Подобно тому, как **Word** был спроектирован для работы с документами, содержащими текст и графику, а **Excel** - с числами и диаграммами. **Access** ориентируется на обработку данных. Данные представляют собой набор фактов, они превращаются в полезную информацию лишь после того, как будучи упорядочены каким-то разумным способом. Из всех приложений **Office** именно **Access** представляет собой инструмент для подобной организаций данных.

База данных состоит из набора таблиц, форм, запросов и отчетов, используемых для обработки и представления данных. В **Access** работа с этими объектами базы данных происходит в окне базы данных. Процесс создания базы данных состоит из следующих этапов:

- проектирование и создание таблиц для хранения данных;
- ввод данных;
- разработка других элементов базы, предназначенных для просмотра**,** редактирования и вывода информации.

1. Основные концепции баз данных

Перед тем как начинать работу с базой данных, необходимо найти какое-то средство для их хранения. В **Access** данные хранятся в специальных объектах — ***таблицах***. Например, одна таблица может содержать данные о студентах, а другая — об учебных курсах, которые они посещают. Эти отдельные таблицы необходимо связать воедино. Комбинация всех таблиц и их взаимных связей составляет «фундамент» базы данных.

1.1. Поля и записи

Таблица **Access** имеет сходство с листом **Excel**. Таблица точно так же делится на строки и столбцы, а значения вводятся в ячейки, разделенные линиями сетки. Однако таблица **Access** отличается от листа **Excel** тем, что каждый столбец таблицы представляет собой *поле,* то есть информационную категорию, а каждая строка — *запись,* то есть один элемент таблицы.

Отличия далеко не исчерпываются терминологией. В отличие от столбца **Excel**, каждое поле таблицы может содержать значения данных только одного типа - текст, числа, даты и т. д. В **Access** каждая запись содержит информацию об одном элементе (например, сведения о студенте или учебном курсе).

При работе с информацией из базы данных довольно часто приходится обрабатывать ее по одной записи. Другими словами, при просмотре данных с помощью *формы* в каждый момент времени пользователь обычно видит содержимое одной записи; каждая строка в *отчете* также обычно представляет собой одну запись (при проектировании форм или отчетов пользователь сам определяет, какие поля записей войдут в них).

Access чрезвычайно гибок в отношении типов данных, которые задаются для полей. Допускается ввод любых разновидностей текста, цифр, дат, времен и денежных сумм. **Access** позволяет использовать специальный тип поля (*Объект OLE*), позволяющий включать в запись любые объекты, которые могут быть представлены в приложениях **Windows** — картинки, звуки,

анимацию, видеоклипы. Впрочем, основная работа все же осуществляется с текстом, числами и датами.

Access имеет ряд отличий в способах хранения информации – одно из которых является то, что вся структура базы данных (таблицы с данными) и средства управления базой представляет собой один файл. Каждый документ **Word** представляет отдельный файл, который не зависит от других документов. Файл **Excel** – рабочая книга - может содержать различные типы листов, но на типы информации в них не накладывается никаких ограничений. Каждый файл **PowerPoint** представляет собой презентацию, в которую могут входить различные типы слайдов.

Каждая таблица базы данных **Access** имеет предельно жесткую структуру, и вся информация в ней должна быть распределена по соответствующим полям. Если какое-либо поле в записи остается пустым, **Access** отображает на этом месте пробелы, несмотря на отсутствие какой-либо информации.

1.2. Системы управления реляционными базами данных

Приложения, работающие с базами данных, ориентируются на один из двух основных их видов: ***плоские таблицы (flat files)*** или ***реляционные базы***. Хотя плоские таблицы были стандартом в течение многих лет, в настоящее время они используются только в приложениях типа **Microsoft Works** или **Microsoft Excel**. В плоской таблице вся взаимосвязанная информация должна находиться в одной таблице. Это означает, что любые данные, повторяющиеся в нескольких записях, должны присутствовать в каждой из этих записей.

По мере развития и усложнения баз данных стало ясно, что такой способ неэффективен при хранении больших объемов информации - в результате возникла идея реляционных баз данных. В реляционной базе используется несколько разных таблиц, между которыми устанавливаются ***связи (relations)***. Они позволяют ввести информацию в одну таблицу и связать ее с записями другой таблицы через специальный идентификатор.

Таким образом, в реляционной базе данных повторяющаяся в обычной плоской таблице информация вводится один раз в отдельную таблицу.

Хранение данных в связанных таблицах обладает рядом преимуществ:

- экономия времени, поскольку одни и те же данные не приходится вводить в нескольких таблицах;
- уменьшение размера базы данных (порой весьма значительное по сравнению с размером плоской таблицы), которое экономит дисковое пространство на вашем компьютере и облегчает перенос базы данных;
- существенное сокращение количества ошибок. Отпадает необходимость ввода одних и тех же данных в большое количество записей. Если же повторяющиеся данные хранятся в связанной таблице, то информацию достаточно ввести всего один раз; после этого в исходной таблице для повторяющихся данных вводится только код (обычно короткая

цифровая или алфавитно-цифровая последовательность). Можно настроить поле так, чтобы выбрать код из списка и обойтись вообще без набора текста.

Обновление данных является одношаговым процессом.

Реляционные базы данных обладают рядом других преимуществ по скорости и производительности, но для понимания работы с **Access** вовсе не обязательно разбираться в теории. Необходимо осознать всего несколько положений:

- *поле* является информационной категорией;
- *значением* называется информация, содержащаяся в определенном поле одной записи;
- *записью* называется совокупность взаимосвязанных значений для одного элемента базы данных (заполняющих строку в таблице);
- *связи между таблицами* создаются с помощью данных, находящихся в специальных полях, и это позволяет обойтись однократным вводом повторяющейся информации.

2. Окно базы данных

Все объекты, относящиеся к конкретной базе данных, **Access** хранит в одном большом файле. Именно с ним пользователь работает при использовании приложения **Access.** Среди объектов, связанных с одной базой данных, могут быть таблицы, в которых содержится информация; формы и отчеты, используемые для представления данных; запросы, с помощью которых запрашиваются различные сведения, модули и макросы для автоматизации работы с базой. Все объекты базы упорядочиваются по своему типу и отображаются на следующих вкладках окна базы:

- Таблицы
- Запросы
- Формы
- Отчеты
- Макросы
- Модули

Каждая вкладка в верхней части окна базы данных отображает отдельный тип объектов в базе. Каждый элемент, представленный на вкладке, отображает отдельный объект.

При закрытии окна базы данных **Access** также закрывает все связанные с ней объекты. Если были внесены изменения в **структуру** объектов базы, но не было выполнено их сохранение, **Access** предлагает это сделать. Необходимо учесть, что любые исправления **данных** автоматически сохраняются при их внесении.

3. Объекты базы данных

Access предоставляет не только возможность хранить набор данных, но и средства, необходимые для получения осмысленной информации. Различные формы, отчеты и запросы, входящие в файл базы данных, не менее важны, чем сами таблицы с данными.

Работа с объектами баз данных **Access** разбивается на две фазы:

- ***фаза конструирования***
- ***рабочая фаза.***

Перед тем как объект базы данных сможет работать с вводимой информацией, его необходимо сконструировать. В отличие от открытия пустых файлов в **Microsoft Word** или пустых листов в **Microsoft Excel**, все объекты баз данных в **Access,** том числе и таблицы для хранения данных, должны быть **сконструированы**. В это понятие входит процесс создания объекта и его последующие изменения в соответствии с требованиями пользователя. В большинстве случаев это происходит в специальном режиме *Конструктора*, в котором имеются средства для подобной работы.

Именно конструкция объекта определяет, что будет происходить с данными, при, работе в этом объекте. Например, при конструировании таблицы задаются типы данных, которые можно будет в нее вводить.

После **фазы конструирования** наступает **рабочая фаза**, во время которой различные объекты базы данных используются для ввода, редактирования информации, преобразование информации из базы при помощи запросов и распечатки отчетов, данные которых представлены в нужном виде. Все это позволяет превратить «сырые» данные в осмысленную информацию.

Во время **рабочей фазы** режим *Конструктора* заменяется режимом таблицы или формы, где осуществляются рассмотренные ниже операции по вводу, редактированию и обработке данных.

При работе с объектами различных типов в приложении **Access**, а также при переходе от конструирования объектов к их использованию, вид меню и панелей инструментов изменяется в соответствии с теми задачами, которые приходится выполнять. При активизации различных окон становятся доступными именно те средства, которые предназначены для работы с текущим окном.

3.1. Таблицы

Как уже упоминалось выше, структура таблицы определяется в режиме конструирования. Ввод и редактирование данных обычно осуществляется в режиме таблицы. Он отчасти напоминает работу с листом **Excel** - данные собраны в строки и столбцы. В **Access** каждая строка соответствует одной *записи*, а каждый столбец — одному *полю*.

3.2. Формы

Формы предоставляют большую свободу в выборе способа представления информации, чем режим таблиц.

Формы в первую очередь предназначены для работы с данными на экране. Обычно в этом режиме выводятся данные из одной записи, а текстовые поля, управляющие кнопки, картинки и прочие служебные объекты облегчают просмотр и обработку данных. Если необходимо просмотреть сразу несколько записей, можно перейти в режим таблицы.

Формы облегчают поочередный просмотр и ввод данных записей. Другое преимущество форм перед режимом таблицы — возможность отображать в одной форме данные из нескольких таблиц.

3.3. Запросы

После того, как выполнено создание таблиц и ввод в них данные, вероятно, самыми важными объектами базы данных могут стать запросы. С их помощью можно выбрать из базы данных определенную информацию и упорядочить ее для использования в отчете или для просмотра на экране в форме или таблице.

Запросы создаются в режиме конструктора запросов и, по сути, являются вопросами о содержимом базы данных. Ответы отображаются в режиме таблицы, который очень похож на аналогичный режим просмотра данных. Принципиальное отличие между режимом таблицы для просмотра данных и вывода результатов запроса заключается в том, что в запросе может содержаться информация из нескольких таблиц, выбранная на основании связанных полей. Чтобы включить результаты запроса в форму или отчет, необходимо сначала убедиться в том, что запрос был сохранен. После этого можно при конструировании формы или отчета выбрать объект-запрос вместо объекта-таблицы.

3.4. Отчеты

Отчеты предназначаются для вывода на печать данных из базы. *Конструктор* не только позволяет вывести информацию в удобном и привлекательном виде, но и (по аналогии с формами) комбинировать данные несколько таблиц, а также упорядочивать записи таблицы на основании данных других таблиц. При выводе отчета на печать, данные из таблиц представляются аккуратно упорядоченными и обработанными.

3.5. Средства программирования

Access также предоставляет средства для создания макросов, аналогичные тем, что имеются в **Word** и **Excel**, а также возможность полноценной работы с языком **Visual Basic for Applications (VBA).** С помощью таких мощных инструментов можно автоматизировать многие из стандартных задач управления базами данных. Примером этого служит создание законченных баз данных с помощью *Мастеров* баз данных **Access**.

3.6. Сохранение данных и объектов

Поскольку в таблице нет ничего важнее данных, **Access** автоматически сохраняет любые изменения, вносимые в содержимое таблицы.

С другой стороны, макеты различных ***объектов,*** создаваемых в базе, должны сохраняться отдельной командой. Например, если сконструирован отчет для обобщения данных из нескольких таблиц, то его необходимо сохранить в базе данных. При этом **Access** просит ввести имя отчета, а также его заголовок для отображения на вкладке *Отчеты* окна базы данных. Обычно сохранение объектов в базе данных во время их конструирования выполняет пользователь, а запись данных выполняется самим приложением **Access**.

4. Работа с базами данных

4.1.Создание новой базы данных

Новая база данных совершенно пуста - в ней нет никаких таблиц, форм, отчетов или каких-либо других объектов. Каждый объект базы создается пользователем.

Новую базу данных можно создать и после запуска **Access**:

4.2. Создание базы данных с помощью *Мастера*

Другая возможность создания базы данных — использование мастеров баз данных. Предлагается шаблоны различных баз данных, что позволяет предельно упростить построение требуемой базы данных.

Заставка большинства *Мастеров* баз данных Access содержит краткий обзор создаваемых таблиц. В дальнейшем предоставляется возможность выбора тех или иных полей из этих таблиц для добавления к создаваемой базе путем установления соответствующих флажков. Также предлагается ввести параметры различных типов объектов, – например, для экранных форм или стиль оформления отчетов. После принятия решений завершается работа ***Мастера***, и Access создает базу данных вместе со всеми таблицами, формами, отчетами и запросами. Разумеется, впоследствии, если это будет необходимо, можно добавить или изменить все эти элементы.

4.3. Открытие базы данных

После того как база данных будет создана, есть несколько возможностей открыть ее в приложении **Access**.

При запуске **Access** из диалогового окна в списке, находящемся в нижней части окна диалога, перечисляются последние использовавшиеся базы данных.

5. Создание таблиц и связей. Проектирование основы базы данных

При создании базы данных с помощью *Мастера* само приложение **Access** создает таблицы и связи между ними. Структура таблиц и связей в базе данных определяет, каким образом впоследствии будет возможно извлекать хранящуюся в таблицах информацию. Однако если помощь *Мастера* оказывается недостаточной ввиду повышенных требований к вновь создаваемой

базе данных, необходимо самостоятельно проектировать, создавать и модифицировать таблицы и связи между ними.

Проектирование основы базы данных

Основное отличие приложения **Access** от других приложений **Microsoft Office** заключается в необходимости выполнения предварительного этапа проектирования базы, до запуска приложения **Access**. Если в приложении **Word** возможно сначала открыть текстовый редактор, ввести примерный текст документа, а затем его отредактировать, или, в приложении **Excel** начать ввод чисел в электронную таблицу, а формулы вставить в самый последний момент, то проектирование базы данных без компьютера все же происходит эффективнее, чем с ним.

Этот процесс состоит из нескольких последовательных этапов.

5.1. Содержимое полей.

Следует определить, какие поля с информацией должны быть включены в базу данных. Необходимо указать все элементы, которые могут понадобиться в работе. Например, при создании базы данных «Книги» необходимо включить в базу данных название книги, автора и тему. Однако можно также рассмотреть возможность включения даты выпуска, названия издательства, числа рисунков, количества страниц, цены, типе обложки и т. д.

5.2. Минимализация полноты данных.

Необходимо соблюдать баланс между полезностью информации и простотой ее обработки. База данных должна соответствовать принципам минимальной избыточности, т.е. содержать необходимый минимум полей данных, достаточно полно характеризующий объекты базы. Иначе, при вводе данных, будет затрачено много времени для заполнения всех дополнительных полей, база будет занимать больший объем памяти, и построение запросов будет усложнено.

Тем не менее, избыток полей все же лучше их недостатка. Если в процессе сбора и записи информации в базу данных возникнет необходимость добавления некоторых полей, трансформация базы данных будет весьма трудоемкой и потребует определенных временных затрат.

5.3. Тип информации в полях.

Далее необходимо определить тип информации, хранящейся в каждом поле. Обычно приходится выбирать между текстом, числами и датами, хотя **Access** предоставляет и несколько других вариантов (различные типы данных см. ниже).

5.4. Структура базы данных.

Проектирование структуры базы данных — это решение о том, как должны быть организованы поля в базе данных. Необходимо решить, каким образом и в каком порядке поля распределяются по таблицам. После того как база данных будет должным образом организована и настроена, можно расставить поля в отчетах и на экране практически любым способом.

5.5. Связи между таблицами.

Если информация в некоторых полях повторяется в ряде записей, необходимо поместить эти поля в отдельную таблицу и установить связи между таблицами. Это позволяет существенно уменьшить объем базы и повысить точность данных. Например, при организации базы данных «Книги» будет введено много записей о книгах одного издательства, вместо повторяющегося ввода информации в каждую запись имеет смысл создать таблицу «Издатели», и включить в нее всю эту информацию, назначив каждому издательству идентифицирующий код. После этого в таблице «Книги» достаточно ввести данный код, чтобы установить связь между книгой и ее издательством. Процесс организации полей и распределения их в одной или нескольких таблицах, а также создания связей называется нормализацией. **Access** несколько облегчает эту сложную задачу при помощи Мастера анализа таблиц.

5.6. Первичные ключи и индексы.

Следующая концепция — назначение в таблицах первичного ключа, или индекса. *Первичным ключом называется одно или несколько полей, которые однозначно определяют каждую запись в таблице*. Наличие индекса помогает **Access** быстрее находить и сортировать записи. Поля, используемые в качестве первичного ключа, индексируются автоматически, но можно составить отдельный индекс и для других полей, которые будут использоваться для поиска или сортировки. Первичные ключи и индексы должны стать частью процесса планирования и обычно указываются в таблице во время ее создания. Идентификация связей между данными в отдельных таблицах, понимание того, когда следует разбивать информацию на две или более таблицы, а также определение первичных ключей и индексов — основные требования для качественного проектирования базы данных.

5.7.Завершение проектирования базы данных.

После выполнения вышеописанных этапов проектирования известно все необходимое о каждой таблице, а также о связях между ними. Тем не менее, процесс планирования еще не закончен. Необходимо проанализировать, как будет происходить ввод данных, будут ли они импортированы из другого компьютерного источника, скопированы из списка. или каждый элемент базы будет введен по отдельности, определить наличие таблиц, которые должны быть заполнены раньше остальных, выявить стандартные значения, которые **Access** назначает самостоятельно и т.д.

6. Создание таблиц

При работе с объектами базы данных в среде **Access**, существуют различные способы создания новых объектов:

Конструктор - Access создает пустую таблицу и открывает ее в режиме конструктора

Ввод данных - Access создает пустую таблицу и открывает ее в режиме таблицы.

Мастер таблиц - таблица создается при помощи Мастера таблиц

6.1 Создание способом Ввод данных

Создание таблицы способом Ввод данных – представляет собой самый простой и в тоже время самый неэффективный способ создания таблицы. Пользователю предлагается пустая таблица, в которую он заносит данные в последовательные поля записей, может переименовать предложенные по умолчанию имена полей (Поле1, Поле2 и т.д.), а после закрытия (или сохранения) задать имя таблицы. При операции закрытия или сохранения Access автоматически, анализируя введенные данные, создает макет таблицы, задавая типы данных, длину полей и другие свойства полей, необходимые для использования таблицы.

6.2 Создание способом Мастер таблицы.

Другим из перечисленных способов является использование Мастера таблиц, который последовательно осуществит процесс создания новой таблицы, основанной на одном из шаблонов Мастера таблиц. В процессе работы в Мастере необходимо выбрать таблицу, которая ближе всего соответствует поставленной задаче, и затем произвести выбор из предлагаемых полей. Используя имеющиеся кнопки, можно добавить выделенное поле, удалить выделенное поле, добавить все поля выделенной таблицы, удалить все поля из новой таблицы. Список Поля новой таблицы - дают возможность быстрого и легкого создания новой таблицы. В новую таблицу можно включить поля из нескольких таблиц - шаблонов. Добавленное в список поле можно переименовать. После завершения отбора полей переходим к следующим окнам Мастера таблиц, в которых можно изменить предлагаемое имя таблицы и задать способ определения ключа с опциями Автоматическое определение ключа и Самостоятельное определение ключа. Последнее окно диалога Мастера таблиц позволяет выбрать действия после создания таблицы - изменение структуры таблиц (в этом случае выполняется переход в режим Конструктора таблиц, в котором можно изменять структуру таблиц), или непосредственный ввод данных в таблицу, или ввод данных в таблицу с помощью формы, предлагаемой Мастером.

После выбора полей в первом окне Мастера таблиц можно в любой момент нажать кнопку Готово и предоставить Мастеру завершить работу с выбранными ранее значениями (а также теми, что приняты по умолчанию для остальных параметров). Если пользователь согласен со всеми значениями по умолчанию, Мастер назначает таблице стандартное имя, определяет в ней первичный ключ, автоматически устанавливает связи и открывает получившуюся таблицу в режиме таблицы для ввода данных.

6.3. Работа с макетом таблицы

Режим *Конструктора* таблиц может применяться как для создания новых таблиц, так и для модификации уже существующих. Он позволяет определять или вносить изменения в структуру таблицы. В нем можно:

1. добавлять, удалять и переставлять поля;

2. задавать имя, тип данных и другие свойства полей;
3. назначать первичный ключ таблицы.

Окно конструктора таблицы в верхней половине содержит поля (ключевое поле помечается специальным значком – ключиком), типы данных и описания. Если режим конструктора используется для создания новой таблицы, то список будет пустым. В нижней половине окна в разделе *Свойства поля* можно задать, просмотреть и изменить свойства выделенного поля. Внизу справа приводится справочная информация для каждого свойства поля.

В режиме конструктора можно добавить, удалить, переставить или скопировать поля в таблице.

Чтобы создать или добавить в таблице новое поле, надо щелкнуть на первой пустой строке столбца Имя поля и ввести имя создаваемого поля. Оно может содержать до 64 произвольных символов, включая пробелы. Однако, желательно, чтобы имя кратко, но содержательно описывало содержимое поля. Затем можно перейти к заданию свойств, значения которых должны отличаться от принятых по умолчанию.

Порядок, в котором поля перечисляются в режиме конструктора (сверху вниз), совпадает с тем, в котором они выводятся в режиме таблицы (слева направо). Определенный порядок полей может выглядеть более логичным и облегчать ввод и изменения данных. Тем не менее, порядок полей в режиме конструктора не влияет на порядок полей в формах, запросах или отчетах.

Вы не можете удалить поле, которое используется для создания связи с другой таблицей; сначала необходимо удалить связь. Если в таблице имеются данные, то **Access** потребует подтверждения перед удалением поля. Необходимо учесть, что вместе с полем удаляются соответствующие данные из всех записей таблицы.

Задание свойств полей.

Каждое поле таблицы описывается набором свойств. Свойства определяют способы хранения, обработки и вывода данных поля. В их число входят: имя поля, тип данных, описание и ряд других параметров (например, размер поля, формат и подпись). Можно просмотреть и изменить значения свойств поля в окне конструктора таблицы (как в решетке в верхней части окна, так и на вкладках внизу). Если курсор находится в текстовом поле свойства, **Access** выводит информацию о данном свойстве в правом нижнем углу окна.

Как было сказано ранее, при создании нового поля в режиме конструктора необходимо задать для него имя. Все свойства нового поля имеют значения, принимаемые по умолчанию. Иногда значение по умолчанию отсутствует, в этом случае соответствующая текстовая область оказывается пустой. Можно изменить значение любого свойства или просто согласиться со значением по умолчанию.

Чтобы изменить имя поля, надо щелкнуть в ячейке с именем (столбец *Имя поля*) и набрать на клавиатуре новое имя. Это никак не повлияет на существующие связи и не изменит содержимого базы данных.

При переименовании поля не всегда меняется имя, находящееся над данным столбцом в режиме таблицы. Имя поля отображается в режиме таблицы лишь в том случае, если не задан специализированный вариант имени в свойстве *Подпись*.

6.4 Выбор типа данных.

Тип данных определяет, какие данные могут вводиться в поле. Для каждого поля в таблице **Access** должен быть задан тип данных. Предусмотрено несколько типов данных. По умолчанию принимается тип *Текстовый*.

Чтобы задать или изменить тип данных, надо щелкнуть в столбце *Тип данных* на том поле, которое требуется изменить, и выбрать новый тип из раскрывающегося списка.

1. *Текстовый* (принимается по умолчанию).

Содержит любые символы — буквы и шифры. Количество хранимых символов (0-255) зависит от значения свойства *Размер поля*. Даже если задать число символов равным 255, **Access** будет использовать ту величину, которая указана в этом свойстве.

2. *Поле MEMO*.

Аналогично типу *Текстовый*, но может содержать до 64К символов.

3. *Числовой*.

Содержит числовые значения, которые могут использоваться в математических выражениях. Числовые поля применяются при вычислениях или в запросах, сравнивающих содержимое поля с числовым значением. Размер и тип хранимого числа определяется текущим значением свойства *Размер поля*.

4. *Дата/Время*.

Содержит любые допустимые календарные даты для годов от 100 до 9999 и времени суток в 12- или 24-часовом формате. Этот тип допускает хронологическую сортировку и вычисления с данными.

5. *Денежный*.

Наиболее точное представление для денежных сумм, используемых в финансовых вычислениях.

6. *Счетчик*.

Поле, значение которого автоматически изменяется при каждом добавлении новой записи. **Access** не использует повторно номера, которые освобождаются при удалении записей. Кроме того, пользователь не может изменить значение поля типа *Счетчик*. Имеются два варианта заполнения поля с типом *Счетчик* - последовательное возрастание значений на 1 и случайные числа.

7. *Логический*.

Наиболее эффективный способ хранения единственного значения, соответствующего выбору *«Да / Нет», «Истина / Ложь»* и т. д. Значение может быть представлено на форме в виде флажка.

8. *Поле объекта* OLE.

Содержит объект OLE (лист **Excel**, документ **Word**, рисунок, звук, анимацию или видеоклип).

9. *Гиперссылка*.

Содержит *Гиперссылку* — то есть ссылку на местонахождение другого объекта базы данных, документа **Office** или страницы **World Wide Web**.

10. *Мастер подстановок*.

Вставляет в поле раскрывающийся список с допустимыми значениями поля, которые взяты из другой таблицы или введены заранее (список, который появляется в ячейках столбца *Тип данных* при выделении, также представляет собой подстановочное поле).

Определение других свойств полей.

При желании можно ввести описание поля в столбце *Описание,* которое будет отображаться в строке состояния внизу таблицы или формы, как пояснение к выбранному полю.

Остальные свойства полей задаются на вкладках *Общие* и *Подстановка* в нижней половине окна конструктора.

Набор свойств меняется в зависимости от типа данных поля. Свойства полей определяют способ ввода и хранения данных в таблице и служат для повышения целостности и последовательности данных. Далее рассматриваются некоторые важнейшие свойства полей.

Размер поля.

Размер поля позволяет указать, сколько места будут занимать данные этого поля. Это свойство задается только для полей типа *Текстовый* или *Числовой*. Для текстового поля оно равно максимальному количеству символов, которые могут в нем храниться; допустимые значения лежат в диапазоне от 0 до 255 (по умолчанию принимается значение 50). Для числового значения следует выбрать из раскрывающегося списка тип величин, хранящихся в поле:

Пример. Принятый по умолчанию тип *Длинное целое* позволяет хранить числа в интервале от - 2,147,483,648 до + 2,147,483,647, а тип *С плавающей точкой* поддерживает хранение дробных величин (например, 3,14).

Формат.

Свойство *Формат* определяет способ отображения на экране хранящейся в поле информации и режим ее вывода на печать. Нужный формат следует выбрать из раскрывающегося списка или создать используя специальные символы (0 # $ % и др.).

Пример. Для поля типа *Дата/время* можно выбрать отображение даты в формате 30-Июн-01, 6/30/01 и т. д.

Число десятичных знаков.

Данное свойство позволяет выбрать количество десятичных знаков, которое выводится для полей типа *Числовой* или *Денежный*. Оно влияет только на способ отображения информации, а не на точность внутреннего представления значений. Определяется выбором из раскрывающегося списка количество знаков или значение Авто (принятое по умолчанию) для отображения того количества, которое указано в свойстве *Формат*.

Маска ввода.

Для большинства типов данных пользователь может определить *маску ввода*, управляющую процессом ввода данных в поле. Преимущество маски состоит в том, что она помогает представлять информацию в заданном виде, сокращая тем самым количество возможных ошибок.

Маска ввода указывается с помощью набора символов-заполнителей.

Пример. Маска ввода для даты хранится в виде 99/99/00. Заполнитель 9 означает, что в данной позиции может быть введена только цифра (хотя ее можно пропустить). В позициях, помеченных символом 0, также допускаются только цифры, но эти позиции обязательно должны быть заполнены. Символы косой черты (/) являются константами. Для получения дополнительной информации об этих символах можно нажать клавишу F1, когда курсор находится в ячейке свойства *Маска ввода*.

В **Access** имеется *Мастер масок ввода*, при помощи которого можно задать маску и избавиться от необходимости ввода символов вручную. *Мастер* запускается маленькой кнопкой с тремя точками, которая находится справа от поля маски.

Подпись.

Свойство *Подпись* представляет собой текст, которым **Access** пользуется для маркировки поля в режиме таблицы, на формах и отчетах. Если значение этого свойства не задано (принимается по умолчанию), то **Access** будет помечать поле его именем. Это удобное средство позволяет разъяснить смысл поля при просмотре базы или вводе данных и обойтись без его переименования.

Значение по умолчанию.

Свойство *Значение по умолчанию* используется при создании базы данных, в которой некоторое поле обычно содержит одно и то же значение. Чтобы назначить значение по умолчанию, надо ввести его в ячейку *Значение по умолчанию* или нажать кнопку с тремя точками справа от ячейки для получения помощи в построении сложных выражений.

Пример. Адресная база данных, в которой все адреса обычно относятся к одному городу. Можно задать для данного поля значение по умолчанию, и тогда **Access** при каждом создании новой записи будет автоматически заносить его в поле, - например, в поле «Город» будет вставлено «Москва».

При необходимости можно изменить текст и ввести название другого города.

Обязательное поле.

Если присвоить данному свойству значение *Да*, то **Access** потребует от пользователя обязательного ввода значения при создании или модификации записи. Если же свойство имеет значение *Нет* (принимается по умолчанию), то поле можно оставить пустым.

Индексированное поле.

Пользователь также может указать, является ли данное поле индексированным. Индексирование поля существенно ускоряет поиск и сортировку, а также выполнение запросов, но

требует дополнительного места для хранения информации и замедляет процессы создания, удаления и обновления записей. Ключевое поле таблицы (см. ниже) всегда индексируется. Из раскрывающегося списка можно выбрать одно из следующих значений:

Да (допускаются совпадения) — поле индексируется, одно и то же значение может присутствовать в нескольких записях.

Да (совпадения не допускаются) — поле индексируется, и все его значения должны быть уникальными, чтобы по ним всегда можно было однозначно определить запись.

Нет — поле не индексируется (принимается по умолчанию).

Чтобы просмотреть список всех индексированных полей таблицы, необходимо выполнить: меню *Вид* - команда *Индексы* или нажать кнопку *Индексы* на *Панели Инструментов*.

Выбор ключевых полей.

Желательно (хотя и не обязательно), чтобы в каждой таблице имелись ключевые поля. **Access** использует их для однозначной идентификации и упорядочения записей в таблице. Если поле сделано ключевым, то свойству *Индексированное поле* присваивается значение *Да (совпадения не допускаются)*, причем пользователь не сможет изменить его. Записи можно быстро отсортировать или выбрать по ключевым полям, и ввод повторяющихся значений в них недопустим. Кроме того, при внесении или изменении данных в таблице **Access** требует задать значение ключевого поля.

В большинстве случаев ключевым можно сделать одно поле, хотя иногда возникают ситуации, при которых данные одного поля не являются уникальными для каждой записи, и тогда приходится назначать ключевыми два и более полей. В таких случаях уникальным должно быть сочетание всех ключевых полей.

Пример. В системе складского учета могут присутствовать поля для категории товара и его номера. По отдельности значения этих полей могут повторяться, но вместе они образуют полное обозначение товара, которое является уникальным.

Если при создании таблицы использовался *Мастер таблиц*, то возможно, **Access** уже назначил ключевые поля. В противном случае (или при изменении ключа, выбранного **Access**) ключ можно задать самостоятельно.

Ключ может быть задан по нескольким полям - в этом случае он называется составной ключ и для его задания необходимо выделить эти поля перед тем, как выполнять установку, для чего необходимо выделить первое поле, а затем, удерживая нажатой клавишу *Ctrl*, поочередно щелкать все остальные поля.

Сохранение структуры таблицы.

После каждого создания или модификации структуры таблицы в режиме конструктора следует сохранить ее перед тем, как приступать к вводу или редактированию данных. **Access** предлагает сохранить внесенные изменения перед выходом из режима конструктора. Тем не менее при серьезных изменениях в таблице следует периодически сохранять ее стандартным для всех приложений **Microsoft Office** способами.

Если в таблице нет данных, **Access** быстро сохраняет структуру, после чего вы можете или продолжить работу с ней или перейти в режим таблицы, чтобы приступить к вводу данных. Если же в таблице присутствуют данные, **Access** проверяет, может ли новая структура содержать старые данные, и не приведут ли изменения к каким-либо конфликтам.

При возникновении проблем появляется сообщение об ошибке. В таком случае следует записать предполагаемые изменения (чтобы реализовать их позже), закрыть таблицу и отказаться от сохранения изменений, после чего перейти в режим таблицы и исправить проблему.

Пример. Если изменить значение свойства *Индексированное поле* на *Да (совпадения не допускаются)*, то **Access** сможет сохранить структуру таблицы только в том случае, если в данном поле не встречаются повторяющиеся значения. Если же поле содержит повторяющиеся значения, то **Access** предложит перейти к таблице и изменить ее содержимое так, чтобы в нем не попадались дубликаты (простая возможность найти их состоит в сортировке таблицы по данному полю).

Импортирование данных.

Таблицы **Access** могут создаваться посредством импортирования данных из других файлов. Таким файлом может быть база данных, созданная в текущей или более ранней версии **Access**, или же в другой программе (например, **Excel**, **dBASE** или **Paradox**). При импортировании базы данных **Access** или листа **Excel** можно выбрать параметры преобразования, а файлы **dBASE** импортируются полностью без участия со стороны пользователя. В диалоговых окнах, отображаемых *Мастером импорта*, требуется задать все необходимые параметры. После чего *Мастер* создает таблицу, содержащую импортированные данные. Если какие-либо из записей не были импортированы правильно, **Access** создает таблицу ошибок, в которой перечисляются все проблемы. Чаще всего следует вернуться к исходному файлу, исправить в нем ошибки и попытаться заново импортировать таблицу.

7. Установка связей между таблицами.

Особенностью реляционных баз данных является наличие отношений между таблицами в базе данных. Необходимо, чтобы такие отношения между полями таблиц были определены в самом **Access**. Такое определение называется связью, а поля называются связанными (как и содержащие их таблицы). После установления связи **Access** помогает обеспечивать целостность связанных данных и облегчает работу с ними.

В подстановочных полях значение можно выбрать из связанного поля, вместо того чтобы запоминать его и вводить вручную.

Связи также позволяют создавать запросы, формы и отчеты, в которых выводится информация сразу из нескольких таблиц (запросы, формы и отчеты будут рассмотрены позднее).

Чтобы просмотреть связи, существующие между таблицами в базе данных, необходимо открыть окно *Схема данных*. Линии между окнами таблиц изображают связи между отдельными полями. Символы рядом с концами линии обозначают тип связи.

При работе с окном *Схема данных* чрезвычайно важно расположить таблицы таким образом, чтобы линии связей были четко видны. Окна таблиц в окне *Схема данных* можно

перетаскивать мышью за заголовок. Необходимо расположить окна таблиц так, чтобы линии не запутывались и были наглядными.

Когда в правой части окна таблицы появляется полоса прокрутки, это значит, что отображаются не все имена полей. Чтобы избавиться от полосы прокрутки и видеть все поля, можно увеличить высоту окна таблицы перетаскиванием его верхнего или нижнего края. Аналогично изменяется и ширина окна таблицы, что позволяет выводить полные имена полей.

Если при создании базы данных используется *Мастер базы данных*, **Access** создает все необходимые связи между таблицами. Кроме того, при создании новой таблицы при помощи *Мастера таблиц* можно установить связь между создаваемой таблицей и существующими таблицами в базе данных. Кроме того, всегда можно запустить *Мастер по анализу таблиц*, который предложит допустимые варианты связей между таблицами. Удобнее возложить на **Access** задачу самостоятельного создания связей, но при желании можно создать, изменить или удалить связь.

Создание, модификация и удаление связей.

В большинстве связей одно из полей в двух таблицах имеет уникальное значение индекса (то есть **Access** требует, чтобы его значение не повторялось в различных записях). Уникальное поле является ключевым, или же его свойство *Индексированное поле* имеет значение *Да (совпадения не допускаются)*. Поле второй таблицы в типичной связи не обладает уникальным значением индекса, то есть его свойство *Индексированное поле* имеет значение *Нет* или *Да (Допускаются совпадения)*. Таблица, содержащая уникальное поле, обычно называется базовой, а вторая таблица — подчиненной. Поскольку одной записи базовой таблицы может соответствовать несколько записей в подчиненных таблицах, подобная связь называется «один – ко - многим».

Если связанное поле базовой таблицы не индексируется, **Access** не может ничего сказать о природе связи, и она описывается как «не определено».

Символ бесконечности над линией связи обозначает подчиненную таблицу (со «многими» записями в отношении «один - ко - многим»).

Символ «1» над линией связи обозначает базовую таблицу (с «одной» записью в отношении «один – ко - многим»).

При установке флажка *Обеспечение целостности данных* **Access** следит за тем, чтобы при вводе или изменении данных не нарушалась связь между таблицами.

После завершения работы со связями надо закрыть диалоговое окно *Схема данных*. Если макет окна был изменен, то есть, изменен состав включенных в него таблиц и их расположение, **Access** предложит сохранить его (при условии, что сами связи были сохранены ранее), что позволяет в дальнейшем воспользоваться тем же самым макетом.

7.1 Связывание данных

При импортировании данных в новую таблицу заносится их копия. Импортированные данные становятся неотъемлемой частью базы и ничем не отличаются от записей, введенных в **Access**. Существует и другая возможность — создать новую таблицу посредством связывания ее с

другим файлом базы данных. Связанные данные не копируются в новую таблицу, а остаются в исходном файле. **Access** хранит ссылку на этот файл, чтобы «связанные» данные можно было просмотреть или изменить при помощи таблицы **Access** (то же самое можно сделать и в программе, в которой был создан этот файл).

7.2 Обеспечение целостности данных

При задании связи между двумя таблицами можно установить флажок *Обеспечение целостности данных*, чтобы **Access** продолжал следить за логической структурой записей в таблицах во время ввода данных и работы с базой. Конкретно это означает, что каждой записи подчиненной таблицы должна быть сопоставлена правильная запись базовой таблицы. Для того чтобы этот флажок можно было установить, необходимо, чтобы типы данных связанных полей совпадали, а связь не была неопределенной. При установке этого флажка **Access** начинает следить за всеми изменениями, вносимыми в связанные поля.

Access не позволяет создать подчиненную запись, для которой не существует соответствующей записи в базовой таблице. Тем не менее, связанное поле в подчиненной таблице можно оставить пустым (это означает, что данная запись не имеет соответствующей).

Access не разрешает изменить значение в связанном поле базовой таблицы, если существуют связанные с ней записи в подчиненной таблице. Тем не менее, можно установить флажок Каскадное обновление связанных полей в диалоговом окне *Связи* — в этом случае изменение значения в связанном поле базовой таблицы приводит к аналогичным изменениям в соответствующих записях подчиненной таблицы, что гарантирует целостность данных в таблицах.

Access не разрешает удалить запись первичной таблицы, если существуют связанные с ней записи подчиненной таблицы. Однако можно установить флажок *Каскадное удаление связанных записей* — в этом случае удаление записи из базовой таблицы приводит к удалению всех связанных с ней записей подчиненной таблицы.

Access рисует толстую линию с символами «1» и «бесконечность» лишь при установке флажка *Обеспечение целостности данных*. При снятом флажке он рисует только тонкую линию без всяких символов.

8. Ввод и просмотр данных в режиме таблицы.

Созданная в режиме Конструктор таблица может использоваться для заполнения дальнейшего редактирования и просмотра записей в Режиме таблицы, который позволяет просмотреть данные, организованные в виде строк и столбцов, и является одним из наиболее распространенных способов работы с информацией в базе данных. В режиме таблицы можно работать и с формами, однако, с ними удобнее работать в соответствующем режиме форм

В таблице, отображаемой в режиме таблицы, каждый столбец соответствует одному полю в таблице, а каждая строка соответствует записи. Несмотря на внешнюю простоту режима таблицы, можно изменить таблицу различными способами и обойтись без создания специальных форм или отчетов. В режиме таблицы можно выполнять следующие операции:

просмотр и редактирование данных;

настройку макета режима таблицы с изменением размера и порядка строк и столбцов;

добавление, удаление и переименование полей в таблице;

быстрое и удобное добавление, удаление и редактирование полей в таблице;

сортировку записей;

поиск по значению поля;

фильтрацию данных, при которой в таблице отображаются только записи, удовлетворяющие определенному критерию.

8.1 Просмотр данных в режиме таблицы.

При открытии таблицы или запроса в окне базы данных, они отображаются в режиме таблицы. Такие приемы изменения макета таблицы, как изменение ширины столбца или высоты строки, а также перестановка столбцов, относятся только к режиму таблицы и никак не влияют на другие режимы или структуру таблицы, запроса или отчета. Другие изменения отражаются на структуре таблицы и на ее внешнем виде в других режимах (например, в режиме конструктора). Среди таких изменений: добавление, удаление и переименование полей. Эти изменения можно внести только в таблицу, но не в форму или запрос.

Многие приемы для изменения макета в режиме таблицы напоминают аналогичные операции с листами **Microsoft Excel**. Ширина столбца изменяется перетаскиванием его границы, аналогичным способом изменяется и высота строки.

Чтобы **Access** автоматически определил оптимальную ширину столбца, надо сделать двойной щелчок на правом крае заголовка столбца. Столбцу будет назначена минимальную ширину, при которой отображается вся содержащаяся в нем информация, включая текст заголовка.

Перестановка столбцов в режиме таблицы не влияет на порядок полей в режиме конструктора или других режимах.

В режиме таблицы в заголовке каждого поля **Access** обычно выводит его имя. Однако при задании свойства *Подпись*, текст, указанный в качестве подписи, выводится вместо имени поля. Если для поля определено свойство *Подпись*, то переименование поля в режиме таблицы приводит к очистке данного свойства.

В режиме таблицы можно добавлять новые и удалять существующие столбцы. **Access** создает новый столбец присваивая ему имя, состоящее из слова Поле, за которым следует число: первый столбец получит имя Поле1, второй — Поле2 и т. д.

Необходимо учитывать, что при удалении столбца из таблицы удаляется соответствующее поле вместе со всеми хранящимися в нем данными! Если после удаления столбца выяснится, что информация из него должна была оставаться на своем месте, придется заново создать поле и повторить ввод всех потерянных данных, возможно, для многих тысяч записей. Поэтому нужна осторожность при удалении столбцов и периодическое выполнение операции резервного копирования базы данных.

Но вместо удаления можно временно скрыть один или несколько столбцов, а позднее скрытые столбцы можно снова вывести на экран.

Access автоматически сохраняет любые изменения, которые влияют на структуру таблицы (переименование, добавление или удаление полей, а также правка записей). Однако при модификации макета таблицы (изменении ширины столбцов и высоты строк, порядка столбцов) необходимо сохранить изменения стандартным образом.

Если этого не сделать сразу, то **Access** предложит сохранить новый макет при выходе из режима таблицы. Если изменения окажутся ненужными при следующем переходе в режим таблицы все окажется в первоначальном состоянии.

8.2 Ввод и редактирование данных в режиме таблицы.

В последнюю строку таблицы можно добавлять новые записи. Она помечена символом звездочки (в крайнем левом столбце), показывающим, где появится новая запись.

При вводе информации в запись, значок в крайнем левом столбце (селекторе записей) заменяется карандашом с двумя точками. При вводе данных в новую запись последней строки **Access** немедленно создает пустую запись под той, с которой пользователь работает в настоящий момент.

В некоторых полях имеется раскрывающийся список или список с полем для выбора значения из заданного перечня. Когда курсор оказывается в таком поле справа от поля, отображается кнопка с указывающей вниз стрелкой, щелчком по которой можно открыть этот список. Входящие в список варианты создаются *Мастером подстановок* и являются единственно допустимыми значениями поля. Чтобы ввести данные, надо выбрать один из элементов списка или ввести его имя с клавиатуры так, как оно представлено в списке.

Использование форматированных полей.

Если свойство *Формат* было задано, то достаточно ввести значение и нажать клавишу *Tab* для перехода к следующему полю. **Access** немедленно отформатирует введенное значение.

Пример. Если для поля задан денежный тип данных, после чего было введено число 2,1 и нажата клавиша *Tab*, **Access** преобразует информацию в денежный формат, то есть к виду $2.10.

Удаление записи.

Чтобы удалить запись, ее надо выделить щелчком в заголовке слева от строки, после чего выполнить команду *Удалить запись*. Запись исчезает из таблицы, а на экране открывается окно диалога, в котором точно указывается удаляемый объект. Если удаление ошибочно, надо щелкнуть по кнопке *Нет*, чтобы **Access** восстановил запись. Чтобы продолжить удаление, надо щелкнуть по кнопке *Да*.

После того, как запись будет удалена, и щелчком по кнопке *Да* будет выполнено подтверждение на удаление, восстановить запись будет невозможно. Команда *Отменить* из меню *Правка* не отменяет операции удаления записей. Чтобы вернуть удаленную запись в таблицу, придется вводить информацию заново.

Необходимо учесть, что при удалении строки (записи) из базы пропадают значения всех полей, относящиеся к одному информационному объекту (например, клиенту), тогда как удаление столбца (поля), являющегося информационной категорией (например, поле *Адрес*), удаляет хранимые в нем сведения из всех записей таблицы. Хотя любых ошибочных удалений желательно избегать, в общем случае все же бывает проще заново ввести одну запись (все сведения об одном клиенте), чем поле (адрес всех клиентов в базе).

При необходимости можно выделить группу записей и удалить их одной командой.

Если запись удаляется из таблицы, связанной с другой таблицей, может возникнуть необходимость в удалении одной или нескольких записей в другой таблице (для сохранения целостности данных). **Access** уведомляет об этом соответствующим сообщением.

Связи, допускающие каскадные удаления, приведут к удалению в данной и в связанных с ней таблицах следующего числа записей:

Необходимо решить, стоит ли удалять дополнительные записи, которые не видны в настоящий момент. При неопределенности по этому вопросу надо нажать кнопку *Нет*, чтобы **Access** отменил удаление, и попытаться определить состав удаляемых связанных записей. В их поиске может помочь окно *Схема данных.*

Задания к лабораторным работам (Access)

Задание 1. ***Создание базы данных***.

1. Создайте новую Базу данных и присвойте имени файла Вашу фамилию.

2. Создайте в режиме КОНСТРУКТОР структуру следующей таблицы:

Коммерческие банки Российской Федерации

Код банка	*Наименование банка*	*Адрес банка*	*Председатель правления*	*Дата регистрации*	*Уставный фонд*

определите типы данных и необходимые свойства и параметры для полей таблицы.

3. Создайте ключевое поле (здесь ключ состоит из одного поля – ***Код банка***)

4. Введите в таблицу «Коммерческие банки Российской Федерации» новые поля перед полем ***Уставный фонд***:

Характеристика банка (тип данных МЕМО);

Телефон справочной службы банка;

5. Создайте в режиме КОНСТРУКТОР структуру следующей таблицы:

Ставки по рублевым вкладам

Код банка	*Вид вклада*	*Срок вклада*	*Дата создания вклада*	*Минимальная сумма вклада*	*Процентная ставка*

6. Создайте составной ключ (здесь ключ состоит из двух полей – ***Код банка***, ***Вид вклада***).

7. Установите следующие условие при вводе данных в таблицы «Ставки по рублевым вкладам» и «Ставки по «коротким» деньгам»:

- поле ***Дата создания вклада*** больше 01 января 2005 г.
- поле ***Срок вклада*** - 1, 3, 6, 12,18, 24 месяцев для «Ставки по рублевым вкладам» и с 1 по 31 дней для “Ставки по «коротким» деньгам»

8. Скопируйте только структуру таблицы «Ставки по рублевым вкладам» в новую таблицу «Ставки по валютным вкладам». Внесите необходимые изменения в свойства соответствующих полей.

Задание 2. *Создание схемы данных.*

1. Создайте схему данных для всех таблиц (меню СЕРВИС):

- связь может быть установлена только при наличии в таблицах полей с совпадающими значениями данных, т.е. типов данных и размеров полей.
- свяжите таблицу «Коммерческие банки Российской Федерации» с таблицами «Ставки по рублевым вкладам», «Ставки по валютным вкладам», связь задать по полям ***Код банка*** - с отношением «**один ко многим**»;

- установите «обеспечение целостности данных»;
- установите «каскадное обновление связанных полей» и «каскадное удаление связанных записей»;

2. Расположите таблицы в схеме данных (т.е. на экране) без пересечений связей и с максимальным просмотром всех имеющихся полей в таблицах (по возможности).

Закрыть схему данных с сохранением изменений.

Задание 3. ***Создание форм.***

1. Создайте следующие формы:

- **форма ввода данных с кнопками** для таблицы «Коммерческие банки Российской Федерации» (используя режим МАСТЕРА) **Коммерческие банки Российской Федерации** в один столбец для всех полей;
  - ✓ В заголовке формы создайте шапку формы с названием заполняемой таблицы и информацию о разработчике формы (фамилия и номер группы);
  - ✓ добавьте в **Область примечаний** формы кнопки перехода к следующей записи, к предыдущей записи, к первой записи, к последней записи, удаления текущей записи, закрыть форму;
- **форма ввода данных с кнопками** отдельные формы для каждой таблицы: «Ставки по валютным вкладам», «Ставки по рублевым вкладам» (используя режим МАСТЕРА): **Ставки по валютным вкладам, Ставки по рублевым вкладам** в один столбец для всех полей;
  - ✓ В заголовке формы создайте шапку формы с названием заполняемой таблицы и информацию о разработчике формы (фамилия и номер группы);
  - ✓ вместо поля КОД БАНКА создайте список, связанный с таблицей «Коммерческие банки Российской Федерации», с полями Код банка и Наименование банка;
  - ✓ вместо поля Срок вклада создайте список со значениями, соответствующими условиям поля в таблицах.
  - ✓ добавьте в Область примечаний формы кнопки перехода к следующей записи, к предыдущей записи, к первой записи, к последней записи, удаления текущей записи, закрыть форму. Там же создайте вычисляемое поле с отображением текущей даты;

В формах расположите поля наиболее удобным для заполнения и просмотра образом и создайте необходимые оформительские графические элементы и отформатируйте элементы формы и саму форму.

Используя созданные формы введите **десять** записей в таблицу «Коммерческие банки Российской Федерации» и по двадцать записей в остальные таблицы (значения поля ***Код банка*** в таблицах «Ставки по рублевым вкладам», «Ставки по валютным вкладам» берется из значений поля ***Код банка***, введенных в таблицу «Коммерческие банки Российской Федерации»), при этом:

- ⇒ Ставки по рублевым вкладам - введите несколько записей с одинаковыми значениями поля ***Код банка***, взятыми из таблицы **Коммерческие банки Российской Федерации,** несколько записей с одинаковыми значениями поля ***Вид вклада***;
- ⇒ Ставки по валютным вкладам - введите несколько записей с одинаковыми значениями поля ***Код банка***, взятыми из таблицы **Коммерческие банки Российской Федерации,** несколько записей с одинаковыми значениями поля ***Вид вклада***.

4. Сохраните **каждую** таблицу базы данных во внешнем файле в формате ***Rich Text Format*** (расширение **.rtf**). Перейдите в редактор ***Microsoft Word*** и проверьте содержимое созданных файлов.

Задание 4. ***Создание запросов.***

1. Создайте следующие запросы и выполните просмотр обрабатываемых данных в таблицах, соответствующих заданному условию:

- **запрос-выборка ПАРАМЕТР** (режим КОНСТРУКТОР для таблиц «Коммерческие банки Российской Федерации» и «Ставки по рублевым вкладам») используемые поля для вывода: ***Наименование банка***, ***Вид вклада***, ***Минимальная сумма вклада***; в качестве условия для поля ***Минимальная сумма вклада*** установить ввод параметра: ВВЕДИТЕ СУММУ ВКЛАДА, добавить вычисляемое поле ***Сумма выплачиваемая вкладчику по вкладу*** и отформатировать его;
- **перекрестный запрос КОЛИЧЕСТВО ВКЛАДОВ** (для таблицы **Ставки по валютным вкладам**, в режиме ЗАПРОС подрежим ВЫБОРКА) (поле для заголовков строк КОД БАНКА, поле для заголовков столбцов МИНИМАЛЬНАЯ СУММА ВКЛАДА, поле для вычислений ВИД ВКЛАДА, функция ЧИСЛО или COUNT);

Задание 5. Создание отчетов.

1. Создайте следующие отчеты:

- **простой отчет** (для таблицы «Коммерческие банки Российской Федерации» режим МАСТЕР) **Коммерческие банки**;
- **табличный отчет** (для таблиц «Ставки по валютным вкладам» и «Коммерческие банки Российской Федерации» режим МАСТЕР) **Ставки по валютным вкладам**, добавив поле ***Наименование банка***, из таблицы «Коммерческие банки Российской Федерации», задать группировку по полу ***Вид вклада***, сортировку по полю ***Наименование банка***.

2. Отформатировать отчеты.

- ✓ Первый лист (заголовок отчета) – титульный, с названием учебного заведения, таблицы, фамилией и группы разработчика.
- ✓ Каждая группа данных должна начинаться с нового листа.
- ✓ Колонтитулы располагаются на всех листах, кроме первого (заголовка) и содержат: верхний – название таблицы, нижний – номер страницы и дату печати.
- ✓ Примечание отчета начинается с новой страницы и содержит текст и вычисляемые поля – общее количество записей по отчету, общее количество страниц в отчете, сумма выплачиваемая по процентам вкладчикам по отчету, количество видов вкладов (групп) по отчету.
- ✓ Добавить различные графические объекты для оформления отчета. Расположить отчет на листе А4 книжной ориентации.

Задание 6. ***Создание управляющей формы.***

- **управляющая кнопочная форма** в режиме КОНСТРУКТОР:
  - ✓ создать кнопки для перехода к созданным ранее формам, запросу; отчетам;
  - ✓ над кнопками добавить текст: «Информационная система по коммерческим банкам российской федерации»;
  - ✓ создать кнопку выхода из базы данных;
  - ✓ добавить внизу формы фамилию разработчика базы данных и добавить вычисляемое поле «текущая дата»;
  - ✓ добавить графические элементы для оформления и выполнить необходимое форматирование формы и элементов формы
  - ✓ установить режим автоматического запуска кнопочной формы при открытии базы данных.

Ход работы прокомментировать в документе «Отчет по лабораторной работе» с использованием скриншотов.