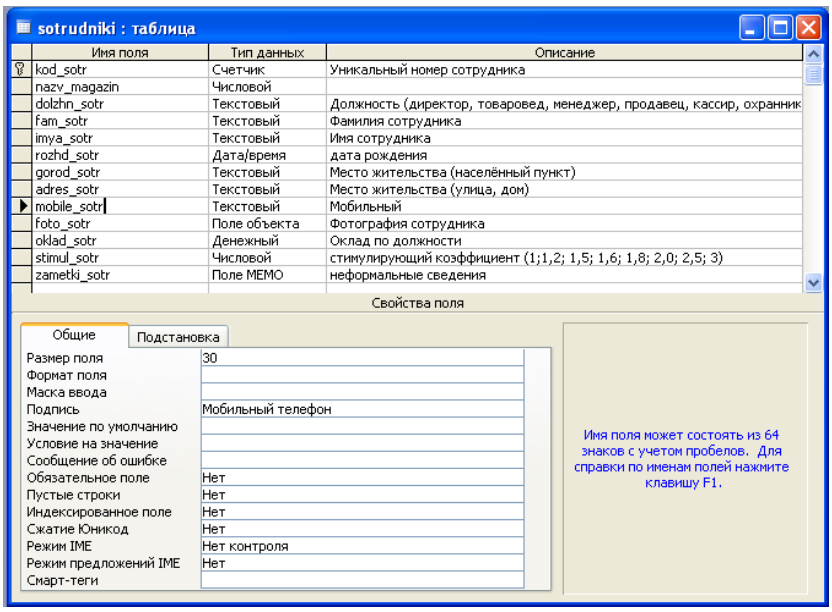
ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
Северо-Кавказский филиал
ордена Трудового Красного Знамени федерального государственного
бюджетного образовательного учреждения высшего образования
"Московский технический университет связи и информатики"

Методические указания
к практическим занятиям

ТЕХНОЛОГИИ БАЗ ДАННЫХ

Проектирование таблиц БД



Ростов-на-Дону
2019

УДК 681.3.06 (076)
ББК 32.07

Чикалов А.Н. Технологии баз данных. Проектирование таблиц БД. Методические указания к практическим занятиям. Ростов-на-Дону: Северо-Кавказский филиал МТУСИ, 2019.- 56 с.

В пособии изложены методические рекомендации, содержательные материалы и контрольные задания для проведения практических занятий по систематизации материала для базы данных, разработке логических моделей реляционных баз данных, созданию таблиц, импорту и экспорту данных. В качестве инструмента использована среда разработки MS Access. Пособие содержит необходимые справочные материалы.

Методические указания предназначены для студентов, обучающихся по направлению подготовки 09.03.01 Информатика и вычислительная техника, профилей Вычислительные машины, комплексы, системы и сети, Программное обеспечение и интеллектуальные системы.

Пособие предназначено для использования при изучении дисциплин Технологии баз данных, а также может быть использовано преподавателями и студентами при изучении родственных дисциплин и в процессе самостоятельной работы.

Учебное пособие обсуждено и одобрено на заседании кафедры ИВТ
Протокол №1 от 26.08.2019

Рецензент Зав. кафедрой ИВТ д.т.н. профессор Соколов С.В.

СОДЕРЖАНИЕ

1. ПРОЕКТИРОВАНИЕ БАЗ ДАННЫХ

Цель

1. Получить представление об этапах и принципах проектирования баз данных от постановки задачи до использования СУБД;
2. Разработать концептуальную модель БД;
3. Разработать логическую модель БД.

Учебные вопросы

1.1. Анализ предметной области;
1.2. Построение концептуальной модели БД;
1.3. Построение логической модели БД.

Литература для подготовки к занятию:

1. Иллюстрированный самоучитель по Access. http://www.selfteachers.ru
2. Вейскас Д. Эффективная работа с Microsoft Access/ - СПб.: Питер Ком, 1999. – 976с. стр.322-354.
3. Работа в Microsoft Access XP. http://www.intuit.ru
4. Уроки программирования. Базы данных. http://www.life-prog.ru
5. Иллюстрированный самоучитель по Access 2002. http://news.claw.ru/Office/Access_2002

Содержание отчета

1. Название работы;
2. Для каждого из заданий: название задания и материалы в объеме, указанном в задании.

Актуальность занятия

База данных – это организованная структура, предназначенная для хранения информации.

Данные и информация – понятия взаимосвязанные, но не тождественные. Сегодня большинство систем управления базами данных (СУБД) позволяют размещать в своих структурах не только данные, но и методы (то есть программный код), с помощью которых происходит взаимодействие с потребителем или с другими программно-аппаратными комплексами, обработка данных в рамках соответствующих задач. Таким образом, можно говорить, что в современных базах данных хранятся отнюдь не только данные, но и методы их обработки для получения соответствующей информация (получения данных, несущих ту или иную смысловую нагрузку).

Это утверждение легко пояснить, если, например, рассмотреть базу данных крупной организации. В ней есть все необходимые сведения: о сотрудниках, об их адресах, перемещении по служебной лестнице, состоянии расчетных счетов, финансовых операциях и т. д.

Доступ к этой базе имеется у достаточно большого количества сотрудников, но среди них вряд ли найдется такое лицо, которое имеет доступ ко всей базе полностью. Кроме данных, база содержит методы и средства, позволяющие каждому из сотрудников, оперировать только с теми данными, которые входят в его компетенцию. В результат взаимодействия данных, содержащихся в базе, с методами, доступными конкретным сотрудникам, образуется информация, которую они потребляют и на основании которой в пределах собственной компетенции производят ввод и редактирование данных.

С понятием базы данных тесно связано понятие **системы управления базой данных (СУБД)**. Это комплекс программных средств, предназначенных для создания структур новой базы, наполнения ее содержимым, редактирования содержимого и визуализации информации. Получение отображаемых данных в соответствии с заданным критерием, их упорядочение, оформление и последующая выдача на устройство вывода или передача по каналам сняли. В мире существует множество систем управления базами данных.

Несмотря на то, что они могут по-разному работать с разными объектами и предоставляют пользователю различные функции и средства, большинство СУБД опираются на единый устоявшийся комплекс основных понятий. Это дает возможность рассмотреть одну систему и обобщить ее понятия, приемы и методы на весь класс СУБД.

БД – это современные, эффективные и широко применяемые инструменты хранения, представления и обработки данных. Поэтому иметь первичные навыки создания и использования баз данных должен иметь каждый современный специалист независимо от сферы своей профессиональной деятельности.

Основные этапы разработки проекта

При разработке базы данных (БД) нельзя начинать ее построение с организации данных по строкам и столбцам. Такой непродуманный подход оправдан только в самых тривиальных ситуациях. Решение реальной задачи требует планирования, в противном случае придется без конца перестраивать свое приложение.

Конечно, реляционные базы данных позволяют легко вносить изменения. Однако гораздо рациональнее на начальном этапе не пожалеть времени на постановку задачи, описание структур данных, необходимых для решения задач, и определения взаимосвязей между различными задачами приложения. Это позволит сохранить значительное количество времени неизбежные на многочисленные переделки, исправления, добавления новых функций.

Существует обширная литература, описывающая этапы разработки базы данных. Они, как правило, отличаются друг от друга. Причина этого кроется в принципах расстановки акцентов, которые хочет подчеркнуть конкретный автор. Однако по своей сути они очень близки, и отражают в себе сочетание известных подходов проектирования информационных систем: сверху вниз и снизу вверх. Чаще всего эти подходы используются в комплек-

се, хотя об этом не сообщается в явном виде, этапы незаметно перетекают друг в друга. Кроме того, авторы непроизвольно учитывают особенности будущих инструментальных средств, которые только будут еще использованы при проектировании. Поэтому всякая последовательность этапов может быть поставлена под сомнение. Однако общий порядок все-таки может быть сформулирован в следующей последовательности:

1. Анализ предметной области;
2. Уточнение задач;
3. Анализ данных;
4. Определение последовательности выполнения задач;
5. Построение концептуальной модели предметной области;
6. Логическое проектирование БД;
7. Разработка макета приложения и пользовательского интерфейса;
8. Создание приложения;
9. Тестирование, отладка, усовершенствование БД.

1) Анализ предметной области

Каждая информационная система в зависимости от ее назначения имеет дело с частью реального мира, которую принято называть предметной областью (ПО) системы. ПО может относиться к любому типу организаций: банк, университет, завод, магазин и т.д.

Предметная область информационной системы - это совокупность реальных объектов (сущностей), которые представляют интерес для пользователей настоящей информационной системы. Иными словами, это часть реального мира, подлежащая изучению для организации управления и автоматизации. Все многообразие объектов реального мира рассматривать, естественно, невозможно.

Объект (сущность) - предмет, процесс или явление, о котором собирается информация, необходимая для решения задачи. Объектом может быть человек, предмет, событие, место.

Каждый объект характеризуется рядом основных свойств (атрибутов). Атрибутом называется поименованная характеристика объекта. Атрибуты показывают, какая информация должна быть собрана об объекте. Например, объект - клиент банка. Атрибуты объекта - номер счета, адрес, сумма вклада. При этом экземпляром сущности называют конкретный объект, с конкретными характеристиками. Например, объект – сотрудник, экземпляр сущности – Иванов Иван Тимофеевич, 1960г.р.

Первым этапом проектирования БД любого типа является анализ предметной области, который в перспективе заканчивается построением информационной структуры (концептуальной схемы). На данном этапе:

- анализируются требования пользователей к будущей БД, их информационные потребности;

- выбираются информационные объекты и выявляются связи между ними;

- уточняются их характеристики, которые предопределяют возможности проектируемой БД. Отсутствие необходимых характеристик в БД, естественно, исключает возможность получения необходимых сведений. Но наличие сведений, которые не используются и не планируются к использованию, просто перегружают БД, ухудшая ее технические характеристики, и необоснованно утомляют персонал.

На основе проведенного анализа структурируется предметная область. Анализ предметной области не зависит от программной и технической сред, в которых будет реализовываться БД.

Требования пользователей к разрабатываемой БД представляют собой список вопросов к БД с указанием их интенсивности и объемов данных. Эти сведения разработчики БД получают в диалоге с ее будущими пользователями. Здесь же выясняются требования к вводу, обновлению и корректировке информации. Можно говорить о том, что диалог осуществляется в свободной форме, и пользователи формулируют запросы в форме "хочу, чтобы БД …". Требования пользователей уточняются и дополняются при анализе имеющихся и перспективных задач.

Допустим, необходимо разработать базу данных, обеспечивающую накопление, сохранение и анализ данных для организации, осуществляющей подготовку и переподготовку кадров из различных организаций на договорной основе. Приложение должно позволять пользователям, имеющим соответствующий допуск, вводить и обновлять данные, получать обобщающие результаты, осуществлять поиск необходимых данных, распечатывать результаты запросов в необходимом виде.

Для данной предметной области вопросы пользователей могут быть, например, следующие:

1. Показать перечень обучавшихся организаций, сотрудников;
2. Показать количество обучавшихся организаций и сотрудников;
3. Каково количество человек, прошедших обучение, по годам, месяцам, специальностям, организациям?
4. У каких организаций или специалистов истекает срок очередной переподготовки?
5. Какие организации не завершили оформление договоров?
6. Осуществить поиск организации, сотрудника, адреса по имеющимся неполным сведениям и т.д.

Вторая фаза анализа предметной области состоит в выборе информационных объектов, задании необходимых свойств для каждого объекта, выявлении связей между объектами, определении ограничений, накладываемых на информационные объекты, типы связей между ними, характеристики информационных объектов.

При выборе информационных объектов нужно постараться ответить на ряд вопросов:

1. На какие классы можно разбить данные, подлежащие хранению в БД?
2. Какое имя можно присвоить каждому классу данных?
3. Какие наиболее интересные характеристики (с точки зрения пользователя) каждого класса данных можно выделить?
4. Какие имена можно присвоить выбранным наборам характеристик?

В рассматриваемом примере наиболее существенными объектами могут быть: обучающиеся сотрудники, организации, в которых они работают, специальности, по которым они обучаются, оформляемые договора, протоколы о сдаче итогового экзамена и т.д.

Атрибутами для обучающихся сотрудников являются: фамилия, имя, отчество, дата рождения, серия и дата выдачи паспорта, контактный телефон, должность и т.д.

Атрибутами организации – название, город, улица, номер дома и номер офиса размещения, рабочий телефон, координаты директора и т.д.

Атрибутами специальности – название специальности, объем программы в часах, дата и фамилия утвердившего программу, стоимость обучения, начало использования программы, характеристики программы.

Для выявления связи между информационными объектами следует ответить на следующие вопросы:

1. Какие типы связей существуют между объектами? Существуют ли ограничения на характеристики объектов?
2. Какое имя можно присвоить каждому типу связей?
3. Какие типы связей могут быть использованы в будущем?
4. Имеют ли смысл какие-нибудь комбинации типов связей?

Ограничениями являются некоторые логические границы, которые можно накладывать на конкретные данные (характеристики) или объекты. Система ограничений образует понятие целостности данных, характеризующее неизменность таких ограничений при каждом значении свойств. При наличии связей, они также включаются в это понятие и определяют правила изменения и удаления данных, объединенных этими связями. В целом целостность обеспечивает неизбыточность, непротиворечивость и адекватность данных, позволяет предотвратить ошибки ввода и случайные потери данных. Кроме того, ограничения на данные могут помочь спрогнозировать размеры создаваемых объектов и ограничить объемы выделяемой памяти.

Так, например, количество различных специальностей не может быть больше десяти, длина фамилии, имени и отчества не может превышать тридцати символов каждая, дата рождения не может соответствовать возрасту более ста лет и т.д.

Связями называют соответствия, отношения, возникающие между объектами предметной области. Естественно, все объекты связаны между собой

множеством связей. Однако существует устоявшаяся система формализованных связей, которые можно будет реализовать с помощью системы управления базой данных (СУБД): один к одному, один ко многим, многие ко многим.

Связь один к одному (1:1) предполагает, что в каждый момент времени одному экземпляру информационного объекта А соответствует не более одного экземпляра информационного объекта В и наоборот. Например, конкретный обучаемый обычно сдает один квалификационный экзамен и получает соответствующий рейтинговый результат.

Связь один ко многим (1:∞) характеризуется тем, что одному экземпляру информационного объекта А соответствует 0, 1 или более экземпляров объекта В, но каждый экземпляр объекта В связан не более, чем с 1 экземпляром объекта А. Например в одной организации работают один или много сотрудников, по одной теме обучалось много сотрудников, и т.д.

Связь многие ко многим (∞:∞) характеризуется тем, что одному экземпляру информационного объекта А соответствует 0, 1 или более экземпляров объекта В и наоборот. Например, одна и та же специальность может присутствовать во многих договорах на обучение. Вместе с тем в одном договоре может присутствовать заявка на обучение по многим специальностям.

2) Уточнение задач

После сбора первичных сведений следует переходить к составлению списка задач, которые должно выполнять приложение БД. Здесь вполне могут присутствовать и перспективные задачи, которые не актуальны сегодня, но могут потребоваться в будущем. Все задачи будут представлены в формах и отчетах приложения. Поэтому часть из этих задач станут основными и будут решаться с применением некоторой формы, созданной для реализации этой задачи (например, для ввода данных об обучаемых). Другие задачи, вероятно, станут подзадачами для этой задачи и будут выполняться в той же форме или в подчиненной форме (например, подсчета общего количества обучаемых, поиска обучаемых и т.д.).

Для разрабатываемой БД список задач может быть следующим:

1. Ввод данных о специальностях;
2. Ввод данных об обучаемых;
3. Ввод сведений об организации, заключающей договор на обучение;
4. Ввод данных об оформленном договоре на обучение;
5. Ввод сведений об итоговом протоколе;
6. Ввод сведений о сдаче квалификационного экзамена;
7. Поиск организаций;
8. Поиск сотрудников;
9. Поиск организаций с истекающим сроком переподготовки;
10. Печать документов о прохождении подготовки и повышении квалификации.

3) Анализ данных

После формирования списка задач наиболее важным этапом является составление подробного перечня данных, необходимых для решения каждой задачи. При этом определяется название данных, формат данных, краткое описание. Для каждой задачи данные могут иметь свои признаки:

- входные или исходные - данные, которые меняться не будут. Они используются для расчетов необходимых сведений или извлекаются из БД без изменения. Например, фамилия, имя, возраст и т.д.;
- выходные - данные, которые вводятся в этой задаче, возможно изменяются в ходе решения задачи на основе входных данных, а затем сохраняются. Например, в протокол вносятся фамилия обучаемого, организация, специальность, рейтинговая оценка;
- удаляемые - данные, удаляемые в ходе решения задачи;
- изменяемые - данные, добавляемые в ходе решения задач или изменяемые и записываемые заново. Например, заменяемый на новый адрес организации;
- вычисляемые - данные вычисляемые, которые рассчитываются, отображаются, печатаются, но в БД не сохраняются.

Например, для задачи ввода данных об обучаемых требуются следующие основные данные:

Элемент данных	Признак	Описание	Объект
Fam	Вых	Фамилия обучаемого	Klient
Imja	Вых	Имя обучаемого	Klient
Otch	Вых	Отчество обучаемого	Klient
Date	Вых	Дата рождения	Klient
Pas_Serija	Вых	Серия и номер паспорта	Klient
Pas_Vyd	Вых	Кем выдан паспорт	Klient
Pas_Data	Вых	Дата выдачи паспорта	Klient
Kl_Tel	Вых	Контактный телефон	Klient
Kl_Dolgn	Вых	Должность обучаемого	Klient
Kl_Org	Входные	Место работы обучаемого	ORG
Kl_Prim	Вых	Комментарии	Klient

4) Определение последовательности выполнения задач

Чтобы приложение работало логично и удобно, лучше всего объединить основные задачи в тематические группы и затем упорядочить задачи каждой группы так, чтобы они располагались в порядке их выполнения. Например, следует отделить задачи, имеющие отношение к кадрам, от задач, связанных с финансовыми процедурами. Группа задач, связанных с оформ-

лением итоговых документов об обучении, должна реализовываться после ввода данных о договорах, протоколах и экзаменах. Группировка задач и графическое представление последовательности их выполнения помогут определить естественный порядок следования задач, который затем можно отразить во взаимных связях форм и отчетов в приложении.

Если некоторый элемент данных указан в качестве входной информации, то должна существовать предшествующая задача, для которой этот элемент является выходным.

Так, например, для объекта Klient, содержащего сведения об обучаемых существуют сведения об организации, в которой он работает. Эти данные должны уже существовать к моменту ввода данных об обучаемых. Аналогичным образом связаны данные и других объектов. Все определенные для разрабатываемого приложения задачи полезно представить в виде диаграммы, отражающей порядок их реализации. Часть задач может иметь признаки выполняемых при желании или потенциально возможных. Они на диаграмме связаны пунктирными линиями. В частности, поиск обучаемого в имеющемся списке может осуществляться при отработке очередного договора, могут данные об обучаемом вводиться как первый раз, возможен вариант поиска обучаемого для получения справочных данных о нем. Диаграмма задач для разрабатываемой БД показана на рисунке 1.1.

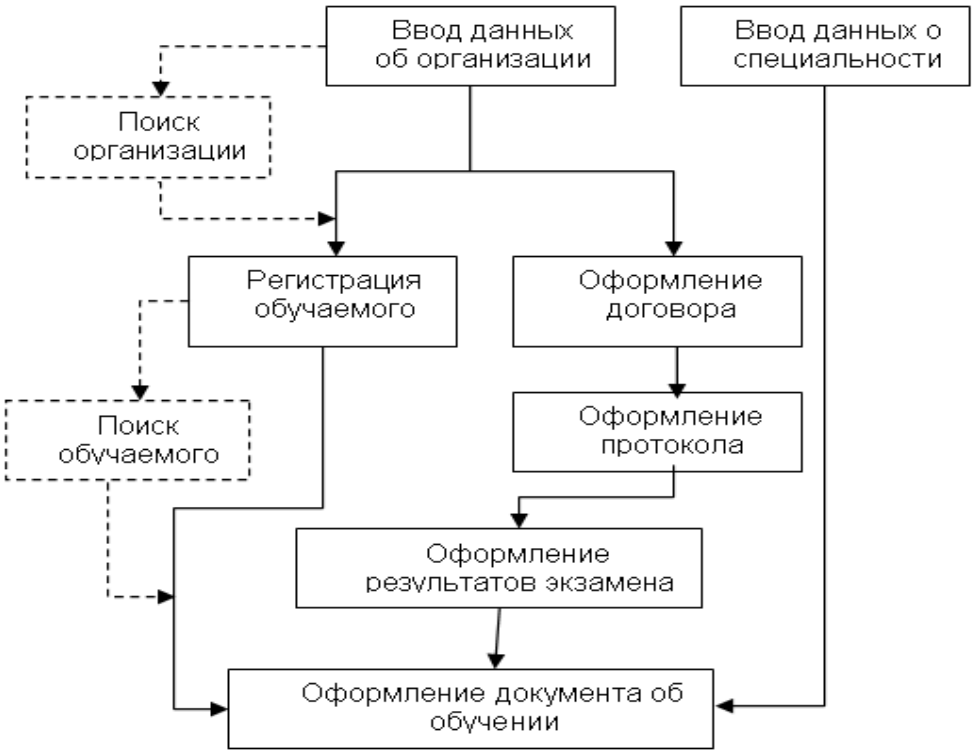


Рис.1.1. Диаграмма задач БД

5) Построение концептуальной модели предметной области

Заключительная фаза анализа предметной области состоит в проектировании её информационной структуры или концептуальной модели. Концептуальная модель включает описания объектов и их взаимосвязей, представляющих интерес в рассматриваемой предметной области (ПО) и выявляемых в результате анализа данных.

Концептуальная модель применяется для структурирования предметной области с учетом информационных интересов пользователей системы. Она дает возможность систематизировать информационное содержание предметной области, сгруппировать элементы данных в объекты, которые впоследствии станут основой для создания таблиц в проектируемой БД. Этот этап еще не привязан к программному обеспечению.

Концептуальная модель является представлением точки зрения пользователя на предметную область и не зависит ни от программного обеспечения СУБД, ни от технических решений.

Концептуальная модель должна быть стабильной. Могут меняться прикладные программы, обрабатывающие данные, может меняться организация их физического хранения, концептуальная модель остается неизменной или увеличивается с целью включения дополнительных данных.

Одной из распространенных моделей концептуальной схемы является модель "сущность – связь". Основными конструкциями данной модели являются сущности и связи. Аналогом сущности является объект, имеющий, как и сущность, атрибуты и ключевые атрибуты, делающие объект уникальным. Связь определяет отношения между сущностями (объектами). Связи имеют конкретное содержание (отражаемое в названии) и характеризуются типом: "один к одному", "один ко многим" и "многие ко многим".

Межу сущностями (объектами) в рассматриваемом примере можно определить следующие связи

Название связи	Связанные сущности
Работает	Client - Org
Заключают	ORG - Dogovor
Отражает выполнение	Dogovor - Protocol
Включает сотрудников	Protocol - ORG
Экзамен	Protocol - Client
Прошел обучение	Protocol - Tema
Изучает	Client - Tema

В виде графической диаграммы такая модель показана на рисунке 1.2.

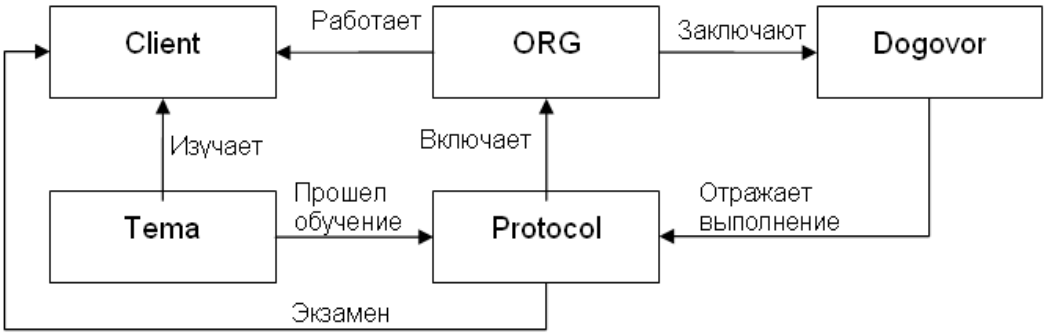


Рис.1.2. Типы связей между объектами БД

6) Логическое проектирование БД

После предварительного анализа всех необходимых для приложения данных необходимо упорядочить их по объектам и соотнести объекты с таблицами и запросами БД. Здесь же осуществляется нормализация, после которой разрабатывается логическая модель данных, соответствующая реляционной модели БД.

При накоплении данных без анализа задач и выделения самостоятельных объектов всегда присутствуют типичные недостатки:

- многократное сохранение одних и тех же данных. Например, для занесения данных о нескольких обучаемых требуется указывать для каждого из них подробные данные о его организации, данные протокола, специальности и т.д.;

- одна и та же организация может обучать сотрудников несколько раз. Заранее неизвестно, сколько будет заключено договоров, поэтому требуется зарезервировать максимальное количество областей для их фиксации, но это опять неэффективное расходование памяти. Кроме того, если появится хотя бы одна организация, которой не хватит и этого максимума, то придется переделывать весь проект;

- часть сведений возможно совсем не хранить в БД, а вычислять на момент распечатки отчета. Например, количество обучаемых, включенных в конкретный протокол;

- наличие в таблице поля, содержащего несколько элементов данных, значительно затрудняет поиск и сортировку по этим элементам. Например, адрес без разделения, города, улицы, номера дома и офиса, не позволяет отобрать адреса по конкретному городу.

Для преодоления названных недостатков используется процесс, называемый нормализацией. Он заключается в проведении анализа результатов проектирования по ряду правил и принятии мер по устранению выявленных нарушений этих правил.

Правило №1. Уникальность полей: каждое поле таблицы должно представлять уникальный тип информации. Это означает, во-первых, отсутствие повторяющихся по смыслу полей в одной записи а, во-вторых, разделение составных полей на несколько отдельных элементов данных. В соответствии с этим правилом не должно быть данных о нескольких договорах в одной записи, а адрес должен быть разделен на отдельные составляющие. В таблицы, созданные для повторяющихся данных, следует включить ключевые данные из основной таблицы, чтобы обеспечить связь основной и новыми таблицами.

Новые таблицы получаются проще, поскольку на каждую организацию требуется только одна запись. Для поиска организации достаточно просмотреть только таблицу, в которой каждая организация представлена только один раз.

Правило №2. Первичные ключи: каждая таблица должна иметь уникальный идентификатор. Он может состоять из одного или нескольких естественных полей, которые однозначно идентифицируют каждую запись. Такая совокупность полей называют первичным ключом (или просто ключом).

Ключи необходимы для организации связей. Если они занимают достаточно много места, то использование таких ключей не снимает проблему дублирования данных, потому что они должны быть включены в связываемые таблицы. Поэтому для сокращения размера данных применяют в качестве ключей малоразмерные естественные цифровые коды, за уникальностью которых следит сама СУБД автоматически. Если в качестве ключа использовать название организации, то ключ займет 30-100 символов, причем технически не удастся сделать длину ключа изменяемой. Для цифрового поля длина его будет примерно пять цифр.

Так, для связи таблицы DOGOVOR и таблицы с данными организаций ORG используют ключевое поле Kod_ORG, которое размещается в таблице DOGOVOR (в этой таблице оно не является ключевым). Связь по типу получается один ко многим (рис.1.3).

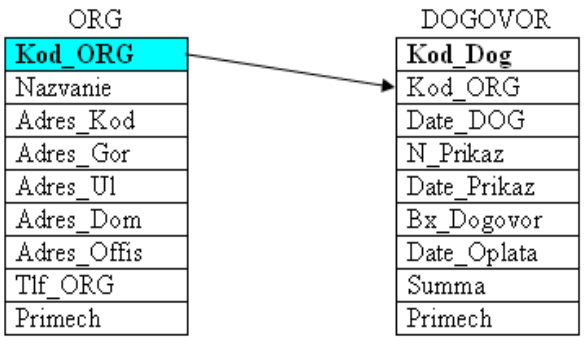


Рис.1.3. Связь таблицы ORG с таблицей DOGOVOR

Правило №3. Функциональная зависимость. Для каждого значения первичного ключа значения в столбцах данных таблицы должны относиться к объекту таблицы и полностью его описывать. Другими словами, каждое поле записи функционально зависит от этого ключа: в таблице отсутствуют данные, не относящиеся к объекту, а данные максимально полные. Так, например, в каж-

дом договоре требуются сведения об обучаемых, но данные о них должны находиться в отдельной таблице. Аналогично, договор должен содержать сведения об организации, которая его заключает, но сведения о ней образуют свою отдельную таблицу, в которой имеются все сведения об организации (см. рис.1.3). С другой стороны, включение в таблицу поля даты оплаты Date_Oplata делает ее более полной, т.к. проведение оплаты не всегда совпадает по времени с датой заключения договора.

Правило №4. Независимость полей. Должна быть возможность изменения значений любого поля кроме ключевого без воздействия на данные других полей.

Например, в таблице ORG возможно изменение телефона организации. Он включен в соответствии с правилами №2 и №3 в таблицу организации ORG. Если для полного описания договора эти данные были бы включены в таблицу договора DOGOVOR, то потребовалось бы заменить номер телефона во всех записях заключенных договоров. Аналогичная ситуация получится и с названием организации в случае ошибки при вводе. А если Неверно указана сама организация, то придется переписывать все ее координаты. Этого не надо делать, если организации описаны в отдельной таблице. Достаточно изменить просто ссылку на организацию (поле Kod_ORG) в таблице DOGOVOR.

Одним из способов проверки независимости полей является просмотр записей на наличие повторяющихся данных. Для внесения данных о договоре требуется сохранить данные об обучаемых, указанных в этом договоре. Поэтому с одним номером договора требуется внести много записей с разными обучаемыми. Значительно проще в договоре указать только организацию, а обучаемых внести в отдельную таблицу CLIENT. В этом случае связи будут организованы так, как показано на рис.1.4.

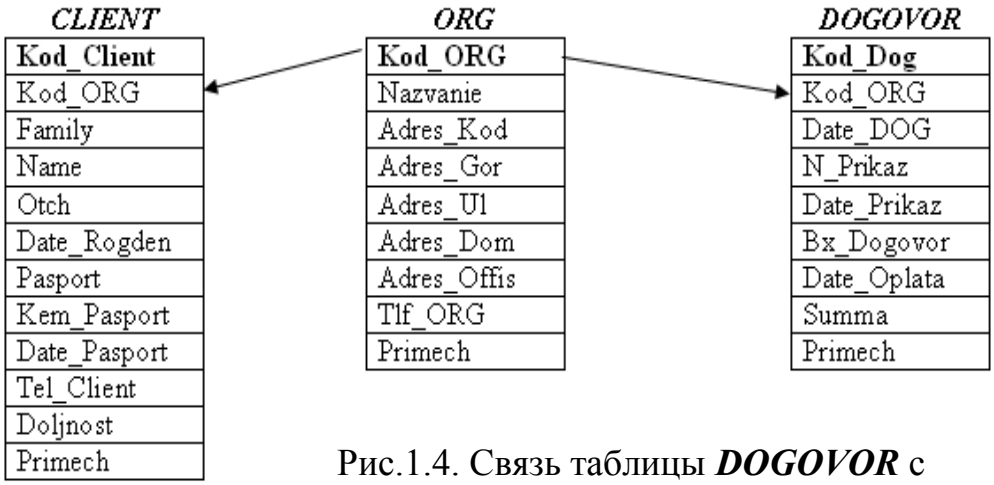


Рис.1.4. Связь таблицы ***DOGOVOR*** с таблицей ***CLIEN*T**

В результате нормализации получается Множество отдельных таблиц. В хорошо спроектированной БД использование чужих ключей для связи обеспечивает эффективность работы приложения. Для ускорения поиска соз-

дают также индексы. Индекс – это внутренняя таблица, состоящая из двух столбцов: значения выражения, содержащего все поля, включенные в Индекс и, и местоположение каждой записи таблицы с данными значениями индексного выражения. .СУБД обеспечивает кроме всего целостность данных.

Несмотря на то, что многие отношения между объектами существуют по схеме "многие ко многим", СУБД, как правило, не позволяют задать их напрямую. В этом случае создаются дополнительные таблицы пересечения, с помощью которой одна связь "многие ко многим" будет сведена к двум связям типа "один ко многим". Такое замещение создается при разбиении связи между таблицами ***DOGOVOR*** и ***CLIENT.*** Один сотрудник может обучаться несколько раз, поэтому должен быть включен в несколько договоров. Наоборот один договор объединяет насколько обучающихся сотрудников. Поэтому связь имеет тип "многие ко многим". Но организация между ними таблицы (таблицы пересечения) ***ORG*** позволяет свести все к двум связям типа "один ко многим".

Таблицы для рассматриваемой организации по переподготовке кадров показаны на рис.1.5.

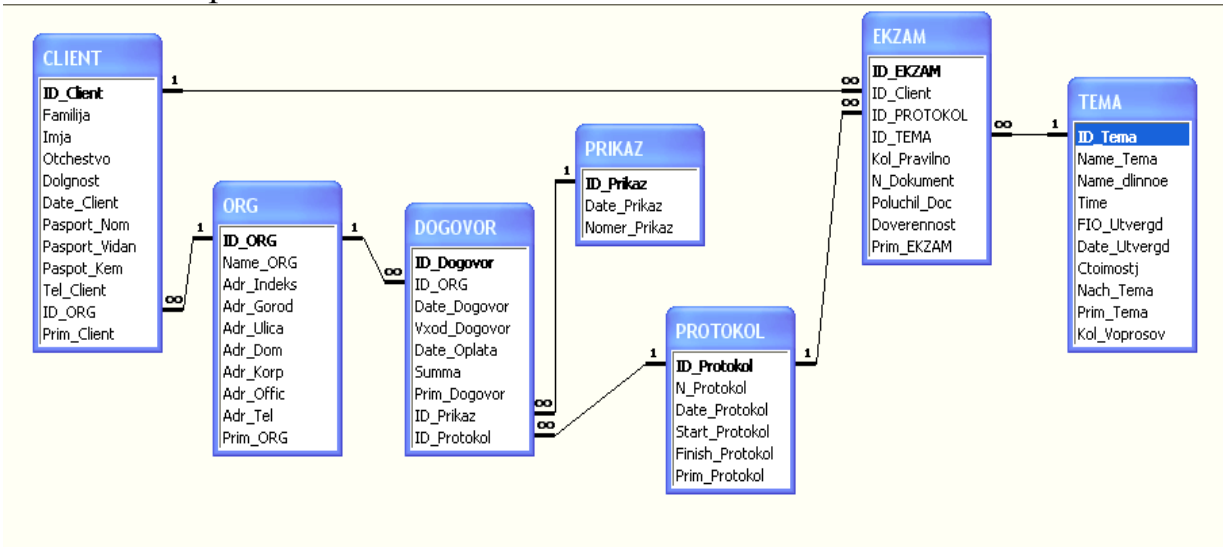


Рис.1.5. Схема данных БД

7) Разработка макета приложения и пользовательского интерфейса

Задав структуры таблиц, в Access необходимо создать макет приложения с помощью форм и связать их между собой, используя несложные макросы или процедуры обработки событий VBA. При этом появляется возможность просмотра на мониторе достаточно реальных форм и отчетов, переключаясь из режима формы в режим просмотра. Такие приемы позволяют достаточно объективно оценить результат разработки приложения еще до детальной реализации задач приложения.

8) Создание приложения

В случае относительно простых задач созданный макет является практически законченным приложением. Однако довольно часто приходится писать процедуры, позволяющие полностью автоматизировать решения требуемых задач. Это требует специальных связующих форм для транзита между задачами. Это могут быть, например, кнопочные формы, выполняющие функцию диспетчера, окна диалога для ввода параметров для конкретной задачи, специальные меню.

9) Тестирование, отладка, усовершенствование БД

После завершения работ по отдельным компонентам приложения необходимо проверить функционирование приложения в каждом из возможных режимов. Для этого существуют различные режимы отладки, в том числе пошаговые, позволяющие проверить работу приложения, выявить и исправить ошибки.

Отладка приложения, и тем более усовершенствование, обычно, не заканчивается после передачи приложения в эксплуатацию. Источником необходимых доработок являются как пользователи приложения, уточняющие и переценивающие существующие возможности, так и сами разработчики, выявляющие более удачные технические решения. Этот процесс требует повторения соответствующих этапов проектирования.

Задание 1.1. Анализ предметной области

В отчете представить:
1. Вопросы пользователей к БД;
2. Состав информационных объектов и их свойств (полей);
3. Состав связей между объектами;
4. Состав задач БД.

Задание 1.2. Построение концептуальной модели БД

В отчете представить:
1. Диаграмму задач БД;
2. Типы связей в виде схемы, аналогичной рис.1.2;

Задание 1.3. Построение логической модели БД

В отчете представить:
1. Результаты нормализации данных в виде итоговых таблиц;
2. Схему данных реляционной БД.

Вопросы для самопроверки
1. Каковы этапы разработки проекта?

2. С какой целью проводится анализ предметной области?
3. Что является результатом анализа данных?
4. Для чего выявляется последовательность выполняемых задач?
5. Что называется концептуальной моделью данных?
6. Какие категории образуют концептуальную модель данных?
7. Каковы задачи логического проектирования?
8. В чем отличие концептуальной и логической модели?
9. Какие типы связей предполагаются в БД?
10. В чем суть метода нормальных форм при проектировании БД?
11. Сформулируйте суть трех первых нормальных форм.

2. РАЗРАБОТКА ТАБЛИЦ БАЗЫ ДАННЫХ

Цель

1. Выработать первичные навыки создания и использования Системы управления базами данных (СУБД);

2. Получить навыки работы с основными элементами СУБД при создании и редактировании таблиц БД.

Учебные вопросы

2.1. Создание таблицы с помощью конструктора;
2.2. Ввод и редактирование данных в таблице БД;
2.3. Организация связей между таблицами БД

Литература

Литература аналогична той, которая указана в разделе 1.

Задание 2.1. Создание таблицы с помощью конструктора

Задание выполняется в соответствии с выданным индивидуальным вариантом.

Рабочий вариант задания имеет следующий вид.

1. Разработать таблицу, выбрать тип для следующих данных. Имя полей сделать по образцу. Дать название таблице - ***Sotrudniki:***

Имя поля	Назначение поля
Kod_sotr	Уникальный номер сотрудника
Dolzhn_sotr	Должность (из списка: директор, товаровед, менеджер, продавец, кассир, охранник)
Fam_sotr	Фамилия сотрудника
Imya_sotr	Имя сотрудника
Gorod_sotr	Место жительства (населенный пункт)
Adres_sotr	Место жительства (улица, дом)
Mobile_sotr	Мобильный телефон
Foto_sotr	Фотография сотрудника
Oklad_sotr	Оклад по должности
Stimul_sotr	Стимулирующий коэффициент (из списка: 1; 1,2; 1,5; 1,6; 1,8; 2,0; 2,5; 3)
Zametki_sotr	Неформальные сведения (примечания)

2. Разработать таблицу, выбрать тип для следующих данных. Имя полей сделать по образцу. Дать название таблице - ***Mag***:

Имя поля	Назначение поля
Kod_magazin	Уникальный номер магазина
Gorod_magazin	Город размещения магазина

Vid_magazin	Из списка: РиК_Фото, РиК_Комп, РиК_Мобил, Сопутств.
Imya_ magazin	Название магазина
Adres_magazin	Адрес магазина
Arenda_ magazin	Стоимость аренды (месяц)
Kommun_ magazin	Стоимость коммунальных платежей (месяц)
Otkr_ magazin	Время открытия магазина
Zakr_ magazin	Время закрытия магазина
Rabota_ magazin	Работает ли магазин (Да/Нет)
Primech_ magazin	Дополнительная информация (например, почему не работает)

В отчете представить:
1. Форматы таблиц с именами, именами полей, типами данных;
2. Файл с разработанными таблицами.

Загрузка MS Access

Для загрузки MS Access необходимо последовательно выполнить: ***Пуск – Программы – Microsoft Office – Microsoft Access*** (рис. 2.1).

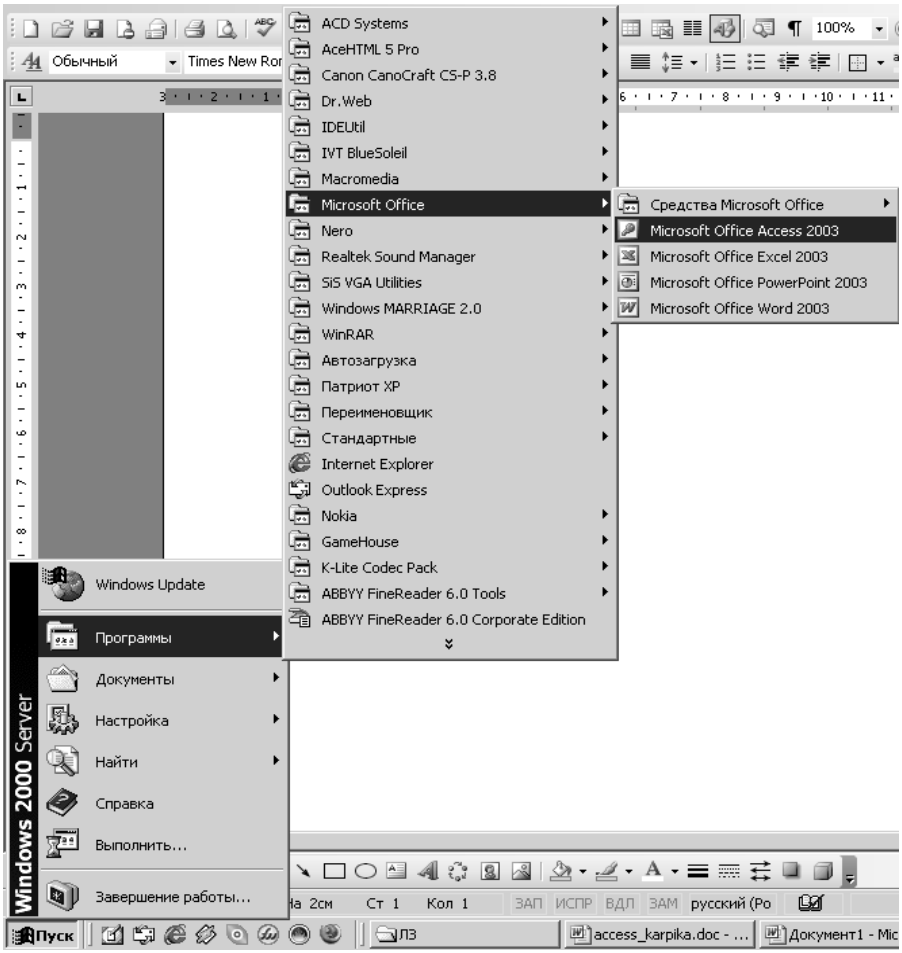


Рис.2.1. Запуск MS Access

В результате на экране последовательно появляются окно создания новой, или изменения существующей БД (рис.2.2 слева) и окно контейнера БД (рис.2.2 справа).

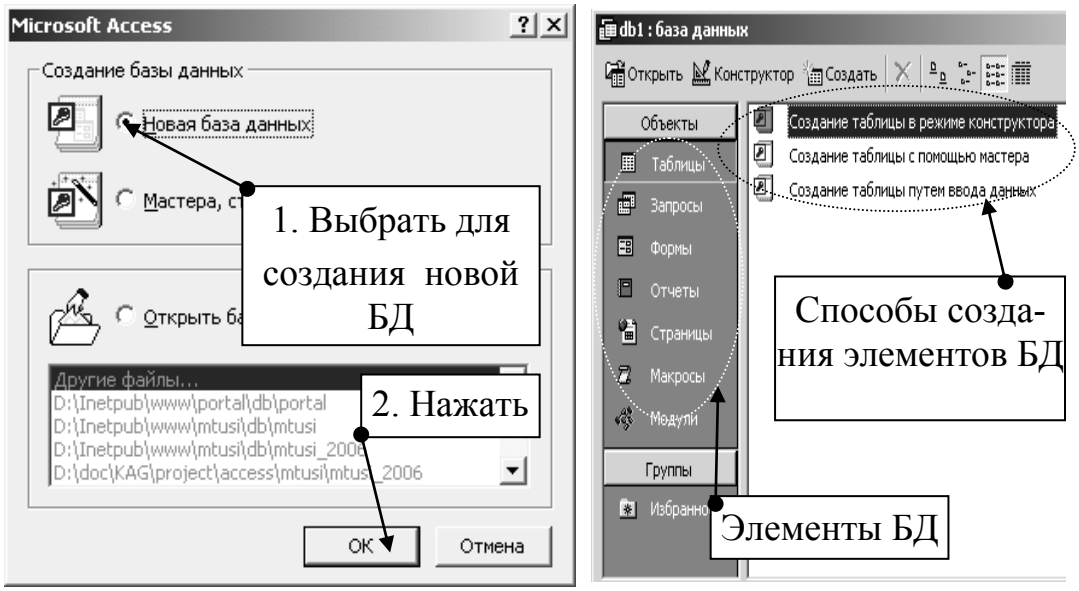


Рис.2.2. Окно создания БД и Контейнера БД

Назначение объектов БД

В окне контейнера БД выбираются элементы БД, имеющие следующие назначения:

1. **Таблицы** – физические объекты, структуры, предназначенные для хранения данных;

2**. Запросы** – средство выборки данных по определяемым пользователем условиям (фактически представляют собой конструкции на языке SQL Structural Query Language – Структурированный язык запросов);

3**. Формы** – программные конструкции, организующие интерфейс между таблицами и пользователем;

4. **Отчёты** – средство представления данных, хранящихся в таблицах и выбираемых запросами в виде, пригодном для вывода на печать (документирования);

5. **Страницы** – средство представления данных в виде, пригодном для публикации в Интернет;

6. **Макросы** – простейшее средство автоматизации работы с БД, не требующее знаний программирования;

7. **Модули –** средство, предназначенное для написания приложений для БД. Требует навыков программирования в среде VBA (Visual Basic for Application).

Создание спецификации таблицы БД

В таблицах БД хранятся данные. В этом заключается их основное назначение. При этом необходимо различать процесс создания таблицы (по аналогии с таблицей на листе – это заготовка шапки таблицы, в которой обозначены колонки, подобран их размер для будущих данных, но самих данных еще нет) и процесс заполнения и редактирования таблицы. Первый этап еще называют разработкой спецификации (описания) таблицы.

Создание таблицы можно выполнить тремя способами:

1. С помощью *конструктора* (наиболее гибкий и правильный с точки зрения полноты контроля со стороны пользователя способ);

2. С помощью *мастера* (пользователь отвечает на вопросы мастера и создаёт таблицу на основе типовых вариантов, но, как правило, малопригодную для его конкретных нужд, т.к. она обязательно потребует доработки в режиме конструктора);

3. Путём *ввода данных* (в этом режиме СУБД Access анализирует вводимые пользователем данные и пытается определить тип полей самостоятельно, но делает это достаточно "грубо").

В дальнейшем вся работа с БД будет осуществляться через окно контейнера БД. Отсюда осуществляется доступ ко всем объектам путем выбора соответствующей закладки и нажатия требуемой кнопки (Открыть, Конструктор, Создать).

Создание таблицы с помощью конструктора

Пусть требуется создать таблицу учёта успеваемости курсантов 111 курса. При этом будут интересовать следующие данные:

- *код_курсанта* – уникальный идентификатор (Счётчик, ключевое поле);
- *номер_уч_гр* (числовая величина);
- *воин_зван* (текстовая величина);
- *фио* (текстовая величина);
- *дата_рожд* (дата/время);
- *оц_физ* (числовая величина);
- *оц_мат* (числовая величина).

После нажатия кнопки ***Создать*** на закладке ***Таблицы*** окна контейнера БД, будет предоставлена возможность выбора одного из пяти вариантов действий. После выбора режима ***Конструктор*** и нажатия кнопки ***Ок*** появится окно конструктора, в котором необходимо самостоятельно создавать поля, выбирать типы данных для полей и размеры полей. Каждая строка спецификации определяет характеристики одного поля записи.

Для создания таблицы необходимо выполнить несколько действий:

- создать поля;
- выбрать тип данных для каждого поля;
- определить ключевое поле (поля);
- сохранить таблицу с уникальным именем.

Пример: Создание таблицы 111_kurs в режиме конструктора

В контейнере базы данных (рис.2.2) выбрать пункт ***Создание таблицы в режиме конструктора.*** В открывшемся окне конструктора таблицы создать поля, определённые выше (рис.2.3).

В первой колонке необходимо ввести название (имя) поля. О правилах записи имен полей в Access можно узнать, нажав клавишу F1. После набора имени поля нажать клавишу ***Enter*** или установить курсор на вторую колонку в той же строке и щелкнуть мышью.

Во второй колонке нужно выбрать тип данных из списка, который появляется в выпадающем меню после нажатия на кнопку со стрелкой на правой границе колонки ***Тип данных***. Для каждого типа поля в левой нижней части экрана высвечивается свой набор свойств.

В третьей колонке можно ввести описание поля. Вся эта информация является справочной для разработчика и БД не используется.

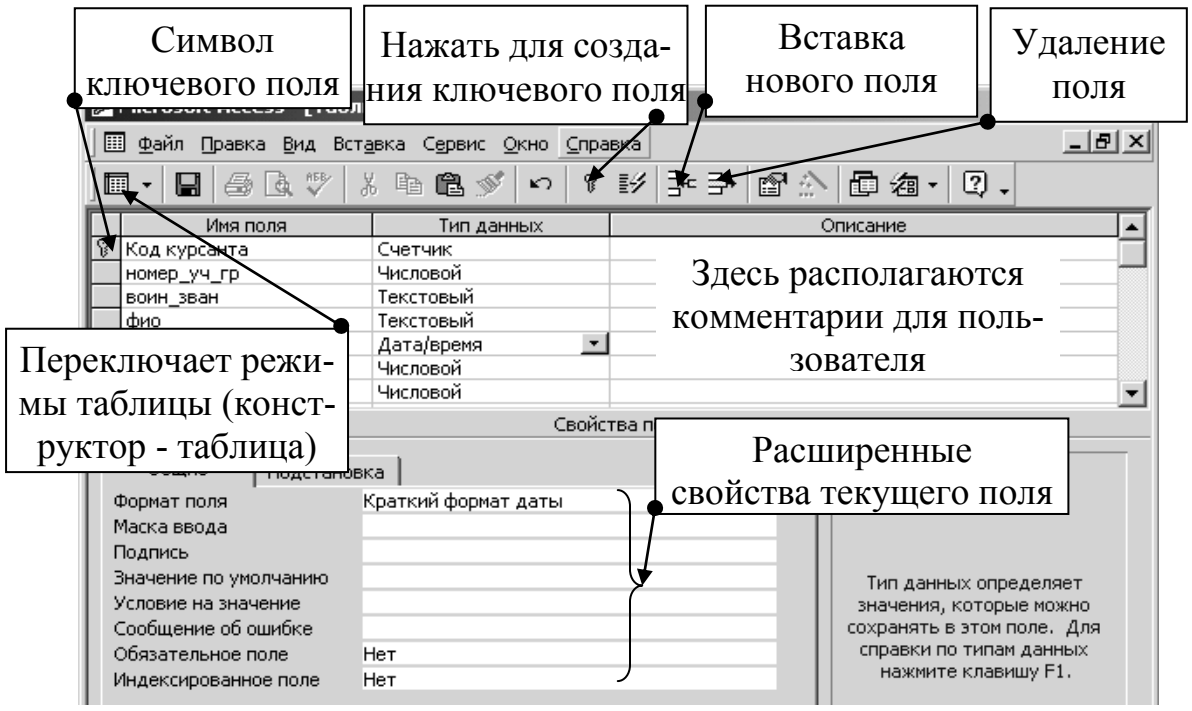


Рис.2.3. Таблица в режиме конструктора

Рекомендации по созданию полей:

1. Имя поля не должно содержать пробелов, точек, запятых и др. служебных символов;

2. Имена полей не могут быть одинаковыми;

3. Имена полей должны быть понятные (должны отражать суть хранимых в них данных);

4. Создание ключевых полей логично только если предполагается организовывать связь этой таблицы с другой таблицей;

5. Правильно определяйте тип данных каждого поля. Это обеспечит правильное выполнение операций с этими данными. Например, задание даты в виде текста не позволит сортировать даты, выполнять с ними математические операции для расчета возможных сроков;

6. Используйте при необходимости для уточнения характеристик данных ***Расширенные свойства поля*** (см. рис.2.3).

Первичный ключ или ключевое поле - это поле или совокупность полей, значения которых однозначно идентифицируют записи, хранящиеся в таблице. Каждая таблица должна иметь первичный ключ.

Если при создании таблицы использовался Мастер таблиц, то возможно Acess уже назначил ключевые поля. Для того, чтобы изменить ключевое поле или создать его самостоятельно, в режиме Мастера таблиц выберите нужное поле, щелкнув на нем мышкой. Далее в меню ***Правка*** активизируйте команду ***Ключевое поле***. Другой путь - щелкните правой кнопкой мыши по выбранному полю. В открывшемся меню выберите команду ***Первичный ключ.***

Тип данных

Для правильного определения типа данных каждого поля необходимо рассмотреть типы данных, поддерживаемые таблицей БД.

Таблица 2.1

Типы данных таблицы БД ACCESS

№	Тип данных	Пример данных	Примечание
1	Текстовый	Курсант	Может содержать любые данные ограниченной длины, но автоматически преобразует их в текст
2	Поле Мемо	Большой текстовый блок	Может содержать любые данные практически неограниченной длины, но автоматически преобразует их в текст
3	Числовой	2; 34.567; -0.003	Числа в любом формате, определяемом в расширенных настройках
4	Дата/время	23.02.2007, 23 февр. 2007	Даты в различных форматах, определяемых в расширенных настройках. По своим свойствам это формат числовой
5	Денежный	5000 р. 100 €	Финансы в различных форматах, определяемых в расширенных настройках. По своим свойствам это формат числовой
6	Счётчик	*Как правило числовые*	Позволяет автоматически увеличивать, или уменьшать значения своего поля в каждой новой записи (не позволяет вводить дан-

			ные вручную). Это обеспечивает уникальность поля, используется по этой причине в качестве ключевого поля
7	Логический	Истина; Ложь; 1; 0	Фиксирует наличие, или отсутствие чего-либо (использует булевы значения)
8	Поле объектов OLE	*различные*	Содержит большие объекты, поддерживающие технологию OLE (изображения, звуки и др.). В таблице не отображаются, но отображаются в формах
9	Гиперссылка	http://web44	Ссылки на другие документы, в том числе находящиеся в сети.
10	Мастер подстановок	*различные*	Создаёт связь с другими таблицами и может отображать данные, хранящиеся в них, кроме того, может содержать фиксированный набор заданных ранее значений (список)

Расширенные свойства поля

Для тонкой настройки полей необходимо использовать некоторые из возможных настроек расширенных свойств. Рассмотрим смысл этих настроек.

1. .*Размер поля* – определяет:

- для числового поля – формат числа (рис.2.4)
- для текстового поля – максимальное число символов, которое может быть введено в поле.

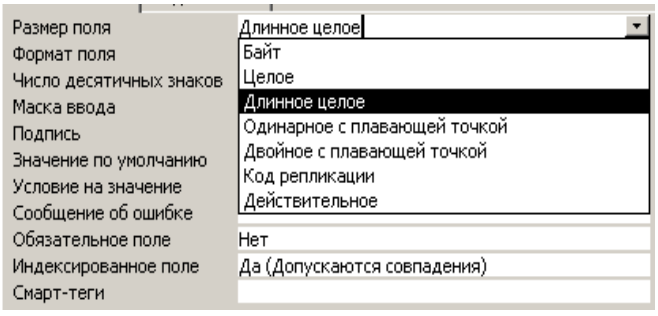


Рис.2.4. Размер поля

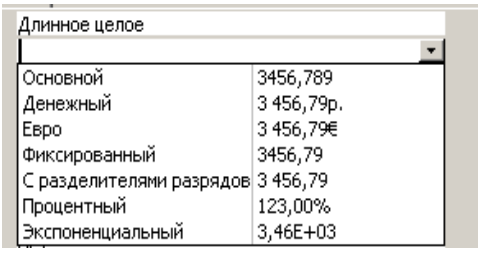


Рис.2.5. Формат поля

2. *Формат поля* – детализирует выбранный тип данных, позволяя более гибко определить характер хранимых данных. На рис.2.5 – детализация числового типа.

3. *Число десятичных знаков* – определяет количество цифр после запятой в дробном десятичном числе.

4. *Маска ввода* – предоставляет пользователю шаблон для ввода данных с целью избежать некорректного ввода. Желательно свойство использовать всегда, т.к. каждый формат предполагает свой шаблон, несоблюдение которого приводит к ошибкам.

5. *Подпись* – изменяет отображение названия поля при просмотре таблицы в режиме ***Таблица***. Подпись не изменяет имени поля в таблице БД.

6. *Значение по умолчанию* – значение, автоматически вставляемое в поле, если пользователь не ввёл в него свои данные.

7. *Условие на значение* – позволяет контролировать корректность данных на этапе их ввода пользователем (используется построитель выражений).

8. *Сообщение об ошибке* – выводит это сообщение, вместо стандартного сообщения БД при нарушении пользователем условия на значение вводимых данных.

9. *Обязательное поле* – Если "Да", то при вводе данных это поле не должно остаться пустым. В противном случае сохранить таблицу не удастся. СУБД выдает напоминающее сообщение.

10. *Индексированное поле* – если поле индексировано, то в этом поле в различных записях не должно быть одинаковых данных.

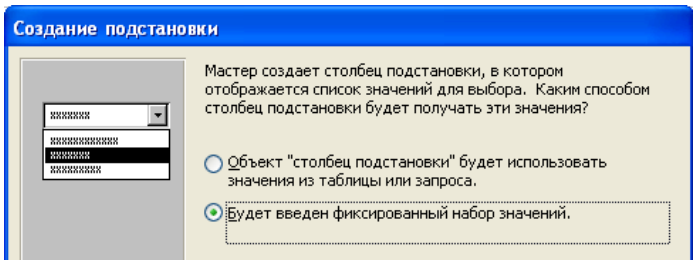
11. *Пустые строки* – Разрешает, или запрещает ввод пустых строк в текстовое поле.

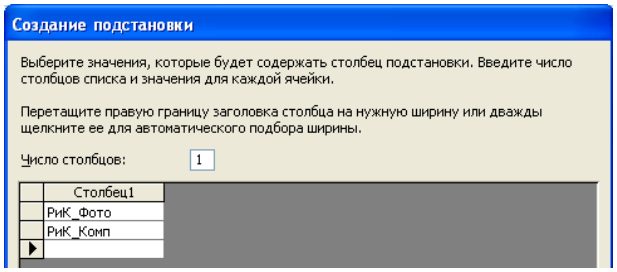
Создание списка для заполнения поля

При заполнении отдельных полей целесообразно выбор значений для записей осуществлять из подготовленного заранее списка. Это позволяет ограничить вводимые данные фиксированным количеством вариантов, и, кроме того, исключить возможные ошибки при вводе.

Если список предполагается создавать из текстовых значений, то начинать процедуру можно при наличии в поле типа данных – "текстовые". Если список будет формироваться из числовых значений, то необходимо установить соответствующий числовой формат данных.

Для создания списка необходимо выбрать тип данных ***Мастер подстановки*** и следовать указаниям мастера. Важно в первом же окне Мастера установить флажок ***Будет введен фиксированный набор значений***. Дальнейшие шаги Мастера приведут к записи значений из создаваемого списка. Каждое значение вводится в отдельную строку, при этом свободное поле образуется автоматически. Последующие шаги Мастера предложат отсортировать элементы списка по возрастанию или по убыванию значений. После реализации всех шагов Масте-





ра нажать кнопку ***Готово***. Список будет состоять из элементов того типа, который был установлен для этого поля перед запуском мастера.

Для работы с этим полем в режиме таблицы (при вводе данных) будет достаточно установить курсор на данное поле, справа появится стрелочка для показа выпадающего списка. В списке курсором указать требуемый элемент, после чего он будет занесен в поле таблицы.

Сохранение спецификации таблицы

Подготовленную при проектировании спецификацию таблицы следует сохранить - на ее основе будет строиться вся таблица. Для этого следует в меню ***Файл*** активизировать команду ***Сохранить.*** Если таблица сохраняется впервые, то на экран будет выведено окно Сохранить как... с запросом имени таблицы. Можно согласиться с предложенным Access именем, но лучше набрать свое по смыслу данных в таблице. Затем необходимо сделать щелчок на кнопке ***Ок*** - таблица запишется на диск.

Задание 2.2. Ввод и редактирование данных в таблице БД

В отчете представить:
1. Файл с заполненными таблицами.

Ввод и редактирование данных в таблице БД

В окне контейнера БД надо выделить в списке объектов требуемую таблицу и щелкнуть по кнопке ***Открыть.*** Выбранная таблица откроется в режиме заполнения (в режиме таблицы). Изначально она состоит из одной пустой записи, представленной в виде строки таблицы. Имена полей образуют заголовки колонок этой таблицы. Данные вводятся непосредственно в поля очередной записи. По мере надобности переход к следующему полю или записи осуществляется нажатием клавиши ***Tab.***

Внешний вид созданной в конструкторе таблицы (рис.2.2) представлен на рис.2.6. После создания полей сохранить таблицу под именем "*111_kurs*" и закрыть таблицу.

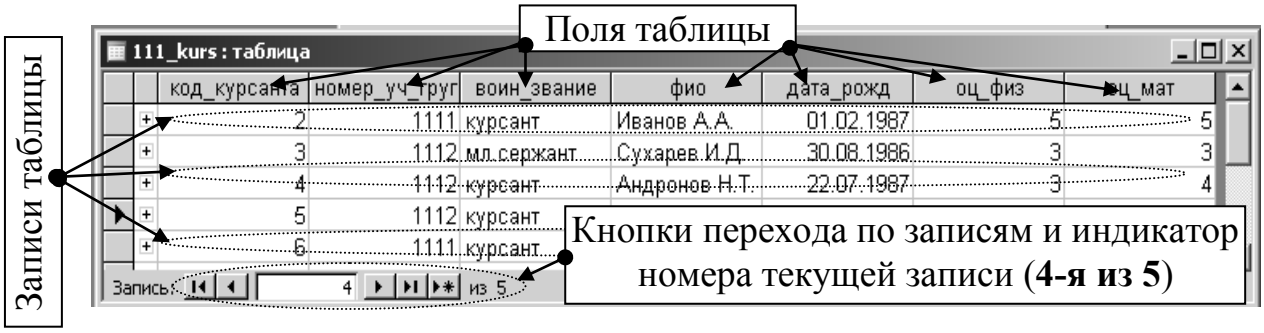
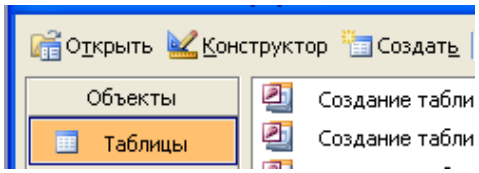
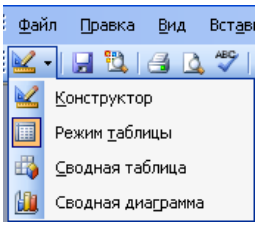


Рис.2.6. Вид таблицы в режиме таблицы

Переход из режима конструктора в режим таблицы производится или через контейнер базы данных, или с помощью раскрывающегося окна ***Вид*** панели инструментов.

Более удобно использовать для ввода данных специально подготовленную форму (см. соответствующий раздел).





Добавление записей в таблицу БД. Access-таблица всегда содержит одну пустую запись. Если таблица только что создана, то единственная запись, входящая в нее пуста. В заполненной таблице пустая запись располагается в конце. Именно в нее и вводится информация. Рекомендуется для добавления записей использовать форму.

Удаление записей из таблицы БД. При необходимости удалить одну или несколько записей, их следует маркировать, т.е. выделить щелчком мыши в селекторном столбце (крайнем слева) и вызвать команду ***Удалить запись*** меню ***Правка*** или нажать клавишу ***Del.***

Access в отдельном диалоговом окне запросит подтверждение удаления записей. После щелчка по кнопке ***Ок*** записи будут удалены.

Добавление столбца. Столбец добавляется слева от столбца, в котором установлен курсор, путем выбора команды ***Столбец*** в меню ***Вставка***. Столбец получает имя ***Поле 1***. Для изменения его имени и типа данных необходимо войти в режим конструктора таблиц.

Удаление столбца. Вначале столбец маркируется, для чего курсор мыши выставляется на заголовке столбца (при этом он превращается в направленную вниз стрелку) и нажимается на кнопку мыши. Затем в меню ***Правка*** следует выбрать команду ***Удалить столбец***. После подтверждения (кнопка Да) столбец будет удален.

Редактирование данных. Для изменения отдельных данных курсор выставляется в нужной клетке таблицы. Значение поля изменяется средствами клавиатуры.

Задание 2.3. Организация связей между таблицами БД

В отчете представить:

1. Схему данных (распечатка);

2. Схему данных в файле БД.

Организация связей между таблицами БД

В действительности базы данных из одной таблицы, практически, никогда не встречаются. Как правило, в БД входит несколько таблиц, которые связаны друг с другом посредством ключевых полей. Связывание позволяет с одной стороны обеспечить непротиворечивость данных, а с другой – целостность БД. В качестве примера, поясняющего сказанное, рассмотрим две таблицы: ***Kurs*** и ***Ocenki*** (рис.2.7).

kurs

kursant_id	kursant_fio
1012	Петров И.Т.
1013	Королёв С.П.

ocenki

kursant_id	oc_fiz	oc_mat	oc_inf

Рис.2.7. Связываемые таблицы

Анализ таблиц показывает, что для работы с этими двумя таблицами целесообразно обеспечить три условия:

1. Заполнение поля ***kursant_id*** таблицы ***Ocenki*** должно происходить с участием поля **kursant_id** таблицы ***Kurs***, иначе рано или поздно пользователь может ввести в поле **kursant_id** таблицы **ocenki** фамилию несуществующего курсанта, либо ошибиться при вводе. В этом случае БД расценит эту запись как нового курсанта.

2. При удалении курсанта из таблицы ***Kurs*** данные о нём должны автоматически удалиться из таблицы ***Ocenki.***

3. При изменении данных в поле **kursant_id** таблицы ***Kurs*** данные в поле **kursant_id** таблицы ***Ocenki*** должны адекватно измениться, иначе пользователь будет вынужден, например, при изменении фамилии курсанта в таблице ***Kurs*** изменить вручную данные в поле **kursant_id** таблицы ***Ocenki*** во всех записях, касающихся этого курсанта.

Таким образом, решив поставленные задачи, мы подойдём вплотную к созданию так называемой реляционной базы данных. В реляционной базе данных предполагается, что таблицы делятся на две категории: ведущие (master) и детализирующие (detailed). Это означает, что одной записи в ведущей таблице может соответствовать одна, или множество записей в детализирующей таблице. В нашем примере каждому курсанту таблицы ***Kurs*** может соответствовать много записей об оценках (полученных в различных семестрах) таблицы ***Ocenki*** (соответствует отношению "один ко многим"). Кроме того, решение задачи 2 означает обеспечение каскадного удаления записей, а решение задачи 3 – обеспечение каскадного обновления записей. Перечисленные выше три задачи обеспечивают целостность данных.

Пример заполненных таблиц, связанных так как описано выше представлен на рис.2.8.

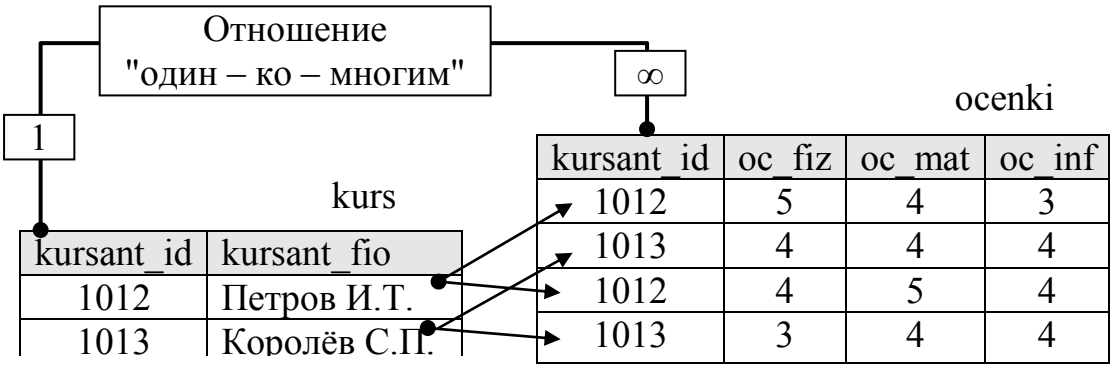


Рис.2.8. Связанные таблицы

При удалении из таблицы ***Kurs*** записи №1 (курсант Петров И.Т.) из таблицы № 2 удалятся все записи, относящиеся к удаляемой записи. Результат удаления отображён на рис.2.9.

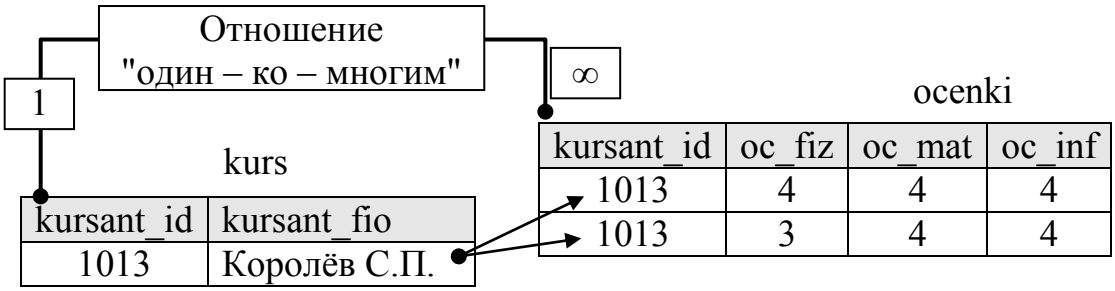


Рис. 2.9. Таблицы после удаления записи № 1012

Наиболее целесообразным и понятным способом связывания таблиц является следующий. В процессе создания таблицы ***Ocenki*** с помощью мастера подстановки установить связь с полем **kursant_id** таблицы ***Kurs*** (рис.2.10). Далее открыть схему данных и настроить вид связи.

Для связывания таблиц БД в обеих таблицах должны быть ключевые поля, данные в которых не должны повторяться (должны быть уникальны).

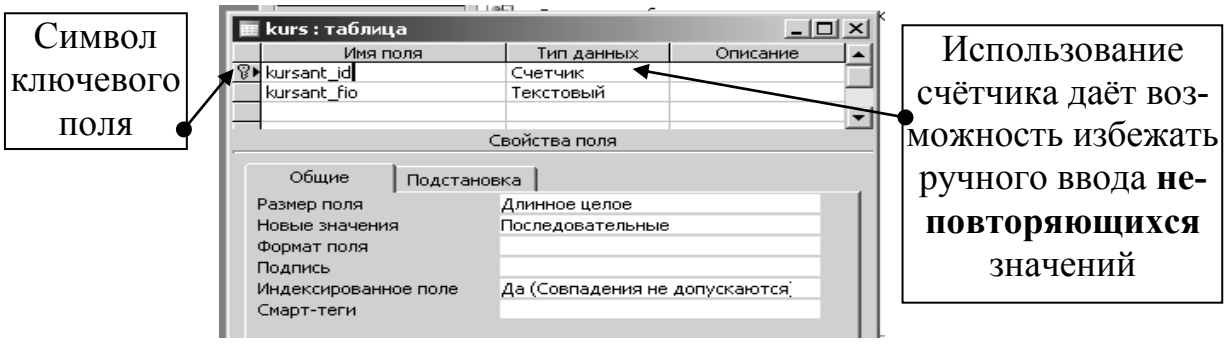


Рис.2.10. Ключевое поле таблицы ***Kurs***

Этап 1. Создание таблицы ***Kurs*** (см.рис.2.10).

Этап 2. Создание таблицы ***Ocenki*** (см. этапы, представленные на рис.2.11).

В процессе создания таблицы устанавливаем связь к полю **kursant_fio** таблицы ***Kurs.***

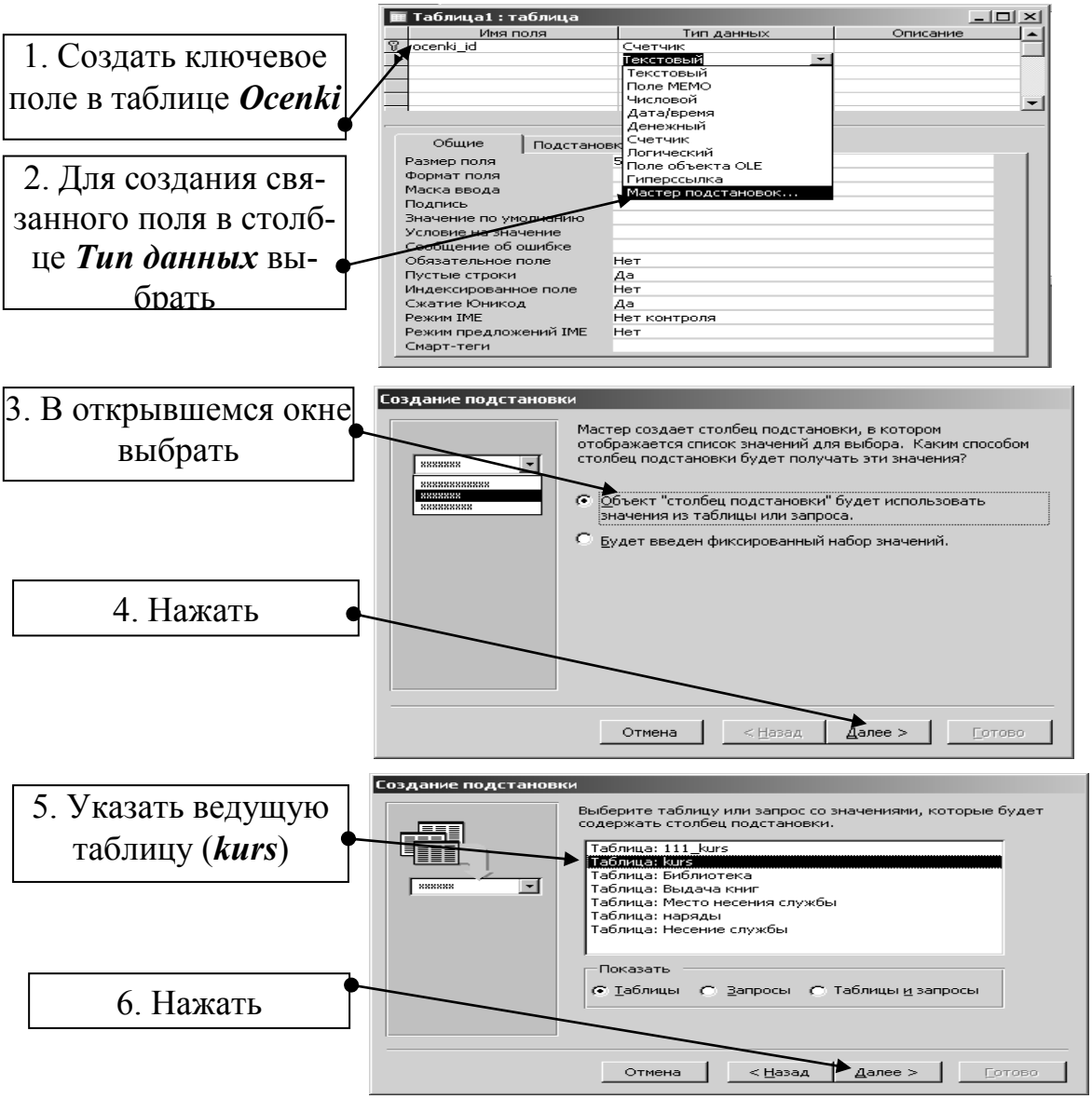


Рис.2.11. Этапы синтеза подчиненной таблицы

Несмотря на то, что мы устанавливаем связь к указанному полю, БД связывает таблицу ***Ocenki*** с полем **kursant_id** таблицы ***Kurs,*** т.к. именно оно является ключевым. (**Связь реализуется только по ключевым полям, но пользователю показывается именно то поле, которое он выбрал**).

В результате две таблицы оказались связанными друг с другом, причём таблица ***Kurs*** стала ведущей, а таблица ***Ocenki*** – детализированной.

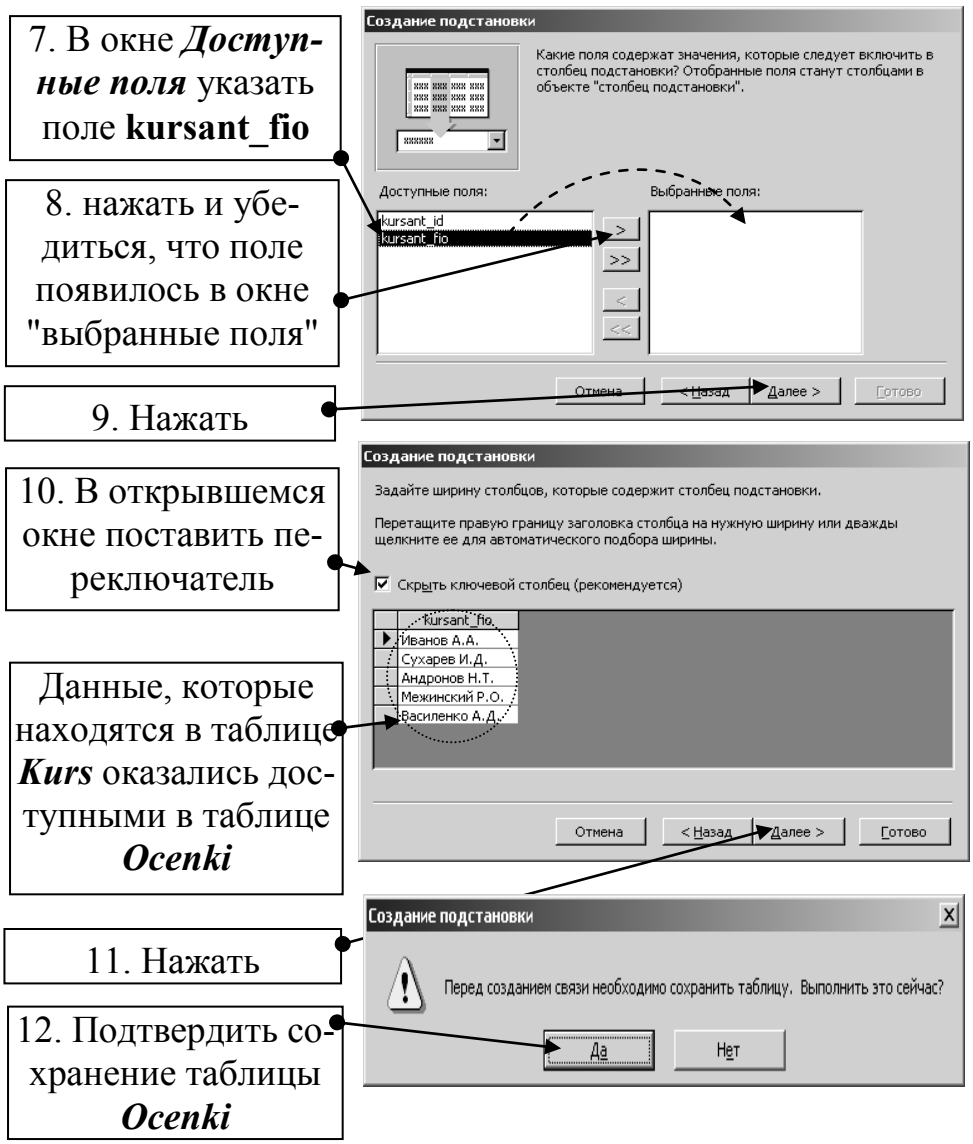


Рис.2.11. (продолжение)

Этап 3. Настройка свойств связи.

Созданная на предыдущем этапе связь между таблицами ***Kurs*** и ***Ocenki*** позволяет использовать данные, находящиеся в поле **kursant_fio** таблицы ***Kurs*** в поле **kursant_id** таблицы ***Ocenki***.

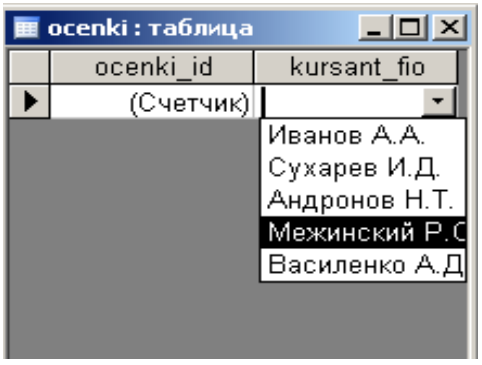


Рис.2.12. Выпадающий список

Внешнее проявление этой связи проявляется в появлении нового элемента в таблице ***Ocenki*** в виде выпадающего списка (рис.2.12).

Естественно предположить, что так как данные для выпадающего списка берутся из таблицы ***Kurs***, то при любых изменениях в поле **kursant fio** таблицы ***Kurs*,** эти изменения немедленно отобразятся в данных из выпадающего списка таблицы ***Ocenki***.

Вместе с тем созданная связь ещё не обеспечивает целостность данных, т.е. каскадное удаление и каскадное обновление связей. Для этого необходимо в главном меню БД выбрать пункт ***Сервис – Схема данных*** и в открывшемся окне изменить связь (рис.2.13).

Если всё сделано верно, то схема данных изменится и будет выглядеть так, как изображено на рис.2.14.

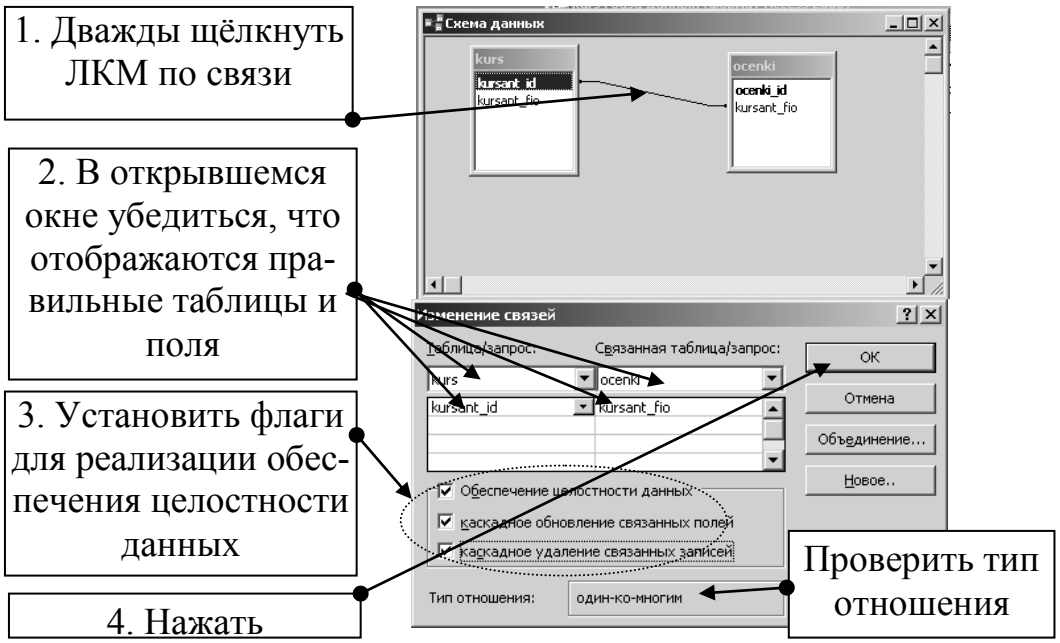


Рис.2.13. Схема данных и изменение связи

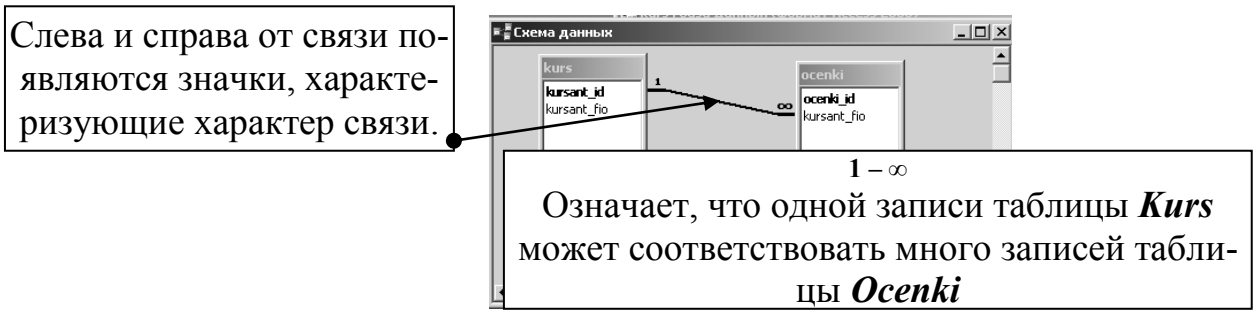


Рис.2.14. Связь с заданными свойствами

Результатом подобной настройки вида связи стала возможность изменения содержимого поля **kursant fio** таблицы ***Kurs*** с автоматическим изменением содержимого поля **kursant fio** таблицы ***Ocenki.***

Кроме того, при удалении записи о каком-либо курсанте из таблицы ***Kurs***, из таблицы ***Ocenki*** автоматически удалятся все записи, касающиеся этого курсанта.

Связи между таблицами помогут автоматически их учитывать в запросах, формах и отчетах.

Для создания ранее отсутствующей связи между таблицами необходимо выполнить следующее.

1. Убедиться, что файл базы данных открыт, но все таблицы в нем закрыты.

2. Открыть окно ***Схема данных***: щелкнуть значок на панели инструментов или щелкнуть правой кнопкой мыши в окне ***Таблицы*** и выбрать команду ***Схема данных***. Кроме того, можно выполнить команду ***Схема данных*** меню ***Сервис***.

3. Убедиться, что в окне ***Схема данных*** находятся нужные вам таблицы (между которыми создаются связи).

4. Если нужно добавить одну или несколько таблиц в окно ***Схема данных***, нажать кнопку ***Добавить таблицу*** на панели инструментов или щелкнуть правой кнопкой мыши в окне ***Схема данных***. В открывшемся меню выбрать команду ***Добавить таблицу***.

5. В окне диалога ***Добавление таблицы*** выделить нужную таблицу и нажать кнопку ***Добавить***, затем кнопку ***Закрыть***. Если нужно убрать таблицу из окна ***Схема данных***, следует щелкнуть по ней в произвольном месте и нажать клавишу ***Delete***. Можно щелкнуть правой кнопкой мыши по таблице и в открывшемся меню выбрать команду ***Скрыть таблицу***. Но при этом никакие связи данной таблицы не удаляются, просто таблица вместе со связями пропадает из виду.

6. Чтобы создать связь - выделить нужное поле одной таблицы и, удерживая нажатой левую клавишу мыши, перетащить его на связываемое поле другой таблицы. Откроется диалоговое окно ***Связи***. Здесь перечислены имена полей, участвующие в создании связи, а также приведены некоторые параметры.

Контрольные вопросы

1. Какие категории характеризуют таблицу?
2. Что такое таблица в БД?
3. Каковы этапы создания таблицы в СУБД?
4. Какие связи может хранить таблица?
5. Сформулируйте технологию создания таблицы с помощью конструктора.
6. Назовите состав расширенных свойств поля.
7. Каким образом создать список для заполнения поля? В чем преимущества списка по отношению к свободному вводу?
8. Каким образом создать связь между таблицами? Для чего они задаются?
9. Какие рекомендации существуют для создания полей?
10. Для чего создаются таблицы?
11. Какие типы данных используются при разработке таблиц?
12. Для каких данных используется каждый тип данных?
13. Какие режимы работы с таблицами существуют?
14. Для какой цели используется ключ?
15. Охарактеризуйте каждый из используемых типов связей.
16. Как создаются связи между таблицами в реляционной БД?

17. В чем заключается и как проявляется свойство "целостность данных"?
18. Какие данные находятся в поле, созданном с помощью мастера подстановки?
19. Чем отличается режим "Конструктор" от режима "Таблица"?

3. ЭКСПОРТ И ИМПОРТ ДАННЫХ

Цель

Приобрести навыки импорта данных в базу данных Access из других приложений и экспорта данных из базы данных.

Учебные вопросы

3.1. Импорт данных из электронных таблиц;
3.2. Импорт данных из текстовых файлов;
3.3. Экспорт данных БД в другие форматы.

Литература

Литература аналогична той, которая указана в разделе 1.

Задание 3.1. Импорт данных из электронных таблиц

Задание выполняется в соответствии с выданным индивидуальным вариантом.

Для выполнения типового задания по этой теме подготовить таблицу со следующей спецификацией:

tovar_sklad : таблица

Имя поля	Тип данных	
kod_tovar	Счетчик	
tip_tovar	Текстовый	Тип товара: РИК_Комп, РИК_Мобил, РИК_Фото, Сопутствующий
nazv_tovar	Текстовый	Наименование товара
cena_tovar	Денежный	Стоимость единицы товара
kolich_tovar	Текстовый	Количество товара на складе
photo_tovar	Поле объекта	изображение товара
primech_tovar	Поле MEMO	Сведения о товаре

1. Выполните импорт данных из всей электронной таблицы в новую таблицу БД. Зафиксируйте схему операций для выполнения импорта.

2. Выполните импорт данных из всей электронной таблицы в имеющуюся таблицу. Добавьте операции в имеющуюся схему операций.

3. Внесите изменение в имя столбца электронной таблицы и выполните импорт. При неудаче – устраните причину. Сделайте вывод о допустимости вариаций в именах таблиц.

4. Поменяйте местами колонки в электронной таблице. Выполните импорт. Сделайте вывод о допустимости несовпадения порядка имен в исходной и конечной таблице.

5. Добавьте новый столбец в электронную таблицу, присвойте ему имя и повторите импорт в имеющуюся таблицу. При неудаче устраните причину и выполните импорт.

6. Вставьте пустой столбец в исходную электронную таблицу. Выполните импорт. При неудаче устраните причину и выполните импорт.

7. Оставьте в электронной таблице только часть столбцов и выполните импорт в имеющуюся таблицу. Сделайте вывод по результатам п.п.5-7 о возможности несовпадения структур исходной и конечной таблицы.

7. Присвойте имена двум диапазонам в электронной таблице. Выполните импорт данных в БД по имени диапазона. Как нужно поступить, если импортировать необходимо только часть электронной таблицы?

Использование внешних данных

При работе с любым приложением обработки данных всегда является актуальным вопрос, как использовать те данные, которые уже были накоплены раньше другими программными средствами и, следовательно, имеют другой формат.

Access позволяет решить эту проблему стандартным способом — путем импорта существующей таблицы базы данных, рабочего листа электронной таблицы или текстового файла, созданных приложениями MS DOS или Windows, во внутренний формат базы данных Access (mdb). Естественно, что Access может также экспортировать данные из таблиц базы данных формата mdb в любой формат, из которого можно импортировать данные.

Однако Access является в этом смысле уникальной системой, т. к. она имеет еще один способ использования данных, которые хранятся в других форматах. Система позволяет присоединять (связывать) таблицы из баз данных других форматов к базе данных Access и работать с ними в исходном формате. После создания связи базы данных с внешней таблицей присоединенную таблицу можно просматривать, изменять ее содержимое, т. е. работать с ней как с внутренней таблицей базы данных Access. При этом другие пользователи могут использовать файл таблицы в своих приложениях.

Помимо файлов баз данных, Access может работать непосредственно с файлами электронных таблиц, текстовыми файлами, документами html, адресными книгами или импортировать данные из этих файлов и документов xml.

Типы файлов, данные из которых могут быть импортированы в базу данных Access или которые могут быть связаны с базой данных Access, можно увидеть, если в меню ***Файл*** (file) выбрать команду ***Внешние данные, импорт*** (get external data, import), а затем щелкнете мышкой по расширению поля ***Типы файлов*** (files of type) в диалоговом окне ***Импорт*** (import).

Форматы, в которые можно экспортировать данные из базы данных Access, можно увидеть, если в меню ***Файл*** (file) выбрать команду ***Экспорт*** (exort) и затем щелкнуть кнопкой мыши по расширению поля ***Типы файлов*** (files of type). В обоих случаях первая строчка содержит шаблоны файлов баз данных и проектов самой Microsoft Access.

Особенности присоединения и импорта таблиц баз данных

Прежде чем перейти к вопросам, связанным с импортом и присоединением таблиц, нужно понять разницу между этими возможностями Access.

При импорте таблицы из другой базы данных в рассматриваемом файле mdb создается новая таблица Access, которая имеет ту же структуру (состав и типы полей), что и исходная таблица, и содержит все данные исходной таблицы. После этого вы уже работаете с новой таблицей в формате Access, которая не зависит от исходной таблицы.

Когда вы присоединяете внешнюю таблицу, вы просто получаете доступ к таблице другого приложения, можете использовать данные из этой таблицы, но при этом она остается в старом приложении, в Access хранится только информация о связи. Оба приложения (то, в котором эта таблица была создана, и приложение Access) могут работать с этой таблицей одновременно. Однако для того, чтобы данные в присоединенной таблице можно было не только просматривать, но и изменять, эта таблица обязательно должна иметь ключевое поле. Обычно ключевое поле определяется в процессе присоединения таблицы.

С присоединенной к базе данных Access внешней таблицей можно работать точно так же, как если бы она содержалась в этой базе. Единственным ограничением является то, что невозможно изменить структуру присоединенной таблицы.

В первых версиях Access для операции присоединения внешней таблицы применялся термин присоединение (attachment). В последних версиях введен термин связывание (linking). Таким образом, внешние таблицы теперь становятся связанными. В этом тексте используется старый термин присоединение и, соответственно, присоединенные таблицы, чтобы не путать со связанными таблицами в одной базе данных, которые содержат связанные данные.

Присоединение внешних таблиц к базе данных Access бывает выгодным в следующих случаях:

- с таблицей работают совместно несколько пользователей;
- невозможен отказ от применения другой СУБД для модификации данных;
- таблица хранится на другой машине (например, на сервере локальной сети) и содержит очень большой объем информации.

Использование импорта внешних таблиц в базе данных Access оправдано в следующих случаях:

- если при разработке приложений необходимо снять ограничения на изменение свойств внешних таблиц;
- отсутствует непосредственный доступ к файлу таблицы;
- необходимо переопределить ключевые поля таблиц Paradox. Такая ситуация может возникнуть в случае, если структура одной или нескольких таблиц серьезно противоречит правилам нормализации;

- в таблицу предполагается вводить повторяющиеся данные, а первичный ключ внешней таблицы этого не допускает.

Импорт таблиц баз данных Access

Допустимо импортировать любой из шести основных типов объектов (таблицы, запросы, формы, отчеты, макросы и модули) из одной базы данных Access в другую.

Импорт таблиц из других баз данных Access выполняется аналогично присоединению, только используется команда меню ***Файл, Внешние данные, Импорт*** (file, get external data, import) и диалоговые окна ***Импорт*** (import) и ***Импорт объектов*** (import objects).

С помощью команды ***Импорт*** возможно копирование нескольких объектов, не переходя каждый раз из одной базы данных в другую.

Для импорта из другой базы данных Access выполните действия:

1. Откройте БД, в которую необходимо импортировать объект;
2. Выберите команду ***Файл,*** В***нешние данные, Импорт***. Откроется окно ***Импорт***;
3. В раскрывающемся списке ***Тип файла*** выберите ***Microsoft Access***, затем найдите необходимый файл, в котором имеются нужные объекты;
4. Нажмите кнопку ***Импорт***, после чего откроется окно ***Импорт объектов*** (рис.3.1)**.** В данном случае объектами называются элементы базы данных, но не объекты технологии OLE;

5. Выделите на требуемой вкладке имя импортируемого объекта. Выделить можно все имена объектов или разные имена на разных вкладках.

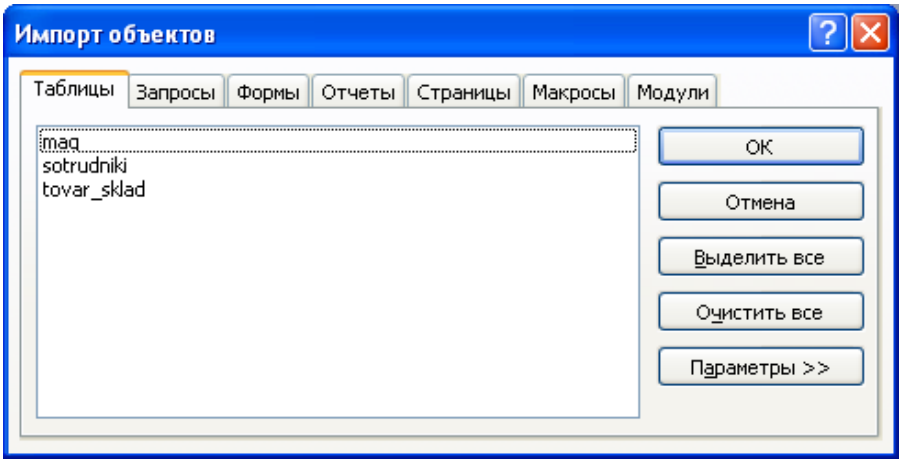


Рис.3.1. Окно ***Импорт объектов***

6. После нажатия кнопки ***Параметры*** становятся доступными дополнительные параметры (рис.3.2). Расставляя флажки можно выбрать:

- ***Схема данных*** - импорт связей выбранных импортируемых таблиц;
- ***Структура и данные*** – импорт данных вместе со структурой таблицы;

- ***Только Структура*** **–** импорт только структуры выбранной таблицы;
- ***Меню и панели*** - импорт их исходной базы данных все специальные меню и панели инструментов;
- ***Спецификации*** – импорт всех спецификаций импорта/экспорта, определенные в исходной базе данных;
- ***Как запросов*** – импорт определение запросов;
- ***Как таблиц*** – импорт результатов запросов в виде таблиц;

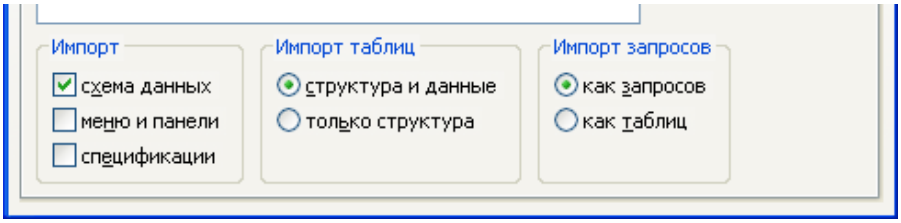


Рис.3.2. Дополнительные параметры при импорте

7. После успешного импорта объекты сохранят свои имена. При совпадении имен автоматически будут добавлены порядковые имена. Поэтому после импорта необходимо проверить ранее установленные ссылки на переименованные объекты.

Импорт электронных таблиц

Access позволяет создавать таблицы в базе данных, импортируя их из рабочего листа Excel. При этом можно импортировать как целый рабочий лист, так и именованный диапазон ячеек этого листа. Кроме того, в процессе импорта может быть создана новая таблицы Access, которая будет содержать все импортируемые данные, или же эти данные будут добавлены к уже существующей таблице Access.

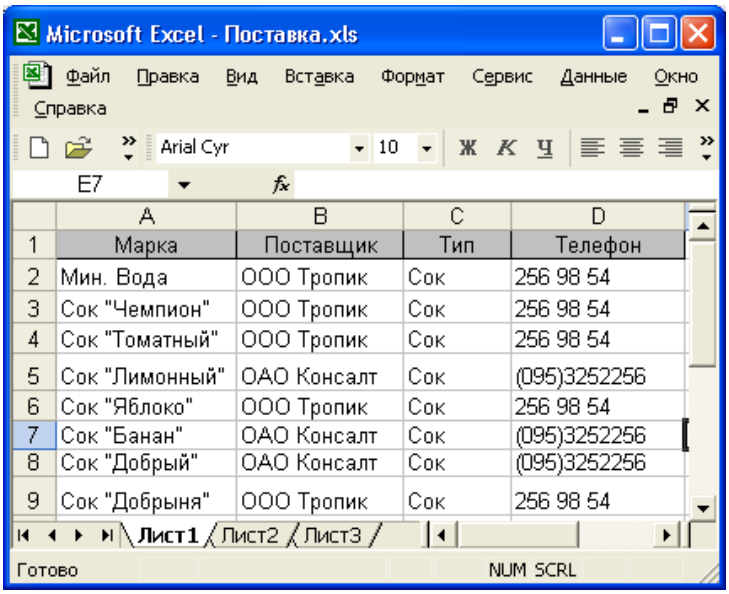
Подготовка электронной таблицы для импорта

Access определяет типы данных для полей новой таблицы, анализируя значения в первых импортируемых строках электронной таблицы. Для того чтобы импорт прошел без ошибок, данные на импортируемом листе должны быть соответствующим образом организованы. На рисунке (рис.3.3) показан формат представления данных на рабочем листе Excel, который наиболее подходит для импорта в базу данных.

Для того чтобы подготовить данные на рабочем листе для импорта, может потребоваться выполнить несколько предварительных действий:

1) запустите Microsoft Excel и откройте рабочий лист, данные из которого нужно импортировать;
2) добавьте, если это необходимо, заголовки столбцов над первой строкой импортируемого диапазона данных. Имена всех полей должны нахо-

дятся в первой строке таблицы, а оставшиеся строки должны содержать данные. Заголовки столбцов будут применяться в качестве имен полей, поэтому в них нельзя употреблять точку (.), восклицательный знак (!) и прямоугольные скобки ([]). Нельзя использовать одно имя дважды. Если в заголовках столбцов встречаются запрещенные символы или один заголовок применяется несколько раз, в процессе импорта данных это вызовет сообщение об ошибке.



	A	B	C	D
1	Марка	Поставщик	Тип	Телефон
2	Мин. Вода	ООО Тропик	Сок	256 98 54
3	Сок "Чемпион"	ООО Тропик	Сок	256 98 54
4	Сок "Томатный"	ООО Тропик	Сок	256 98 54
5	Сок "Лимонный"	ОАО Консалт	Сок	(095)3252256
6	Сок "Яблоко"	ООО Тропик	Сок	256 98 54
7	Сок "Банан"	ОАО Консалт	Сок	(095)3252256
8	Сок "Добрый"	ОАО Консалт	Сок	(095)3252256
9	Сок "Добрыня"	ООО Тропик	Сок	256 98 54

Рис.3.3. Формат листа Excel для импорта

При добавлении данных в существующую таблицу заголовки столбцов рабочего листа Excel должны точно совпадать с названиями полей этой таблицы БД. Их порядок значения не имеет. Однако других имен (или пустых заголовков) внутри полей, а также лишних полей (отличных от полей базы данных) быть не должно;

3) если необходимо импортировать не все данные рабочего листа, выделите нужный диапазон ячеек (вместе со строкой заголовка), а затем выберите команду ***Вставка, Имя, Присвоить*** (insert, name, define) и присвойте имя выделенному диапазону;

4) тип данных в каждой ячейке одного столбца должен быть одинаковым, и в каждой строке должны использоваться одни и те же поля. При импорте электронной таблицы алфавитно-цифровая информация сохраняется в текстовых полях с размером 255 символов, числовые данные – в числовых полях со свойством ***Размер поля,*** установленным в значение ***С плавающей точкой*** (8 байт). Числовые данные в денежном формате сохраняются в денежных полях, значения дат или времени – в полях типа Дата/Время. Если в первых стоках столбца Access обнаружит смешанные данные, он импортирует столбец в текстовое поле.

Если значения в первых строках столбцов не являются представительными для правильного определения типов данных столбцов, вставьте в начале электронной таблицы фиктивную строку со специально подобранными значениями, которые можно легко удалить после импорта таблицы.

Access пытается переводить типы данных при импорте в существующую таблицу в свои форматы, заданные этим полям;

5) сохраните рабочий лист и закройте приложение Excel.

В таком виде таблица на рабочем листе максимально соответствует таблице Access, что позволит упростить процесс импорта.

Если ячейки рабочего листа содержали формулы, по которым вычислялись значения, то в таблицу Access будут импортированы только вычисленные значения.

Импорт из электронных таблиц

После подготовки данные рабочего листа электронных таблиц можно импортировать в таблицу Access. Для этого последовательно необходимо выполнить следующие операции.

1. Запустите Access и откройте базу данных, в которую необходимо импортировать данные. Активизируйте окно база данных (database), щелкнув по его заголовку левой кнопкой мыши.

2. Выберите команду ***Файл, Внешние данные, Импорт*** (file, get external data, import). Появится диалоговое окно ***Импорт*** (Рис.3.4).

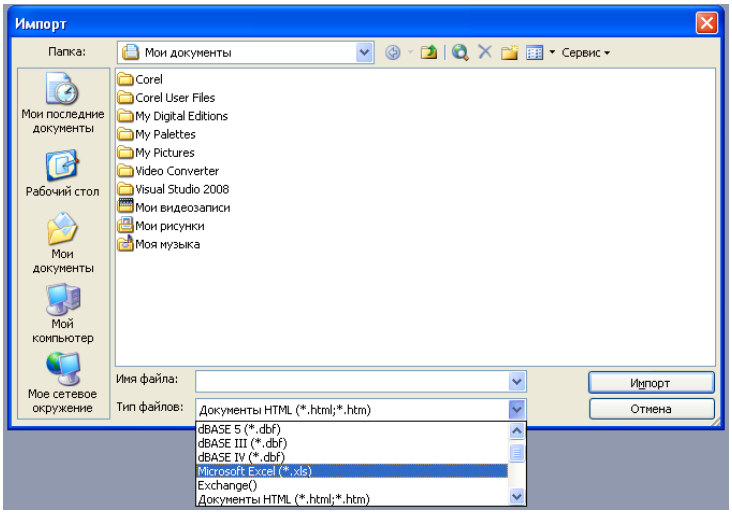


Рис.3.4. Диалоговое окно Импорт

3. Перейдите в папку, содержащую файл рабочего листа Excel. В раскрывающемся списке ***Тип файла*** (file of type) выделите элемент ***Microsoft excel (*.xls).*** Выделите имя файла и нажмите кнопку ***Импорт*** (import) или просто дважды щелкните по имени файла левой кнопкой мыши. Запустится Мастер импорта электронной таблицы (import spreadsheet wizard), первое диалоговое окно которого показано на рис.3.5.

4. Если необходимо импортировать весь рабочий лист, выберите переключатель ***Листы*** (show worksheets). Если же предполагается импортировать именованный диапазон рабочего листа, то выберите переключатель ***Именованные диапазоны*** (show named ranges). В списке первого диалогового окна мастера импорта электронных таблиц будут выведены имена ра-

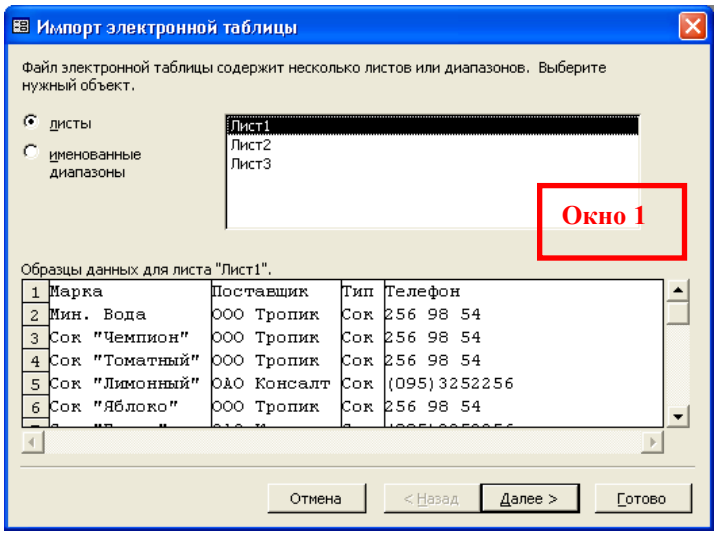


Рис.3.5. Первое окно Мастера импорта

бочих листов или диапазонов, соответственно.

5. Выделите нужное имя рабочего листа или диапазона ячеек. В нижней части первого окна мастера импорта выводится образец данных из выделенного элемента электронной таблицы. Нажмите кнопку Далее (next), чтобы перейти к следующему шагу мастера. Появится следующее диалоговое окно Мастера (Рис.3.6).

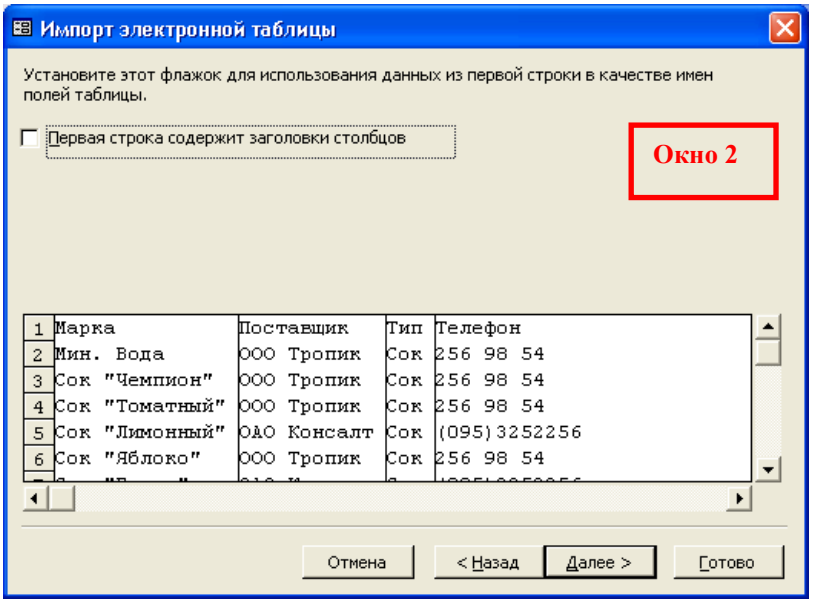


Рис.3.6. Второе окно Мастера импорта

6. Если первая строка импортируемых данных содержит заголовки столбцов, то их можно использовать в качестве имен полей. Для этого установите в окне 2 (см.рис.3.6) флажок ***Первая строка содержит заголовки столбцов*** (first row contains column headings). Снова нажмите кнопку ***Далее*** (next). Появится следующее третье диалоговое окно (Рис.3.7).

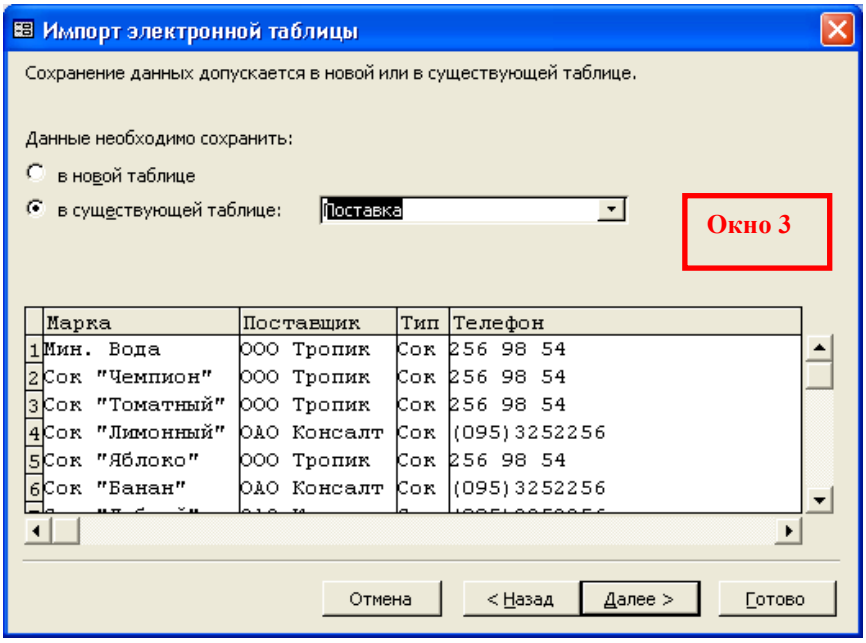


Рис.3.7. Третье окно Мастера импорта

7. В этом диалоговом окне можно указать, где необходимо сохранить импортируемые данные — ***В новой таблице*** или ***В существующей таблице***. Выберите нужный переключатель и, при необходимости, нужную таблицу в списке ***В существующей таблице*** (in ah existing table). Нажмите кнопку ***Далее*** (next).

Если вы добавляете данные к уже существующей таблице, то все промежуточные шаги будут пропущены и появится последнее диалоговое окно мастера (рис.3.10).

Если же вы создаете новую таблицу, то появится диалоговое окно 4 (рис.4.8).

8. В этом окне вы можете выбрать те столбцы, которые нужно импортировать (это не обязательно все столбцы рабочего листа), и ввести некоторые параметры для полей будущей таблицы. Чтобы не импортировать (пропустить) какое-либо поле (столбец рабочего листа), выделите его и установите флажок ***Не импортировать (пропустить) поле*** (do not import field).

9. Мастер импорта электронных таблиц позволяет изменить или добавить (если они не определены в первой строке рабочего листа) имена полей таблицы, соответствующих столбцам рабочего листа (см. рис.3.8). Для этого выделите столбец, щелкнув по нему левой кнопкой мыши, а затем укажите имя в текстовом поле ***Имя поля*** (field name) (горизонтальная полоса прокрутки в диалоговом окне позволяет просмотреть все столбцы листа).

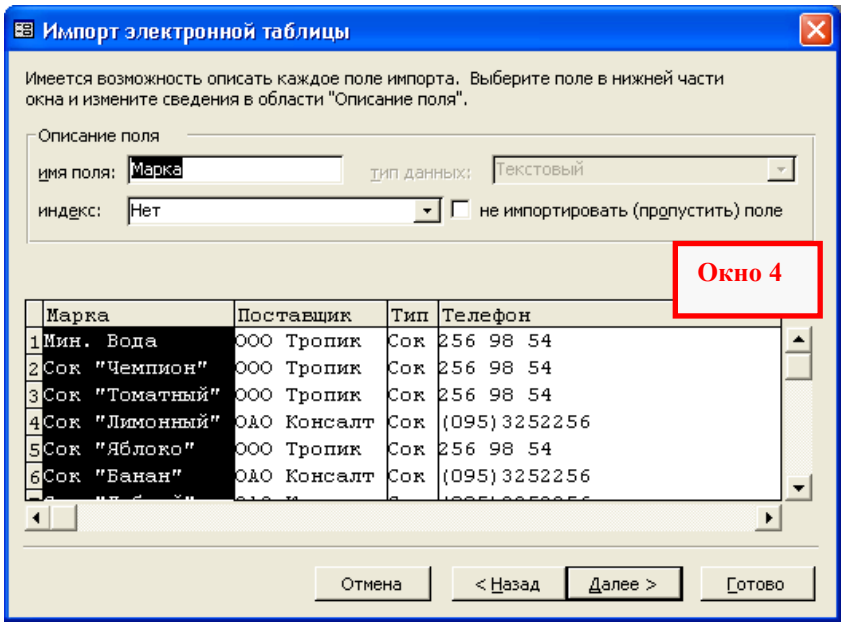


Рис.3.8. Четвертое окно Мастера импорта

10. Если какое-либо поле нужно проиндексировать, в окне 4 выберите требуемый тип индекса в раскрывающемся списке ***Индекс*** (indexed).

11. Если данные рабочего листа не форматировались или были отформатированы как текст, Access позволяет выбрать тип данных для каждого поля при помощи раскрывающегося списка ***Тип данных*** (data type). В окне 4 раскрывающийся список ***Тип данных*** недоступен, поскольку импортируемый рабочий лист содержал форматирование, которое Access смог распознать автоматически.

Повторите шаги 9—11 для каждого столбца импортируемых данных. После того как заданы все имена и типы данных полей, определены проиндексированные поля, а также столбцы, которые нужно пропустить при импорте, нажмите кнопку ***Далее***. Появится следующее диалоговое окно 5 Мастера импорта (рис.3.9).

12. На пятом шаге Мастера импорта электронной таблицы можно определить ключевое поле новой таблицы. Чтобы Access автоматически добавил к импортируемой таблице ключевое поле с типом данных счетчик и присвоил уникальный номер каждой строке таблицы, выберите переключатель ***Автоматически создать ключ*** (let access add primary key). Чтобы определить один из столбцов импортируемого диапазона ячеек в качестве ключевого поля таблицы, выберите переключатель ***Определить ключ*** (choose my own primary key) и выделите в раскрывающемся списке название столбца. Если

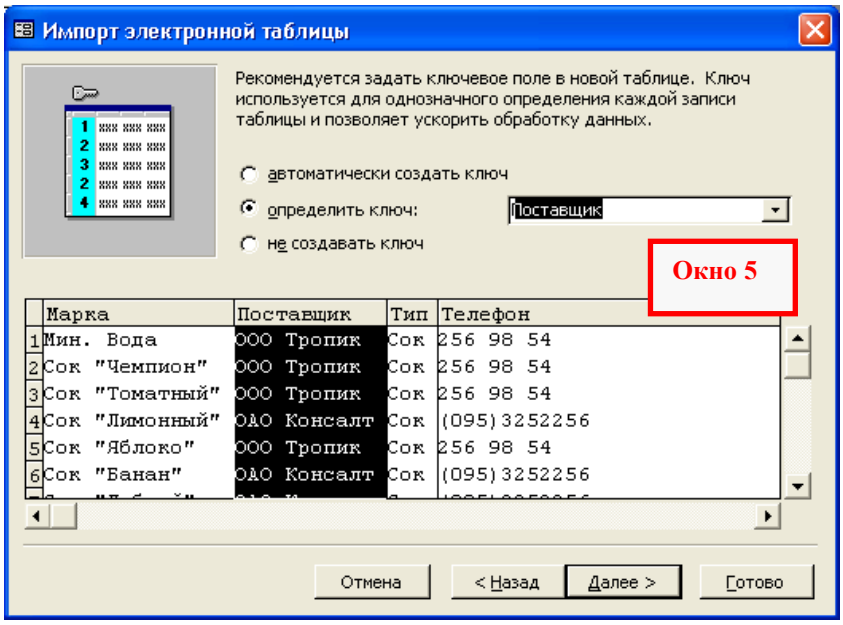


Рис.3.9. Пятое окно Мастера импорта

для импортируемой таблицы не требуется определение ключевого поля, выберите переключатель ***Не создавать ключ*** (no primary key). Прежде чем определить ключевое поле на основе одного из столбцов таблицы, убедитесь в том, что ячейки этого столбца содержат уникальные значения. Нажмите кнопку ***Далее***. Появится последнее диалоговое окно 6 (рис.3.10).

13. В поле ввода ***Импорт в таблицу*** (import to table) укажите имя новой таблицы. По умолчанию Access использует имя импортируемого рабочего листа или диапазона ячеек в качестве имени новой таблицы. Если имя новой таблицы, которое вы указали, совпадет с именем уже существующей в базе данных таблицы, то Access выведет диалоговое окно, в котором спросит у вас, заменить ли уже существующую таблицу новой или переименовать создаваемую таблицу.

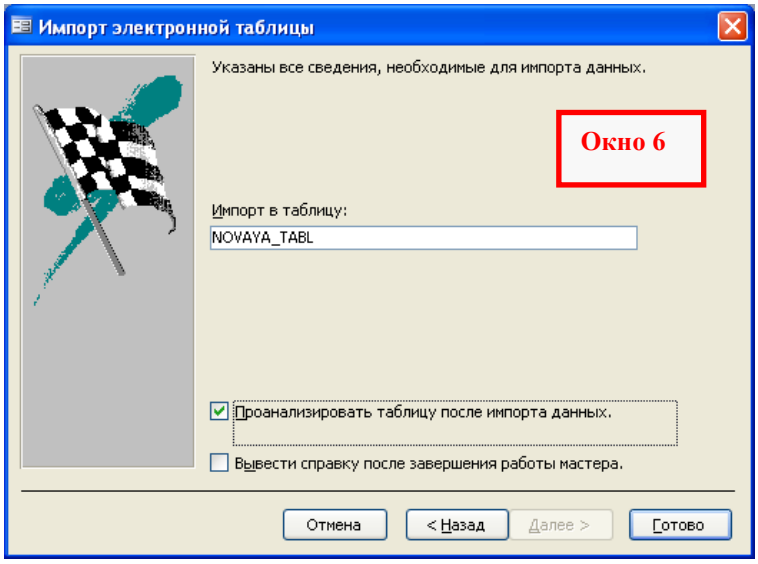


Рис.3.10. Шестое окно Мастера импорта

Если импортировались данные в уже существующую таблицу, то в окошке будет отображено ее имя.

Добавление данных в существующую таблицу производится по строкам свободным от записей. Каждой добавляемой записи счетчиком назначается новый номер ключевого поля.

Нажмите кнопку ***Готово*** (finish), чтобы завершить процесс импорта данных. После того как импорт данных будет завершен, появится сообщение о результате выполнения операции импорта.

В случае неудачи необходимо вернуться в файл электронной таблицы, устранить причины ошибок и повторить импорт.

Исправление ошибок после импорта из электронных таблиц

При автоматическом определении типов данных полей импортируемой таблицы Access просматривает первые 20 строк импортируемого диапазона ячеек. Например, если каждый столбец в импортируемом диапазоне содержит числовые данные, то полю, соответствующему этому столбцу, присвоится один из числовых подтипов данных (выбор подтипа зависит от параметров форматирования рабочего листа Excel). Если же в первых 20 ячейках столбца содержатся числа, а в остальных встречается текст, то Access уже не изменяет тип данных поля, а создает таблицу ошибок импорта, в которой каждая запись соответствует одной ошибке. Система дает возможность либо продолжить импорт, либо прекратить его (Рис.3.11).

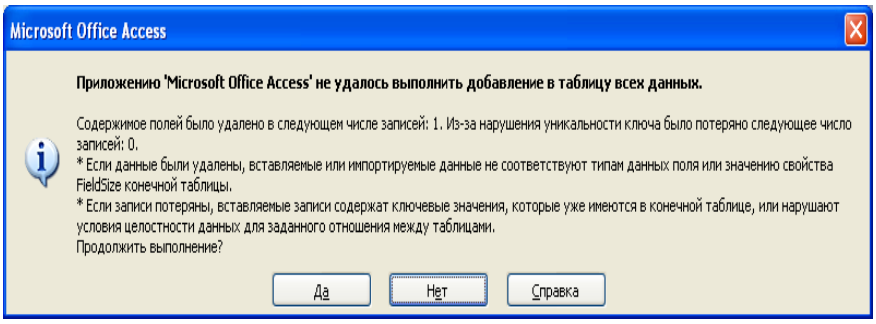


Рис.3.11. Предупреждающее сообщение об ошибках

После окончания процесса импорта в случае возникновения ошибок выдается сообщение об этом и указывается имя таблицы ошибок импорта (рис.3.12).

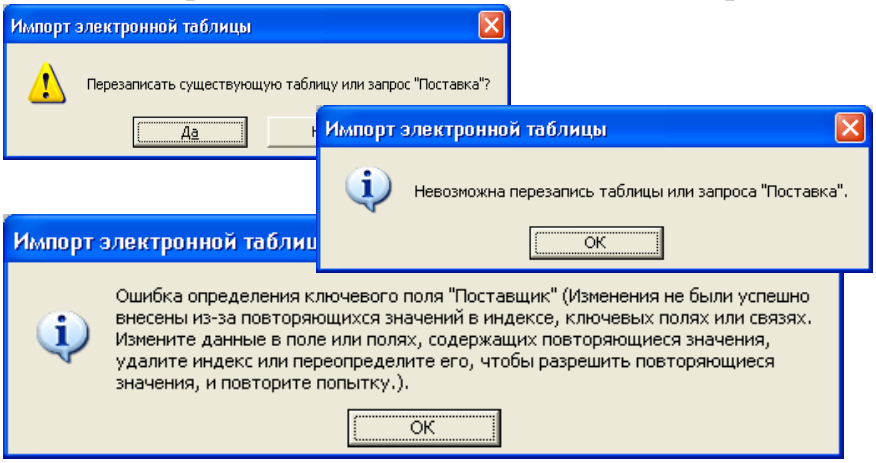
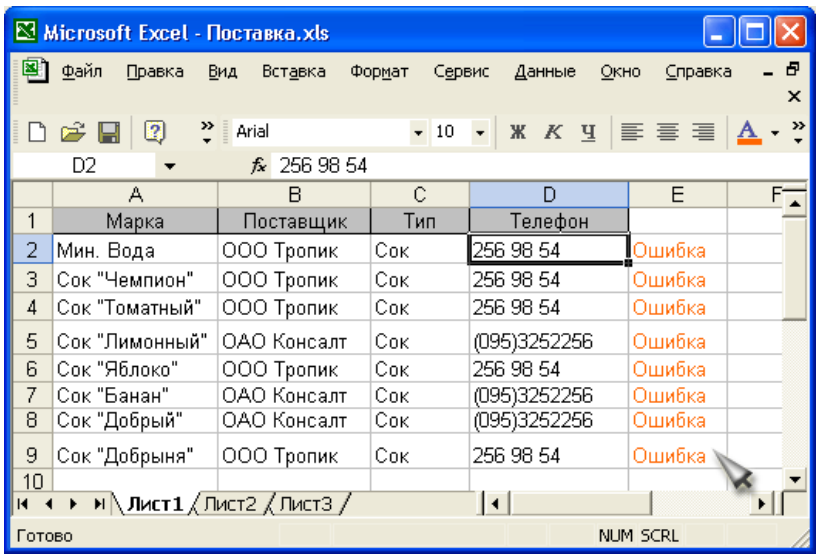


Рис.3.12. Сообщения при аварийном завершении импорта данных

Таблица ошибок импорта (рис.3.13) позволяет найти записи, в которых возникли ошибки, и исправить их. Однако более правильным способом будет исправить рабочий лист Excel и повторить им-



	A	B	C	D	E
1	Марка	Поставщик	Тип	Телефон	
2	Мин. Вода	ООО Тропик	Сок	256 98 54	Ошибка
3	Сок "Чемпион"	ООО Тропик	Сок	256 98 54	Ошибка
4	Сок "Томатный"	ООО Тропик	Сок	256 98 54	Ошибка
5	Сок "Лимонный"	ОАО Консалт	Сок	(095)3252256	Ошибка
6	Сок "Яблоко"	ООО Тропик	Сок	256 98 54	Ошибка
7	Сок "Банан"	ОАО Консалт	Сок	(095)3252256	Ошибка
8	Сок "Добрый"	ОАО Консалт	Сок	(095)3252256	Ошибка
9	Сок "Добрыня"	ООО Тропик	Сок	256 98 54	Ошибка
10					

Рис.3.13. Таблица ошибок импорта

порт данных. После исправления ошибок удалите эту таблицу.

Импортированная таблица появится в окне базы данных. Чтобы проверить, что желаемый результат достигнут, откройте импортированную таблицу в режиме таблицы, дважды щелкнув по ней левой кнопкой мыши.

Чтобы узнать, какие типы данных полей были выбраны при импорте рабочего листа, в окне базы данных откройте созданную таблицу в режиме конструктора. В отличие от присоединенных таблиц, все свойства полей импортированных таблиц можно изменить.

Сразу после завершения импорта данных можно вызвать анализатор таблиц, позволяющий оптимизировать структуру таблицы, например, исключить повторяющиеся данные. В этом случае необходимо установить флажок ***Проанализировать таблицу после импорта данных*** (i would like a wizard to analyze my table after importing the data).

Задание 3.2. Импорт данных из текстовых файлов

1. Подготовьте текстовый файл с разделителями (табуляцией) для импорта в имеющуюся таблицу БД.
2. Выполните импорт данных из всего текстового файла в новую таблицу БД. Зафиксируйте схему операций для выполнения импорта.
3. Выполните импорт данных из всего текстового файла в имеющуюся таблицу. Добавьте операции в имеющуюся схему операций.
4. Повторите операции п.п.1-3 для текстового файла с фиксированной длиной поля.
5. Исказите имя для одного из полей в данных текстового файла. Попытайтесь импортировать файл в имеющуюся таблицу БД. Сделайте вывод о допустимости вариаций в именах полей.
6. . Поменяйте местами колонки в полях текстового файла. Выполните импорт. Сделайте вывод о допустимости несовпадения порядка имен в исходной и конечной таблице.
7. Вставьте вместо одного два разделителя в одной из записей в исходный текстовый файл. Выполните импорт. При неудаче устраните причину и выполните импорт. Сделайте вывод о количестве разделителей в одинаковых полях текстового файла.

Импорт таблиц из текстовых файлов

В Access можно импортировать текстовый файл, хотя его данные, в отличие от Excel, не организованы в строки и столбцы привычным способом. Чтобы Access мог распознавать данные текстового файла, необходимо использовать один из следующих подходов:

- включить в файл специальные символы, разграничивающие поля в каждой записи (называют такой файл - текстовый файл с разделителями, с полями переменной ширины) или

- поместить каждое поле в одну и ту же позицию внутри любой записи, чтобы каждое поле имело определенную фиксированную ширину (называют - текстовый файл с фиксированной длиной записи, с полями фиксированной ширины). Пустые позиции при этом заполняются пробелами.

Подготовка текстового файла для импорта

Большинство текстовых процессоров позволяют сохранить данные документов в текстовых файлах без форматирования, которые затем можно использовать для импорта в таблицы Access. Для этого достаточно сохранить документ как текстовый файл, выбрав команду ***Файл, Сохранить как*** и установив формат ***txt.***

С помощью приложения Блокнот можно быстро определить тип текстового файла (рис.3.14), который предполагается импортировать в таблицу Access. Однако приложение Блокнот может работать только с файлами, размер которых не превышает 60 КБайт. Для просмотра больших текстовых файлов можно использовать редактор Wordpad. Но, работая с этим редактором при просмотре или редактировании текстовых файлов, нельзя применять форматирование и сохранять файлы в формате doc, т. к. в этом случае текстовый файл нельзя будет импортировать.

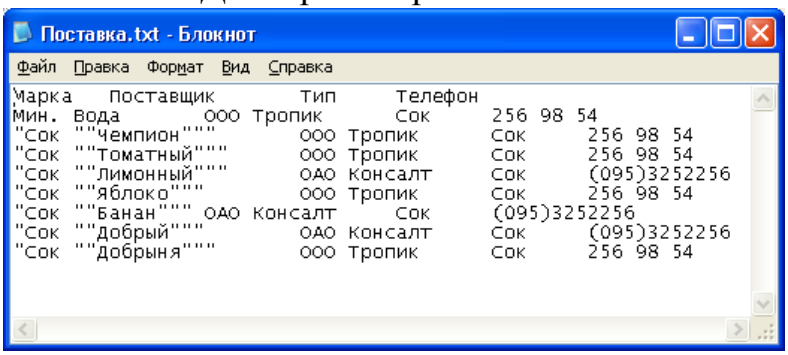


Рис.3.14. Текст без форматирования в редакторе Блокнот

Подготовка текстового файла с разделителями полей

При возможности выбора формата текстовых файлов, предназначенных для импорта в таблицы Access, рекомендуется использовать файлы с разделителями полей.

В качестве символов разделителей полей в текстовых файлах могут использоваться самые разнообразные символы. Access позволяет работать со стандартными символами разделителей (запятыми, пробелами и символами табуляции) или указать символ, используемый в импортируемом файле в качестве разделителя. Однако нестандартные разделители полей (в том числе и пробелы) на практике встречаются крайне редко.

Кроме того, в текстовых файлах могут использоваться двойные кавычки (") для выделения текстовых полей. При импорте данных из таких файлов предполагается, что поля, значения которых не выделены кавычками, являются числовыми (в том случае, если они содержат только числа).

Для текстовых файлов с разделителями полей существует несколько стандартных вариантов разделения полей. Они приведены в таблице.

Формат	Описание
Записи в одну строку, поля разделены запятыми	Разделителем записей является символ новой строки. Некоторые приложения заключают все значения полей в двойные кавычки, другие приложения ограничиваются заключением в кавычки лишь текстовых значений, чтобы отличить их от числовых
Записи в одну строку, поля разделены знаками табуляции	Разделителем записей является символ новой строки. Предполагается, что все значения полей текстовые, а сами поля разделены знаками табуляции. Этот формат поддерживается большей частью текстовых процессоров
Записи в одну строку, поля разделены пробелами	Access позволяет использовать пробел в качестве символа разделителя полей, однако использование такого разделителя очень неудобно для файлов, содержащих текстовые данные (в которые также может входить пробел)

Таким образом, наиболее рациональным разделителем является знак табуляции. Все остальные знаки между полями необходимо удалить.

Подготовка текстового файла с фиксированной шириной полей

Для файлов с фиксированной шириной полей начало каждого поля определяется по его позиции относительно начала строки. Строки имеют одинаковую длину и разделяются символами новой строки (обычно это пара символов — возврат каретки и перевод строки).

Недостатком файлов с фиксированной шириной полей является, во-первых, большой объем, т. к. в них содержится много пробелов, а во-вторых, сложности с определением позиций начала столбцов. Если же все-таки нужно использовать файл с фиксированной шириной полей, будьте внимательны и помните, что Access анализирует только первые 20 строк текстового файла, поэтому позиции разделения полей, определяемые автоматически, могут оказаться неверными.

Для подготовки файла с фиксированной шириной полей необходимо удалить все заголовки и итоговые строки. Он должен содержать только записи с импортируемыми данными, расположенными в фиксированных позициях.

Импорт текстового файла

Процесс импорта данных, хранящихся в текстовом файле, выполняется с помощью мастера импорта текстов.

1. Откройте базу данных, в которую нужно импортировать данные из текстового файла. Активизируйте окно базы данных, щелкнув по его заголовку левой кнопкой мыши.

2. Выберите команду ***Файл, Внешние данные, Импорт*** (file, get external data, import). Появится диалоговое окно импорт (см. рис.3.4).

3. В раскрывающемся списке ***Тип файла*** (files of type) диалогового окна ***Импорт*** выделите элемент ***Текстовые файлы*** (*.txt; *.csv; *.tab; *asc) (text files). Откройте папку, содержащую текстовый файл, который нужно импортировать, и выделите его. Нажмите кнопку ***Импорт*** (import). Появится первое диалоговое окно мастера импорта текстов (рис.3.15).

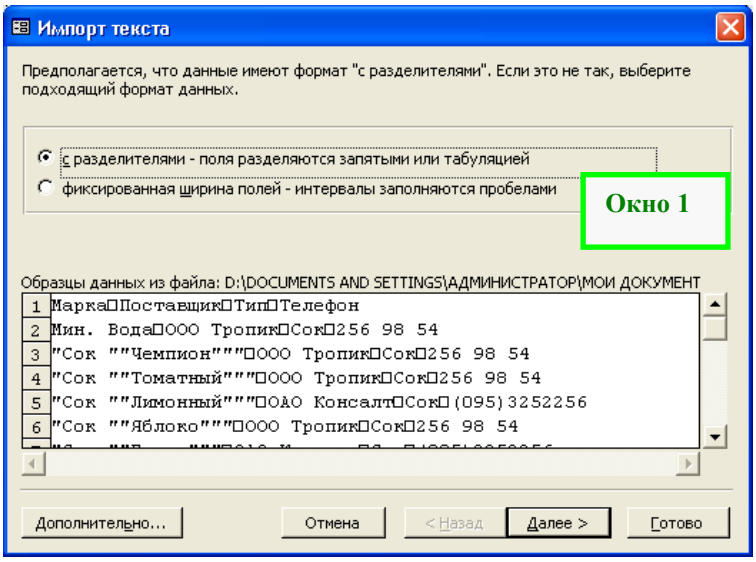


Рис.3.15. Первое окно Мастера импорта текста

4. В этом окне вы должны выбрать один из двух основных форматов текстовых файлов. Если в текстовом файле для определения конца поля используется разделительный символ, выберите переключатель ***С разделителями*** (delimited). Если же текстовый файл содержит записи с полями фиксированной ширины, выберите переключатель ***Фиксированная ширина полей*** (fixed width). В нижней части диалогового окна мастера импорта текстов выводится образец данных из файла, позволяющий визуально определить формат импортируемых данных.

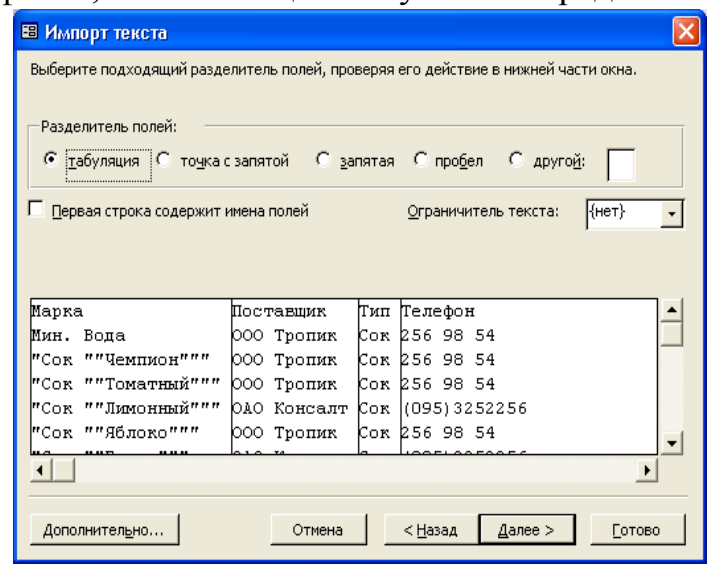


Рис.3.16. Окно Мастера для текста с разделителями

Нажмите кнопку ***Далее*** (next), чтобы перейти к следующему шагу мастера импорта текстов. Если на первом шаге мастера был выбран переключатель ***С разделителями***, появится диалоговое окно, изображенное на рис. 3.16, если же был выбран переключатель ***Фиксированная ширина нолей,*** то на втором шаге мастера появится диалоговое окно, изображенное на рис. 3.17.

5. Если файл содержит разделители полей, то укажите символ разделителя на втором шаге мастера импорта текстов (см. рис.3.16). Если в импортируемом файле

текстовые значения ограничены символом, отличным от двойных кавычек, то укажите ограничитель текстовых значений, выделив его в раскрывающемся списке ***Ограничитель текста*** (text qualifier). Если в первой строке импортируемого файла содержатся заголовки полей, установите флажок ***Первая строка содержит имена полей*** (first row contains field names). В нижней части окна вы будете видеть образец, по которому можно определить правильность сделанного вами выбора. Нажмите кнопку ***Далее*** (next), чтобы перейти к следующему шагу мастера импорта текстов.

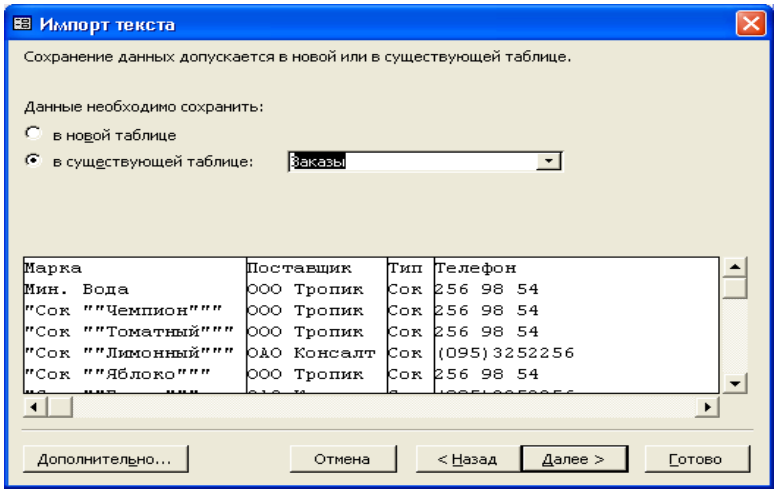


Рис.3.17. Окно Мастера для текста с фиксированной шириной

6. При импорте текстовых файлов с фиксированной шириной полей мастер импорта текстов попытается автоматически разделить поля. Просмотрев образец данных, можно указать дополнительное место разделения полей (черную стрелку), а также переместить или удалить существующий разделитель. После установки ширины всех полей нажмите кнопку ***Далее*** (next) для перехода к следующему шагу мастера импорта.

7. Чтобы создать новую таблицу и поместить в нее данные из текстового файла, выберите переключатель ***В новой таблице*** (in a new table) (рис.3.18). Чтобы добавить импортируемые данные к существующей таблице, выберите переключатель ***В существующей таблице*** (in an existing table) и имя таблицы из раскрывающегося списка. Нажмите кнопку ***Далее***, чтобы перейти к следующему шагу мастера импорта текстов.

При импорте данных в существующую таблицу просматриваются поля слева направо. Поэтому типы данных полей импортируемого текста должны совпадать с типами данных полей таблицы Access. В противном случае импортируемые данные могут попасть не в те поля. При этом чаще всего возникает множество ошибок, которые фиксируются в таблице ошибок импорта.

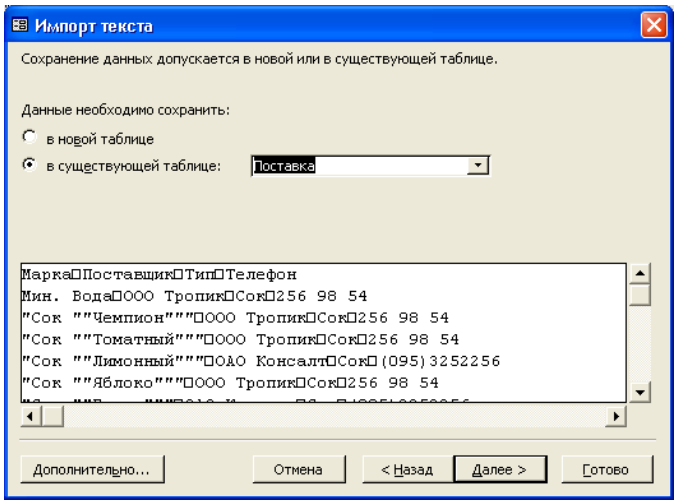


Рис.3.18. Окно выбора таблицы

Пятое диалоговое окно мастера импорта текстов изо-

бражено на рис.3.19. Оно очень похоже на соответствующее диалоговое окно мастера импорта электронных таблиц. В нем можно гибко настроить процедуру импорта: пропустить некоторые поля, изменить их имена или ввести названия полей, если они отсутствовали в текстовом файле, изменить типы данных, присвоенных системой по умолчанию, проиндексировать поля. После того как параметры импорта всех полей будут определены, переходите к следующему шагу мастера импорта.

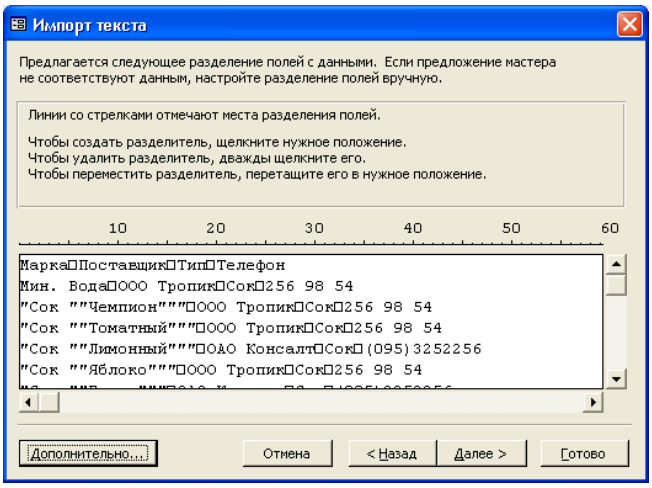


Рис.3.19. Окно настройки процедуры импорта

В окнах также нужно определить ключевое поле новой таблицы, определить имя таблицы, в которую будут помещены импортируемые данные. Затем нажмите кнопку ***Готово*** (finish), чтобы завершить процесс импорта..

В результате будет создана таблица с указанным на последнем шаге названием. Если в процессе импорта текстового файла Access обнаруживает ошибки, то они, как обычно, фиксируются в таблице ошибок импорта. После окончания импорта данных будет выведено сообщение об успешном завершении операции.

Исправление ошибок после импорта текстового файла

Как и при импорте данных из таблицы Excel исправление ошибок импорта целесообразно осуществить путем исправления исходного текстового файла и повторения процедуры импорта.

Задание3.3. Экспорт данных БД в другие форматы

Экспорт данных БД в другие форматы

СУБД Access предоставляет широкие возможности для экспорта данных. Экспорт из таблиц базы данных может осуществляться в любой из форматов, для которых поддерживается операция импорта данных, в том числе в текстовые файлы, электронные таблицы, документы html и xml и составные документы MS Word. Экспортировать данные Access можно не только из таблицы, но и из запроса. Можно экспортировать не только данные, но и объекты Access — формы, отчеты.

Экспорт в электронную таблицу

Для экспорта данных в электронную таблицу Microsoft Excel можно использовать следующую процедуру:

1. Откройте базу данных Access, из которой нужно экспортировать таблицу. Активизируйте окно базы данных, щелкнув по его заголовку левой кнопкой мыши;

2. Раскройте список таблиц, щелкнув по ярлыку таблицы на панели объектов окна базы данных, и выберите в этом списке имя экспортируемого объекта;

3. Выберите команду меню ***Файл, Экспорт*** (file, export) или команду ***Экспорт*** (export) из контекстного меню. Появится диалоговое окно ***Экспорт объекта*** (рис.3.20);

4. В раскрывающемся списке ***Тип файла*** (save as type) этого диалогового окна выберите тип файла, в который вы хотите экспортировать таблицу Access. Найдите папку, в которую нужно сохранить экспортируемый файл, в текстовом поле ***Имя файла*** (file name) введите имя файла и нажмите кнопку ***Экспорт.***

5. Если процедура экспорта закончится успешно, Access создаст новый файл, с которым вы сможете работать с помощью электронной таблицы.

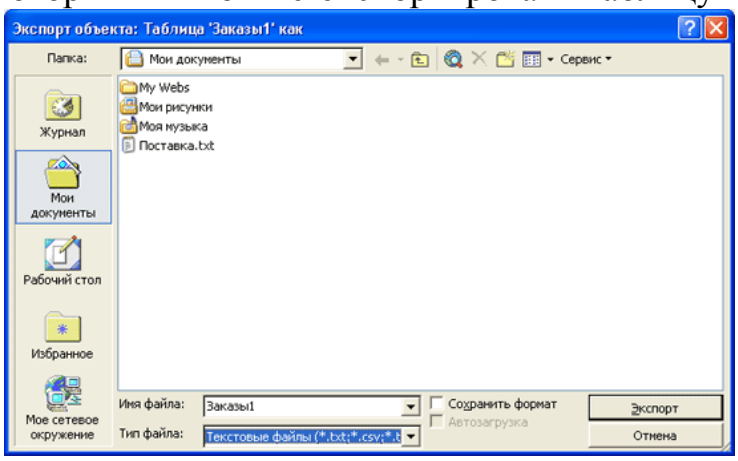


Рис.3.20. Окно ***Экспорт данных***

Экспорт данных в текстовый файл

Экспорт данных из таблицы Access в текстовый формат применяется обычно тогда, когда нужно использовать эти данные в приложении, которое может обмениваться с другими приложениями только через текстовые файлы. Экспортировать можно в двух форматах:

- как в файлы с разделителями, а также как
- файлы с фиксированной шириной полей. Процедура экспорта похожа на описанную выше процедуру экспорта в другие типы файлов:

1. Откройте базу данных Access, из которой требуется экспортировать таблицу. Активизируйте окно базы данных, щелкнув по его заголовку левой кнопкой мыши;

2. Раскройте список таблиц и выделите в нем нужную таблицу;

3. Выберите команду ***Файл, Экспорт*** (file, export). Появится диалоговое окно ***Экспорт объекта*** (export table to) (см. рис.3.20);

4. В раскрывающемся списке ***Тип файла*** (save as type) этого диалогового окна выберите элемент ***Текстовые файлы*** (text files). Перейдите в папку, в которой нужно сохранить экспортируемый файл, в текстовом поле ***Имя файла*** (file name) укажите имя файла и нажмите кнопку ***Экспорт*** (export);

5. Запустится Мастер экспорта текстов, использование которого аналогично использованию Мастера импорта текстов, описанного ранее. Единственным исключением является то, что при экспорте данных нельзя переопределить названия полей. В первом окне мастера в качестве формата экспорта показан файл при выбранном флажке ***С разделителями*** (рис.3.21).

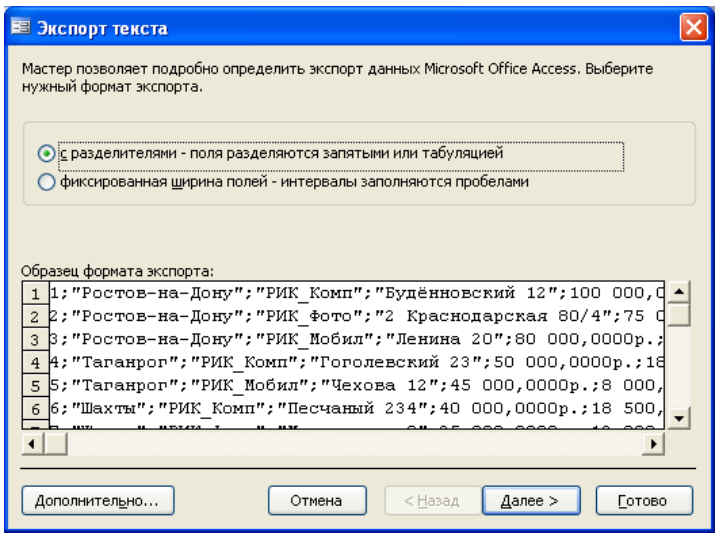


Рис.3.21. Экспорт файла С разделителями

На следующем рисунке (рис.3.22) для того же файла флажок установлен в поле ***Фиксированная ширина полей.***

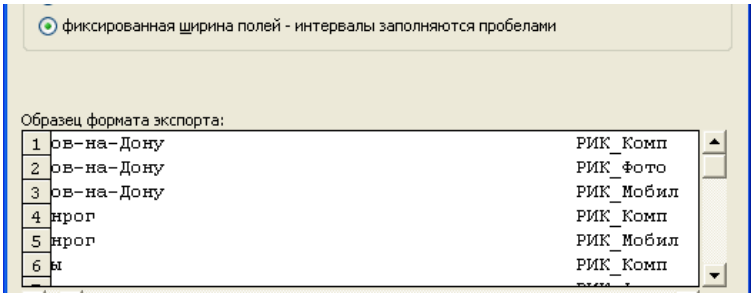


Рис.3.22. Экспорт файла с Фиксированной шириной полей

При нажатии кнопки ***Дополнительно*** откроется окно ***Спецификация экспорта*** (рис.3.23)**.** Оно позволяет выбрать ширину полей, тип разделителя, сохранить спецификацию экспорта/импорта.

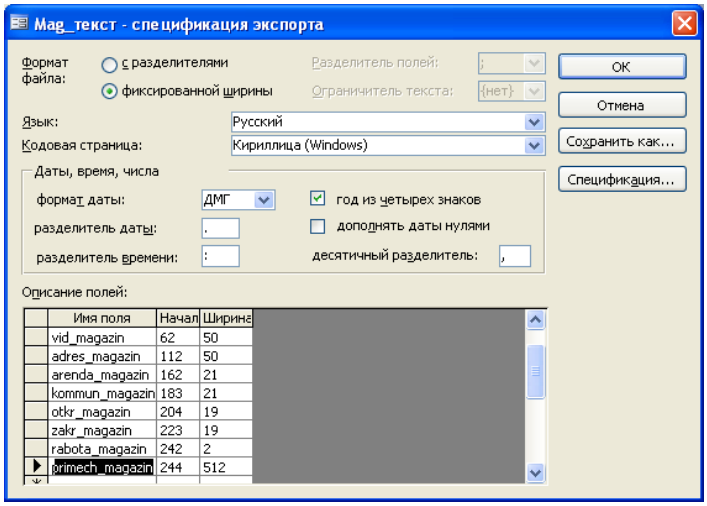


Рис.3.23. Окно спецификации после нажатия кнопки ***Дополнительно***

В последнем диалоговом окне Мастера вы определяете имя текстового файла. После нажатия кнопки ***Готово*** (finish), при успешном выполнении процедуры, вы получите текстовый файл выбранного формата. На рисунке 3.24 показан текстовый файл, экспортированный из таблицы в режиме с фиксированной шириной.

Порядок записей в созданном путем экспорта текстовом файле определяется первичным ключом таблицы. Если у экспортируемой таблицы нет первичного ключа, записи экспортируются в том порядке, в котором они были добавлены в таблицу.

Использование буфера обмена Windows для импорта и экспорта данных

При импорте или экспорте данных, созданных с помощью приложения Windows, можно воспользоваться системным буфером обмена. Это средство позволяет применять стандартные команды Windows и быстро перемещать данные из одного приложения Windows в другое. Это особенно удобно при добавлении или замене существующих данных в таблице Access.

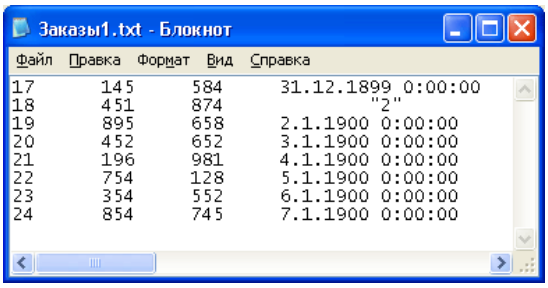


Рис.3.24. Экспортированный файл с разделителями

Однако этот метод имеет одно существенное ограничение — структура таблицы, в которую импортируются данные, должна соответствовать данным, передаваемым через буфер обмена. Например, для правильной вставки данных из строк рабочего листа Excel в таблицу Access необходимо, чтобы типы данных полей таблицы Access соответствовали типам данных каждого столбца, скопированного в буфер обмена.

Чтобы добавить новые записи в таблицу Access при помощи системного буфера обмена:

1. Откройте файл с данными, из которого они будут копироваться в буфер обмена (например, файл Excel);

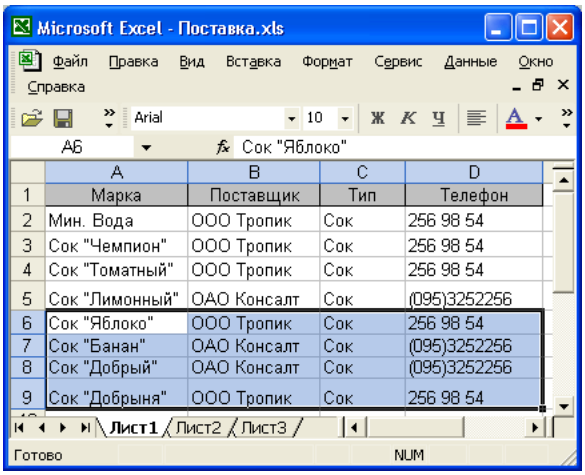


Рис.3.25. Выделение данных для импорта

2. Выделите данные, которые нужно добавить в таблицу Access (рис.3.25). При этом не обязательно выделять все столбцы, соответствующие полям таблицы Access. В столбцы Access, не включенные в выделенный диапазон, будут добавлены пустые значения или значения по умолчанию.

3. Выберите команду ***Правка, Копировать*** (edit, copy) или нажмите комбинацию клавиш ***Ctrl+C***, чтобы скопировать выделенный диапазон ячеек в системный буфер обмена;

4. Откройте таблицу, в которую нужно добавить данные, в режиме таблицы;

5. Выберите команду ***Правка, Добавить из буфера*** (edit, paste append). Если не возникнет ошибок, появится окно, запрашивающее подтверждение на добавление записей в таблицу (рис.3.26). Нажмите кнопку ***Да*** (yes). Новые записи будут добавлены в конец таблицы Access.

Ошибки при добавлении в БД могут возникнуть при несоответствии типов данных или при совпадении данных в ключевом поле.

Простейшим способом экспорта данных из таблицы Access в другое приложение Wndows также является копирование содержимого таблицы Access в буфер обмена с последующей вставкой в документ другого приложения.

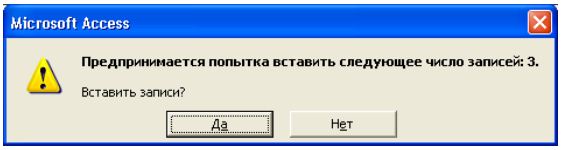


Рис.3.26. Запрос подтверждения при импорте данных

Вопросы для самопроверки

1. Что такое экспорт и импорт данных в БД? С какой целью они выполняются?
2. В чем заключается подготовка данных для импорта из электронных таблиц?
3. Какова технология импорта данных из электронных таблиц?
4. В чем заключается коррекция данных после импорта их электронных таблиц?
5. В чем заключается подготовка данных для импорта из текстового файла?
6. Какова технология импорта данных текстового файла?
7. В чем различие между присоединением таблицы и импортом таблицы БД?
8. Каким образом осуществляется импорт таблиц из БД?
9. Какова технология экспорта данных в электронную таблицу?
10. Какова технология экспорта данных в текстовый файл?