Урок №1

Теория баз данных

Содержание

Введение.

Понятие по БД, СУБД. Типы БД

Сегодня мы начинаем изучение нового курса «Базы данных». С данным понятием Вы уже знакомы, поскольку оно уже вжилось в наше настоящее. Практически каждый из Вас пользуется базой данных каждый день, хотя, возможно, этого и не замечает. Примером баз данных могут служить записная книжка, телефонный справочник, энциклопедии, библиотечный каталог и т.д., в которых сохраняется определенная необходимая информация. Так, в телефонном справочнике содержится информация о людях: их имена, фамилии, телефоны, адреса и т. д. В энциклопедиях — данные по определенной предметной области.

Итак, подытоживая это, можно сказать, что ***база данных (БД)*** — это совокупность данных, связанных между собой определенной тематикой. Эти данные сохраняются на машинных носителях системы в упорядоченном виде.

Наряду с понятием баз данных существует понятие ***системы управления базами данных (СУБД)*** — это компьютерная программа, которая позволяет управлять базой данных, то есть создавать и работать с ней.

Обычно современные СУБД содержат ***следующие составляющие:***

- ядро СУБД, которое отвечает за управление данными и их журнализацию;

- процессор языка базы данных, обеспечивающий оптимизацию запросов и переводом их на машинный язык;
- подсистема поддержки времени выполнения (*runtime*), которая создает пользовательский интерфейс и обеспечивает его взаимосвязь с базой данных;
- сервисные программы, то есть набор утилит, которые предоставляют дополнительные возможности по обслуживанию информационной системы.

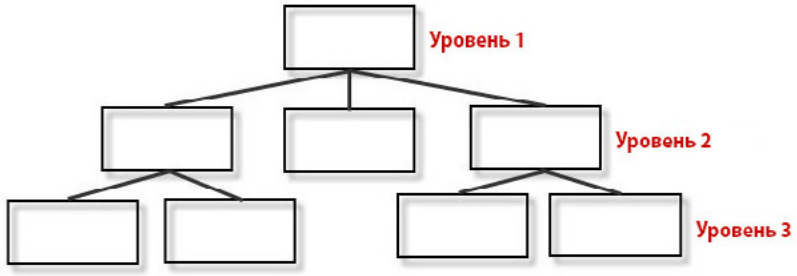
В современных компьютерных информационных системах чаще всего применяются ***4 типа моделей БД и СУБД:***

1) иерархические;
2) сетевые;
3) реляционные;
4) объектно-ориентированные.

Иерархические базы данных

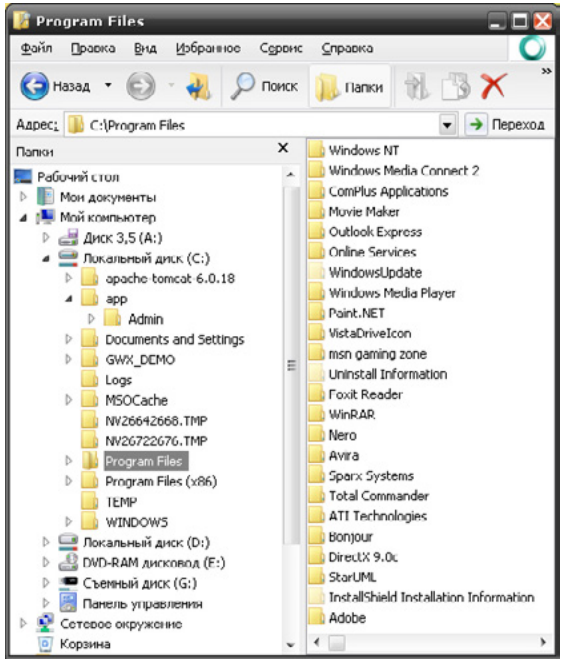
Это самый первый вид баз данных, используемый мирным населением (☺). Такую базу данных графически можно представить в виде дерева, которое состоит из объектов различных уровней. Верхний уровень занимает один объект, второй — объекты второго уровня и так далее. Между объектами существуют связи «родитель-потомок» и поэтому каждый объект может включать в себя несколько объектов более низкого уровня.

Схематически такая модель будет выглядеть следующим образом:



Типичным примером иерархической базы данных является каталог папок Windows — «Проводник». Верхний уровень дерева занимает «Рабочий стол», на втором уровне находятся папки «Мой компьютер», «Мои документы»,

«Сетевое окружение» и «Корзина», которые являются дочерними по отношению к Рабочему столу. Далее размещаются объекты третьего уровня и т.д.



Приведем еще ряд ***примеров иерархических баз данных и СУБД:***

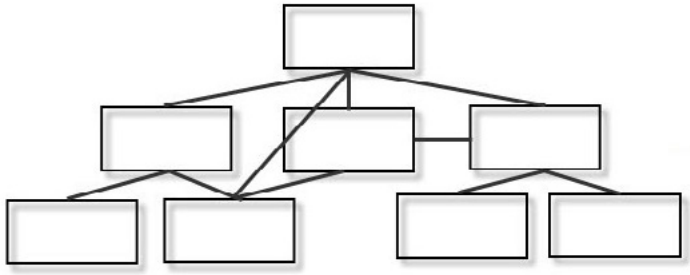
- реестр Windows;
- серверы каталогов, такие как LDAP Active и Directory;
- System-2000 от международной компании SAS-Institute;
- Information Management System (*IMS*) фирмы IBM.

Основным ***недостатком*** таких баз данных является то, что в случае, если данные не имеют такой структуры, то есть их нельзя представить в виде дерева, возникает много сложностей при построении базы данных. При этом такая база данных будет непродуктивной.

К недостаткам следует отнести и тот факт, что в случае поиска данных нельзя направить поток поиска снизу вверх.

Сетевые базы данных

Являются обобщающей моделью иерархических. Каждый объект в такой базе данных может иметь более одного родительского элемента. Простыми словами, в сетевой базе данных элементы могут быть связаны между собой произвольным образом. Схематическое изображение данной модели представлено ниже:



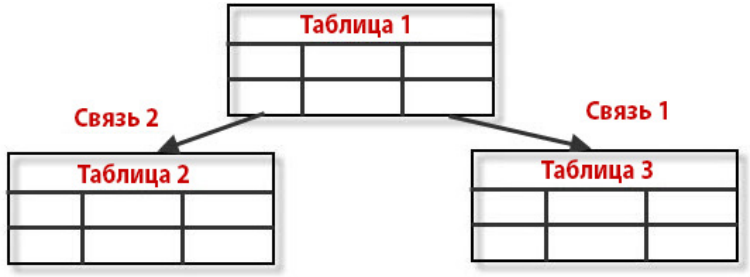
Характерным примером такой базы данных является Глобальная сеть Интернет, в которой гиперссылки связывают между собой сотни миллионов документов. Все эти связанные документы представляют собой единую распределеную сетевую базу данных.

Примерами сетевой модели могут служить такие базы данных как СООБЗ (сетевая объектно-ориентированная база знаний) Cerebrum и CronosPRO.

Недостатком такой модели является низкая производительность, которая приводит к тому, что даже простые запросы выполняются достаточно сложно. Кроме того, при работе с такой базой данных, прикладной программист должен знать много терминов, изучить несколько внутренних языков баз данных, подробно представлять логическую структуру базы данных для навигации среди различных экземпляров, наборов, записей и тому подобное.

Реляционные (табличные) базы данных

Слово «реляционная» происходит от английского «*relation*» (*связь*). В такой модели, все данные в базе данных хранятся в виде двухмерной таблицы, состоящие из столбиков и строк. Каждая таблица содержит совокупность данных одного типа, которые имеют одинаковые свойства. Столбики такой таблицы называются полями или атрибутами и содержат информацию о свойствах объекта, а строки (*записи, кортежи*) содержат значение конкретного свойства.



Примеры реляционных СУБД:

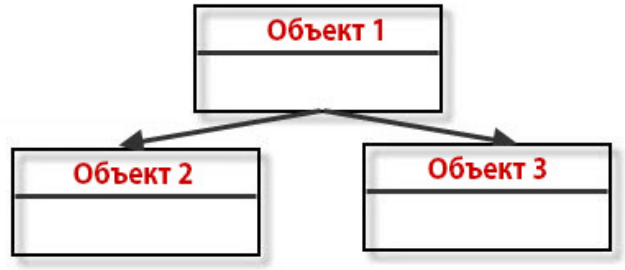
- Microsoft SQL Server;
- Oracle;
- Interbase/Firebird;
- Microsoft Access;
- MySQL.

Основное преимущество этой модели данных — это простота и наглядность базы данных при ее проектировании. Именно поэтому она является самой распространенной моделью БД и именно на такой модели мы и остановимся при изучении данного курса.

Объектно-ориентированные базы данных

Данные в такой модели базы данных оформлены в виде моделей объектов, которые используют приложения, управляющие внешними событиями. Стоит отметить, что ряд объектно-ориентированных баз данных разработаны для тесного взаимодействия с объектно-ориентированными языками программирования (C++, C #, Java, Python, Visual Basic .NET, Smalltalk и т.д.) и используют такую же модель, что и остальные. Такие базы данных

обычно рекомендованы для тех случаев, когда требуется высокопроизводительная обработка данных, имеющих сложную структуру.



Объектно-ориентированные базы данных должны соответствовать двум основным критериям: система должна быть объектно-ориентированной и представлять собой базу данных. Фактически, такая модель сочетает в себе характеристики ОО языков программирования (наследование, инкапсуляцию, полиморфизм) и баз данных (целостность данных, безопасность, стабильность, транзакции, архивирование, восстановление из резервной копии, запросы и т.д.).

Примеры объектно-ориентированных СУБД:

- IBM Lotus Notes/Domino;
- Jasmine;
- ObjectStore;
- db4objects;
- ODB-Jupiter.

12 правил Кодда,

которым должна соответствовать реляционная СУБД

Поскольку реляционная модель баз данных является самой распространенной за счет простоты построения и работы с базой данных, то именно на ней мы и сосредоточим свое внимание. Но перед тем как начать усвоение практических навыков, нужно усвоить основные правила, которым должна соответствовать любая реляционная база данных. Эти правила были предложены в 1970 г. английским математиком Е.Ф. Коддом из компании IBM.

В терминологии Кодда таблицы реляционной базы данных называются ***отношениями*** (*relation*) — отсюда и название самой модели. Всех правил — 12 (на самом деле 13). Основное правило, которое не входит в перечень, говорит о том, что ***любая реляционная СУБД должна полностью управлять базой данных, используя связи между данными***.

Все остальные 12 правил представлены ниже:

1. ***Правило информации*** (*The Information Rule*). Вся информация в базе данных должна быть представлена в виде данных, хранящихся в ячейках таблиц. Причем, порядок записей (строк) в реляционной таблице не должен влиять на суть хранимых данных.

2. ***Правило гарантированного доступа*** (*Guaranteed Access Rule*). Доступ к данным в реляционной БД должен обеспечиваться путем использования комбинации имени таблицы, первичного ключа и имени столбца.
3. ***Правило поддержки отсутствующих значений*** (*Systematic Treatment of Null Values*). Указывает на то, что любые отсутствующие данные можно представить с помощью недействительного значения NULL.
4. ***Правило интерактивного каталога, основанного на реляционной модели*** (*Active On-Line Catalog Based on the Relational Model*). Указывает на то, что реляционная база данных должна сама себя описывать. Иными словами, база данных должна содержать набор реляционных системных таблиц, описывающих структуру самой базы данных. При этом нужно, чтобы доступ к этим таблицам был аналогичный доступу к обычным таблицам базы данных.
5. ***Правило исчерпывающего подъязыка данных*** (*Comprehensive Data Sublanguage Rule*). Реляционная СУБД должна поддерживать хотя бы один реляционный язык, который имеет:
   - линейный синтаксис;
   - может использоваться как интерактивно, так и для прикладных приложений;
   - поддерживает операции работы с данными (вставка, удаление, обновление), обеспечивает их целостность и тому подобное.

6. ***Возможность модификации представлений*** (*View Updating Rule*). Все представления (*view*) должны поддерживать операции манипулирования данными, подобно реляционным таблицам: выборка данных, вставка, обновление и их удаление.
7. ***Правило высокоуровневых операций вставки, обновления и удаления*** (*High-Level Insert, Update, and Delete*). Здесь акцентируется внимание на том, что базы данных по своей природе ориентированы на множества. Оно требует, чтобы операции вставки, удаления и обновления можно было выполнять как над единичными данными, так и над множествами записей (строк).
8. ***Правило независимости физических данных*** (*Physical Data Independence*). Указывает на то, что данные базы данных не должны зависеть от используемых способов сохранения данных на носителях, от аппаратного обеспечения компьютера, на которых установлена СУБД.
9. ***Правило независимости логических данных*** (*Logical Data Independence*). Представление данных в прикладных приложениях не должно зависеть от структуры реляционных таблиц. Иными словами, если в базе данных произошла смена ее структуры (некоторые таблицы были разделены на несколько, добавлены другие таблицы и т.д.), то в прикладной программе отображение данных не должно измениться.
10. ***Правило независимости контроля целостности*** (*Integrity Independence*). Вся информация, необходи-

мая для обеспечения целостности данных должна храниться в системных таблицах. Используя эти данные, СУБД должна проверять все входящие данные на соответствие их условиям целостности.

11. ***Правило независимости распределения*** (*Distribution Independence*). Позволяет базе данных быть распределенной, то есть находиться на разных компьютерах. Причем, перенос данных на другой компьютер не должен повлиять на ее работоспособность.
12. ***Правило согласованности языков разных уровней*** (*The Nonsubversion Rule*). Если используется низкоуровневый язык для доступа к данным, то он не должен игнорировать правила безопасности и целостности данных, которые поддерживаются языком более низкого уровня.

Стоит отметить, что на самом деле, из-за суровости данных правил, не все современные реляционные СУБД их придерживаются.

Начало работы с Microsoft Access.

Достоинства и недостатки

Одной из самых наглядных и простых реляционных СУБД на сегодняшний день является СУБД Microsoft Access, которая входит в состав пакета продуктов Microsoft Office. Не вдаваясь в лабиринты истории создания данной СУБД, рассмотрим ее преимущества и недостатки, а также основные принципы ее строения.

Основное ***преимущество*** MS Access — это ***визуальность***. Средствами данной СУБД можно легко просмотреть структуру базы данных, содержимое таблиц, удобно и быстро осуществить модификацию данных и тому подобное. Однако, MS Accesss является desktop-приложением и поэтому становится невозможным организация клиент-серверного взаимодействия приложения и базы данных. Это основной недостаток данной СУБД. К существенным недостаткам MS Access также следует отнести:

- Невозможность сохранения данных большого объема;
- Низкий уровень безопасности;
- Неполноценная поддержка языка данных (*SQL*).

Фактически, MS Access предназначен для персонального использования, а в случае необходимости разработки базы данных, например, крупного предприятия, лучше воспользоваться серверной СУБД, например, Oracle, MS SQL Server, MySQL и тому подобное.

Как вы знаете, сама по себе база данных — это файл (или группа файлов), записанных в специальном формате. Диспетчером данных в MS Access, который выполняет загрузку и сохранение данных в базе данных пользователя, является ядро базы данных ***Microsoft Jet***. Начиная с MS Access 2000, использовалось ядро Microsoft Jet версии 4.0. Эта версия Jet поддерживает двухбайтовое представление символов (формат Unicode), улучшена поддержка языка SQL спецификации ANSI SQL 92 (но она еще далека от классической реализации SQL), реализована встроенная поддержка интерфейсов OLE DB, благодаря которой Microsoft Access может быть использован для разработки клиентских приложений, взаимодействующих с другими СУБД и др.

Начиная с MS Access 2007, используется улучшенная версия ядра Microsoft Jet — ***Microsoft Data Engine (MSDE)***. Это обновленное ядро обеспечивает интеграцию с Windows SharePoint Services 3.0 и Microsoft Office Outlook 2007, а также новые возможности (например, создание полей подстановок, которые одновременно допускают несколько значений).

Проектирование базы данных

Прежде чем начать строить свою базу данных, ее необходимо спроектировать, то есть определить, для сохранения каких именно данных она предназначена, кто будет ею пользоваться и тому подобное. Сам процесс построения базы данных имеет свой жизненный цикл, который проходит ***ряд этапов:***

1. ***Анализ данных*** включает в себя этапы формулировки и анализа требований.
2. ***Проектирование базы данных*** предусматривает непосредственное создание объектов и связей базы данных.
3. ***Ввод в эксплуатацию*** включает:
   - перенос базы данных на рабочую машину;
   - написание технической документации;
   - передача пользователю;
   - анализ функционирования, поддержка и модификация базы данных.

Этап анализа данных является самым сложным и самым главным. Как правило, он осуществляется в тесном взаимодействии с заказчиком. Его можно разделить еще на ряд ***подэтапов***:

1. ***Концептуальное проектирование***, включающее сбор, анализ и редактирование требований к данным. Для этого осуществляются следующие действия:

- исследование предметной области, изучение ее структуры;
- определение пользователей будущей базы данных;
- определение требований пользователей к базе данных, то есть, что каждый из пользователей хочет получить в результате создания БД;
- определение информационных объектов, которые должны присутствовать в базе данных и связей между ними.

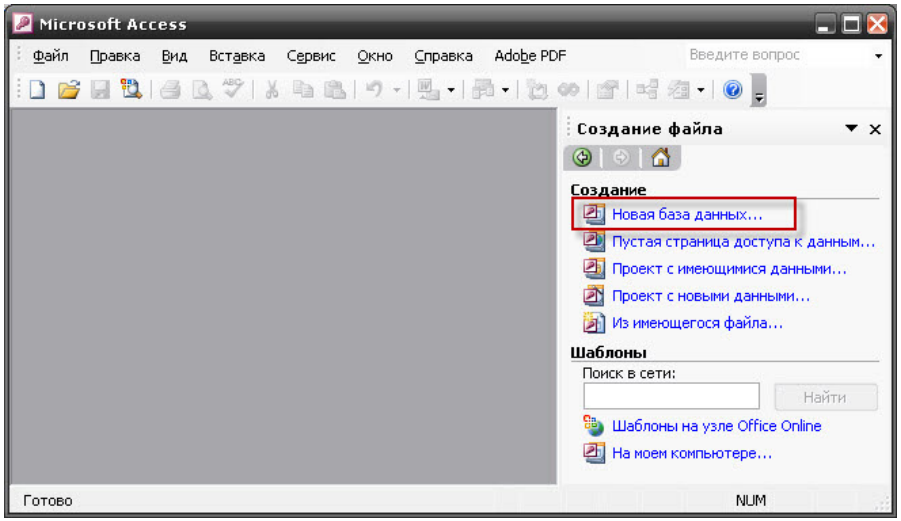
2. ***Логическое проектирование*** — преобразование требований к данным в конкретные объекты данных. В результате, на выходе мы получаем ориентировочную структуру базы данных.
3. ***Физическое проектирование*** — определение особенностей хранения данных (осуществляется группировка данных, создаются индексы), методов доступа и т.д.

Фактически, можно сказать, что концептуальный уровень проектирования позволяет представить базу данных с точки зрения аналитика, логический — с точки зрения программиста, а физический — администратора.

Создание базы данных

Время перейти к практике. После того, как Вы спроектировали базу данных, ее необходимо создать. Для демонстрации работы с MS Access мы воспользуемся версией, которая входит в состав пакета Microsoft Office 2003. Принцип создания базы данных в низших и старших версиях немного отличается, но в целом процесс похож. Поэтому, разобравшись с какой-либо одной версии, разобраться в другой будет не слишком сложно.

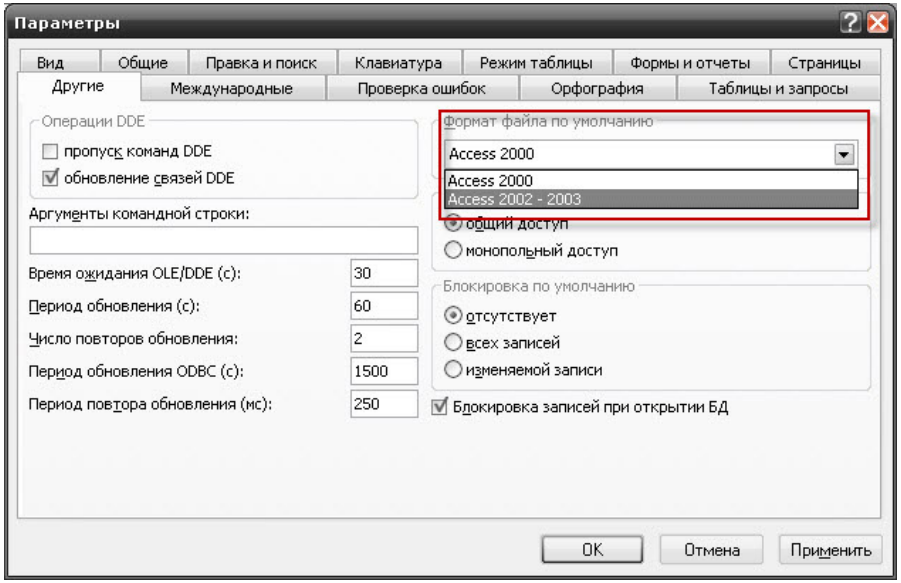
Итак, для того, чтобы создать базу данных средствами MS Access нужно прежде всего его запустить. После этого нужно выбрать пункт меню ***Файл> Создать*** (***Ctrl + N***). Перед Вами появится новая боковая панель с меню, в котором необходимо выбрать пункт ***"Новая база данных"***.



Это позволит создать новую пустую базу данных, которая будет храниться в файле с именем, которое Вы ей зададите на следующем шаге, с расширением ***.mdb*** и размером около 60 Kb.

Для открытия уже существующей базы данных следует воспользоваться пунктом меню ***Файл-> Открыть (Ctrl + O)***.

После создания новой базы данных, Вы можете обеспечить ее совместимость с базами данных других (низших) версий СУБД MS Access. Для этого нужно изменить параметры вновь БД с помощью пункта главного меню ***Сервис-> Параметры***. Затем выбрать закладку «***Другие***» и в разделе «***Формат файла по умолчанию***» выбрать необходимую версию.



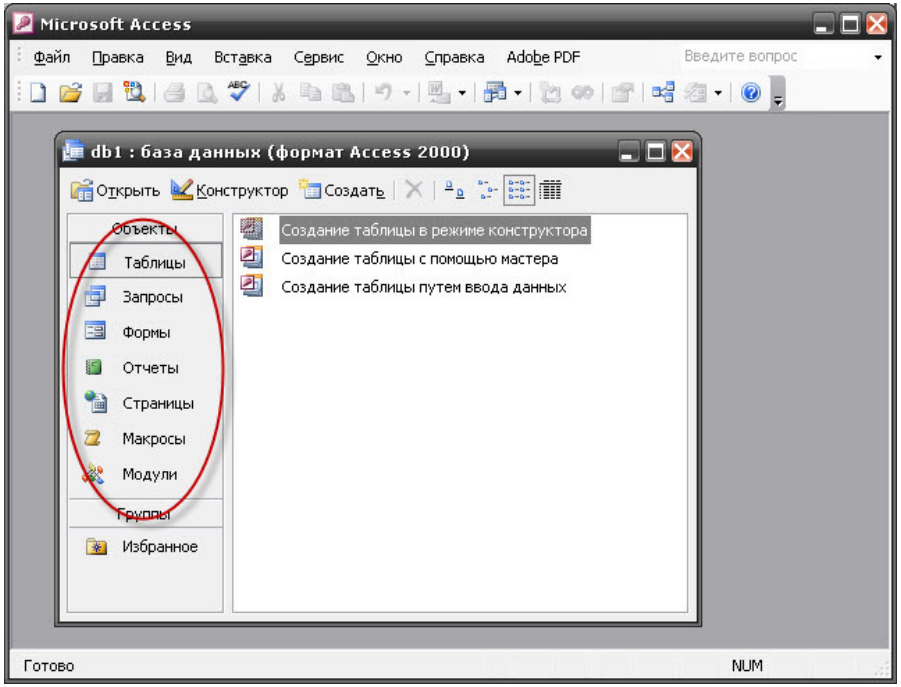
Следует отметить, что данная возможность существует, начиная с версии MS Access 2002.

После того, как Вы определились с выбором и проделали все необходимые действия, перед Вами должен появиться новая база данных. С чем Вас и поздравляем (☺).

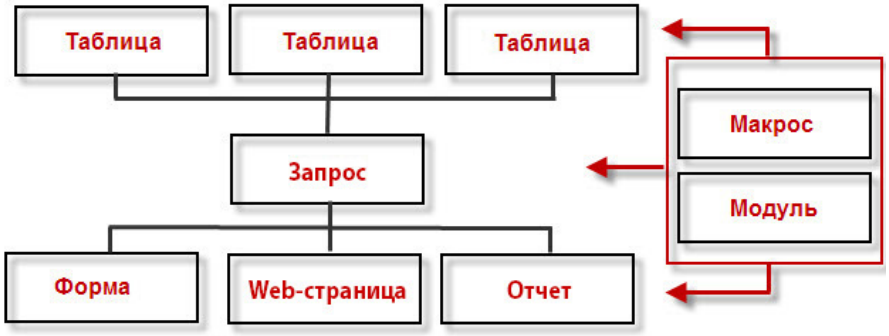
Составляющие части базы данных MS Access

СУБД MS Access — это совокупность объектов различного типа и назначения, используемых для хранения, отображения и вывода на печать данных базы, а также могут содержать созданный Вами код. К объектам СУБД MS Access ***относятся***: таблицы, запросы, формы, отчеты, страницы, макросы и модули.

Все объекты представлены в окне базы данных MS Access, с которого и начинается Ваша работа в среде.



Таблицы являются местом хранения данных, все остальные обьеты — это инструменты, которые предназначены для работы с таблицами и для создания пользовательского интерфейса. Схематично взаимодействие объектов СУБД можно изобразить следующим образом:



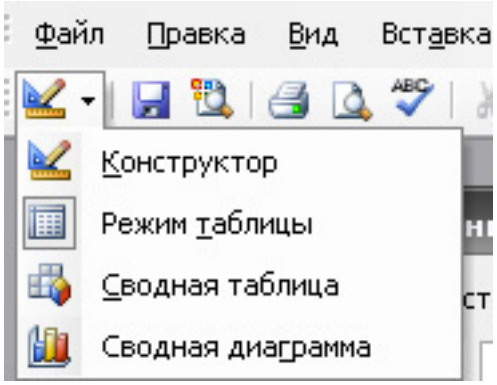
- ***Таблицы*** — это объекты, которые создает пользователь для хранения информации, то есть свои данные. Access позволяет создавать и редактировать таблицы. Таблицы состоят из строк (записей, кортежей) и столбцов (полей, атрибутов). В свою очередь, каждый столбик имеет свой формат. Сразу отметим, что таблицы можно импортировать или делать ссылки на таблицы, размещенные в других СУБД.
- ***Запросы*** создаются пользователем для выборки необходимых данных из одной или нескольких связанных таблиц. Результатом выполнения запросов является таблица, которая может быть использована вместе с другими таблицами БД при обработке данных. В MS Access запрос может формироваться в варианте ***Query By Example*** (*запрос по образцу*) или с помощью язы-

ка структурированных запросов ***Structured Query Language***. Отметим, что с помощью запросов можно также управлять построением БД, например, создавать новые таблицы, на основе уже существующих.

- ***Формы***. Позволяют создать интерфейс, с помощью которого пользователи могут работать с таблицами в более удобной форме. Пользовательский интерфейс можно создавать как на основе таблиц, так и на основе запросов (в основном).
- ***Отчеты*** — это объект, который используется для вывода данных на печать в удобном виде. Стоит отметить, что отчеты строятся преимущественно на основе запросов.
- ***Страницы*** — это HTML-страницы, которые позволяют представить данные в окне браузера, создать Web-интерфейс доступа к данным. Это позволяет обращаться к БД через Internet.
- ***Макросы*** — это простые программы, которые позволяют с помощью готовых команд автоматизировать часто повторяющихся операций. Каждое действие реализуется макрокомандой.
- ***Модули*** — это объекты БД, содержащие программный код, написанный на языке VBA (Visual Basic for Application). Как правило, они используются как замена макросам, или же для написания собственного кода реакции на события.

Режимы работы MS Access

MS Access позволяет работать с каждым объектом базы данных в различных режимах. Стоит отметить, что в основном эти режимы индивидуальные для каждого объекта, но есть ряд общих. Режим работы можно изменить с помощью кнопки на панели инструментов «Вид» в любой удобный для Вас момент времени



Рассмотрим кратко существующие режимы работы.

Таблицы базы данных имеют 4 режима работы:

1. ***Режим конструктора***, в котором Вы можете изменять структуру таблицы и ее внешний вид. Этот режим предназначен для программиста баз данных и в пользовательских приложениях его нужно блокировать.
2. ***Режим таблицы***. Позволяет просмотреть данные в привычном для пользователя режиме таблицы.
3. ***Режимы итоговой таблицы и итоговой диаграммы.***

Предусматривает просмотр данных в соответствующем виде.

Объект базы данных «***Запросы***» имеет дополнительный режим работы — ***Режим SQL***, который позволяет перейти в окно для построения запросов на языке SQL. С этим режимом Вы познакомитесь через несколько пар более подробно.

Объект «Формы» имеет режимы работы, аналогичные объекта «Таблицы», и включает дополнительный ***Режим формы***. В этом режиме вы можете посмотреть, как будет выглядеть форма для пользователя. Режим конструктора для формы позволят с легкостью разместить необходимые элементы управления, через которые будет происходить взаимодействие пользователя с данными базы данных.

Отчеты имеют 3 режима работы:

1. ***Режим конструктора*** — аналогичный предыдущим.
2. ***Режим предварительного просмотра*** (*Print Preview*) позволяет просмотреть отчет в таком виде, который он будет иметь при печати.
3. ***Режим просмотра образца*** (*Layout Preview*) очень удобный для проверки структуры и макета отчета. В этом режиме на экран выводятся только те данные, которые необходимы для заполнения каждого из элементов отчета.

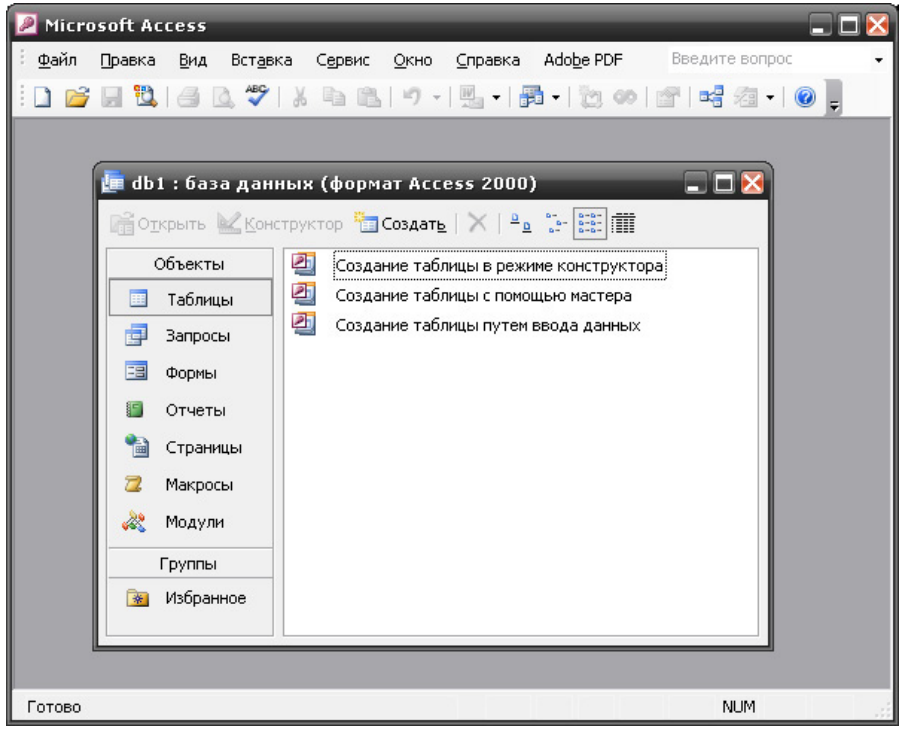
Страницы имеют также 3 режима:

1. ***Режим конструктора***.
2. ***Режим просмотра страницы***, который позволяет

просмотреть внешний вид страницы в окне базы данных.

3. ***Режим предварительного просмотра веб-страницы***, который запускает страницу в браузере для просмотра.

Стоит отметить, что в MS Access выделяют еще один режим работы — ***начальный***. Этот режим работы касается базы данных в целом. Он устанавливается в самом начале работы, и вернуться к нему можно только закрыв все таблицы, формы, отчеты, запросы и т.д., если они открыты. Он выглядит следующим образом:



Таблицы.

Порядок и методы создания таблиц в MS Access

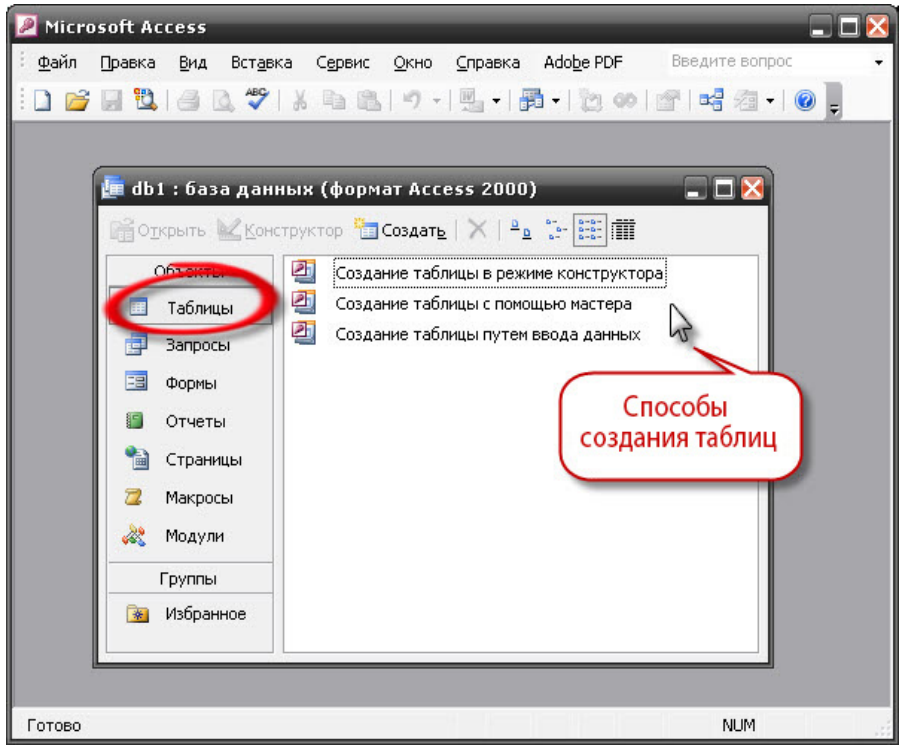
Создав базу данных, нужно заполнить ее данными. Данные, как уже было не раз сказано, хранятся в таблицах, которые связаны между собой связями. Все данные должны быть организованы в таблицах таким образом, чтобы обеспечить объединение разнородной информации, исключить дублирование данных и обеспечить быстрый и эффективный доступ к данным. Важной особенностью таблиц является то, что представленная в них информация описывает, как правило, однотипные предметы или объекты. По сути, это список объектов, которые имеют общие характеристики. Характеристики объектов указываются в полях таблицы, а их значение в соответствующих ячейках строки таблицы, образуя запись.

СУБД MS Access позволяет создавать таблицы одним из ***трех способов***:

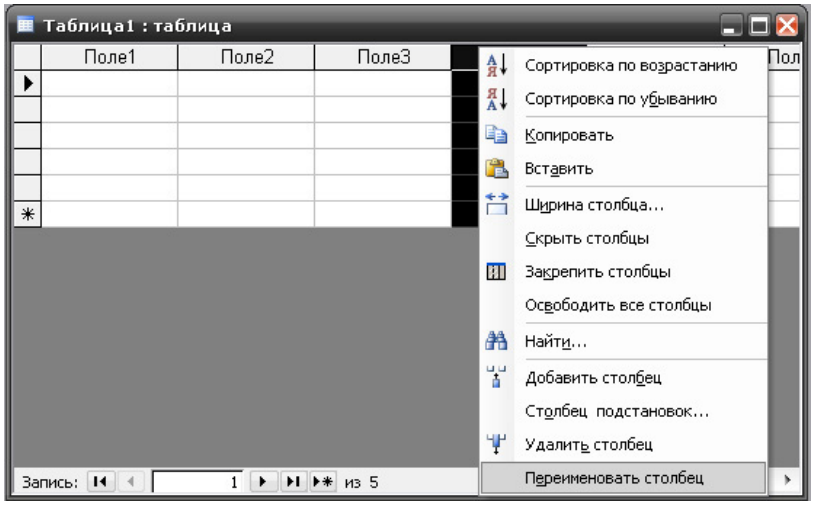
1. В режиме конструктора;
2. С помощью Визарда, то есть на основе шаблона;
3. Путем ввода данных в таблицу

Выбрать необходимый способ можно с помощью необходимого ярлыка в окне активной базы данных,

наряду со списком созданных объектов, или выбрав пункт меню «Создать» на панели инструментов этого же окна и выбрав необходимый список в списке



Рассмотрим все три метода и начнем с самого простого способа создания таблиц — путем ввода данных. При выборе данного режима перед Вами появится пустая таблица, имеет вид обычной электронной таблицы MS Excel. Полям данной таблицы с помощью контекстного меню можно предоставить необходимые названия характеристик, а в записи таблицы ввести необходимые данные.

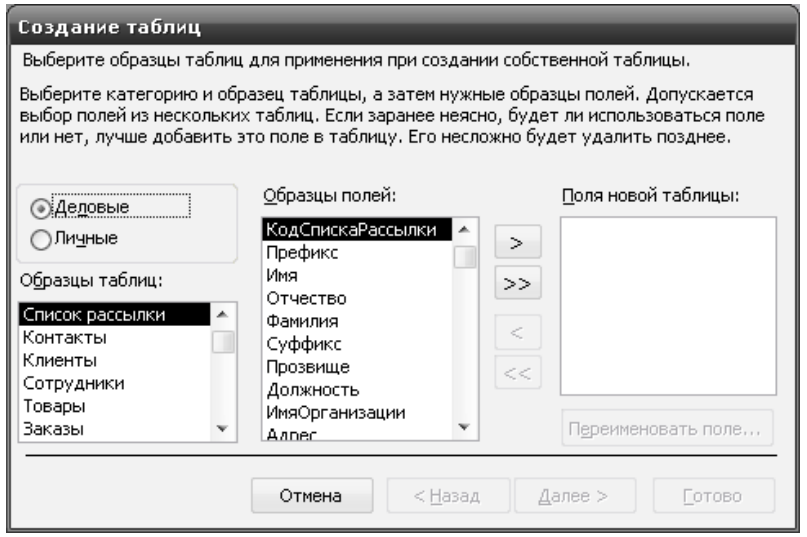


Практическая часть №1

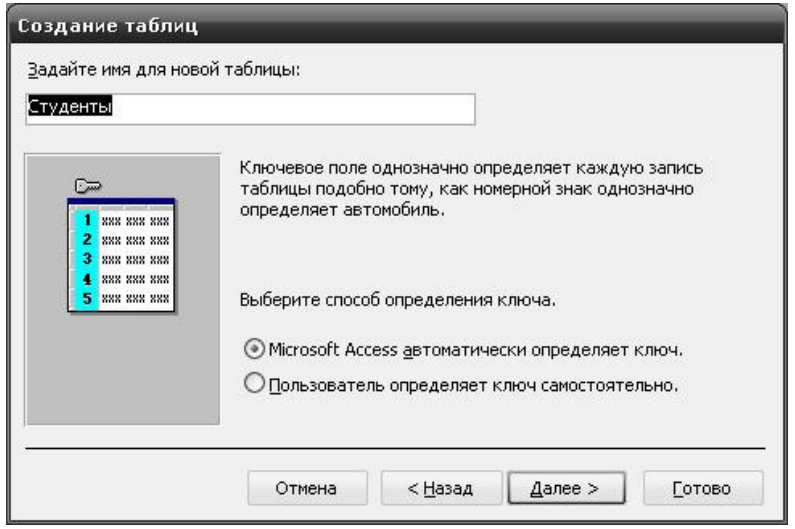
Чтобы Вы более подробно представляли себе данный процесс создания таблицы этим способом, создайте таблицу с названием Student, содержащую информацию о студенте и имеющую следующую структуру: name, surname, dateOfBirth, course.

Следующий способ — ***создание таблицы на основе шаблона***. Выбираем нужный нам метод создания и наблюдаем следующее за этим действие — запускается окно Визарда, которое предлагает выбрать шаблон для создания таблицы (см. ниже).

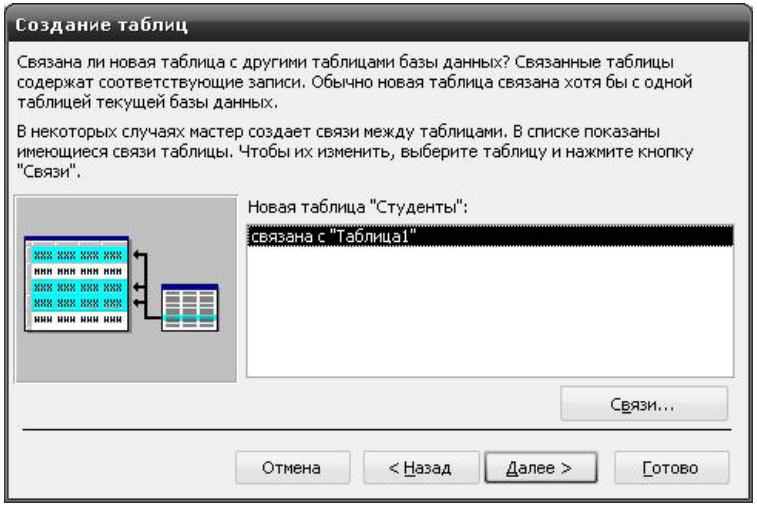
В поле ***"Образцы таблицы"*** выбираем шаблон будущей таблицы. В поле ***"Образцы полей"*** выбираем необходимые поля и переносим их с помощью стрелок в поле под названием ***"Поля новой таблицы"***. Кстати, названия выбранных полей можно изменить путем выбора кнопки ***"Переименовать поле"***.



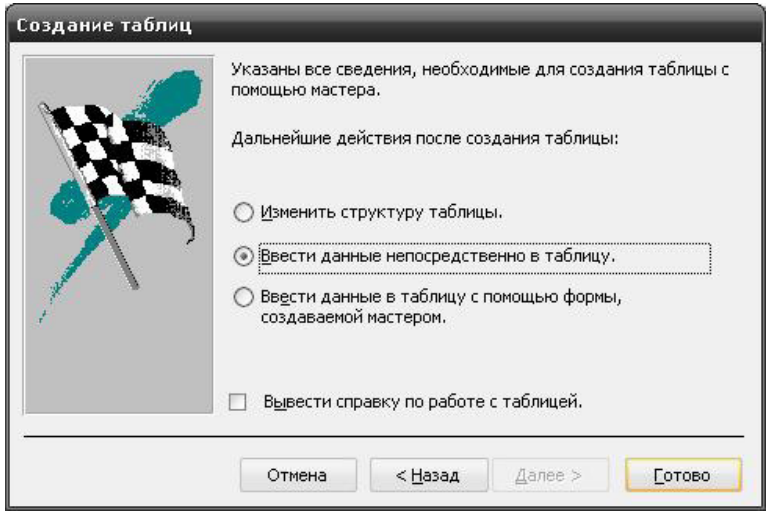
На следующем шаге можно задать имя новообразованной таблицы, если Вы не согласны с именем, которое предлагает Визард, а также первичный ключ. О ключе мы поговорим с Вами несколько позже, поэтому оставляем эту закладку без изменений и идем дальше.



Далее можно задать связь с какой-то уже существующей таблицей, а нажав кнопку «Связи», выбрать тип связи.



На последнем шаге необходимо указать действие, которое будет иметь место после создания данной таблицы. Выбираем необходимое и завершаем работу Визард.



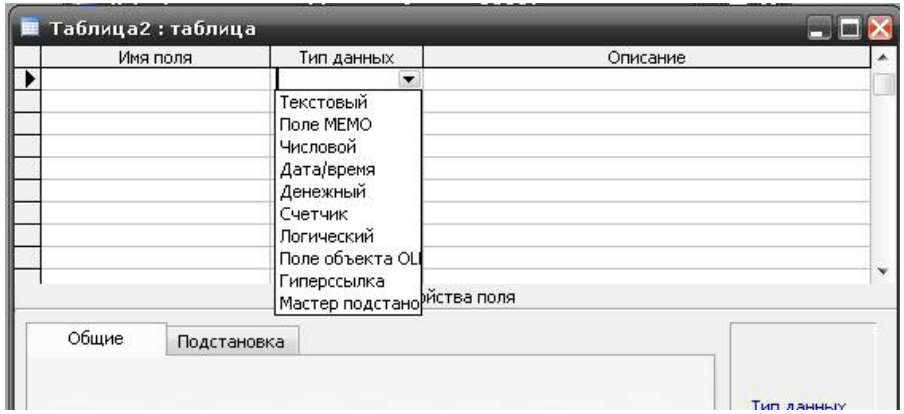
Практическая часть №2

Попробуйте создать любую таблицу на основе предложенных шаблонов, переименовать некоторые поля и тому подобное.

Кроме рассмотренных выше двух способов создания таблицы БД, существует третий — с помощью конструктора. Он предоставляет полный контроль над полями создаваемой таблицы, простой и удобный в работе. Поскольку данный способ чаще всего используют программисты баз данных, то именно его мы и рассмотрим более подробно и будем использовать в нашей дальнейшей работе при добавлении таблиц в базу данных.

Итак, выбираем объект ***"Таблицы"*** и вариант ***"Создание таблицы в режиме конструктора"***, или выбираем кнопку на панели инструментов ***"Создать"*** и вариант ***"Конструктор"***. То есть выполняем стандартный набор действий по созданию таблицы.

После этого перед Вами появится окно конструктора таблицы. Проанализируем его содержание.



В верхней части окна есть три колонки: ***Имя поля*** (*название поля таблицы*), ***Тип данных*** (*тип данных, с которыми мы будем работать: текст, числа и т.д.*) и ***Описание*** (*короткий комментарий к данному полю*).

В нижней части окна, представленны свойства данного поля в виде двух закладок: Общие (позволяет настроить данное поле. Например: для текстового типа данных можно указать его размер, для типа Дата / Время — формат работы с датой и т.д.) и ***Подстановка*** (позволяет ассоциировать данное поле с некоторым элементом управления, который будет использован при создании формы таблицы).

Чтобы лучше понять основной принцип построения таблицы в режиме конструктора нужно для начала более подробно познакомиться с понятием типа полей, поэтому перейдем к этому ознакомлению.

Типы полей

СУБД MS Access поддерживает 10 типов полей, краткие характеристики которых представлены ниже.

- ***Текстовый*** (*Text*) — тип данных по умолчанию. Текст или числа, не принимают участия в расчетах. Количество символов в поле не должно превышать 255. Максимальное количество символов, которое можно ввести в поле, задается в свойстве Размер поля. Пустые символы в неиспользуемых полях не сохраняются.
- ***Поле MEMO*** (*MEMO*) — длинный текст, например, описание определенного объекта или примечание. Максимальная длина 64000 символов.
- ***Числовой*** (*Number*) — числовые данные, используемые в математических расчетах. Конкретные варианты числового типа и их длина задаются в свойствах Размер поля.
- ***Денежный*** (*Currency*) — денежные значения и числовые данные, используемые в математических расчетах.
- ***Точность*** — до 12 знаков в целой и до 4 знаков в десятичной части. Длина поля 8 байт.
- ***Дата / Время*** (*Date/Time*) — значение даты или времени, относятся к годам с 100 по 9999 включительно. Длина поля 8 байт.
- ***Счетчик*** (*Счетчик, AutoNumber*) — тип данных поля, в которое для каждой новой записи автоматически вводятся уникальные целые числа в диапазоне от 1 до двух с лишним миллиардов. Значение этого типа не-

возможно изменить или удалить. Используется для определения уникального ключа таблицы, поэтому данный тип может иметь только одно единственное поле таблицы.

- ***Логический*** (*Yes / No*) — логические данные, то есть поле, которое может содержать одно из двух возможных значений: Да / Нет; Истина / Ложь; Вкл. / Выкл., и тому подобное
- ***Поле объекта OLE*** (*OLE object*) — объект (например, рисунок, звукозапись, электронная таблица Microsoft Excel и т.д.), связанный или встроенный в таблицу MS Access. Длина поля — до 1 Гб. Для полей типа OLE и MEMO не допускается сортировка и индексирование.
- ***Гиперссылка*** (*Hyperlink*) — в качестве гиперссылки можно указывать путь к файлу на диске или URL.
- ***Максимальная длина*** — 64000 символов.
- ***Мастер подстановок*** (*Lookup Wizard*) — создает поле, в котором предлагается выбор значений из списка, или же с поля со списком, содержащего набор постоянных значений или значений из другой таблицы. Выбор данного типа поля запускает Визард, который строит для поля список значений на основе поля или полей из другой таблицы.

Домашнее задание

Создать базу данных предприятия, которая состоит из трех таблиц, содержащих информацию о:

- работниках;
- отделах;
- продукции.

Продумать структуру каждой таблицы и создать их различными способами.