Автономная некоммерческая организация высшего образования
«МОСКОВСКИЙ МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ»

Кафедра «Естественно-научных дисциплин»

А.Б.Фролов

СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ ACCESS 2016
Часть I. Таблицы и запросы.

Конспект лекций по дисциплине
«Информационные системы и базы данных»
для студентов очно-заочного и заочного отделений

МОСКВА – 2018

УДК 681.3.06

Рецензенты:

д.ф.- м.н., проф. Е.А.Лопаницын (Салют);

к.ф.- м.н., доцент Е.А.Коган (Московский политехнический университет).

Фролов А.Б. Система управления базами данных Access 2016. Часть I. Таблицы и запросы. Конспект лекций по дисциплине «Информационные системы и базы данных» для студентов очно-заочного и заочного отделений. М.: АНОВО Московский международный университет, 2018, 70 с.

Лекции ориентированы на изучение основ решения практических задач в программ MS Access 2016. Рассматриваются возможности работы программы по построению таблиц баз данных, модели данных, использованию фильтров и запросов, а также фильтрации данных в таблицах.

СОДЕРЖАНИЕ

1. Введение.

1.1. Понятие базы данных

База данных (БД) — это систематизированное хранилище информации, которое может относиться к различным сферам человеческой деятельности. Типичные примеры такой информации: телефонный справочник, сведения о студентах вуза, записи о заказах товаров и т.д.

До появления компьютеров вся эта информация хранилась в папках или картотеках. На каждом листе бумаги или на карточке был напечатан бланк формы, в котором были оставлены пустые места для заполнения данными. Весьма затруднителен был поиск нужной информации, когда для получения справки приходилось перебирать сотни личных карточек.

На начальных этапах использование компьютеров позволило устранить многие проблемы, характерные для некомпьютерных БД. С помощью компьютера можно быстро найти нужные сведения, причем критерий поиска может быть весьма сложным. Резко упростились подготовка и печатание различных отчетов и информационных справок. Однако работа с информацией, содержащейся в изолированных файлах, весьма затруднительна.

Для того чтобы возможности компьютера были использованы в полной мере, необходимо:

- пользоваться программным обеспечением, специально предназначенным для этих целей;
- соблюдать определенные правила организации информации.

Для координации доступа к БД из различных прикладных программ, обеспечения целостности данных, их модификации, создания, добавления и удаления необходим один «хозяин» хранилища данных. Таким «хозяином» является специальная

прикладная система программ, называемая ***системой управления базой данных (СУБД)***.

СУБД- это прикладное программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также осуществлять к ней контролируемый доступ.

По универсальности различают два класса СУБД общего назначения и специализированные.

СУБД общего назначения- это очень сложные программные комплексы, предназначенные для выполнения всей совокупности функций по созданию и эксплуатации баз данных. Основные разработки СУБД принадлежат фирмам Microsoft (Foxpro-DOS/WIN, Access) и Borland (Paradox-DOS/WIN).

Специализированные СУБДсоздаются в тех случаях, когда невозможно или нецелесообразно использовать СУБД общего назначения, например, информационно-поисковые системы (Консультант+; Кодекс; Гарант).

Правила организации информации называют ***модель данных***,под которойпонимают совокупность структур данных и операций над ними. Существует три типа моделей данных иерархическая, сетевая и реляционная.

1. ***Иерархическая модель***- ориентированный граф. Основная идея - каждая запись имеет свой путь от корневой записи. В иерархической модели данных используются только вертикальные линии связи подчинения между узлами данных на разных уровнях. Каждый узел может иметь любое количество связей с подчиненными узлами на нижнем уровне и только одну связь с родительским узлом на верхнем уровне.

2. ***Сетевая модель***- неориентированный граф. Основная идея - каждая запись может быть связана с другой записью. В сетевой

модели используются вертикальные и горизонтальные связи подчинения между узлами данных на разных уровнях. Каждый узел может быть связан с любым другим узлом на любом уровне.

3. ***Реляционная модель***– таблица, которая представляет совокупность записей, которые являются совокупностью именованных полей. Основная идея – представить произвольную структуру данных в виде двумерных таблиц. Понятие реляционный модели (***relation- отношение***) связано с разработками известного американского специалиста в области баз данных Кодда.

В настоящее время наибольшее распространение получили БД, основанные на реляционной модели.

Базы данных классифицируются и по другим признакам: по месту хранения, способу доступа к данным и архитектуре.

Классификация баз данных по месту хранения

По технологии обработки базы данных могут подразделяться на централизованные и распределенные.

Централизованная база данных хранится на одной ЭВМ, которая является компонентом сети.

Распределенная база данных состоит из нескольких, возможно пересекающихся или даже дублирующих друг друга, хранимых на разных ЭВМ вычислительной сети.

Классификация баз данных по способу доступа

По способу доступак данным базы данных разделяются на ***базы данных с локальным доступом*** и ***базы данных с удаленным (сетевым) доступом***.

Классификация баз данных по архитектуре

Локальная архитектура реализует централизованное хранениеи локальный доступк данным. Обычно это БД на персональном компьютере.

Архитектура файл-сервер реализует централизованное хранениеи сетевой доступк данным. Данные хранятся на центральной ЭВМ, которая называется ***Сервером файлов***, и совместно используются пользователями ***Рабочих станция***, для которых реализован сетевой доступ с помощью локальной сети. При большой интенсивности доступа к одним и тем же данным производительность информационной системы падает.

Архитектура клиент-сервер реализует хранение распределенной базы данных в памяти нескольких мощных серверов баз данных, и совместное использование данных ***Клиентами***, которые работают на рабочих станциях. Запросы клиентов к сетевой базе данных обрабатываются на ***Сервере приложений***. Извлеченные данные транспортируются по сети на рабочие станциидля формирования отчетов, форм и диаграмм с помощью сетевых приложений баз данных ***Генераторов отчетов***. Архитектура клиент-сервер характеризуется высокой производительностью обработки и передачи данных и надежностью функционирования.

На лекциях мы сосредоточимся на изучении наиболее распространенной сегодня ***СУБД Access из MS OFFICE***.

СУБД MS Access– прикладной пакет программ, который предназначен для создания и управления локальными и сетевыми реляционными базами данных, их обработки, сортировки, фильтрации, группировки, агрегирования, визуального представления с помощью экранных форм и текстовый отчетов.

1.2. Реляционные базы данных

В реляционных БД все данные структурированы (т.е. хранятся) в виде двумерных таблиц, между которыми установлены связи. Каждая таблица представляет собой (т.е. моделирует) информационный объект, называемый сущностью.

Сущность, а, следовательно, и таблица, моделируют множество предметов реального мира (предметной области), которые описываются одним и тем же набором характеристик и следуют одним и тем же правилам, и линиям поведения.

Пример сущности – стол. Это понятие описывает множество столов, существующих в реальной предметной области, причем каждый из столов может быть описан одним и тем же набором характеристик (инвентарный номер, высота, ширина, длина, назначение) и каждый стол следует одним и тем же правилам, и линиям поведения (ни один экземпляр из этого множества не может летать). А вот стол с инвентарным номером 91235 и конкретными другими характеристиками (высота=0.8 м, ширина=1.2м и т.д.) является ***экземпляром данной сущности***.

Таким образом, каждая таблица в БД моделирует сущность из описываемой предметной области.

Каждая строка таблицы (называемая ***записью***) описывает экземпляр этой сущности.

Каждый столбец таблицы (называемый ***полем***) моделирует одну характеристику сущности. Каждый столбец таблицы предназначен для хранения данных определенного типа: текстовых, числовых, дат, времени, денежных и т.д.

Несмотря на то, что сущностей в реальном мире существует бесконечное множество, большинство сущностей можно отнести к одной из следующих категорий:

- ***Реальные сущности***– это объекты реального мира, существующие физически. Например, стол, самолет, человек, насос и т.д.
- ***Ролевые сущности (роли)***– это термины, определяющие назначение отдельного человека, организации, оборудования и

т.д. Например, мастер цеха, студент, бухгалтерия и т.п.

— ***Инциденты***– это события, случающиеся в реальной жизни. Например, заказ товара, оплата счета, отгрузка товара, получение путевки и т.д.
— ***Взаимодействия***– это сущности, возникающие из отношений (связей) сущностей друг с другом.
— ***Спецификации***– это сущности, используемые для представления правил, стандартов или критериев качества.

Поля в таблицах могут относиться к одному из следующих видов:

— ***Идентифицирующие (или первичные ключи)*** – это поле или набор полей, однозначно идентифицирующий запись таблицы (экземпляр сущности). Значение первичного ключа в таблице БД должно быть уникальным, т.е. в таблице не должно существовать двух или более записей с одинаковым значением первичного ключа. Каждая таблица **обязательно** должна иметь первичный ключ. Первичные ключи обеспечивают установление связи между таблицами.
— ***Описательные***– это поля, содержащие значения, присущие отдельным экземплярам сущности.
— ***Вспомогательные (или внешние ключи)*** – это поля, добавляемые в таблицу для организации её связи с другой таблицей. Эти поля добавляют к таблице только на этапе организации связей между таблицами.

Для любого поля таблицы (или группы полей) может быть назначен ***индекс,*** облегчающий поиск нужных строк (записей) таблицы. При индексировании полей таблицы СУБД создает дополнительную служебную таблицу, которая содержит упорядоченные значения индексированных полей и физические адреса записей. Поэтому поиск значений по индексированным полям

таблицы осуществляется значительно быстрее, чем по неиндексированным полям. Однако избыточное количество индексов в таблице существенно замедляет процессы записи и модификации такой таблицы, т.к. СУБД вынуждена модифицировать не только саму таблицу, но и соответствующие ей индексные таблицы. Поэтому злоупотреблять индексами при создании таблиц базы данных не следует.

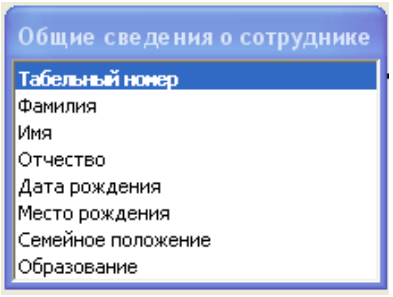
1.3. Реляционные отношения между таблицами базы данных

Записи различных таблиц могут быть связанными друг с другом логически, т.е. одной записи из первой таблицы могут соответствовать одна или несколько записей из другой таблицы.

На этапе построения реляционной модели данных какой-либо предметной области в целях более компактного представления вместо реальных таблиц будем пользоваться их макетами. На рис.1 представлена таблица с данными (а) и макет (б). Название таблицы указывается в заголовочной области макета, а первичный ключ в макете таблицы выделяется жирным шрифтом.

Фамилия	Имя	Отчество	Дата рождения	Место рождения	Семейное положение	Образование
Иванов	Иван	Иванович	20.07.1985	г.Москва	Холост	Среднее
Петрова	Мария	ВасПетров	31.07.1975	г.Краснодар	Замужем	Высшее

а)



б)

Рис.1. Пример таблицы с данными и её макета.

Всё многообразие логических отношений между таблицами в

реляционных базах данных моделируется тремя видами ***связей***:

- ***Один-к-одному (1:1).***
- ***Один-ко-многим (1:∞).***
- ***Многие-ко-многим (∞:∞).***

Одна из связанных таблиц (на стороне «один») называется ***главной***таблицей, а вторая называется ***подчиненной*** таблицей. Для реализации связей между таблицами используются их поля: *первичный ключ* в главной таблице и *внешний ключ* в подчиненной таблице.

Связь Один-к-одному (1:1)означает, что одной записи из главной таблицы соответствует одна запись из подчиненной таблицы. Эта связьмежду таблицами на практике встречается чрезвычайно редко, т.к. этой связи можно избежать простым слиянием двух таблиц. Однако, если таблица будет содержать слишком много столбцов (например, свыше 15-20 столбцов), то в этом случае целесообразно создать две таблицы, связав их отношением 1:1. Для реализации связи 1:1 в подчиненной таблице в качестве внешнего ключа необходимо использовать первичный ключ. Например, в информационных системах кадрового состава организаций для учета личных сведений о сотруднике используется около 100 показателей. Поэтому личные сведения о сотруднике целесообразно разнести по нескольким таблицам, связав их затем отношением 1:1. Для связи можно использовать поле «Табельный номер», которое однозначно идентифицирует каждого сотрудника организации. Пример такой связи приведен на рис.2.

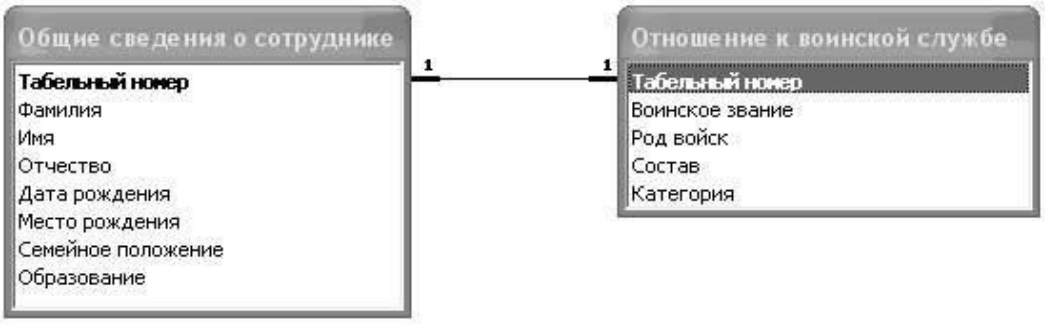


Рис.2. Пример связи 1:1 между таблицами

Связь Один-ко-многим (1:∞)означает, что одной записи из главной таблицы соответствует много записей из подчиненной таблицы. На практике это наиболее распространенный вид связи. Для реализации этой связи необходимо первичный ключ из главной таблицы добавить в качестве внешнего ключа к подчиненной таблице. При этом добавленное поле в качестве внешнего ключа к подчиненной таблице обязательно надо индексировать (совпадения допускаются) при создании таблицы в выбранной СУБД. Кроме того, необходимо отслеживать, чтобы тип поля первичного ключа главной таблицы совпадал с типом поля внешнего ключа подчиненной таблицы.Например, таблицы «Континенты» и «Страны» связаны отношением 1:∞, т.к. одной записи из таблицы «Континенты» соответствует несколько записей из таблицы «Страны» (рис.3).

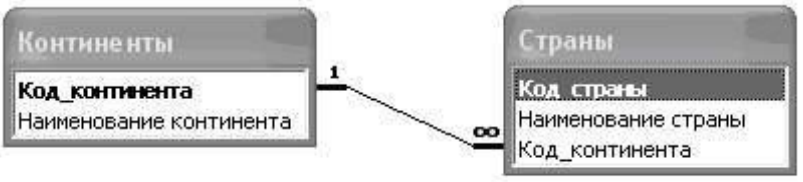


Рис.3. Пример связи 1:∞ между таблицами

Для реализации связи 1:∞ на рис.3 к таблице «Страны» (подчиненная таблица) добавлено поле «Код_континента» в качестве внешнего ключа, являющегося первичным ключём в таблице «Континенты» (главная таблица).

Связь Многие-ко-многим (∞:∞)означает, что одной записи из первой таблицы соответствует много записей из второй таблицы, и в то же время одной записи из второй таблицы соответствует много записей из первой. Такой вариант логической связи между сущностями (таблицами) на практике встречается довольно часто. Однако, ни одна из современных реляционных СУБД не имеет средств реализации такого рода связи.Для реализации связи ∞:∞ её надо разбить на две связи 1:∞, добавив в базу данных дополнительную таблицу. Эта дополнительная таблица называется ***ассоциативной***и относится к категории сущностей – «Взаимодействия». Несмотря на то, что ассоциативная таблица добавляется в базу данных исключительно для реализации связи ∞:∞, она в базе данных обладает равными правами с другими таблицами, т.е. должна обязательно иметь первичный ключ и может участвовать в свою очередь в других связях с таблицами базы данных, т.е. может иметь и внешние ключи.

Например, к рассмотренным выше таблицам «Континенты» и «Страны» добавим таблицу «Полезные ископаемые». Как установлено выше, таблицы «Континенты» и «Страны» связаны отношением 1:∞. Определим связь между добавленной таблицей «Полезные ископаемые» и таблицей «Страны»: каждая страна может иметь множество полезных ископаемых в своих недрах, в то же время одно и то же полезное ископаемое может добываться в различных странах. Следовательно, таблицы «Страны» и «Полезные ископаемые» связаны отношением ∞:∞. Для реализации этой связи в базу данных следует добавить дополнительную (ассоциативную) таблицу. Назовем эту таблицу «Полезные ископаемые в странах». Добавленная таблица связана отношением 1:∞ с таблицей «Страны» и отношением 1:∞ с таблицей «Полезные ископаемые». Из таблицы

«Страны» добавим в ассоциативную таблицу в качестве внешнего ключа поле «Код страны», а из таблицы «Полезные ископаемые» добавим поле «Код полезного ископаемого» также в качестве внешнего ключа. Кроме этих полей, включим в ассоциативную таблицу «Полезные ископаемые в странах» еще описательное поле «Разведанные запасы».

Поскольку в ассоциативной таблице поля «Код страны» и «Код полезного ископаемого» будут повторяться, но их сочетание будет уникальным, то в качестве первичного ключа в ассоциативной таблице выберем эти два поля, т.е. первичный ключ в ассоциативной таблице будет составным. Полученная реляционная модель данных для этого примера представлена на рис.4, где связь ∞:∞ между таблицами «Страны» и «Полезные ископаемые») разбита на две связи 1:∞ через ассоциативную таблицу «Полезные ископаемые в странах».

Рис.4. Пример связи ∞:∞ между таблицами «Страны» и «Полезные ископаемые»

1.4. Нормализация базы данных

Итак, после определения таблиц, полей, индексов и связей между таблицами следует посмотреть на проектируемую базу данных в целом и проанализировать , используя правила нормализации, с целью устранения логических ошибок. Главная цель нормализации

базы данных - устранение избыточности и дублирования информации.

Теория нормализации реляционных баз данных была разработана в конце 70-х годов 20 века. Согласно ей, выделяются шесть нормальных форм, пять из которых так и называются: первая, вторая, третья, четвертая, пятая нормальные формы, а также нормальная форма Бойса-Кодда.

Первая нормальная форма:

- запрещает повторяющиеся столбцы (содержащие одинаковую по смыслу информацию)
- запрещает множественные столбцы (содержащие значения типа списка и т.п.)
- требует определить первичный ключ для таблицы, то есть тот столбец или комбинацию столбцов, которые однозначно определяют каждую строку

Вторая нормальная форма

Требует, чтобы не ключевые столбцы таблиц зависели от первичного ключа в целом, но не от его части.

Третья нормальная форма

Требует, чтобы не ключевые столбцы в ней не зависели от других не ключевых столбцов, а зависели только от первичного ключа. Самая распространенная ситуация в данном контексте - это расчетные столбцы, значения которых можно получить путем каких-либо манипуляций с другими столбцами таблицы.

Нормальная форма Бойса-Кодда

Требует, чтобы в таблице был только один потенциальный первичный ключ. Чаще всего у таблиц, находящихся в третьей нормальной форме, так и бывает, но не всегда. Если обнаружился второй столбец (комбинация столбцов), позволяющий однозначно

идентифицировать строку, то для приведения к нормальной форме Бойса-Кодда такие данные надо вынести в отдельную таблицу.

Четвертая нормальная форма

Для приведения таблицы, к четвертой нормальной форме необходимо устранить имеющиеся в ней многозначные зависимости. То есть обеспечить, чтобы вставка / удаление любой строки таблицы не требовала бы вставки / удаления / модификации других строк этой же таблицы.

Пятая нормальная форма

Таблицу, находящуюся в четвертой нормальной форме и, казалось бы, уже нормализованную до предела, в некоторых случаях еще можно бывает разбить на три или более (но не на две!) таблиц. Получившиеся в результате такой, как правило, весьма искусственной, декомпозиции таблицы и называют находящимися в пятой нормальной форме. Формальное определение пятой нормальной формы таково: это форма, в которой устранены зависимости соединения.

Главное, чего мы добьемся, проведя нормализацию базы данных – это сокращение вероятности появления противоречивых данных, облегчение администрирования базы и обновления информации в ней, сокращение объема дискового пространства. Однако зачастую, чтобы извлечь информацию из нормализованной базы данных, приходится конструировать очень сложные запросы, которые работают довольно медленно - из-за большого количества соединений таблиц. Поэтому, чтобы увеличить скорость выборки данных и упростить программирование запросов, нередко приходится идти на выборочную денормализацию базы.

База данных считается нормализованной, если ее таблицы (по крайней мере, большинство таблиц) представлены как минимум в

третьей нормальной форме.

2. Построение реляционной модели данных

Модель данных является основой проекта базы данных определенной предметной области. Построение реляционной модели осуществляется в следующей последовательности:

Шаг 1.Определение всех сущностей предметной области, которые необходимо отразить в модели для поставленной задачи. Этот этап завершается составлением списка сущностей, которые необходимы в модели для решения поставленной задачи. Каждая сущность в базе данных будет представлена отдельной таблицей.

Шаг 2.Определение всех атрибутов каждой сущности. Для каждого атрибута каждой сущности на этом этапе необходимо указать:

– Тип данных атрибута (числовой целый, числовой действительный, текстовый с указанием количества символов и т.д.).

– Обязательный или необязательный, т.е. может ли атрибут содержать пустые значения.

– Формат ввода (например, для дат формат может иметь вид: дд.мм.гггг и т.д.).

На этом же этапе в каждой сущности отмечаются идентифицирующие атрибуты – первичные ключи. Результаты данного этапа можно представить в виде таблицы с описанием атрибутов каждой сущности.

Шаг 3.Определение связей между сущностями. На этом этапе к сущностям, участвующим в связи добавляются вспомогательные атрибуты – внешние ключи. При необходимости в модель добавляются ассоциативные сущности со своими атрибутами. На данном этапе разработчик периодически возвращается к шагу 1 (добавление ассоциативных сущностей) или к шагу 2 (добавление и описание внешних ключей и атрибутов ассоциативных сущностей),

т.е. процесс построения модели имеет итеративный характер. Шаг завершается схематическим построением реляционной модели.

2.1. Пример построения реляционной модели данных

Рассмотрим следующий учебный пример построения модели данных: построить реляционную модель данных для диспетчера отдела продаж автомобильных шин

Шаг 1.Прежде все определим необходимые сущности для очерченной в условии примера предметной области:

- Категория автошин– эта сущность необходима для облегчения поиска шин в системе и анализа состава заказов. Экземплярами этой сущности будут: зимние шины, летние шины, всесезонные шины и т.п.
- Шины – эта сущность не требует пояснений.
- Покупатели – эта сущность содержит список всех покупателей автомобильного салона и сведения о них, которые необходимы для выполнения заказа.
- Сотрудники – эта сущность содержит сведения о сотрудниках автомобильного салона, принимающих заказ. Эта сущность необходима для персональной ответственности сотрудников.
- Заказы – эта сущность содержит информацию о заказе: заказчик, дата принятия заказа, дата исполнения заказа, сведения о сотруднике, принявшем заказ.

Таким образом, составленный список сущностей для рассматриваемого примера определяет таблицы в проектируемой базе данных.

Шаг 2.Определяем атрибуты каждой сущности и их характеристики.

Таблица 1.

Атрибуты сущности «Категория автошин» и их характеристики.

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код категории	Целое число	Без дробной части	Да	Первичный ключ, индексируется (совпадения не допуск.)
Сезонность	Текст, 50 знак.		Да	

Таблица 2.

Атрибуты сущности «Шины».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код шины	Целое число	Без дробной части	Да	Первичный ключ, индексируется (совпадения не допуск.)
Производитель	Текст, 30 знаков		Да	
Радиус	Текст, 5 знаков		Да	
Ширина	Целое число	Без дробной части	Да	
Высота	Целое число	Без	Да	

профиля		дробной части		
Цена	Денежный		Да	

Таблица 3.

Атрибуты сущности «Покупатели».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код покупателя	Целое число	Без дробной части	Да	Первичный ключ, индексируется (совпадения не допуск.)
Фамилия	Текст, 30 знак.		Да	
Имя	Текст, 20 знак.		Да	
Отчество	Текст, 20 знак.		Да	
Телефон	Текст, 15 знак.		Нет	

Таблица 4.

Атрибуты сущности «Сотрудники».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Табельный номер	Целое число	Без дробной	Да	Первичный ключ, индексируется

		части		(совпадения не допуск.)
Фамилия	Текст, 30 знак.		Да	
Имя	Текст, 20 знак.		Да	
Отчество	Текст, 20 знак.		Да	
Должность	Текст, 20 знак.		Да	

Таблица 5.

Атрибуты сущности «Заказы».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код заказа	Целое число	Без дробной части	Да	Первичный ключ, индексируется (совпадения не допускаются)
Дата приема	Дата	дд.мм.гг	Да	
Дата исполнения	Дата	дд.мм.гг	Да	

Шаг 3.На данном этапе построения модели необходимо определить логические связи между введенными в модель сущностями.

И так. Между сущностями «Категория автошин» и «Шины»: каждому экземпляру сущности «Категория автошин» соответствует множество экземпляров сущности «Шины», в то же время каждый экземпляр сущности «Шины» входит только в один экземпляр сущности «Категория автошин». Следовательно, между этими сущностями существует связь 1:∞, причем сущность «Категория

автошин» находится на стороне «1» связи, а сущность «Шины» - на стороне «∞». Для реализации этой связи к сущности «Шины» надо добавить в качестве внешнего ключа атрибут «Код категории», являющийся первичным ключом сущности «Категория автошин».

Дополнительные атрибуты сущности «Шины».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код категории	Целое число	Без дробной части	Да	Внешний ключ, индексируется (совпадения допускаются)

Далее переходим к рассмотрению связи между сущностями «Заказы» и «Покупатели». Поскольку каждый заказчик может сделать множество заказов (в разные дни и разное время), а каждый заказ принадлежит только одному заказчику, то между этими сущностями имеется связь 1:∞, причем сущность «Покупатели» находится на стороне «1» связи, а сущность «Заказы» - на стороне «∞». Для реализации этой связи к сущности «Заказы» добавляем в качестве внешнего ключа атрибут «Код покупателя», являющийся первичным ключом сущности «Покупатели».

Дополнительные атрибуты сущности «Заказы».

Код покупателя	Целое число	Без дробной части	Да	Внешний ключ, индексируется (совпадения допускаются)

Аналогичные рассуждения определяют связь между сущностями «Заказы» и «Сотрудники» (эта связь также будет иметь вид 1:∞, причем сущность «Сотрудники» находится на стороне «1» связи, а сущность «Заказы» - на стороне «∞»).

Дополнительные атрибуты сущности «Заказы».

Код сотрудника	Целое число	Без дробной части	Да	Внешний ключ, индексируется (совпадения допускаются)

Теперь о связи между сущностями «Заказы» и «Шины». Один заказ может включать в себя множество шин, в то же время одна и та же шина может входить в разные заказы. Следовательно, между сущностями «Заказы» и «Шины» существует связь ∞:∞. Чтобы реализовать такую связь в модель необходимо добавить ассоциативную сущность, с помощью которой связь ∞:∞ преобразуется в две связи 1:∞. В рассматриваемом примере такая ассоциативная сущность должна быть связана отношением 1:∞ с сущностью «Шины» (ассоциативная сущность на стороне «∞», а сущность «Шины» - на стороне «1») и отношением 1:∞ с сущностью «Заказы» (ассоциативная сущность на стороне «∞», а сущность «Заказы» - на стороне «1»). Каждый экземпляр ассоциативной сущности характеризует какие шины входят в какой заказ и в каком количестве. Назовем вводимую в модель ассоциативную сущность «Состав заказа».

Для реализации двух связей 1:∞ перенесем в сущность «Состав заказа» атрибут «Код заказа», являющийся первичным ключом сущности «Заказы», и атрибут «Код шины», являющийся первичным ключом сущности «Шины». Оба эти атрибута будут в ассоциативной сущности внешними ключами. Каждый из атрибутов («Код заказа» и «Код шины») в таблице «Состав заказа» будут иметь повторяющиеся значения, но сочетание их значений будет для таблицы «Состав заказа» уникальным. Поэтому в таблице «Состав заказа» первичный ключ будет составным, т.е. состоять из двух полей: «Код заказа» и «Код шины».

Таблица 6.

Атрибуты сущности «Состав заказов».

Наименование атрибута	**Тип данных**	**Формат**	**Обязательный**	**Особые отметки**
Код заказа	Целое число	Без дробной части	Да	Первичный ключ, индексируется (сов-падения допускаются)
Код шины	Целое число	Без дробной части	Да	Первичный ключ, индексируется (совпадения допускаются)
Количество	Целое число	Без дробной части	Да	
Отпускная цена	Денежный		Да	

Последний атрибут добавлен в таблицу в связи с тем, что отпускные цены на шины могут изменяться достаточно быстро. Поэтому целесообразно фиксировать в заказе цену на момент оформления заказа.

2.2. Построение модели данных

Таким образом, реляционная модель рассматриваемого примера имеет вид, представленный на рисунке 1.

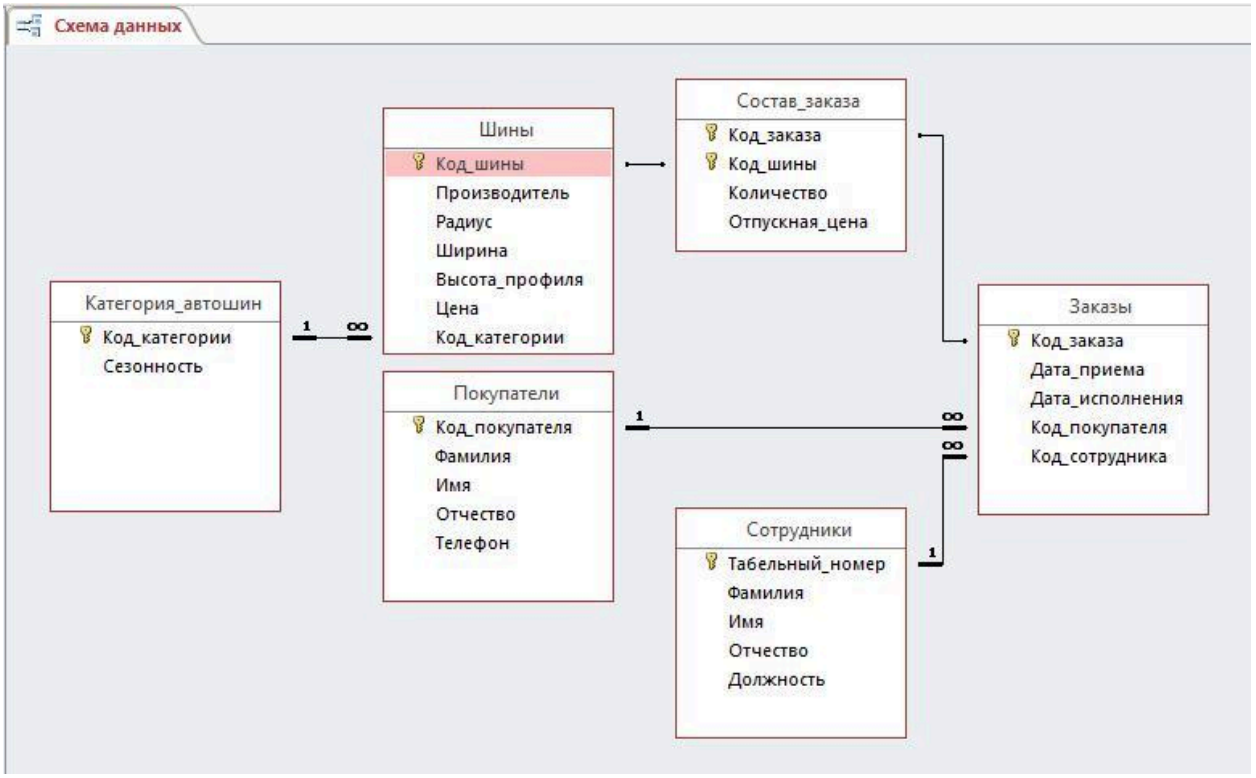


Рис.1. Реляционная модель данных рассматриваемого примера.

3. Физическая реализация базы данных

Запуск Access возможно осуществлять различными способами: из панели MicrosoftOffice, двойным щелчком мыши на ярлыке Access, из главного меню (кнопка ***Пуск ->Программы->MicrosoftOffice ->MicrosoftOfficeAccess 2007***). Общий вид окна программы приведен на рис.1.

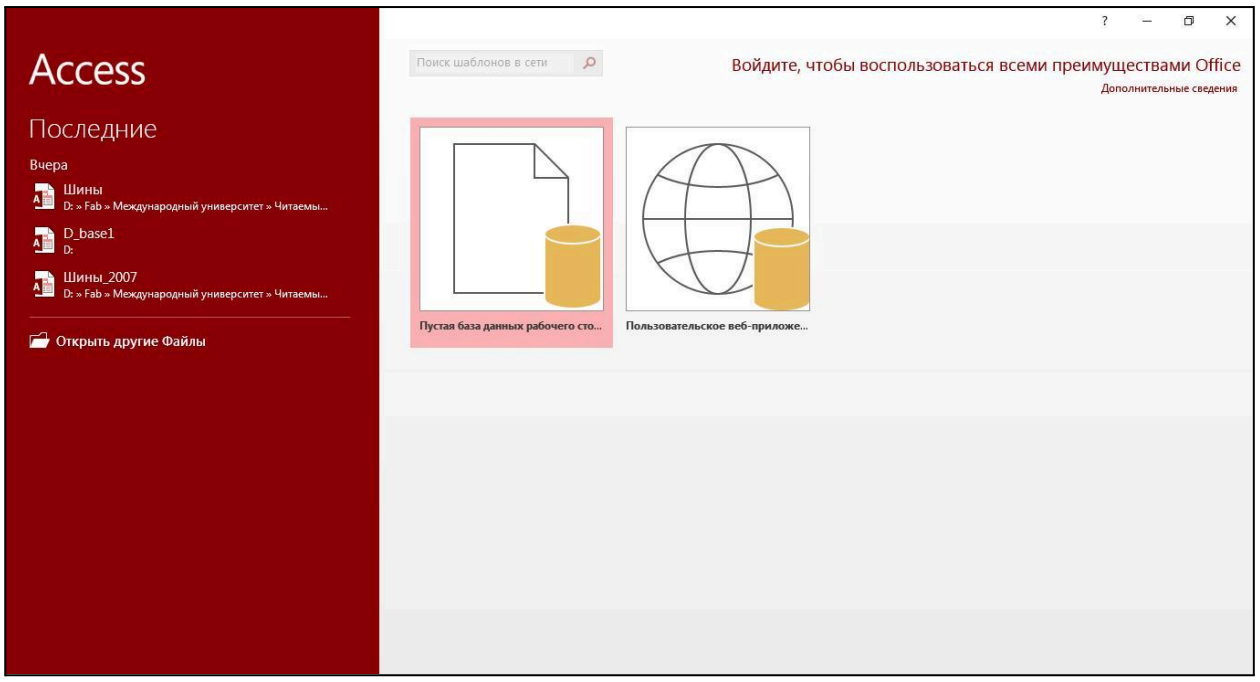


Рис.1. Окно программы Access

После загрузки Access в оперативную память необходимо или создать новый файл .accdb или загрузить в Access уже имеющийся и хранящийся на диске файл.

Для создания нового файла необходимо нажать на значке – Новая База Дан***ных,*** ав поле ***Имя файла*** вписать название создаваемой базы. По умолчанию база данных создается в папке ***Мои документы***, для изменения пути необходимо кликнуть на иконке  слева от поля ***Имя файла***. После чего нажимается кнопка ***Создать***.

3.1. Создание таблиц базы данных

После создания файла новой базы данных (рис .2)

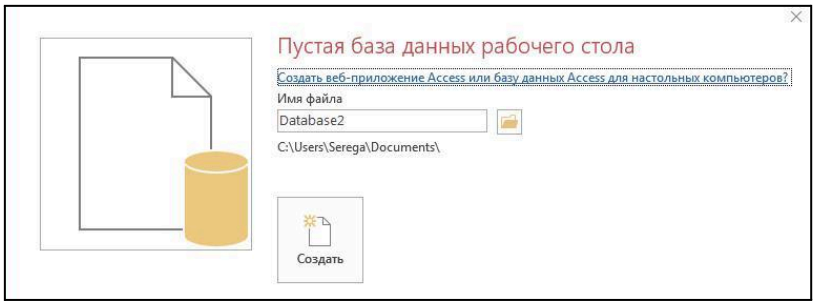


Рис.2. Создания файла новой базы данных

открывается окно ***База данных*** (Рис. 3.) и пользователь может перейти к созданию ее таблиц (сущностей).

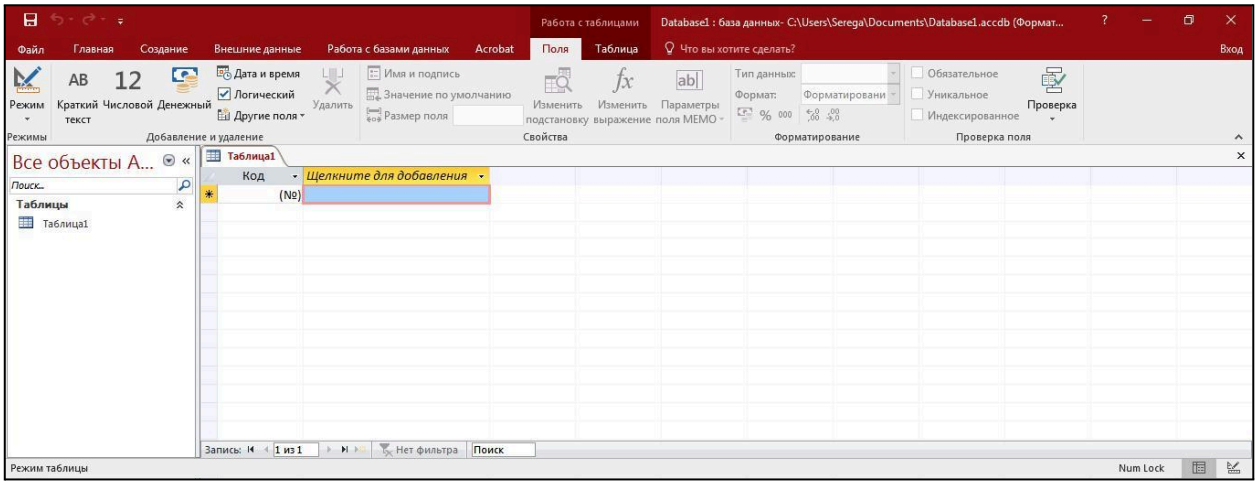


Рис.3. Окно базы данных

Если задача позволяет, то таблицы могут быть построены на базе шаблонов, зарезервированных в программе ***Access***. Для этого достаточно перейти на вкладку ***Создание*** и в меню (рис.3) выбрать пункт ***Части приложения***(рис.4).

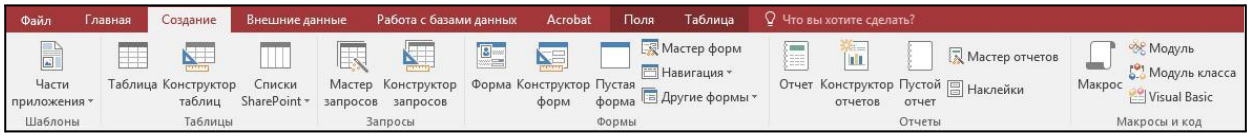


Рис. 3. Меню вкладки ***Создание***

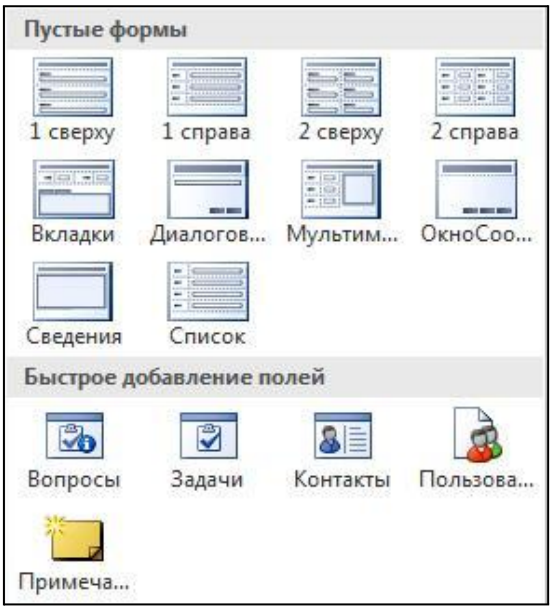


Рис.4. Предлагаемые шаблоны элементов БД

В результате получаем таблицу с готовыми именами столбцов (рис.5).

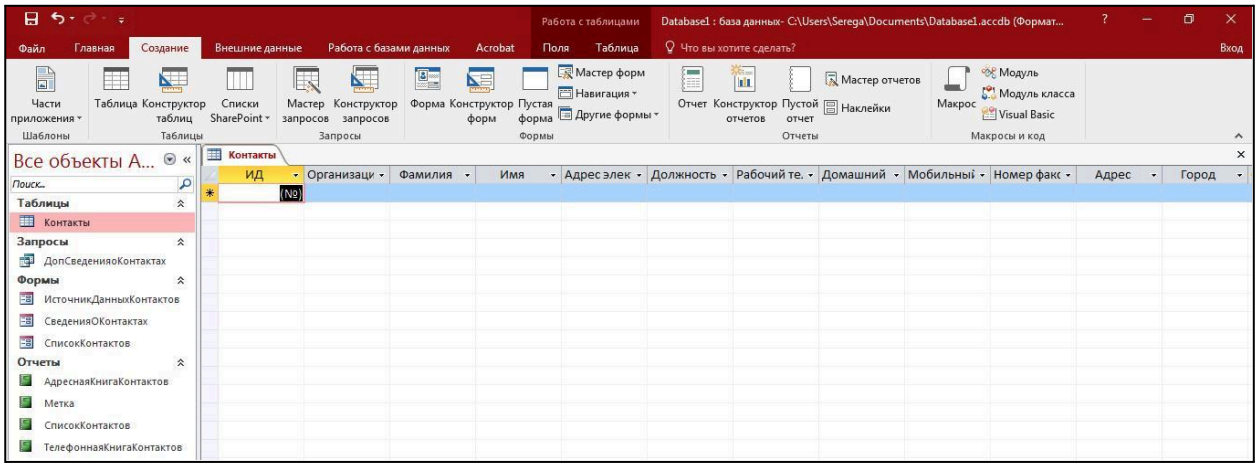


Рис.5. Таблица, построенная по шаблону ***Контакты***.

При необходимости можно добавить в таблицу свои столбцы (двойной клик левой клавишей мыши на последнем столбце ***Добавить поле*** и ввод имени нового столбца), или удалить лишние столбцы (клик правой клавишей мыши на лишнем столбце и выбор пункта контекстного меню – ***Удалить столбец***).

3.2. Создание таблицы с помощью *Конструктора*

Для перехода в режим ***Конструктора таблиц*** необходимо кликнуть на закладку ***Таблица 1*** и выполнить переход, как показано на рис. 6.

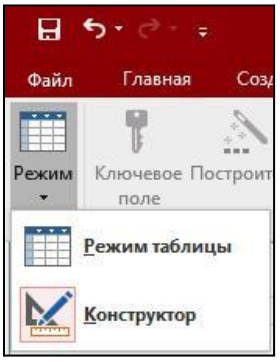


Рис.6. Смена режима.

В режиме ***Конструктора*** сначала предлагается зарезервировать имя будущей таблицы (рис.7),

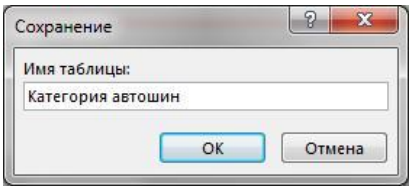


Рис. 7. Ввод имени таблицы.

а, затем, открывается окно создания её макета (рис. 8).

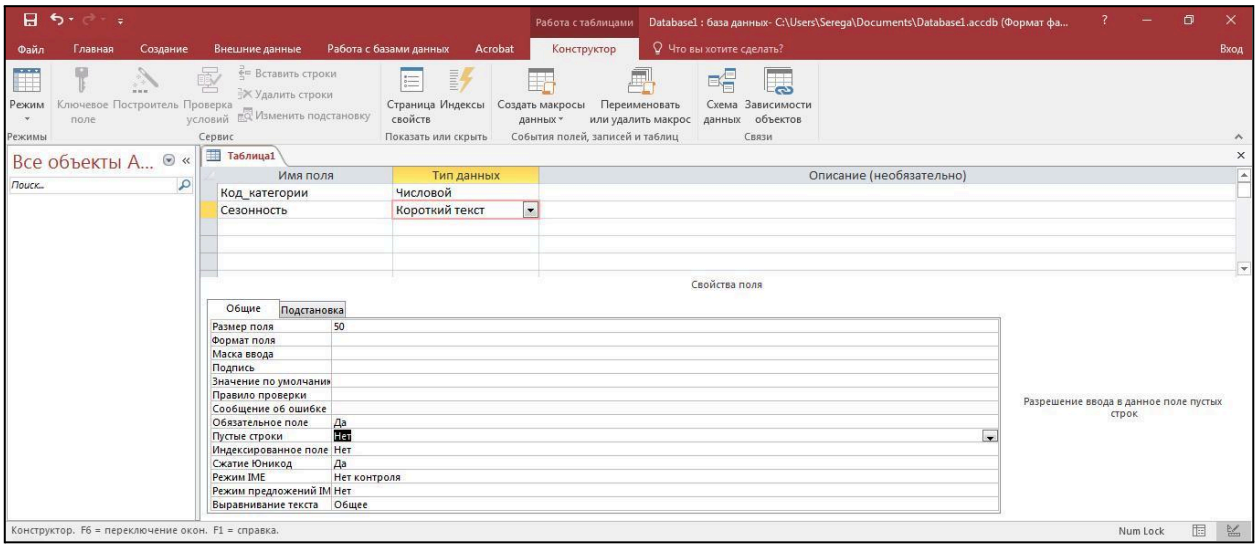


Рис. 8. Окно создания таблицы в режиме ***Конструктора***.

Разработчик должен последовательно вводить в строки окна данные о полях создаваемой таблицы или добавлять поля в уже созданную таблицу.

Имя поля вводится в графу ***Имя поля***. Рекомендуется формировать имя таким образом, чтобы оно представляло собой одно слово. Для этого вместо пробела между словами используют символ ***Подчерк***.

В столбце ***Тип данных*** этой же строки макета таблицы из выпадающего списка необходимо выбрать тип создаваемого поля (рис. 9).

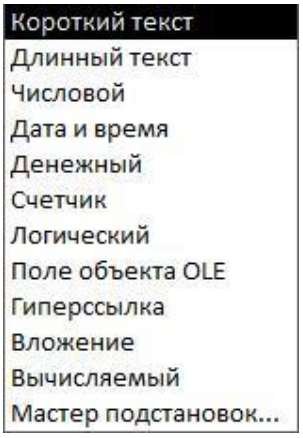


Рис. 9. Допустимые типы данных

Тип данных поля таблицы базы данных определяет, какие данные могут храниться в поле, а также какие операции могут выполняться над ними при их обработке. Эта характеристика является обязательной для всех полей таблицы. В MicrosoftAccess используются следующие типы полей:

Короткий текст.Представляет собой текст или комбинацию текста и чисел. Сохраняет до 255 знаков. Свойство ***Размер поля***определяет максимальное количество знаков, которые можно ввести. По умолчанию задается значение 50.

Длинный текст.Длинный текст или числа, например, примечания или описания. Сохраняет до 65 536 знаков.

Числовой.Данные, используемые для математических вычислений.

Конкретный тип числового поля определяется значением свойства ***Размер поля***. Допустимыми являются следующие значения этого свойства.

Значение	**Описание**	**Дробная часть**	**Размер**
Байт	Числа от 0 до 255	Отсутствует	1 байт
Целое	Числа от -32768 до 32767	Отсутствует	2 байта
Длинное целое	Числа от -2147483648 до 2147483647	Отсутствует	4 байта
Действительное	Числа от -10^{28-1} до 10^{28-1}	28	12 байт
Одинарное с плавающей точкой	Числа от $-3{,}402823_{10}38$ до $3{,}402823_{10}38$	7	4 байта
Двойное с плавающей точкой	Числа от $-1{,}79769313486231_{10}308$ до $1{,}79769313486231_{10}308$	15	8 байт

Дата/время.Значения дат и времени. Сохраняет 8 байтов.

Денежный.Используется для денежных значений и для предотвращения округления во время вычислений. Сохраняет 8 байтов.

Счетчик. Автоматическая вставка уникальных последовательных (увеличивающихся на 1) или случайных чисел при добавлении записи. Сохраняет 4 байта.

Логический.Данные, принимающие только одно из двух возможных значений, таких как «Да/Нет», «Истина/Ложь», «Вкл/Выкл». Пустые значения не допускаются. Сохраняет 1 бит.

Поле объекта OLE.OLE – это технология связи программ, разработанная фирмой Microsoft и позволяющая приложениям совместно использовать данные. Сохраняет до 1 Гигабайта.

Гиперссылка.Это цветной подчеркнутый текст или графический объект, по щелчку на котором выполняется переход к файлу, фрагменту файла или странице HTML в Интернете.

Вложение. Изображения, листы, документы, диаграммы и файлы

других поддерживаемых типов, прикрепленные к записям в БД.

Вычисляемый. Результаты вычисления. В вычислении должны использоваться поля той же таблицы. Для создания вычислений используется ***Построитель выражений.***

Мастер подстановок. Специальная программа, позволяющая подставлять в поле значения из поля другой таблицы или из фиксированного списка.

После выбора типа данных поля в нижней части окна необходимо обратиться к таблице свойств поля (рис. 10).

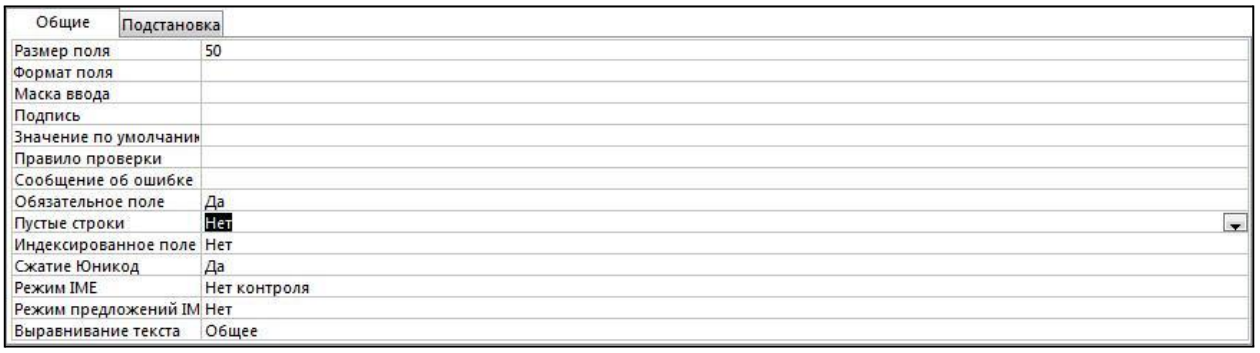


Рис. 10. Свойства поля.

- При создании поля рекомендуется обязательно заполнять следующие свойства: ***Размер поля***, ***Подпись*** - содержит название столбца таблицы в режиме просмотра, ***Обязательное поле***, ***Индексированное поле–*** обязательно индексируются первичные и внешние ключи таблицы, ***Пустые строки***–указывает, может ли поле содержать данные, состоящие из одних пробелов.

Свойство ***Формат поля*** следует задавать из выпадающего списка для типа данных ***Дата/Время***, а для текстовых полей это свойство не заполняется. Для числовых полей свойство ***Формат поля*** можно не заполнять, оставив значение ***Авто*** в свойстве ***Число десятичных знаков***.

Значение свойства ***Значение по умолчанию*** для большинства

полей лучше очистить, а для типа данных ***Дата/Время*** в это свойство обычно вводят функции автоматического заполнения текущей даты (***Date()*** или ***Time()***).

Свойство ***Условие на значение*** может содержать выражение, которому должны удовлетворять значения, вводимые пользователем в поле при заполнении таблицы.

Свойство ***Сообщение об ошибке*** содержит текст, который появляется в окне сообщения, когда пользователь вводит в поле значение, нарушающее свойство ***Условие на значение***.

Значение свойства ***Маска ввода*** вводится обычно только для полей, имеющих тип данных ***Дата/Время*** и текстовых полей. Маска ввода - это синтаксическая конструкция, состоящий из специальных символов маски, указывающих, в какие позиции, в каком количестве и какого типа данные могут быть введены. Маска ввода позволяет усилить контроль за вводимыми данными при эксплуатации базы данных.

Пользователю предлагается ряд готовых масок ввода и возможность сформировать её самому. В приведенной ниже таблице показано, как Access интерпретирует специальные символы, используемые при формировании масок.

Таблица 1

Знаки, используемые для формирования масок ввода.

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 A 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).
Пароль	Создает поле, в котором. любой введенный знак 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 используемых в маске знаков в качестве текстовой константы, необходимо перед ним ввести знак обратной косой черты (\):

Значения остальных свойств поля можно не изменять, оставив значения по умолчанию.

Заполнив значения свойств поля можно вернуться к макету таблицы и в столбце ***Описание*** набрать текст комментария к созданному полю. На этом формирование поля таблицы завершается.

Так последовательно разработчик вводит все поля таблицы.

После создания всех полей таблицы необходимо задать первичный ключ. Для этого в макете таблицы щелкнуть мышью слева от ключевого поля, выделив всю соответствующую строку макета и щелкнуть на кнопке первичного ключа, расположенной на ленте ***Конструктор*** (рис. 11).

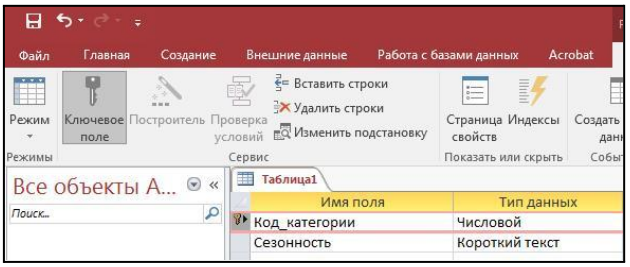


Рис. 11. Создание первичного ключа

Если первичный ключ составной, то необходимо выделить первое

поле ключа, затем нажать клавишу ***Ctrl***и, не отпуская её, выделить мышью другие поля, входящие в состав первичного ключа.

Таким образом, последовательно, вводятся все таблицы формируемой базы данных. Сформировав таблицы базы данных можно перейти в ***Режим таблицы*** и приступить к их заполнению (рис.12).

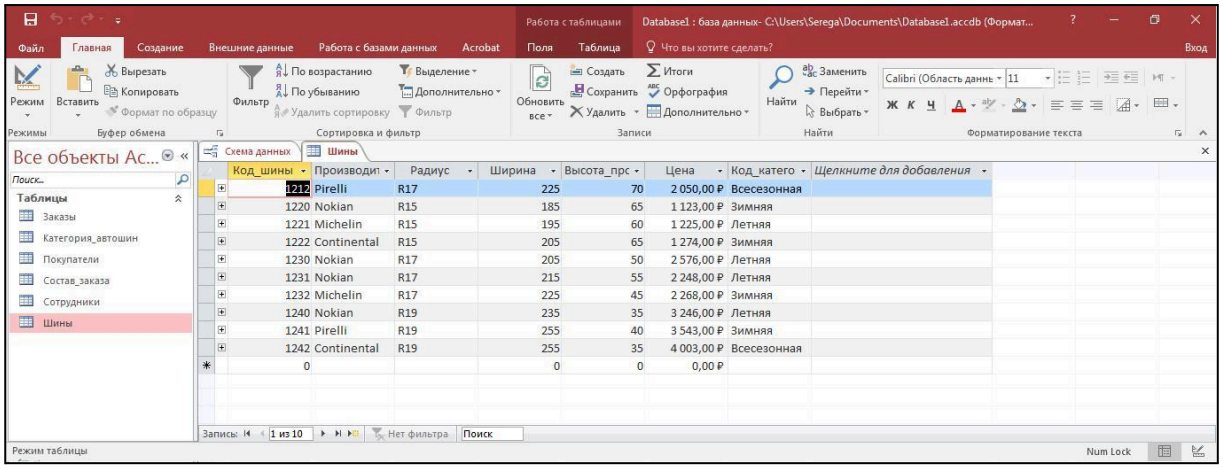


Рис.12. ***Режим таблицы***

3.3. Мастер подстановки

Сделать ввод значений внешних ключей или отдельных полей в таблицу простым и удобным позволяет операция подстановки. Она заключается в том, что значения полей выбираются из списка. Список может быть как фиксированным, так и строиться на базе значений первичного ключа главной таблицы.

Вернемся к нашему примеру. При создании поля ***Код_категории*** таблицы ***Шины*** выберем ему тип данных ***Мастер подстановки***, поскольку в это поле нужно заносить данные, содержащиеся в таблице ***Категория_автошин***(рис.1).

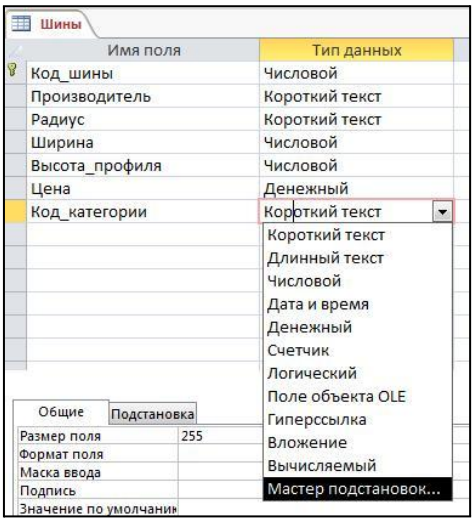


Рис.1.

После выбора типа данных ***Мастер подстановки*** сразу запускается специальная программа, называемая ***Мастер,*** которая предлагает последовательность окон построения поля подстановки. Переход к следующему окну ***Мастера*** осуществляется кнопкой ***Далее >***, а возврат к предыдущему окну – кнопкой ***<Назад***. После завершения построения (в последнем окне мастера) надо щелкнуть кнопку ***Готово***. Последовательность окон программы Мастер для поля ***Код_категории***приведена на рис. 2.а. - 2.ж.

Первое окно ***Мастера*** запрашивает информацию о типе создаваемого набора значений: фиксированный или считываемый из ключевого столбца.

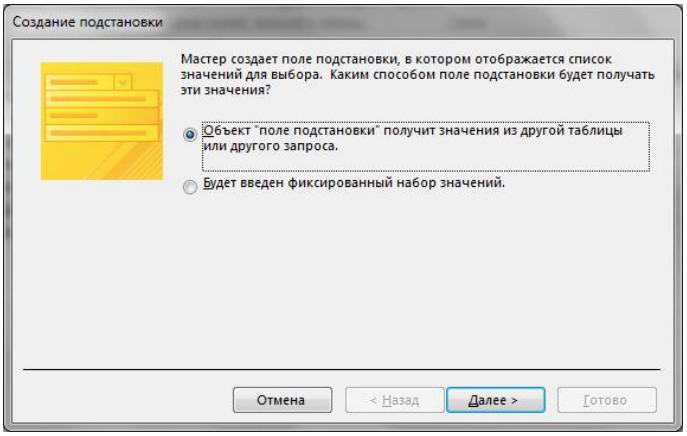


Рис. 2а.

Далее требуется выбрать таблицу, значения поля которой будет содержать столбец подстановки

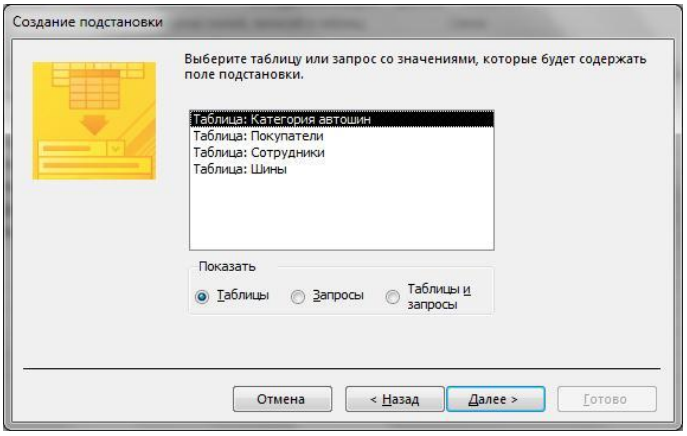


Рис. 2б.

В третьем окне осуществляется выбор полей, значения которых будут показаны при использовании операции подстановки.

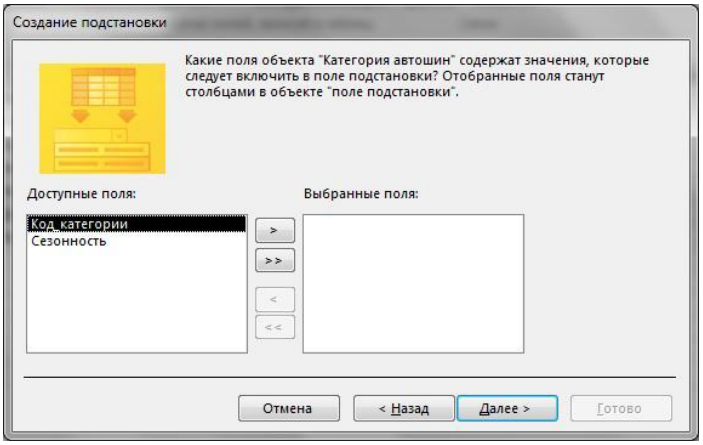


Рис. 2в.

Необходимо используя кнопку >, перевести названия выбранных полей из области ***Доступные поля*** в область ***Выбранные поля***. Кнопка>> перемещает сразу все поля. Отмена действий выполняется аналогичными кнопками < и<< .

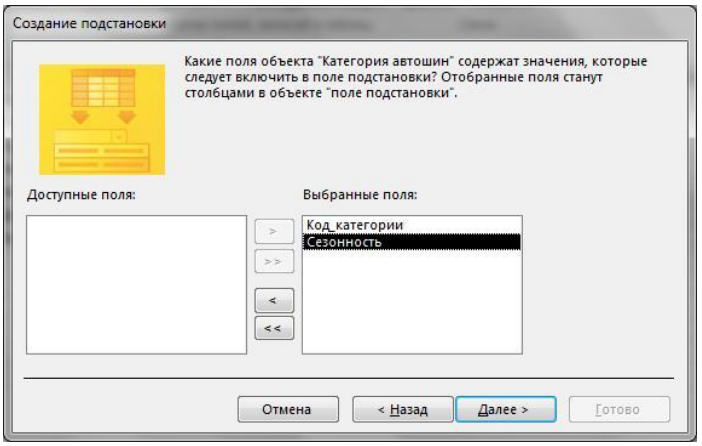


Рис. 2г.

Для удобства подстановки значений их можно упорядочить. При этом допускается упорядочивание информации по четырем полям.

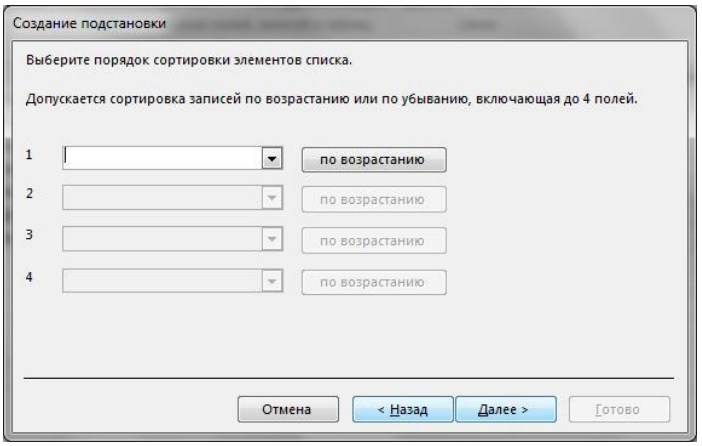


Рис. 2д.

Пятое окно позволяет настроить внешний вид окна подстановки путем задания ширины столбцов. Требуется с помощью мыши установить удобную для пользователя ширину столбцов. Выбранная ширина столбцов запоминается и используется в процессе подстановки значений.

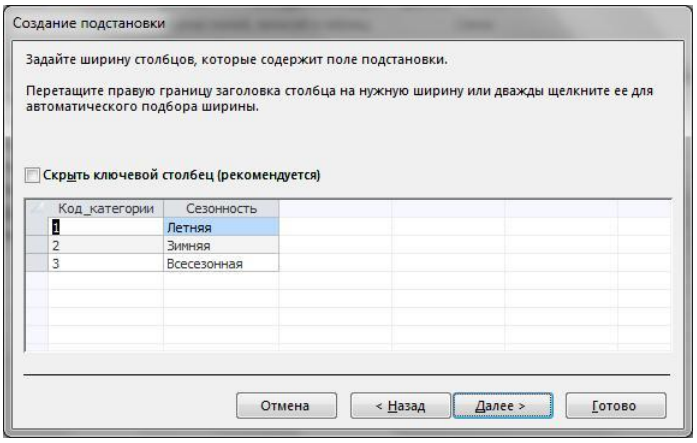


Рис. 2е.

Так как было выбрано несколько столбцов, то в следующем окне необходимо указать столбец, значение которого будет использовано для подстановки. В нашем случае используем поле ***Код_категории***.

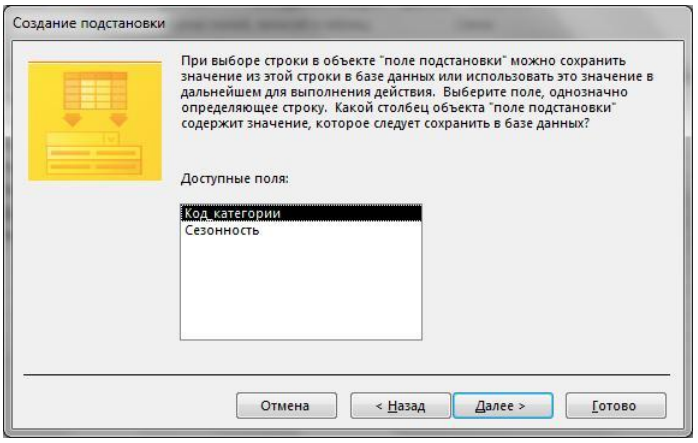


Рис. 2ж.

И наконец, в последнем окне задается параметр ***Подпись*** для поля подстановки. Обычно на этом окне ничего менять не надо, так как программа Мастер сама корректно выбирает имя. В этом же окне рекомендуется включать проверку целостности данных для таблицы.

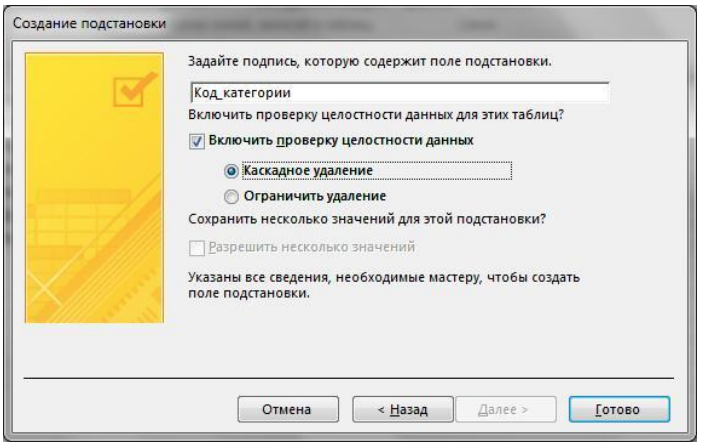


Рис. 2з.

Функционирование подстановки значений в поле ***Код_категории*** таблицы ***Шины*** из таблицы ***Категория_автошин*** (рис.3) представлен на рис.4.

Категория_автошин

Код_катего	Сезонность
1	Летняя
2	Зимняя
3	Всесезонная
0	

Рис.3. Таблица ***Категория_автошин***

Шины

Код_шины	Производит	Радиус	Ширина	Высота_прс	Цена	Код_катего	Щелкните для
0	Nokian		0	0	0,00 ₽		
1212	Pirelli	R17	225	70	2 050,00 ₽	3	
0			0	0	0,00 ₽		

1 Летняя
2 Зимняя
3 Всесезонная

Рис.4. Подстановка значения в поле ***Код_категории***

Для подстановки может использоваться и фиксированный список значений. Например, для заполнения поля ***Производитель*** таблицы ***Шины***.

В данном случае после выбора типа данных ***Мастер подстановки***выполняется последовательность операций, представленная на рис.5а – 5в

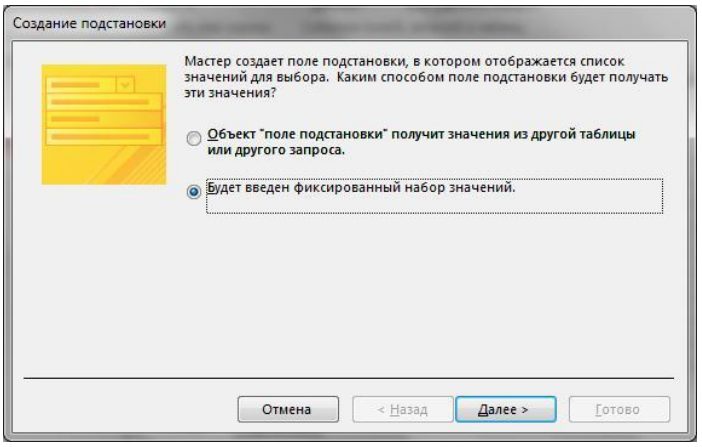


Рис.5а.

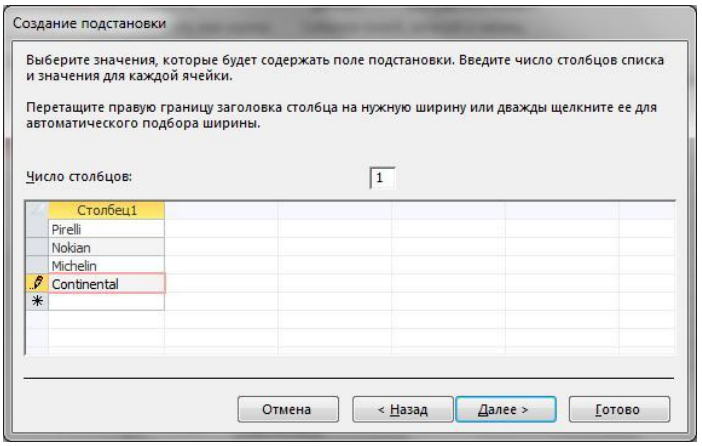


Рис.5б.

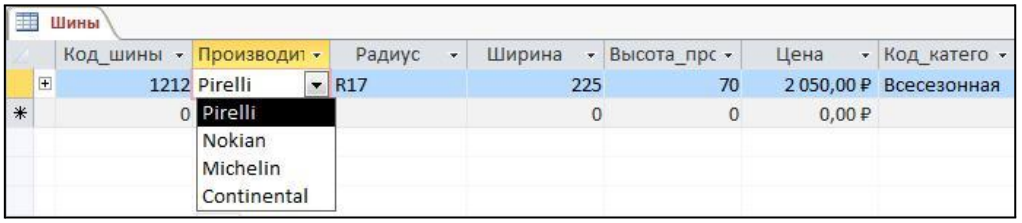


Рис.5в.

3.4. Построение схемы данных

После создания всех таблиц базы данных необходимо установить связи между ними. Для этого нужно перейти на закладку ***Работа с базами данных*** и нажать на кнопку ***Схема данных***. В результате

откроется окно ***Добавление таблиц***, которое позволит отобразить макеты всех необходимых таблиц на закладке ***Схема данных*** (рис.6).

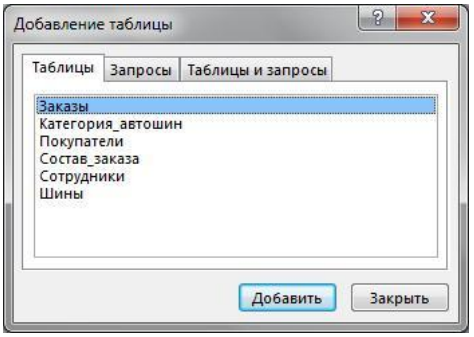


Рис. 6. Добавление таблиц

Последовательно добавляем макеты всех таблиц базы данных.

Если одно из полей таблицы было заполнено с помощью ***Мастера подстановок*** с использованием другой таблицы, то такие таблицы сразу отображаются связанными.

Макеты таблиц на закладке ***Схема даны х***можно расположить удобным образом с помощью мыши.

Для установления связи между двумя таблицами требуется выполнить следующие действия:

а) Щелкнуть мышью на первичном ключе таблицы, находящейся на стороне «1» связи, не отпуская её, перетащить ключ в подчиненную таблицу (в поле внешнего ключа), где отпустить кнопку мыши.

В открывшемся окне ***Изменение связей*** будут высвечены имена полей, по которым связываются таблицы. Необходимо отметить пункт ***Обеспечение целостности данных***. При этом Access сделает невозможным запись в подчиненную таблицу такого значения общего поля, которого нет в главнойтаблице (рис. 7).

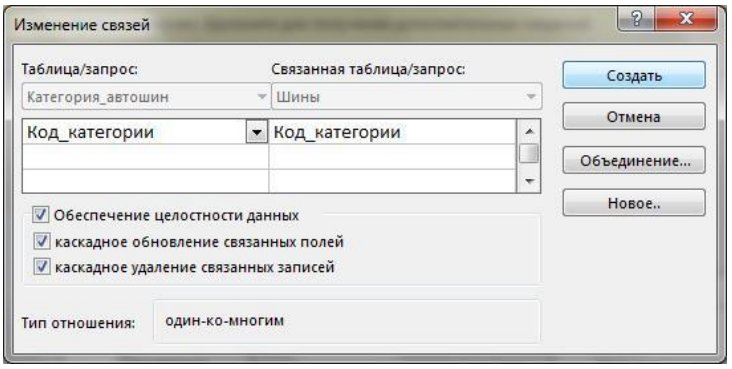


Рис. 7. Установление связи между таблицами.

Целостность данных включаются две дополнительные опции ***Каскадное обновление связанных полей*** и ***Каскадное удаление связанных полей***. Первая позволяет при обновлении значения ключевого поля в главной таблице автоматически изменить его во всех подчиненных таблицах. Вторая опция действует аналогично – удаляя записи из подчиненных таблиц, соответствующих удаленному ключу главной таблицы. Отметив эти пункты необходимо нажать на кнопку ***Создать***.

На ***Схема данных***между таблицами устанавливается связь в виде линии. Для удаления связи линию выделяют и нажимают клавишу ***Delete***.

Схема данных, для рассматриваемого примера, представлена на рис. 8.

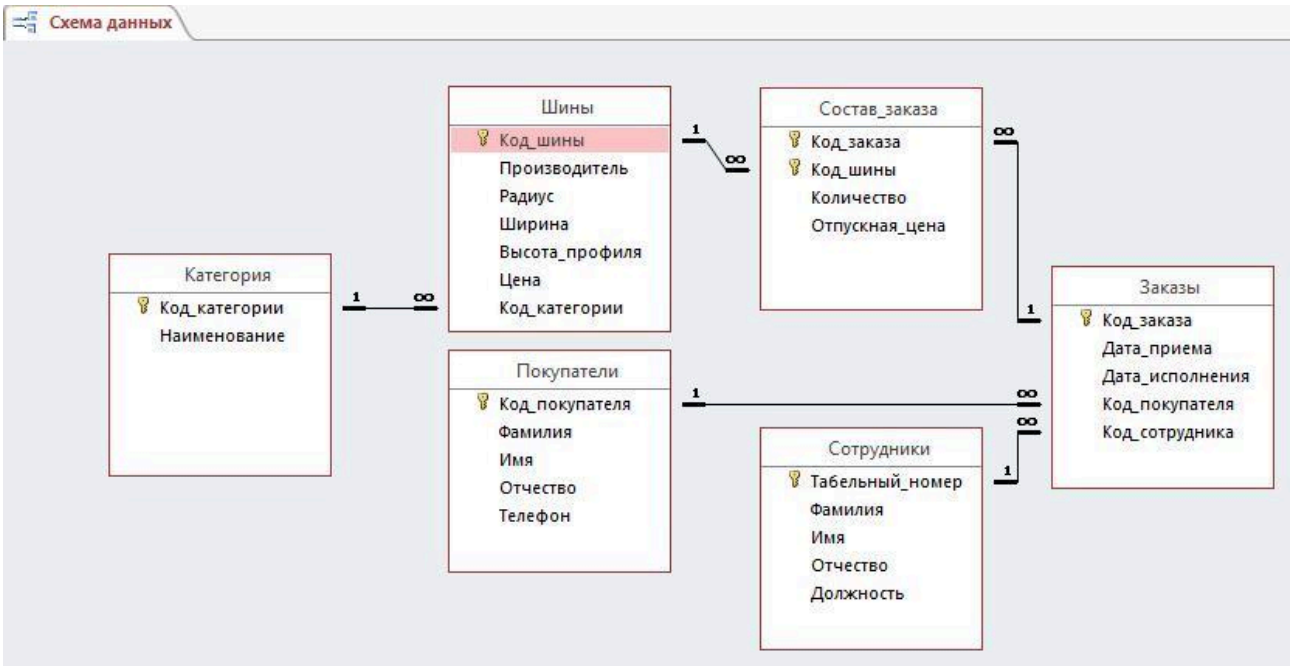


Рис. 8. Схема данных рассматриваемого примера.

4. Поиск информации в БД

4.1.Сортировка информации в таблицах

Можно сделать просмотр записей таблицы более комфортным, если упорядочите их по значениям какого-либо поля. Например, в таблице ***Шины*** записи можно отсортировать в порядке убывания их радиуса. Для этого следует выбрать сортируемое поле и на закладке ***Главная*** в группе ***Сортировка и фильтр*** (рис. 1) потребуется нажать на одну из кнопок сортировки: по возрастанию или по убыванию.

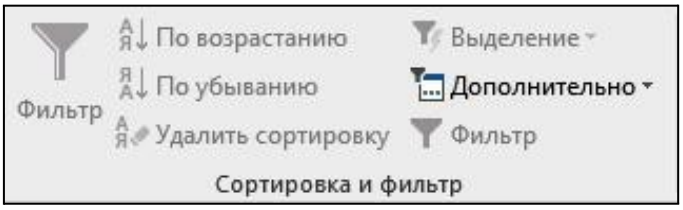


Рис. 1. Группа ***Сортировка и фильтр***

Другой способ выполнения этой операции: щелкнуть правой кнопкой мыши по любой строке нужного столбца и выбрать из контекстного меню соответствующую команду (рис. 2).

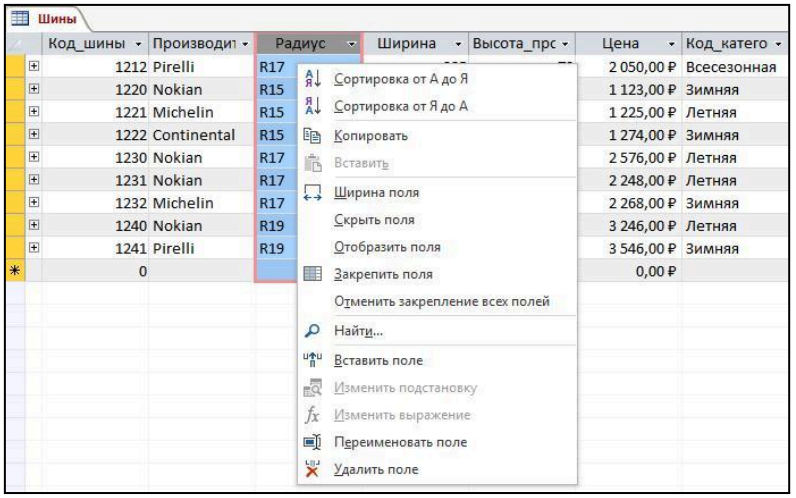


Рис. 2. Контекстное меню

Результат сортировки зависит от типа данных поля:

– значения текстовых полей упорядочиваются по алфавиту;
– числа и денежные суммы упорядочиваются по величине;

– даты упорядочиваются в хронологическом порядке;
– при сортировке логических значений по возрастанию первыми следуют истинные значения, при сортировке по убыванию первыми следуют ложные значения.

Для полей типа MEMO, гиперссылки или объекты OLE сортировка не поддерживается.

Чтобы правильно применять сортировку, нужно знать несколько простых правил:

При сортировке в возрастающем порядке записи, содержащие пустые поля, указываются в списке первыми.

Числа, находящиеся в текстовых полях, сортируются как строки символов, а не как числовые значения. Если нужно отсортировать их в числовом порядке, все текстовые строки должны содержать одинаковое количество символов. Если строка содержит меньшее количество символов, то сначала нужно вставить незначащие нули.

Чтобы сохранить созданную сортировку при следующем открытии таблицы требуется сохранить изменения сочетанием клавиш Ctl+S.

Результат сортировки таблицы ***Шина***по полю ***Радиус*** представлен на рис. 3.

Шины

Код_шины	Производит	Радиус	Ширина	Высота_прс	Цена	Код_катего
1222	Continental	R15	205	65	1 274,00 ₽	Зимняя
1221	Michelin	R15	195	60	1 225,00 ₽	Летняя
1220	Nokian	R15	185	65	1 123,00 ₽	Зимняя
1212	Pirelli	R17	225	70	2 050,00 ₽	Всесезонная
1232	Michelin	R17	225	45	2 268,00 ₽	Зимняя
1231	Nokian	R17	215	55	2 248,00 ₽	Летняя
1230	Nokian	R17	205	50	2 576,00 ₽	Летняя
1241	Pirelli	R19	255	40	3 546,00 ₽	Зимняя
1240	Nokian	R19	235	35	3 246,00 ₽	Летняя
0			0	0	0,00 ₽	

Рис. 3. Сортировка таблицы ***Шина***по полю ***Радиус***

Иногда нужно выполнить сортировку по значению нескольких полей. Для этого лучше сначала переместить сортируемые столбцы таким образом, чтобы они оказались, во-первых, рядом, а во-вторых, с

учетом приоритетов, — приоритеты устанавливаются слева направо, т. к. первыми будут сортироваться значения в крайнем левом столбце. После этого нужно выделить все столбцы и нажать, соответственно, кнопку ***Сортировка по возрастанию***или ***Сортировка по убыванию***на панели инструментов.

Для восстановления порядка отображения записей, используется кнопка ***Удалить сортировку***.

4.2. Фильтры

Работая с таблицей в оперативном режиме, можно установить фильтр, т.е. задать, логическое выражение, которое позволит выдавать на экран только те записи, для которых это выражение выполняется.

В Access предусмотрено несколько способов отбора записей с помощью фильтров: ***простой фильтр***, ***фильтр по выделению***, ***фильтр по форме*** и ***расширенный фильтр***:

Простой фильтр и ***фильтр по выделению*** обеспечивают отбор записей по значениям одного столбца.

Создание простого фильтра заключается в выборе одного из встроенных критериев отбора (набор критериев зависит от типа данных столбца). В таблице можно применить одновременно несколько простых фильтров, каждый для своего столбца.

Для создания простого фильтра, необходимо следующие:

1. Щелкнуть кнопкой мыши на черную стрелку в заголовке столбца (***Ширина***) (рис. 4).

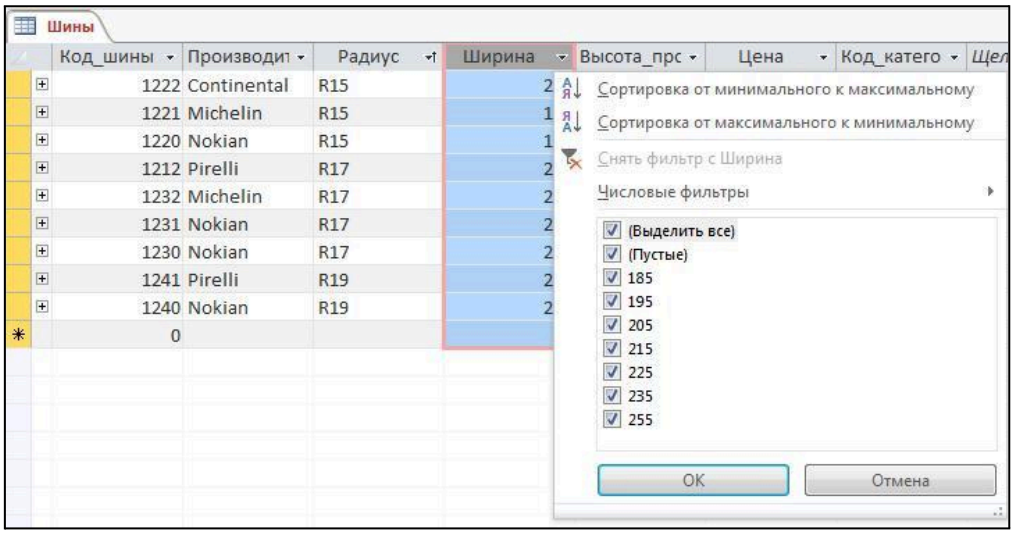


Рис. 4. Окно сортировки и фильтрации

2. Создать условие отбора одним из двух способов:

– в перечне всех значений поля снять флажок ***Выделить все***. Затем установить флажки для тех значений поля, которые необходимо включить в фильтр. Нажать кнопку **ОК**. В результате будут отобраны те записи, в которых значение поля совпадает с одним из значений, отмеченных флажком;

– щелкнуть кнопкой мыши на пункте меню, расположенном непосредственно над перечнем значений: в зависимости от типа данных этот пункт может называться ***Текстовые фильтры***, ***Числовые фильтры***или ***Фильтры дат***. В подменю выбрать один из встроенных фильтров (рис. 5),

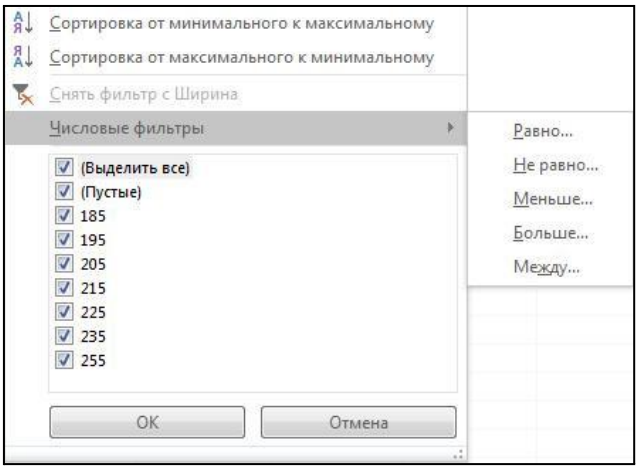


Рис. 5. Подменю встроенных фильтров

затем, ввести константы, с которыми будут сравниваться значения поля.

Чтобы использовать ***фильтр по выделению***, необходимо:

1. В режиме ***Таблицы*** найти значение поля, которое предполагается использовать в качестве основы для фильтрации.
2. Выделить это значение и нажать кнопку ***Выделение*** на вкладке ***Главная***в группе ***Сортировка и фильтр***.
3. Выбрать одну из предлагаемых в подменю команд.

Эти команды доступны также в контекстном меню поля по щелчку поля правой кнопкой мыши.

Пример фильтрации приведен на рис. 6.

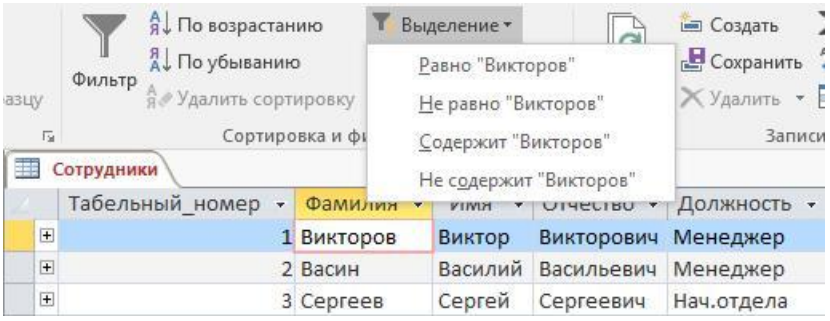


Рис. 6. Фильтрация фамилий сотрудников.

После того как для поля создан ***простой фильтр*** или ***фильтр по выделенному***, справа от названия поля отображается кнопка.

Если требуется удалить фильтр для одного из столбцов, нужно выделить его и в группе ***Сортировка и фильтрация*** выбрать пункт ***Снять фильтр***. При этом отсутствует возможность снова включить фильтр, его придется создать заново.

Фильтр по форме позволяет отбирать записи по значениям нескольких столбцовили при указании нескольких условий отбора с помощью логических операторов ***И***и***Или***.

Чтобы применить фильтр, необходимо:

1. Открыть таблицу в режиме ***Таблицы***.
2. Нажать кнопку ***Дополнительно***на вкладке ***Главная***в группе ***Сортировка и фильтр***. В появившемся подменю выбрать пункт ***Фильтр по форме*** В результате откроется специальная форма изменения фильтра (рис. 7).

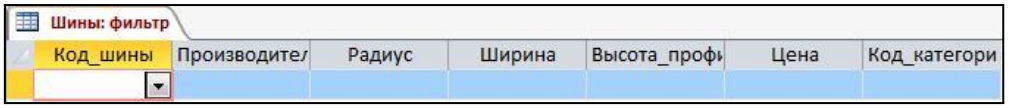


Рис. 7. Форма изменения фильтра

Форма содержит линейку полей таблицы. В любое из этих полей можно ввести или выбрать из списка значение, которое и будет являться условием отбора. Если условия ввести в несколько полей, они будут объединяться с помощью логического оператора ***И***(рис. 8). Для того чтобы объединить условия по ***ИЛИ***, нужно раскрыть другую вкладку формы, щелкнув по ярлычку ***Или*** в её нижней части.

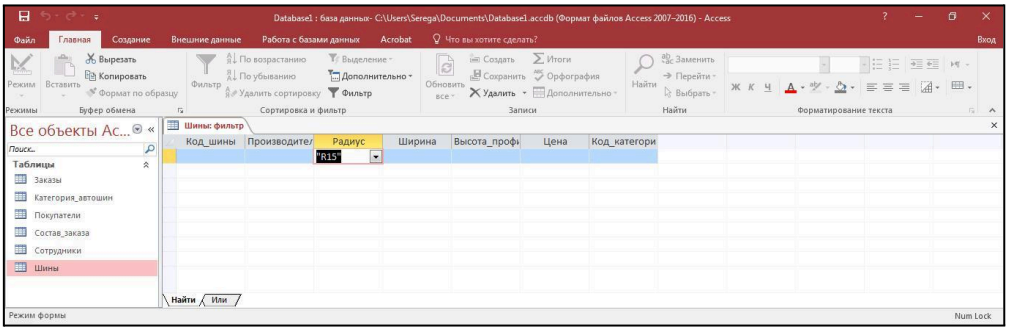


Рис. 8. Ввод условий отбора

Для выполнения фильтрации данных необходимо нажать кнопку ***Применить фильтр***на вкладке ***Главная***в группе ***Сортировка и фильтр***. Результат отбора информации ***"R15"ИЛИ "R17"***представлен на рис. 9.

Шины

Код_шины	Производит	Радиус	Ширина	Высота_прс	Цена	Код_катего
1222	Continental	R15	205	65	1 274,00 ₽	Зимняя
1221	Michelin	R15	195	60	1 225,00 ₽	Летняя
1220	Nokian	R15	185	65	1 123,00 ₽	Зимняя
1212	Pirelli	R17	225	70	2 050,00 ₽	Всесезонная
1232	Michelin	R17	225	45	2 268,00 ₽	Зимняя
1231	Nokian	R17	215	55	2 248,00 ₽	Летняя
1230	Nokian	R17	205	50	2 576,00 ₽	Летняя
0			0	0	0,00 ₽	

Рис. 9. Результат применения фильтра

Расширенный фильтр предоставляет пользователю огромные возможности по фильтрации информации. Он позволяет применить фильтр, отсутствующий в списке обычных фильтров, когда придется написать условие для фильтра самостоятельно.

Для формирования условия расширенного фильтра нажать кнопку ***Дополнительно*** на вкладке ***Главная***в группе ***Сортировка и фильтр*** и выбрать в открывшемся подменю пункт ***Расширенный фильтр.***

По данной команде открывается окно, в верхней части которого расположен макет таблицы, а в нижней – таблица формирования фильтра. Для формирования критерия отбора информации перетаскивают с помощью мыши имя поля (например,***Ширина***) и вводят в строку ***Условие отбора*** критерий выбора (>225). По второму полю (например,***Цена***) можно ввести критерий сортировки – ***по возрастанию*** и условие отбора <4000 (рис. 10)

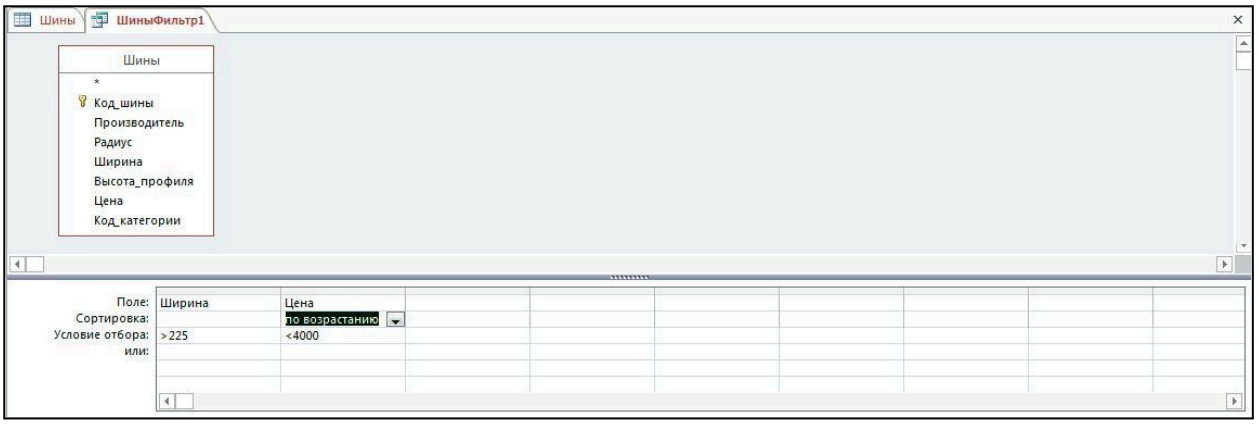
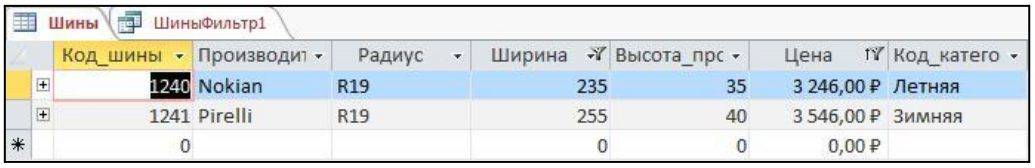


Рис. 10. Настройка расширенного фильтра

После выполнения команды ***Применить фильтр*** будут показаны записи, удовлетворяющие указанному условию (рис. 11).



Код_шины	Производит	Радиус	Ширина	Высота_прс	Цена	Код_катего
1240	Nokian	R19	235	35	3 246,00 ₽	Летняя
1241	Pirelli	R19	255	40	3 546,00 ₽	Зимняя
0			0	0	0,00 ₽	

Рис. 11. Результат работы фильтра.

При фильтрации нельзя подавить отображение отдельных полей и выполнить вычисления.

5. Запросы

Пользоваться сортировкой и фильтрацией очень удобно, но при работе с несколькими таблицами их возможностей недостаточно. В таких случаях применяются запросы. Самый распространенный тип запросов отображает записи, удовлетворяющие определенным условиям. К ним относятся:

- запрос на выборку извлекает данные из одной или нескольких таблиц и представляет их в табличном виде. Этот тип запроса можно использовать для группировки записей, вычисления сумм, средних величин и других итоговых значений. Работая с результатами запроса, можно одновременно редактировать данные из нескольких таблиц.
- параметрический запрос запрашивает ввод значений параметров, определяющих условия выборки (например, начальную и конечную дату). Этот тип запросов часто используется для получения отчетов за определенный период времени.
- перекрестный запрос выполняет расчеты и группирует данные для анализа информации. Для элементов, расположенных в левом столбце и в верхней строке результатов запроса, могут вычисляться итоговые значения (сумма, количество или средняя

величина). Ячейки на пересечении строк и столбцов также содержат вычисляемые значения.

– запрос на действие, которое вносит множественные изменения за одну операцию. Собственно, это запрос на выборку, который выполняет определенные действия над результатами отбора. Возможны четыре типа действий: обновление, удаление и добавление записей и создание таблицы.

5.1. Создание простого запроса с помощью *Мастера запросов*

Наиболее просто создается запрос при помощи Мастера запросов. Чтобы создать простой запрос нужно перейти на закладку ***Создание*** и нажать кнопку ***Мастер запросов*** в группе ***Запросы*** (рис. 12).

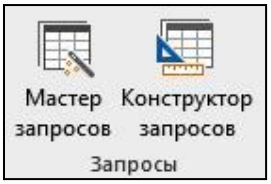


Рис. 12. Вызов ***Мастера запросов***

Работа программы ***Мастер запросов*** выполняется в несколько последовательных шагов. На первом шаге требуется определить вид создаваемого запроса (рис. 13).

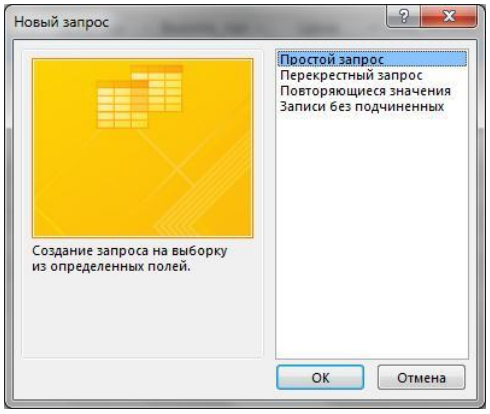


Рис. 13. Выбор вида создаваемого запроса.

Выбирается «Простой запрос» о нажимается кнопка «Ok». На втором шаге требуется указать исходную таблицу или запрос и

выбрать поля, информация которых будет отобрана в процессе выполнения запроса (рис. 14).

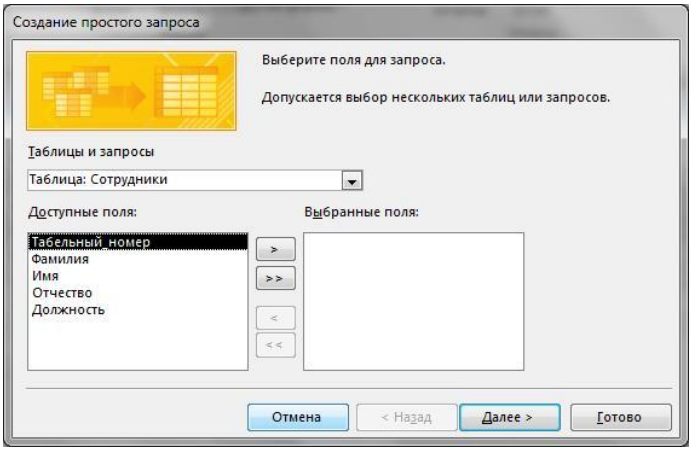


Рис. 14. Выбор таблицы и полей.

С помощью стрелок вправо и влево переместить из списка ***Доступные поля*** в список ***Выбранные поля*** те поля, которые необходимы в запросе. Для включения в запрос всех поля, нажать кнопку *с* двумя стрелками (рис. 15).

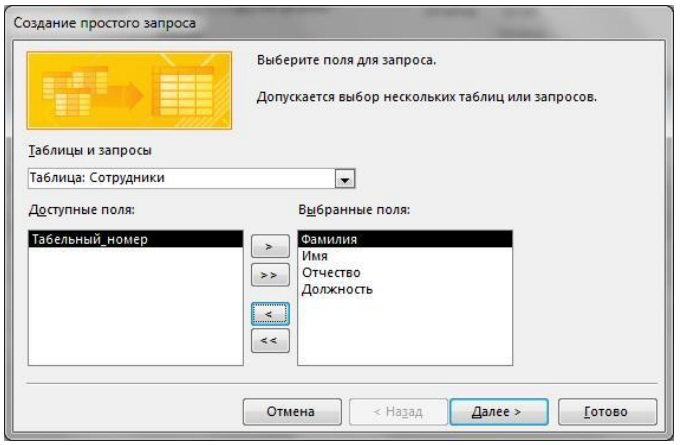


Рис. 15. Выбор таблицы и полей

На последнем шаге требуется ввести имя создаваемого запроса (рис. 16) в поле ***Задайте имя запроса*** и выбрать дальнейшие действия: ***Открыть запрос для просмотра данных***или***Изменить макет запроса***.

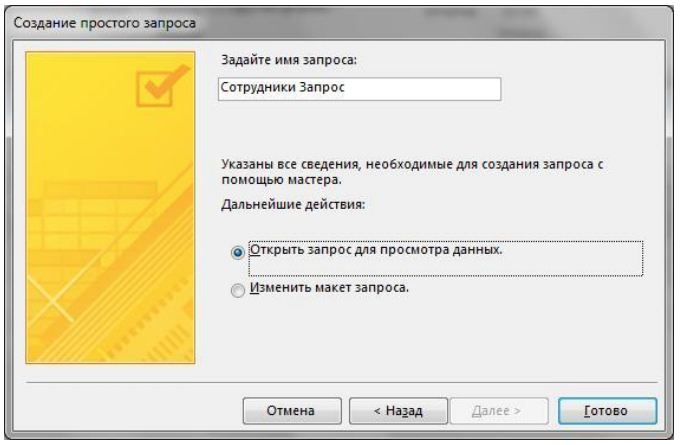


Рис. 16. Ввод имени запроса.

Результат выполнения запроса представлен на рис. 17.

Сотрудники Запрос

Фамилия	Имя	Отчество	Должность
Викторов	Виктор	Викторович	Менеджер
Васин	Василий	Васильевич	Менеджер
Сергеев	Сергей	Сергеевич	Нач.отдела

Рис. 17. Итог выполнения запроса.

5.2. Создание запросов режиме конструктора

Конструктор запросов, обеспечивает полное управление параметрами запроса и построение сложных условий отбора данных. Он вызывается кнопкой ***Конструктор запросов*** на закладке ***Создание*** (см. рис. 12).

Нажатие на эту кнопку приводит к открытию главного окна ***Конструктора запросов*** и вспомогательного окна ***Добавление таблицы*** (рис.1).

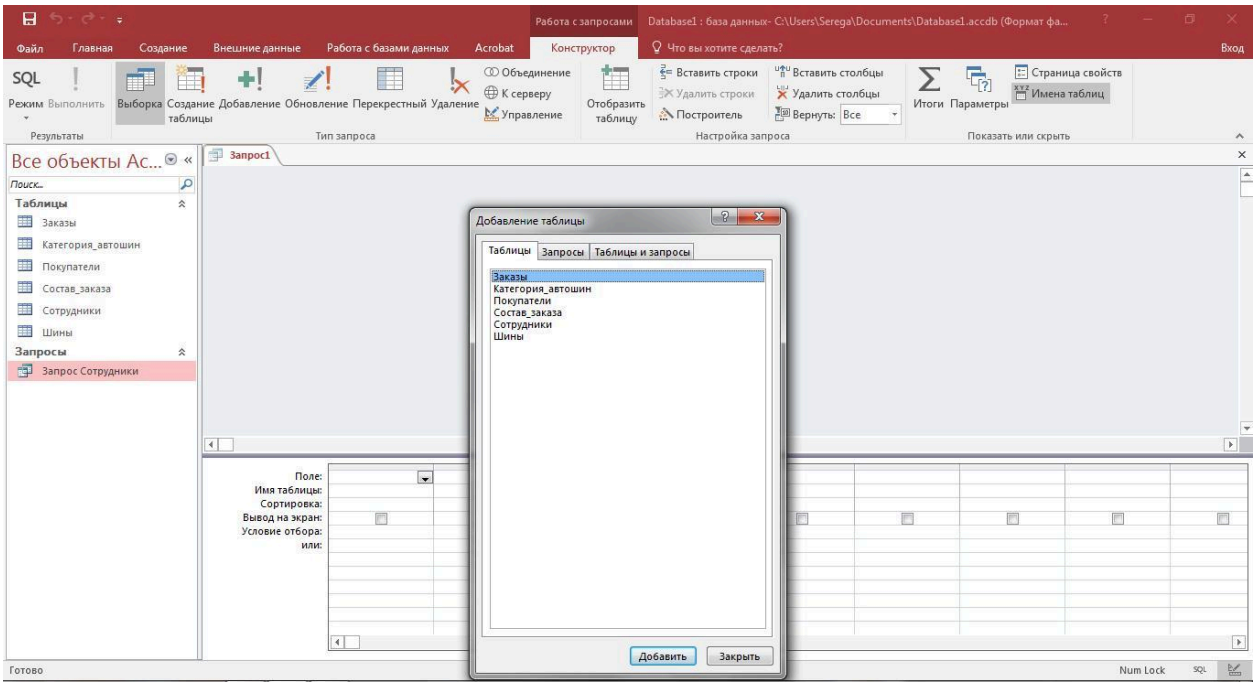


Рис. 1. Окно ***Конструктора запросов***.

Окно ***Конструктора запросов*** имеет две области. В верхней области окна отображаются таблицы, которые формируют информационную базу создаваемого запроса. В нижней области находится ***бланк запроса***– таблица, ячейки которой используются для формирования запроса. На бланке должны быть отображены все столбцы, включенные в результирующее множество запроса.

Перечень необходимых таблиц и запросов формируется с использованием окна ***Добавление таблицы*** (см. рис. 2). Чтобы добавить их в окно запроса достаточно выделить имя таблицы и щелкнуть на кнопку ***Добавить***. Макет таблицы будет отображен в верхней части окна (рис. 3).

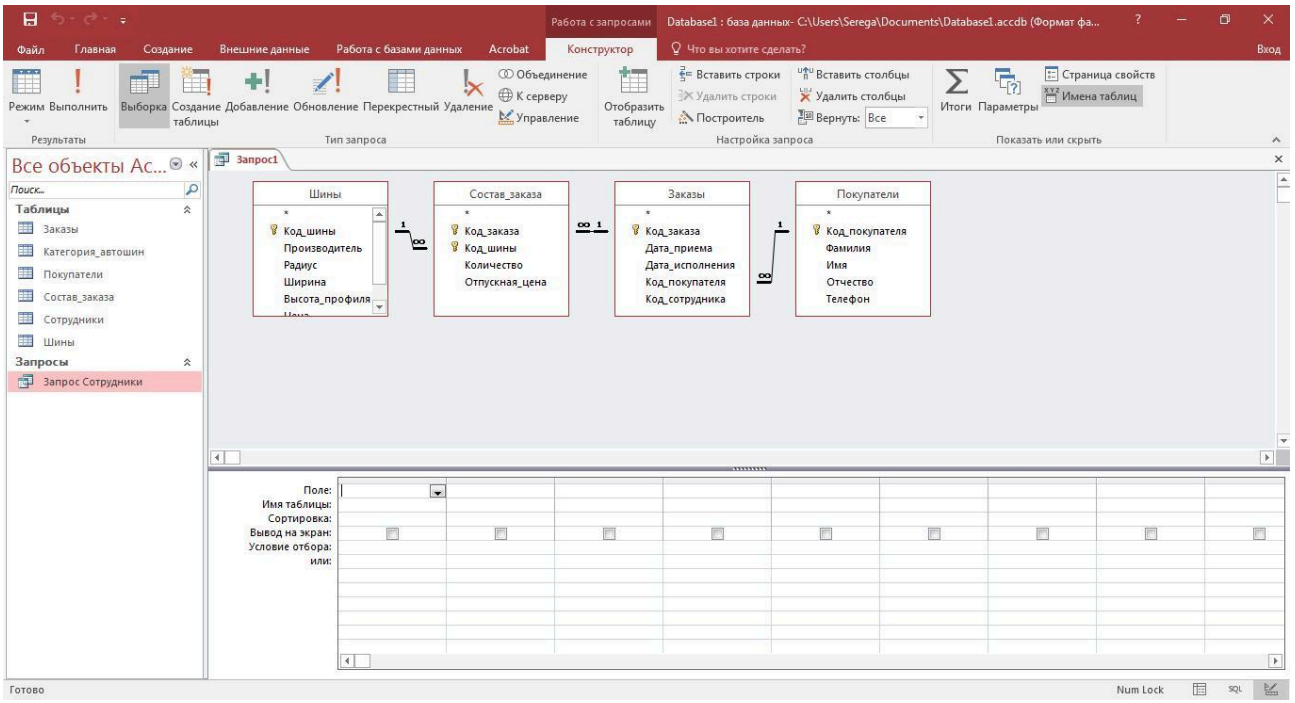


Рис. 3 Макеты таблиц и информационные связи

Формирование запроса начинается с указания, какие поля из базовых таблиц будут в нем отображаться. Включать в запрос можно поля из любой таблицы. Способов включения полей в запрос существует несколько: дважды щелкните левой кнопкой мыши на выделенном поле; перетащить поле в первую строку бланка; использовать раскрывающийся список в строке ***Поле*** бланка запроса.

Чтобы удалить поле из запроса, выделите нужный столбец в бланке запроса, а затем нажмите клавишу <Delete>. Чтобы выделить столбец, пользуйтесь областью выделения столбцов – узкой серой полоской над столбцами.

В режиме ***Конструктора запросов*** можно изменять имена полей запроса. Чтобы переименовать поле, необходимо установить курсор в ***бланке запроса*** перед первой буквой его имени и ввести новое имя и символ двоеточия. Изменение имени поля в бланке запроса приводит к изменению заголовка столбца при просмотре запроса в режиме таблицы. Имя поля базовой таблицы при этом не изменяется.

Строка ***Сортировка***позволяет указать поле и принцип сортировки информации в нем по возрастанию или убыванию.

В строке ***Условие отбора*** и в строке ***Или*** указываются условия отбора записей. Такими условиями могут быть логические выражения. Например, (>30), (='Иванов'), (=10) и т. п.

Условия, находящиеся в одной строке, но в разных столбцах бланка, объединяются по логическому оператору **And (И).** Если нужно объединить условия отбора по логическому оператору **Or (ИЛИ),** разместите эти условия в ответствующей строке ***бланка запроса***.

В запросах для описания критериев выборки записейактивно используются***Выражения***. ***Выражение***– это последовательность операндов (констант, идентификаторов и функций) соединенных знаками операций, указывающая какие действия требуется выполнить над данными.

Для создания выражений в Access существует следующие категории операторов: ***арифметические***, ***логические***, ***конкатенации*** и ***сравнения***.

Арифметические операторы

Арифметические операторы выполняют сложение, вычитание, умножение и деление. Они оперируют только с числовыми значениями.

Оператор	***Описание***	***Пример***
+	Сложение	[Итог] + [Надбавка]
-	Вычитание	Date () – 7
*	Умножение	[Коробок] * [Цена коробки]
/	Деление	[Количество] / 12.55
\	Целочисленное деление	[Коробок] \ 2
Mod	Остаток от деления	15 Mod 12

Оператор	*Описание*	*Пример*
^	Возведение в степень	[Размер]^2

Логические операторы

Логические (булевы) операторы используются для объединения результатов двух или более выражений сравнения в единое целое

Оператор	*Описание*	*Пример*
And	Конъюнкция (логическое И)	A And B
Or	Дизъюнкция (логическое ИЛИ)	A Or B
Not	Логическое отрицание	Not A
Xor	Исключающее ИЛИ	A Xor B
Eqv	Логическая эквивалентность	A EqvB
Imp	Логическая импликация	A Imp B

Операторы слияния строковых значений (конкатенации)

Стандартный значок оператора конкатенации – амперсант (&) выполняет операцию объединения двух текстовых строк в одну.

Операторы сравнения

Операторы сравнения соотносят значения двух операндов и возвращают логические значения ***True*** или ***False***.

Оператор	*Описание*	*Пример*
<	Меньше	[Количество] < 15
<=	Меньше либо равно	[Размер]<= 50
=	Равно	[Сумма] = 500
>=	Больше либо равно	[Процент]>= 25
>	Больше	[Цена]> 100
<>	Неравно	[Итог]<> [Сумма]

Операторы сравнения с образцом

Эти операторы возвращают **True**или **False**, в зависимости от соответствия значения в поле выбранной спецификации оператора.

Оператор	***Описание***	***Пример***
Between	Определяет, находится ли числовое значение в диапазоне значений	Between (-100) And (100)
Is	При использовании вместе с Null определяет, является ли значение Null или NotNull	Is Null
In	Определяет, является ли строковое значение элементом списка	In ("Москва", "Киев")
Like	Определяет, включает ли строковое значение указанные символы	Like"Ив*" Like"db??"

Символ «*» замещает любое число знаков, а символ «?» замещает только один знак. Символы шаблона «*» и «?» могут стоять в любом месте строки.

5.3. Построитель выражений

Ввод выражений возможен вручную и с помощью инструмента, называемого ***Построитель выражений***.

Если в бланке запроса щелкнуть правой кнопкой мыши на ячейке ***Условие отбора*** и выбрать в контекстном меню команду ***Построить***, то откроется окно ***Построителя выражений*** (рис. 4). Аналогичный результат может быть получен при использовании кнопки ***Построитель*** закладки ***Конструктор*** в группе ***Настройка запроса.***

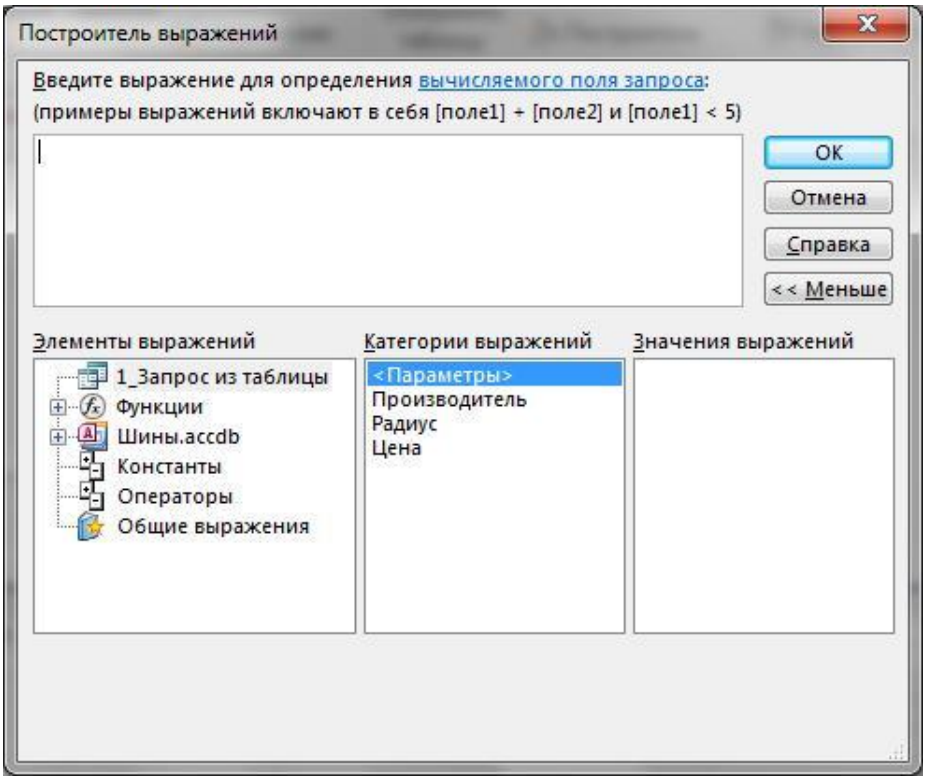


Рис. 4. Окно***Построитель выражений***

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

В средней части окна построителя находятся кнопки с часто используемыми операторами. При нажатии на одну из этих кнопок построитель вставит соответствующий оператор в текущую позицию поля выражения.

В нижней части окна построителя находятся три поля:

В левом поле выводятся папки, содержащие таблицы, запросы, формы, объекты базы данных, встроенные функции, константы, операторы и общие выражения.

В среднем поле задаются определенные элементы или типы элементов для объектов, задаваемых в левом поле. Например, если выбрать в левом поле ***Встроенные функции***, то в среднем поле появится список всех типов функций MicrosoftAccess.

В правом поле выводится список значений для элементов, заданных в левом и среднем полях. Например, если выбрать в левом поле ***Встроенные функции*** и тип функции в среднем, то в правом поле будет выведен список всех встроенных функций выбранного типа (рис.5).

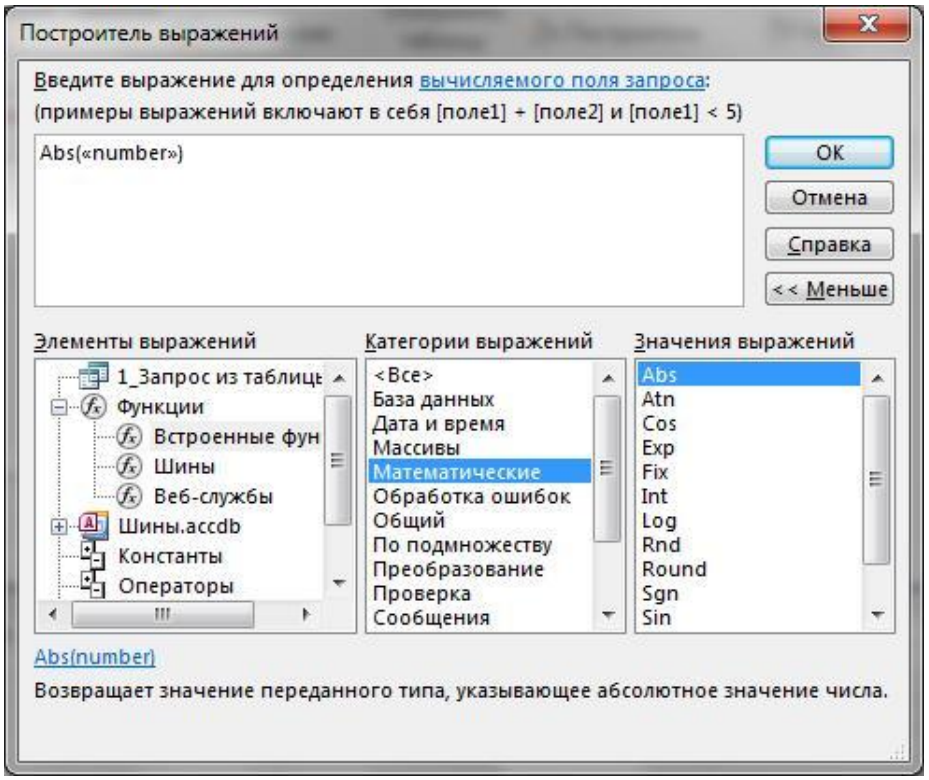


Рис. 5. Выбор встроенной математической функции.

Созданное выражение вставляется в ту позицию, из которой был вызван ***Построитель выражений***.

Построение запроса можно рассмотреть на следующем примере: выяснить фамилию, имя и отчество покупателей, которые приобретали шины Nokia, а также даты приема заказов и их исполнения. Фамилии покупателей вывести в алфавитном порядке.

Для построения запроса потребуется использовать таблицы ***Шины***, ***Состав заказа***, ***Заказы*** и ***Покупатели***.

На поле ***Производитель*** таблицы ***Шины*** наложено условие отбора информации – Like"Nokia" (возможный вариант Like("Nokia")). Так

как значение этого поля во всех отобранных записях будет одинаковым, его лучше не выводить в результирующей таблице. Сортировка по возрастанию наложена поле ***Фамилия*** таблицы ***Покупатели***.

Сформированный таким образом в ***Конструкторе*** запрос представлен на рисунке 6.

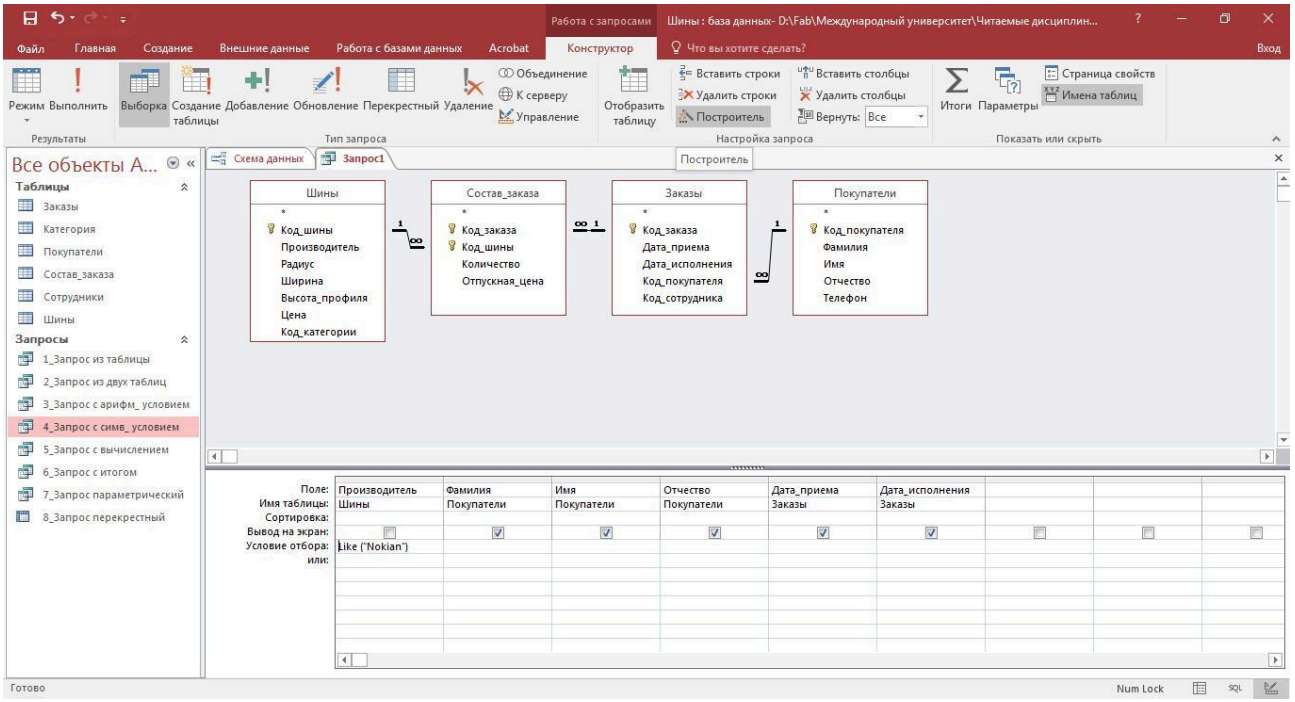
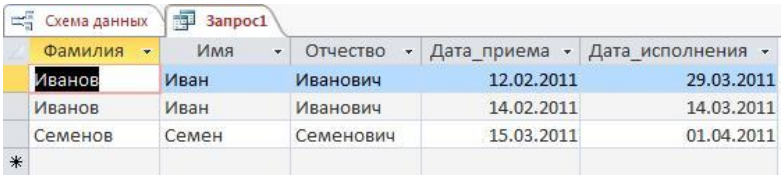


Рис. 6.Сформированный в Конструкторе запрос

На рисунке 7 приведена результирующая таблица выполнения запроса



Фамилия	Имя	Отчество	Дата_приема	Дата_исполнения
Иванов	Иван	Иванович	12.02.2011	29.03.2011
Иванов	Иван	Иванович	14.02.2011	14.03.2011
Семенов	Семен	Семенович	15.03.2011	01.04.2011

Рис. 7. Результат выполнения запроса

5.4. Создание вычисляемых полей в запросах

Запросы в приложении позволяют в виде одной таблицы представить данные из нескольких связанных таблиц и отобрать нужные записи из этих таблиц. Кроме того, можно создавать столбцы в запросе, которые являются результатом вычислений над значениями других столбцов. Такие столбцы называются ***вычисляемыми***. Это существенно расширяет возможности запросов. Простейшим примером вычисляемого поля в запросе может быть поле, которое объединяет имя, отчество и фамилию человека. На рис. 1 показан пример такого поля в запросе, созданном на базе таблиц «Сотрудники» и «Заказы».

Чтобы создать вычисляемое поле, нужно ввести выражение, которое вычисляет требуемое значение, в строку ***Поле*** свободного столбца бланка запроса. В данном примере это выражение представляет собой конкатенацию полей, содержащих имя, отчество и фамилию сотрудника, с пробелом между ними. В этом выражении используются ссылки на поля таблицы, которые в выражении заключаются в квадратные скобки. Перед выражением нужно написать имя поля: ФИО и отделить его двоеточием от выражения.

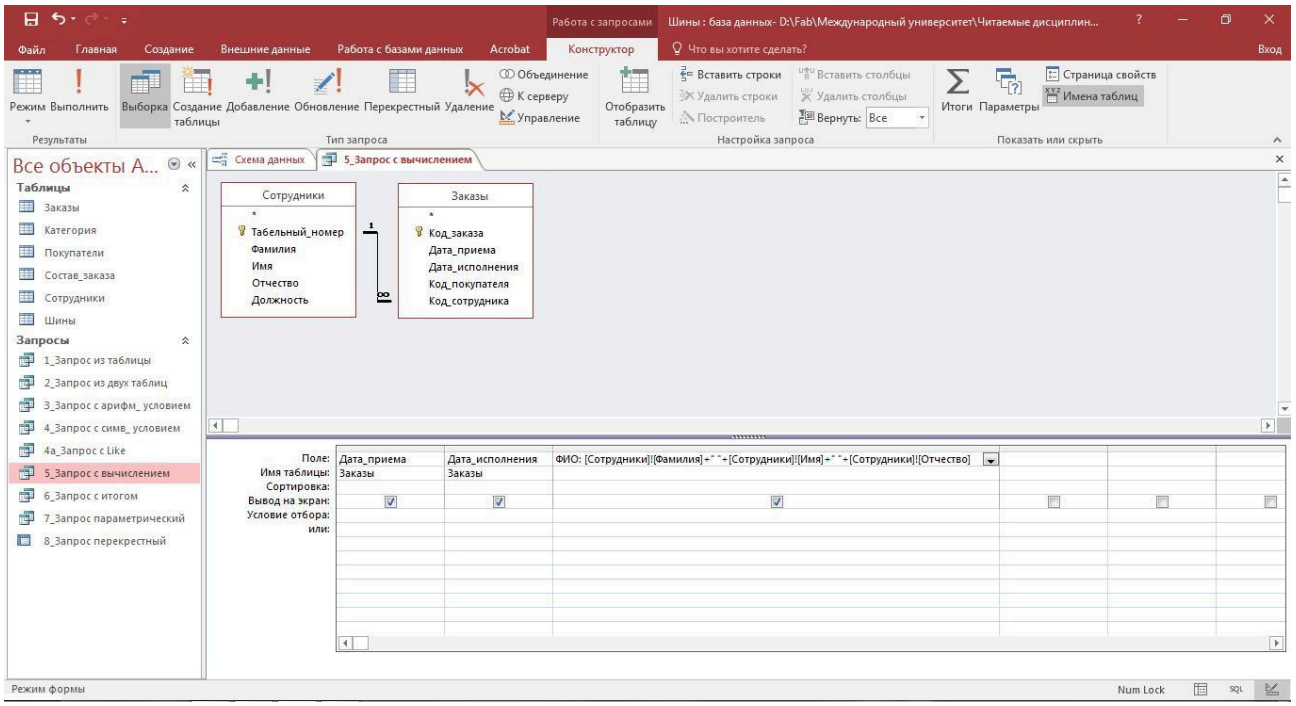


Рис. 1. Вычисляемое поле в запросе

Если выражение длинное и его неудобно писать в строке ***Поле***, нажмите комбинацию клавиш **<Shift>+<F2>**. Появится диалоговое окно ***Область ввода*** (рис. 2), в котором вводить выражение удобнее. Можно также использовать ***Построитель выражений***.

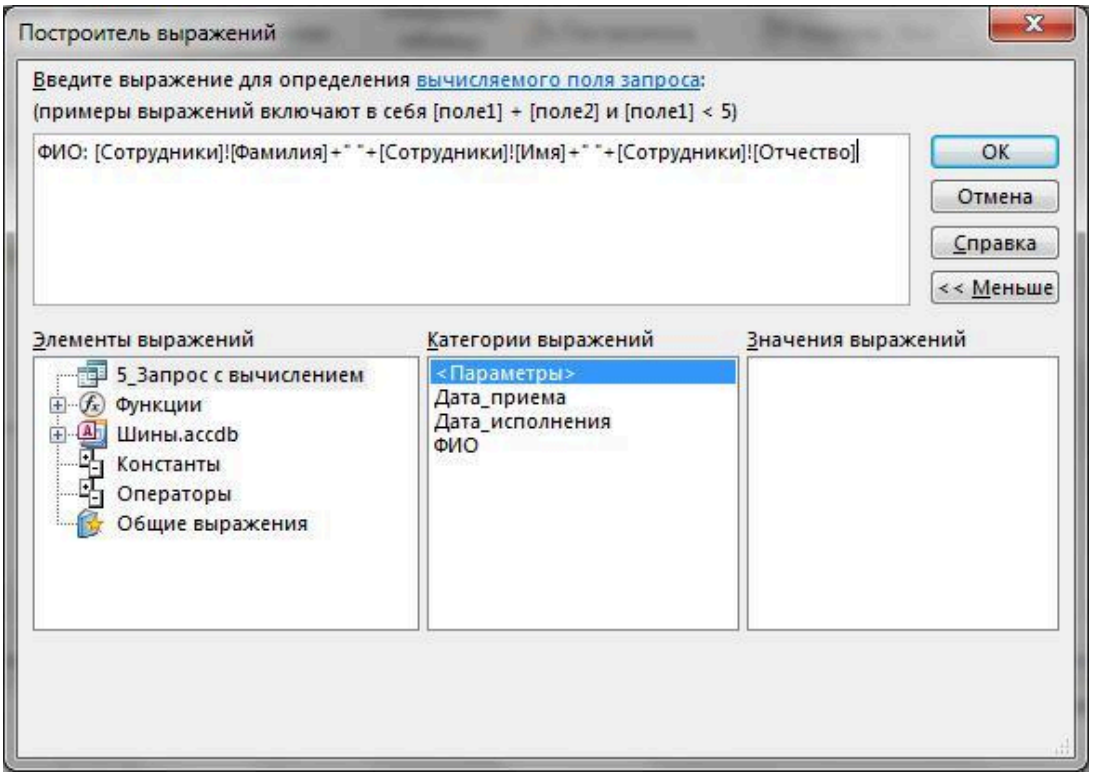
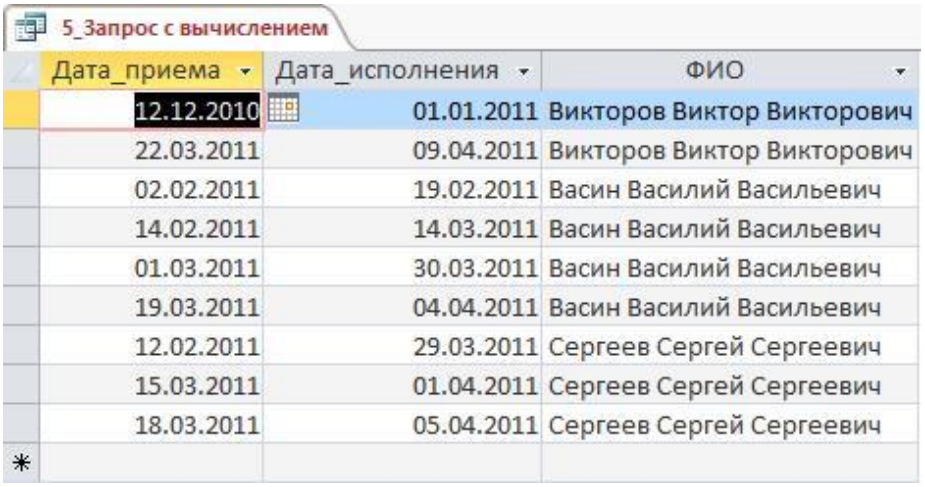


Рис. 2. Использование ***Построителя выражений***

Результат запроса на рисунке 3.



5_Запрос с вычислением

Дата_приема	Дата_исполнения	ФИО
12.12.2010	01.01.2011	Викторов Виктор Викторович
22.03.2011	09.04.2011	Викторов Виктор Викторович
02.02.2011	19.02.2011	Васин Василий Васильевич
14.02.2011	14.03.2011	Васин Василий Васильевич
01.03.2011	30.03.2011	Васин Василий Васильевич
19.03.2011	04.04.2011	Васин Василий Васильевич
12.02.2011	29.03.2011	Сергеев Сергей Сергеевич
15.03.2011	01.04.2011	Сергеев Сергей Сергеевич
18.03.2011	05.04.2011	Сергеев Сергей Сергеевич

Рис. 3.

5.5. Создание параметрического запроса

Запрос в Access является объектом, который сохраняется в файле базы данных и может многократно повторяться. Если требуется повторять запрос с различными значениями в условиях отбора, его нужно открыть в режиме Конструктора, изменить условие и выполнить. Чтобы не делать многократно этих операций, можно создать запрос с параметрами. При выполнении такого запроса выдается диалоговое окно ***Введите значение параметра***, в котором пользователь может ввести конкретное значение и затем получить нужный результат.

Чтобы определить параметр запроса, введите в строку ***Условие отбора*** для соответствующего столбца вместо конкретного значения слово или фразу и заключите их в квадратные скобки, например, [Поставщик:]. Эта фраза будет выдаваться в виде приглашения в диалоговом окне при выполнении запроса.

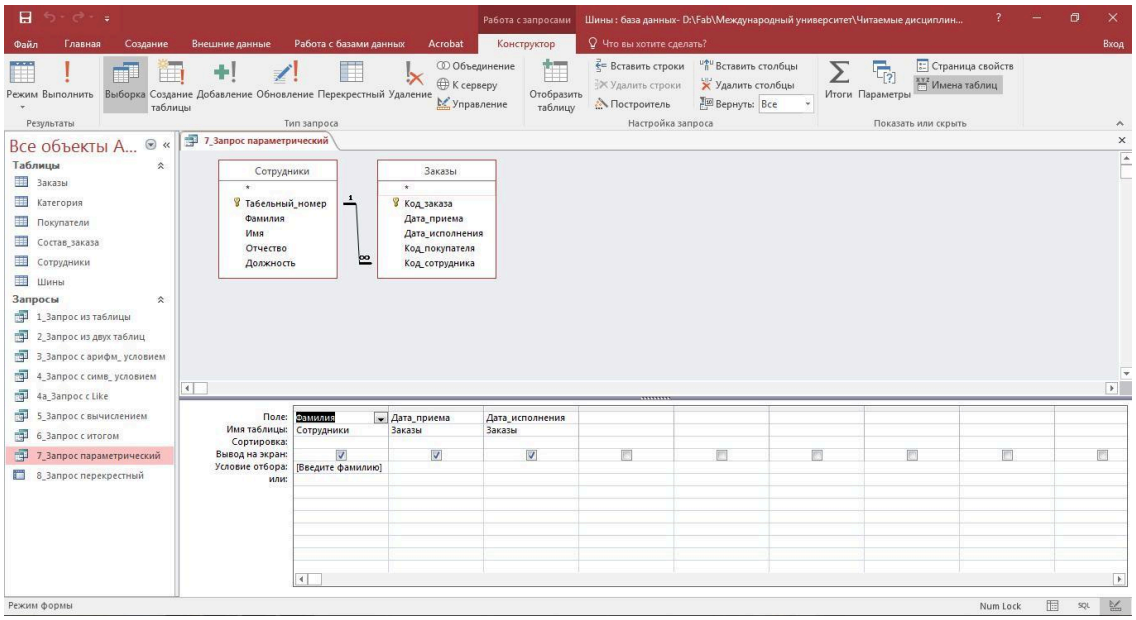


Рис. 4.

Чтобы Access проверял данные, вводимые в параметр, необходимо на вкладе ***Конструктор***выбрать команду ***Параметры***. Появляется диалоговое окно ***Параметры запроса***(рис. 5).

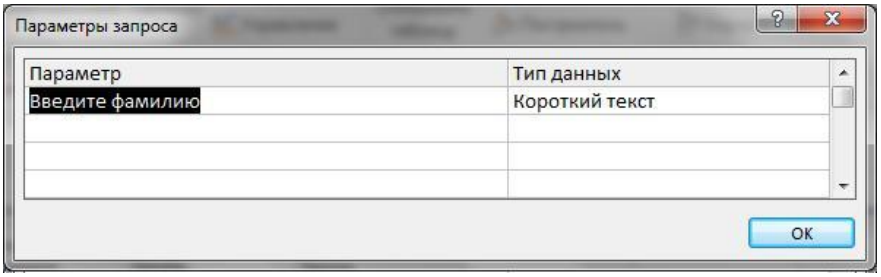


Рис. 5. Диалоговое окно ***Параметры запроса***

В столбец ***Параметр*** нужно ввести название параметра точно так, как он определен в бланке. В столбце ***Тип данных***надо выбрать из раскрывающегося списка необходимый тип данных. В одном запросе можно ввести несколько параметров. При выполнении такого запроса для каждого из параметров будут поочередно выводиться диалоговые окна ***Введите значение параметра***в том порядке, в котором параметры перечислены в бланке запроса.

При выполнении запроса появляется диалоговое окно ***Введите значение параметра***, в которое нужно ввести значение (рис. 6).

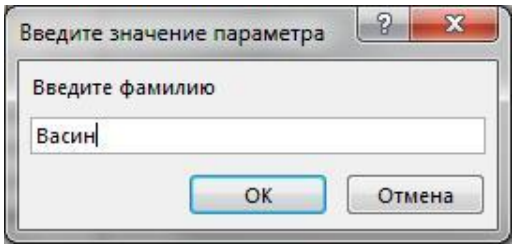


Рис. 6.

Результаты работы (рис. 7).

7_Запрос параметрический

Фамилия	Дата_приема	Дата_исполнения
Васин	02.02.2011	19.02.2011
Васин	14.02.2011	14.03.2011
Васин	01.03.2011	30.03.2011
Васин	19.03.2011	04.04.2011

Рис. 7.

5.6. **Перекрестный запрос**

Перекрестный запрос - это операция построения таблицы для вычисления итоговых значений на основе существующей таблицы или запроса. Перекрестный запрос создается в предположении, что исходная таблица содержит необходимые данные для формирования заголовков строк и столбцов новой таблицы. Если такая таблица отсутствует, то необходимо создать запрос, позволяющий объединить всю необходимую информацию.

Для иллюстрации формирования перекрестных запросов можно построить перекрестный запрос, показывающий как сотрудники отдела продаж продавали имеющиеся в наличии автомобильные шины.

Так как таблицы содержащей в себе всю необходимую информацию нет, её требуется создать, используя простой запрос на выборку (рис. 8).

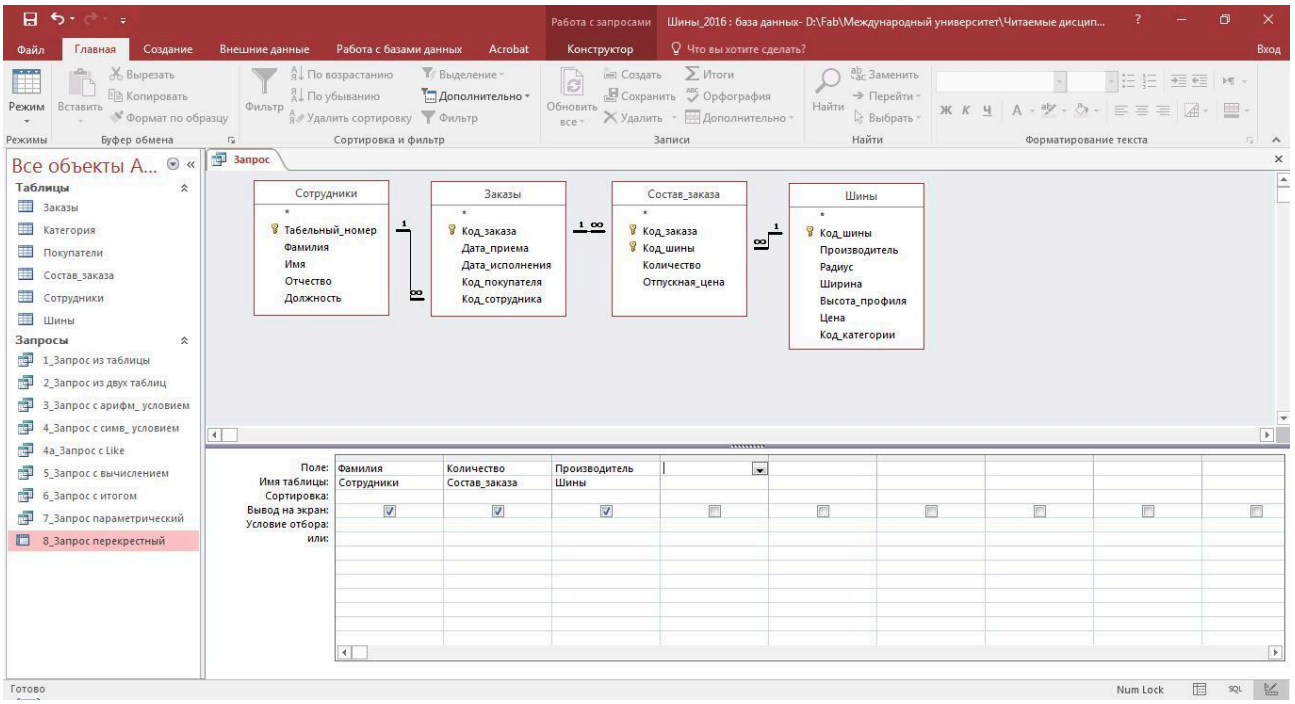


Рис.8. Формирование обобщающего запроса

В результате работы этого запроса формируется таблица, отвечающая всем требованиям, предъявляемым к исходной таблице для формирования перекрестного запроса (рис. 9).

Запрос

Фамилия	Количество	Производитель
Викторов	4	Pirelli
Васин	2	Continental
Сергеев	4	Nokian
Васин	4	Nokian
Васин	5	Michelin
Сергеев	5	Nokian
Сергеев	4	Michelin
Васин	5	Michelin
Викторов	5	Pirelli

Рис. 9. Результат работы созданного запроса

Для создания перекрестного запроса на базе созданного требуется выбрать пункт ***Перекрестный***на вкладке ***Конструктор*** врежиме***Работа с запросами*** (рис. 10).

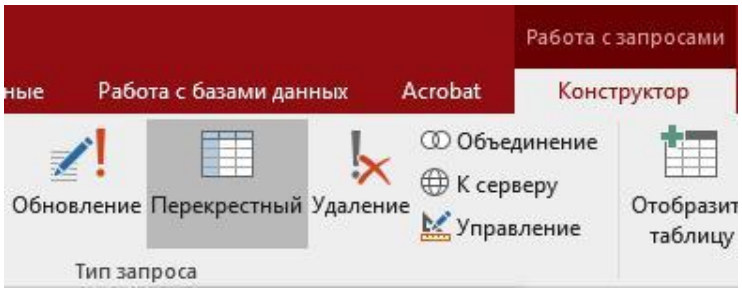


Рис.10.

В результате в свойствах таблиц появляется пункт ***Перекрестная таблица***(рис.11).

Поле:	Фамилия	Количество	Производитель
Имя таблицы:	Сотрудники	Состав_заказа	Шины
Групповая операция:	Группировка	Группировка	Группировка
Перекрестная таблица:			
Сортировка:			
Условие отбора:			
или:			

Рис.11.

Используя этот пункт формируем структуру перекрестной таблицы (рис.12).

Поле:	Фамилия	Количество	Производитель
Имя таблицы:	Сотрудники	Состав_заказа	Шины
Групповая операция:	Группировка	Группировка	Группировка
Перекрестная таблица:	Заголовки строк	Значение	Заголовки столбцов
Сортировка:		Заголовки строк	
Условие отбора:		Заголовки столбцов	
или:		Значение	
		(не отображается)	

Рис.12.

Учитывая, что необходимо аккумулировать сведения по всем договорам, в столбце ***Количество*** в пункте ***Групповая операция***требуется указать функцию суммирования (рис.13).

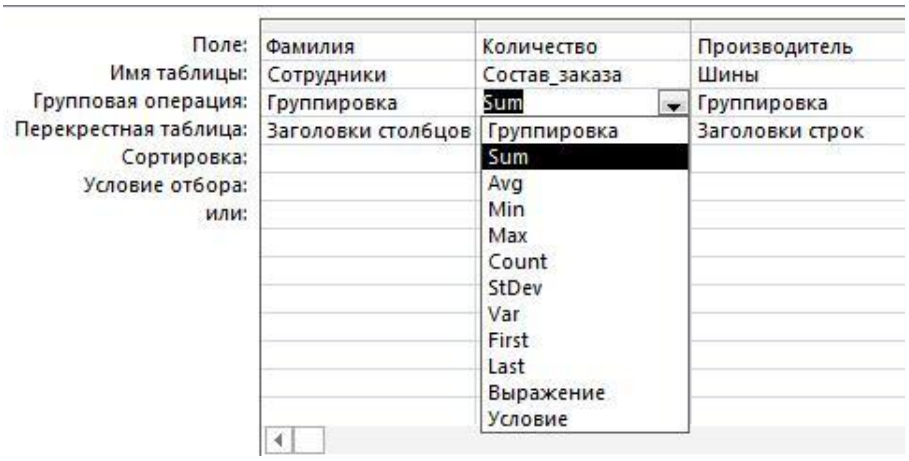


Рис.13.

Полученный перекрестный запрос показан на рис.14.

8_Запрос перекрестный

Производитель	Васин	Викторов	Сергеев
Continental	2		
Michelin	10		4
Nokian	4		9
Pirelli		9	

Рис.14. Результат работы перекрестного запроса.

Список литературы

1. Бекаревич, Ю. Б. Самоучитель Microsoft Access 2013 / Ю. Б. Бекаревич, Н. В. Пушкина. – СПб.: БХВ-Петербург, 2014. – 464 с.
2. Вишневецкий В.Ю., Старченко И.Б., Ледяева В.С. Работа с Microsoft Office 2016: Access, Visio. Методическое руководство к выполнению лабораторных работ по курсу «Информационные технологии». – Ростов–на–Дону: Изд–во ЮФУ, 2016. – 39 с
3. Андерсен В. Базы данных Microsoft Access. Проблемы и решения: Практ. пособ. /Пер. с англ. – М.: Издательство ЭКОМ, 2001. – 384 с

Александр Борисович Фролов

Система управления базами данных Access 2016. Конспект лекций по дисциплине «Информационные системы и базы данных» для студентов очно-заочного и заочного отделений. М.: АНОВО Московский международный университет, 2018, 70 с

Подписано в печать Заказ

Формат $60 \times 84\ ^1/_{16}$ Печ.л. Уч.-изд.л. Тираж

ММУ Москва Ленинградский проспект, 17.