выберите **Свойства**. Окно свойств откроется для того элемента формы, который был активизирован. Активизировать всю форму, а не её отдельные элементы можно нажав на квадратик в левом верхнем углу или на темно-серый фон на макете формы («Область формы» на рисунке 7.4). Также нужный элемент можно выбрать из списка непосредственно в окне свойств.
- 5 В окне свойств для объекта «Форма» выберите закладку **Данные** и в поле **Источник записей** – таблицу **Поставщики**. В меню **Вид** выберите **Список полей** (этот список может появиться автоматически после выбора источника записей) и Панель элементов (см. рисунок 7.4).

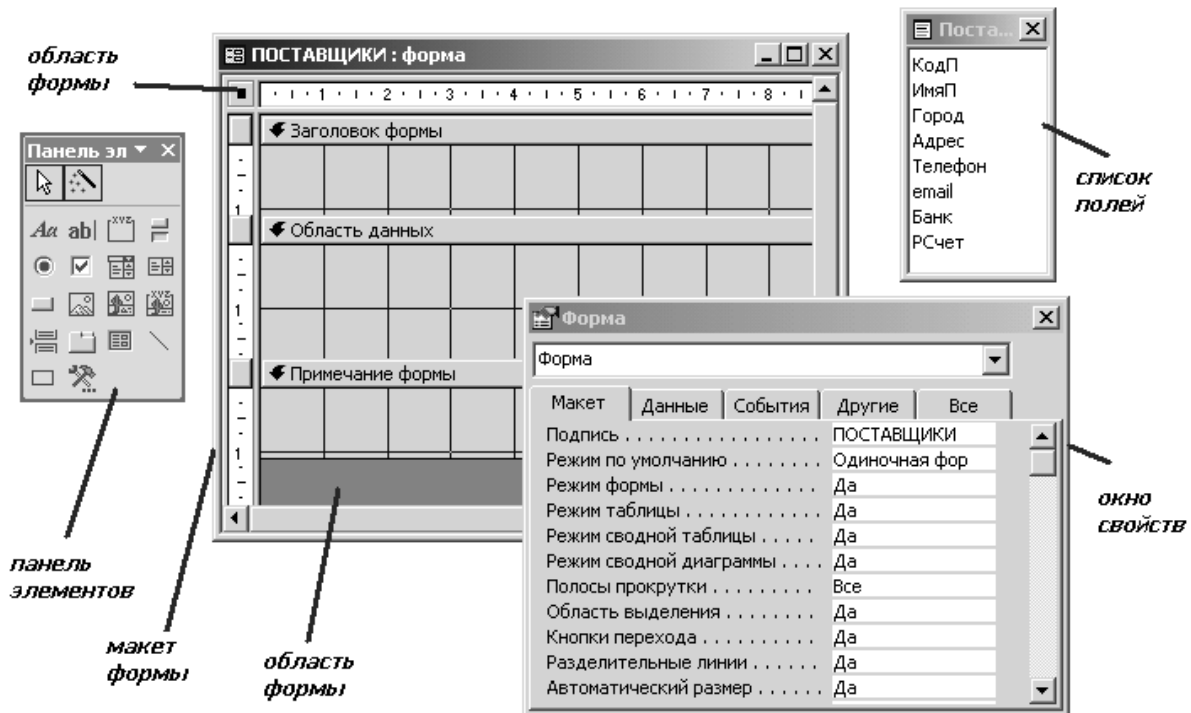


Рисунок 7.4 – Инструменты создания форм в режиме конструктора

- 6 Выделите в списке полей все атрибуты и перетащите их мышкой на форму в область данных.

Каждый атрибут на формы представлен с помощью двух элементов: **Надпись** и **Поле**. Надпись можно редактировать, выделив её и нажав клавишу F2, или в окне свойств: **Макет\Подпись**. Если необходимо переместить только **Надпись** или только **Поле**, то это можно сделать, удерживая нужный элемент за левый верхний угол. Для перемещения всей пары **Надпись–Поле** удерживайте мышкой за периметр любого из двух элементов. Эlemen-

ты **Надпись** и **Поле** находятся в группе **Элементы управления** на закладке конструктора форм. Для подчиненной формы используется элемент **Подчиненная форма/отчет**.

### Задание 2

- 1 Для своего варианта создайте простые и составные формы.
- 2 На формах поместите кнопки для открытия и закрытия форм, перехода по записям, поиска записей.
- 3 Создайте главную кнопочную форму для работы со всей базой данных.

## 8 Создание и использование макросов

**Цель работы:** научиться автоматизировать функции базы данных с помощью макросов.

Макрос – это инструмент, позволяющий автоматизировать задачи и добавлять функции в формы, отчеты и элементы управления. Макросы Access можно рассматривать как упрощенный язык программирования, код на котором создается в виде списка необходимых действий. Создавая макрос, пользователь выбирает каждую макрокоманду из раскрывающегося списка, а затем вводит для нее необходимую информацию. С помощью макросов можно добавлять функции в формы, отчеты и элементы управления без необходимости писать код в модуле VBA.

Организуем диалог пользователя с базой данных следующим образом (рисунок 8.1): есть список товаров, о которых у нас уже есть информация; по мере выбора какого-либо товара из данного списка выводится список имен поставщиков, которые этот товар привезли и в каком количестве. Также есть поле, в котором выводится общее количество этого товара в наличии и средняя его цена, вычисляемая по всем поставкам. Выбирая запись для конкретного поставщика, можно получить по нему более подробную информацию, нажав на кнопку **«Поставщики»** (рисунок 8.2).

Сначала нужно создать две формы: одну (**ФПоставка**) – на основе подчиненной таблицы **Поставка**, добавив для наглядности поле с именем поставщика из таблицы **Поставщик** (рисунок 8.2); другую (**ФПоставщик**) – на основе таблицы **Поставщик** (рисунок 8.3).

Формы создаются с помощью **Мастера**.

При создании формы **ФПоставка** сначала выбираете все поля из таблицы **Поставка**, затем поле **ИмяП** из таблицы **Поставщик**. Для формы выбираете ленточный вид. После создания формы открываете её в режиме конструктора и в примечание добавляете кнопку (никакие действия пока для неё не назначаете, если запустится мастер кнопок, нажмите **Отмену**). Должны получить форму как на рисунке 8.2.



ТОВАРЫ

Средняя цена: 70,00

В наличии:

КодТ: ТЗ

№	Дата	КодП	ИмяП:	КОЛИЧЕСТВО	Цена	Стоимость
8	29.09.2004	П1	Иванов	50	90,00	4 500,00
9	05.10.2004	П3	Васильев	100	50,00	5 000,00

Поставщик

Запись: 1 из 2

Нет фильтра Поиск

Запись: 3 из 5

Нет фильтра Поиск

Рисунок 8.1 – Пример формы с элементами управления

Поставка

№	Дата	КодП	ИмяП:	КОЛИЧЕСТВО	Цена	Стоимость
1	12.08.2004	П1	Иванов	50	238,00	11 900,00
5	14.09.2004	П1	Иванов	300	70,00	21 000,00
8	29.09.2004	П1	Иванов	50	90,00	4 500,00
10	05.10.2004	П1	Иванов	250	190,00	47 500,00
2	15.08.2004	П2	Петров	30	250,00	7 500,00
6	15.09.2004	П2	Петров	400	65,00	26 000,00

Поставщик

Запись: 1 из 16

Нет фильтра Поиск

Рисунок 8.2 – Форма «Поставка» ленточного типа

Поставщик

КодП: 1

ИмяП: Иванов

Город: СПБ

Адрес: Зимний дворец

Телефон: 222333

email: p1@tut.by

Банк: Славнефтебанк

РСчет: 255005450

Запись: 1 из 1

С фильтром Поиск

Рисунок 8.3 – Форма «Поставщик»

Для формы **ФПоставщик** выбираете одиночный тип формы (см. рисунок 8.3). Теперь нужно создать главную форму.

Выберите создание формы в режиме конструктора, в пункте свойств формы *Источник* записей выберите таблицу **Товар**.

В меню **Вид** выберите пункт **Список полей**. Из открывшегося списка перетащите на форму все четыре поля.

На **Панели элементов** выберите элемент *Подчиненная форма/отчет* и поместить его на форму. В его свойствах задайте ему имя *Fslave* (закладка *Другие*), на закладке *Данные* в качестве Объекта-источника выберите форму **ФПоставка**. Связь между главной и подчиненной формой осуществляется по ключевым полям *КодТ*, что и должно быть указано в пунктах *Основные поля* и *Подчиненные поля* (рисунок 8.4).



Рисунок 8.4 – Окно свойств для подчинённого объекта

Сохраните форму под именем **ФТовар** и откройте для просмотра. Сначала форма откроется для первой записи. На главной форме будут видны номер и название товара, и в окне подчиненного объекта будет выведена дополнительная информация о том, кто привез именно этот товар. При переходе к другой записи будет меняться и информация на подчиненной форме.

Добавьте на форму элемент *Поле со списком*, задайте ему имя *list* (**Свойства/закладка Другие / Имя**). Если запустится мастер – отмените. Далее нужно выбрать источник строк для этого списка. В окне свойств объекта на закладке *Данные* в строке *Тип источника записей* должен быть выбран тип *Таблица/Запрос*. В поле *Источник строк* с помощью построителя запросов, либо записи на SQL выберите из таблицы **Товар** поля *КодТ* и *НазваниеТ*. Также для данного списка установите следующие параметры:

Закладка **Данные**:

Присоединенный столбец	1
Значение по умолчанию	“Т1”

Закладка **Макет**:

Число столбцов	2
Ширина столбцов	0см;4см

Так как теперь поля с номером и наименованием товара больше не нужны, однако к ним придется в последствии обращаться, поэтому их можно скрыть под внедренным объектом.

Теперь нужно сделать так, чтобы при выборе какой-либо записи из списка обновлялась информация на подчиненной форме, а при нажатии на кнопку Поставщики открывалась форма **ФПоставщик**, но только для конкретного поставщика. Для этого создаются макросы.

В окне базы данных на закладке Макросы выберите [Создать]. Появится окно для написания макроса. По умолчанию оно имеет два раздела: Макрокоманда и Примечание. В нижней части экрана находится область аргументов макрокоманды. В меню Вид добавьте еще один раздел – **Именамакросов**. На первой строке напишите имя макроса 1. В колонке Макрокоманда из списка выберите команду *КЭлементуУправления*. В аргументах макрокоманды напишите имя элемента, которому передаете управление. В данном случае Вы переходите от списка к ключевому полю КодТ, по которому и осуществляется связь между главным и подчиненным объектами. Далее нужно найти запись, где значение ключевого поля КодТ совпадает со значением в поле со списком list. На следующей строке колонки Макрокоманда выберите следующую макрокоманду *НайтиЗапись*. В строке *Образец поиска* напишите =[list].

Далее напишите имя нового макроса 2, который будет открывать форму **ФПоставщик**. В меню Вид добавьте еще один раздел – **Условия**. В колонке Макрокоманда выберите команду ОткрытьФорму.

В аргументах макрокоманды напишите имя формы **ФПоставщик**, а в условиях отбора следующую строку:

[КодП]=[Forms]![ФТовар]![Fslave]![КодП].

В колонке **Условия** запись [Forms]![ФТовар]![Fslave]![КодП] IsNotNull обеспечивает корректное выполнение макрокоманды. Данная команда не будет выполняться для пустой строки. Конструктор макросов представлен на рисунке 8.5.

Сохраните книгу макросов под именем *Макрос1* и вернитесь к конструктору формы **ФТовар**. Первый макрос должен запускаться каждый раз после обновления значений списка. На закладке **События** окна свойств для поля со списком **list** подключите созданный макрос *Макрос1.1* к событию *После обновления* (рисунок 8.6).

Откройте форму **ФПоставка** в режиме конструктора и *Макрос1.2* подключите к событию для кнопки *Нажатие кнопки*. Сохраните и закройте форму **ФПоставка**. Переведите форму **ФТовар** в режим просмотра. Теперь при выборе какой-либо записи из списка обновлялась информация и на подчиненной форме а при нажатии на кнопку Поставщики открывается форма **ФПоставщик** для конкретного поставщика.



```

☐ Вложенный макрос: 1
  КЭлементуУправления
    Имя элемента КодТ
  НайтиЗапись
    Образец поиска =[list]
    Совпадение Поля целиком
    С учетом регистра Нет
    Область поиска Все
    С учетом формата поля Нет
    Только в текущем поле Да
    Первое вхождение Да
  Конец вложенного макроса
☐ Вложенный макрос: 2
  ☐ Если [Формы]![ФТовар]![Fslave]![КодП] Is Not Null to
    ОткрытьФорму
      Имя формы ФПоставщик
      Режим Форма
      Имя фильтра
      Условие отбора =[КодП]=[Формы]![ФТовар]![Fslave]![КодП]
      Режим данных
      Режим окна Обычное
    Конец блока "Если"
  Конец вложенного макроса

```

Рисунок 8.5 – Конструктор макросов

### Задание

Создать не менее двух макросов для автоматизации работы базы данных из индивидуального задания.

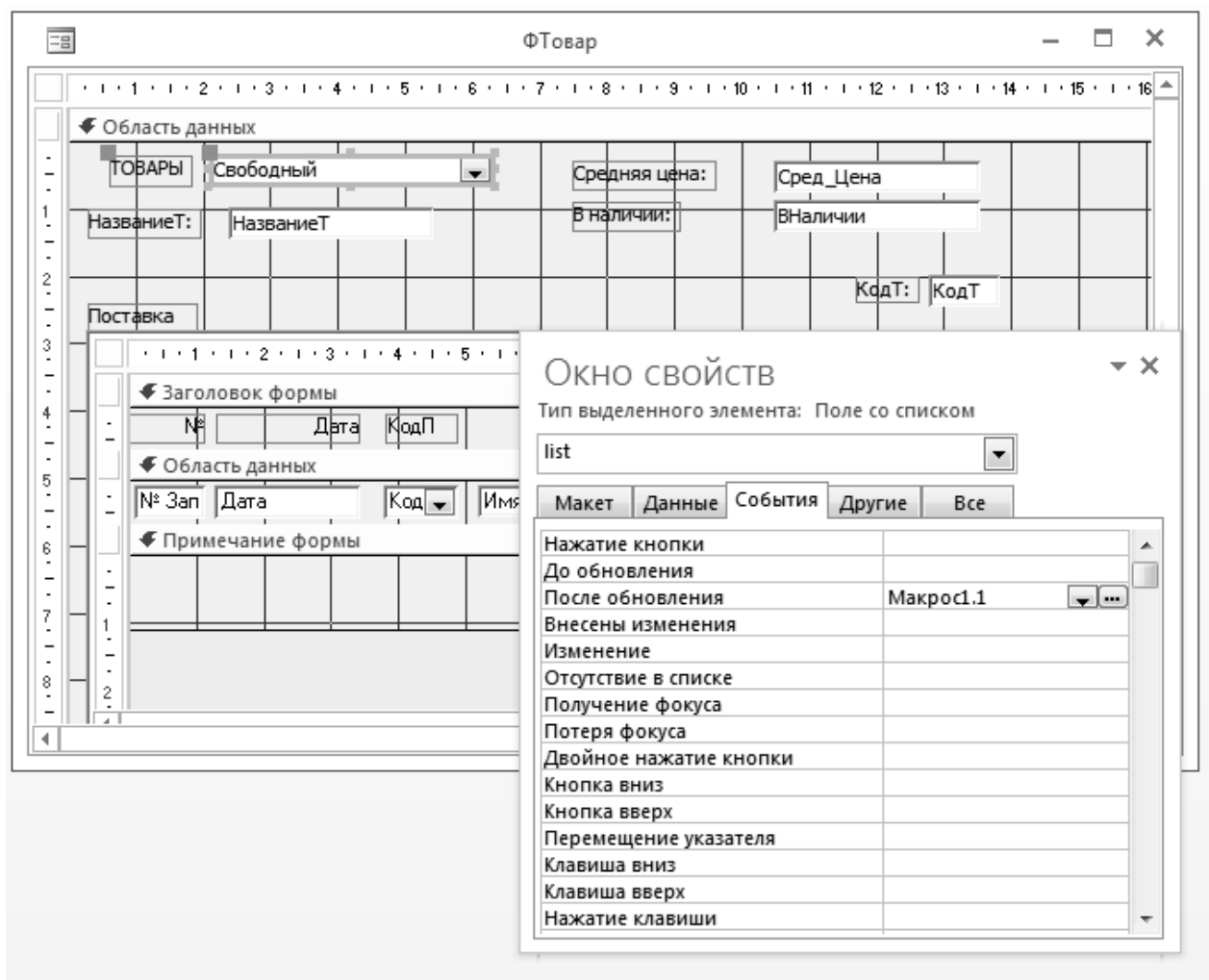


Рисунок 8.6 – Назначение макроса для элемента управления

## 9 Обработка событий на VisualBasic

**Цель работы:** научиться создавать процедуры в MS Access на Visual Basic.

Как и макросы, код VBA используют в Access для автоматизации и добавления функций.

Самый быстрый способ приступить к написанию программного кода VBA – создать макрос Access, а затем преобразовать его в код VBA. В этом случае будет создан модуль VBA, который выполняет те же операции, которые определены в макросе. Кроме того, откроется редактор Visual Basic, так что Вы сможете приступить к редактированию процедуры.

В Access можно автоматически преобразовать макросы в модули VBA или модули классов.

Преобразование макросов, прикрепленных к форме или отчету.

Этот процесс преобразует в VBA все макросы, на которые ссылается форма, отчет или любой из их элементов управления (либо которые внедрены

в форму, отчет или элемент управления), а затем добавляет код VBA в модуль класса формы или отчета. Модуль класса становится частью формы или отчета и сохраняется при их перемещении или копировании.

В области навигации щелкните форму или отчет правой кнопкой мыши и выберите пункт **Конструктор**.

На вкладке **Конструктор** в группе **Сервис** нажмите *Преобразовать макросы формы* или *Преобразовать макросы отчета*.

В диалоговом окне **Преобразование макросов** укажите, нужно ли добавить к создаваемым функциям код обработки ошибок.

Нажмите кнопку **Преобразовать**.

Чтобы просмотреть и изменить код VBA, нужно открыть форму в режиме конструктора. На вкладке **Событие** страницы свойств выбрать поле свойства, в котором отображается процедура обработки событий, и нажать кнопку **Построить**. Чтобы просмотреть свойства событий для определенного элемента управления, щелкните его, чтобы выбрать. Чтобы просмотреть свойства событий для всей формы или отчета, в верхней части окна свойств выберите в раскрывающемся списке пункт **Форма** или **Отчет**.

Откроется редактор VisualBasic с процедурой обработки событий в соответствующем модуле класса.

### Задание

- 1 Преобразуйте все макросы базы данных в код VBA.
- 2 Поставьте на форме кнопку и напишите на VBA процедуру для закрытия формы.

## 10 Создание отчетов в MS Access

**Цель работы:** научиться создавать и редактировать отчёты для вывода информации из базы данных на печать.

Отчет – это объект базы данных, который используется для вывода на экран, в печать или файл структурированной информации. Отчёты позволяют извлечь из таблиц или запросов базы данных необходимую информацию и представить ее в виде, удобном для восприятия. Отчёт содержит заголовок, область данных, верхний и нижний колонтитулы, примечание и разбит на страницы.

В Microsoft Access для создания отчетов можно использовать различные средства:

- мастер отчетов;
- конструктор отчетов;
- инструмент Report;
- пустой отчёт.

Создание отчётов во многом идентично созданию форм.





В отчётах имеется возможность выполнять многоуровневую группировку данных и для каждого уровня подводить промежуточные итоги.

### **Задание**

Создать не менее трёх отчётов с итогами для базы данных из индивидуального задания.

## **11 Публикация базы данных в Интернете**

**Цель работы:** научиться создавать объекты базы данных для публикации данных в Интернете.

Термин «публикация» означает предоставление данных, хранящихся в БД, широкому кругу пользователей Интернета.

Одной из основных задач публикации является преобразование объектов базы данных в Web-страницы.

При публикации БД в Интернете с помощью Microsoft Access 2000 можно создавать следующие три разновидности Web-страниц:

- 1) статические страницы HTML;
- 2) динамические (серверные) страницы HTML;
- 3) страницы доступа к данным.

**Статические страницы HTML** создаются из таблиц, запросов, форм и отчетов. Они не требуют подключения к источнику данных (т. е. к самой БД), содержат в себе всю необходимую информацию для отображения, при этом нет никакой необходимости в дополнительной настройке при публикации на Web-сервере в сети Интернет. Чтобы сделать статические файлы HTML доступными в Интернете, следует опубликовать их в папках Web или на Web-сервере.

Однако такие страницы содержат лишь те данные, которые существовали в базе данных на момент публикации, и, конечно, эти данные доступны только для просмотра в браузере, а не для редактирования.

Использование статических страниц целесообразно только в тех случаях, когда данные в БД изменяются очень редко. При изменении БД нужно вновь экспортировать статические страницы на Web-сервер для просмотра новых данных в обозревателе.

Динамические страницы HTML создаются из таблиц, запросов и форм. Они используются, когда информация в БД часто изменяется. При этом связь с БД организуется с помощью ODBC-интерфейса или ADO-интерфейса.

Чтобы постоянно предоставлять пользователям Web актуальную информацию, достаточно один раз создать страницу и экспортировать ее на Web-сервер. И каждый раз после изменений в источнике (в БД) новые данные будут попадать в нее автоматически. Иными словами, заполнять (динамически генерировать) страницу будет Web-сервер по запросу обозре-



вателя любого пользователя Интернета. При этом пользователи Интернета могут только просматривать эти страницы, однако редактировать данные, содержащиеся на динамической странице (как и на статической), нельзя.

**Страницы доступа к данным** представляют собой полноценный интерактивный интерфейс к данным в базе. Это специальный тип Web-страниц, предоставляющий пользователям Web интерфейс форм и отчетов Access для доступа к данным из базы данных Microsoft Access. Страницы доступа к данным представляют собой текстовые файлы с исходным кодом, соответствующим расширенному стандарту HTML – XML (Extended Markup Language).

Как уже отмечалось выше, формат динамических страниц HTML, генерируемых сервером, позволяет получать актуальные данные, но не позволяет редактировать их.

Страницы доступа к данным дают возможность делать и то, и другое. И еще с их помощью можно добавлять, удалять, сортировать и группировать записи. Каждый раз после изменений в базе данных новые данные автоматически попадут на страницу доступа к данным. А если отредактировать данные на странице доступа к данным, изменения автоматически попадут в присоединенную базу данных.

Преобразование объектов базы данных в форматы публикуемых страниц (HTML) производится с помощью одной и той же команды Файл/Экспорт (Export), которая может быть выбрана из окна БД Access (рисунок 11.1).

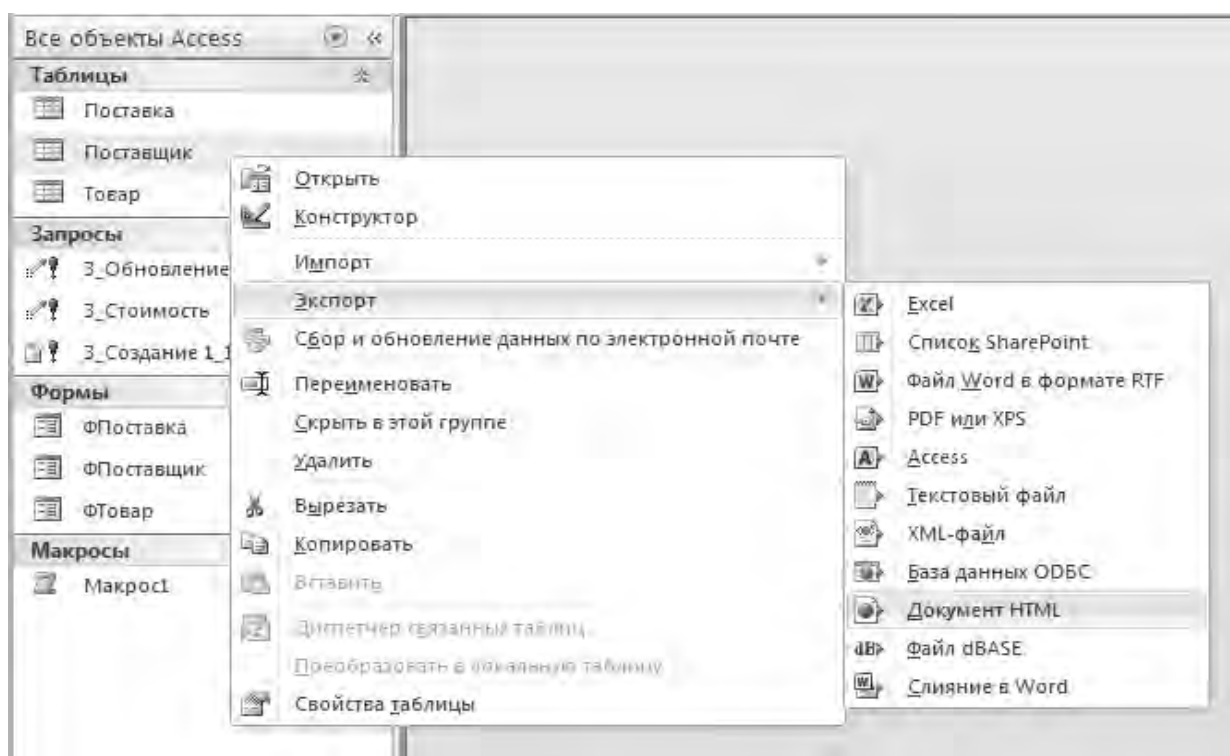


Рисунок 11.1 – Инструменты для преобразования объектов базы данных

Экспортировать любую созданную страницу или саму БД на Web-сервер проще всего при помощи Проводника Windows.

**Задание**

Выполнить экспорт нескольких объектов базы данных в формат, предназначенный для публикации в Интернете.

## **12 Импорт и экспорт данных, сжатие и восстановление. Связи с Office**

**Цель работы:** изучить возможности обмена информацией между СУБД MS Access и другими приложениями MS Office и возможности сжатия данных.

**Задание**

- 1 Создать новую базу данных, сохранив ее под именем. Импортировать в нее все таблицы из созданной ранее базы данных.
- 2 Импортировать электронную таблицу в базу данных.
- 3 Импортировать в новую базу данных текстовые файлы с разделителями и с фиксированной длиной записей.
- 4 Связать таблицу предыдущей базы данных с новой.
- 5 Удалить связанную таблицу из новой базы данных.
- 6 Экспортировать в новую базу данных формы.
- 7 Выполнить слияние любого запроса базы данных с документом MS Word.
- 8 Выполнить команды сжатия и восстановления базы данных.

**Контрольные вопросы**

- 1 Как импортировать таблицы одной базы данных MS Access в другую?
- 2 Как импортировать данные в Access из электронных таблиц?
- 3 Как импортировать данные в Access из текстового файла?
- 4 Как установить связь с таблицами из других баз данных MS Access?
- 5 Как использовать диспетчер связанных таблиц?
- 6 Как экспортировать данные в другую базу MS Access?
- 7 Как экспортировать данные из MS Access в электронную таблицу или в файлы других СУБД?
- 8 Как связать данные таблицы или набора записей запроса с документом Microsoft Word?
- 9 Как сжать (восстановить) базу данных MS Access?



## Список литературы

1 Центр справки и обучения Office [Электронный ресурс]. – Режим доступа: <https://support.office.com>. – Дата доступа: 25.03.2019.

2 **Кабанов, В. А.** Практикум Access] / В. А. Кабанов. – Москва: ИНФРА-М, 2015. – 55 с.

3 **Кузин, А. В.** Основы работы в Microsoft Office 2013: учебное пособие / А. В. Кузин, Е. В. Чумакова. – Москва: ФОРУМ, ИНФРА-М, 2015. – 160 с.

4 **Федотова, Е. Л.** Информационные технологии в профессиональной деятельности: учебное пособие / Е. Л. Федотова. – Москва: ИНФРА-М, 2015. – 368 с.

