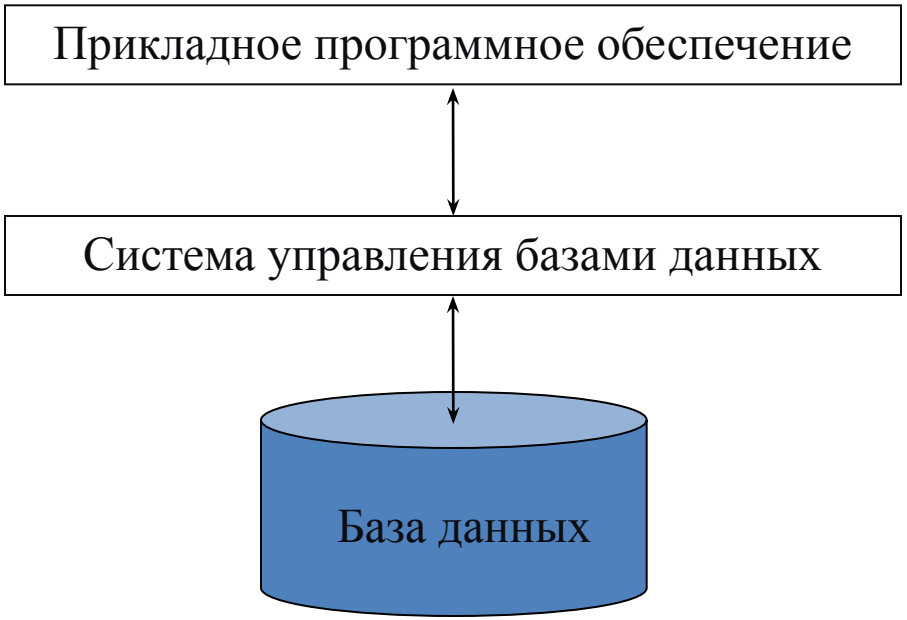
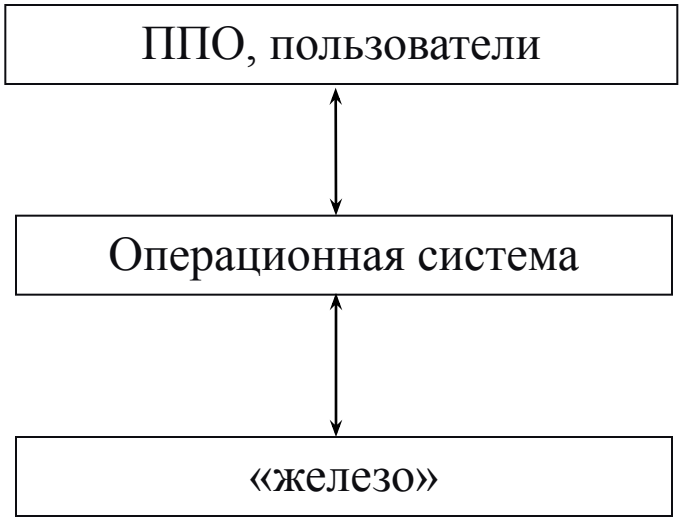
Реляционные системы управления базами данных

Основные концепции.

Основные функции СУБД





- Обеспечение доступа ППО к базе данных
- Управление базой данных

СУБД

Программные составляющие СУБД включают в себя ядро и сервисные средства (утилиты).

- **Ядро СУБД** – это набор программных модулей, необходимый и достаточный для создания и поддержания БД, то есть универсальная часть, решающая стандартные задачи по информационному обслуживанию пользователей.
- **Сервисные программы** предоставляют пользователям ряд дополнительных возможностей и услуг, зависящих от описываемой предметной области и потребностей конкретного пользователя.

Системой управления базами данных называют программную систему, предназначенную для создания на ЭВМ общей базы данных для множества приложений, поддержания её в актуальном состоянии и обеспечения эффективного доступа пользователей к содержащимся в ней данным в рамках предоставленных им полномочий.

Классификация СУБД

По степени универсальности СУБД делят на два класса:

1. СУБД общего назначения (СУБД ОН)
2. специализированные СУБД (СпСУБД).

Специализированные СУБД создаются в тех случаях, когда ни одна из существующих СУБД общего назначения не может удовлетворительно решить задачи, стоящие перед разработчиками. Причин может быть несколько:

- не достигается требуемого быстродействия обработки данных;
- необходима работа СУБД в условиях жёстких аппаратных ограничений;
- требуется поддержка специфических функций обработки данных.

СпСУБД предназначены для решения конкретной задачи.
Создание СпСУБД – дело весьма трудоёмкое.

Классификация СУБД

По методам организации хранения и обработки данных СУБД делят на

- **Централизованные**
- **Распределённые**.

Первые работают с БД, которая физически хранится в одном месте (на одном компьютере). Большинство централизованных СУБД перекладывает задачу организации удалённого доступа к данным на сетевое обеспечение, выполняя только свои стандартные функции.

По модели данных различают **иерархические**, **сетевые**, **реляционные**, **объектно-реляционные** и **объектно-ориентированные** СУБД.

Требования к реляционным СУБД (по Кодду)

1. **Явное представление данных** (The Information Rule). Информация должна быть представлена в виде данных, хранящихся в ячейках. Данные, хранящиеся в ячейках, должны быть атомарны. Порядок строк в реляционной таблице не должен влиять на смысл данных.
2. **Гарантированный доступ к данным** (Guaranteed Access Rule). К каждому элементу данных должен быть гарантирован доступ с помощью комбинации имени таблицы, первичного ключа строки и имени столбца.
3. **Полная обработка неизвестных значений** (Systematic Treatment of Null Values). Неизвестные значения (**NULL**), отличные от любого известного значения, должны поддерживаться для всех типов данных при выполнении любых операций.

Требования к реляционным СУБД (по Кодду)

4. **Доступ к словарю данных** в терминах реляционной модели (Dynamic On-Line Catalog Based on the Relational Model). **Словарь данных** должен сохраняться в форме реляционных таблиц, и СУБД должна поддерживать доступ к нему при помощи стандартных языковых средств.
5. **Полнота подмножества языка** (Comprehensive Data Sublanguage Rule). Система управления реляционными базами данных должна поддерживать **единственный язык запросов**, который позволяет выполнять все операции работы к данным:

- операции определения данных,
- операции манипулирования данными,
- управление доступом к данным,
- управление транзакциями.

Требования к реляционным СУБД (по Кодду)

6. Поддержка **обновляемых представлений** (View Updating Rule). Обновляемое представление должно поддерживать все операции манипулирования данными, которые поддерживают реляционные таблицы: операции выборки, вставки, модификации и удаления данных.
7. Наличие **высокоуровневых операций** управления данными (High-Level Insert, Update, and Delete). Операции вставки, модификации и удаления данных должны поддерживаться не только по отношению к одной строке реляционной таблицы, но по отношению к любому множеству строк.

Требования к реляционным СУБД (по Кодду)

8. **Физическая независимость данных** (Physical Data Independence). Приложения не должны зависеть от используемых способов хранения данных на носителях, от аппаратного обеспечения компьютеров, на которых находится реляционная база данных.
9. **Логическая независимость данных** (Logical Data Independence). Представление данных в приложении не должно зависеть от структуры реляционных таблиц.

Требования к реляционным СУБД (по Кодду)

10. **Независимость контроля целостности** (Integrity Independence). Вся информация, необходимая для поддержания целостности, должна находиться в словаре данных. СУБД должна выполнять проверку заданных **ограничений целостности** и автоматически поддерживать целостность данных.
11. **Независимость от распределенности** (Distribution Independence). База данных может быть распределенной, может находиться на нескольких компьютерах, и это не должно оказывать влияние на приложения.
12. **Согласование языковых уровней** (Non-Subversion Rule). Не должно быть иного средства доступа к данным, отличного от стандартного языка работы с данными. Если используется низкоуровневый язык доступа к данным, он не должен игнорировать правила безопасности и целостности, которые поддерживаются языком более высокого уровня.

Основные функции реляционной СУБД

1. Поддержка многопользовательского доступа.
2. Обеспечение физической целостности данных.
3. Управление доступом.
4. Настройка РСУБД.

- подключение внешних приложений к БД;
- модификация параметров организации среды хранения;
- изменение структуры хранимых данных или их размещения в среде хранения (**реорганизация БД)**;
- модификацию концептуальной схемы данных (**реструктуризация БД)**.

Требования к составу и функциям СУБД

1. Хранение, извлечение и обновление данных.
2. Каталог (ССД), доступный конечным пользователям.
3. Поддержка транзакций.
4. Служба управления параллельной работой.
5. Службы восстановления.
6. Службы контроля доступа к данным.
7. Службы поддержки целостности данных.
8. Службы поддержки независимости от данных.
9. Вспомогательные службы.

Администрирование базы данных

В задачи администратора входит выполнение нескольких групп функций:

1. Администрирование предметной области
2. Администрирование БД:
   - изменения в структуре хранимых данных
   - изменения способов размещения данных в памяти
   - изменения используемых методов доступа к данным
3. Администрирование приложений
4. Администрирование безопасности данных

Вспомогательные службы

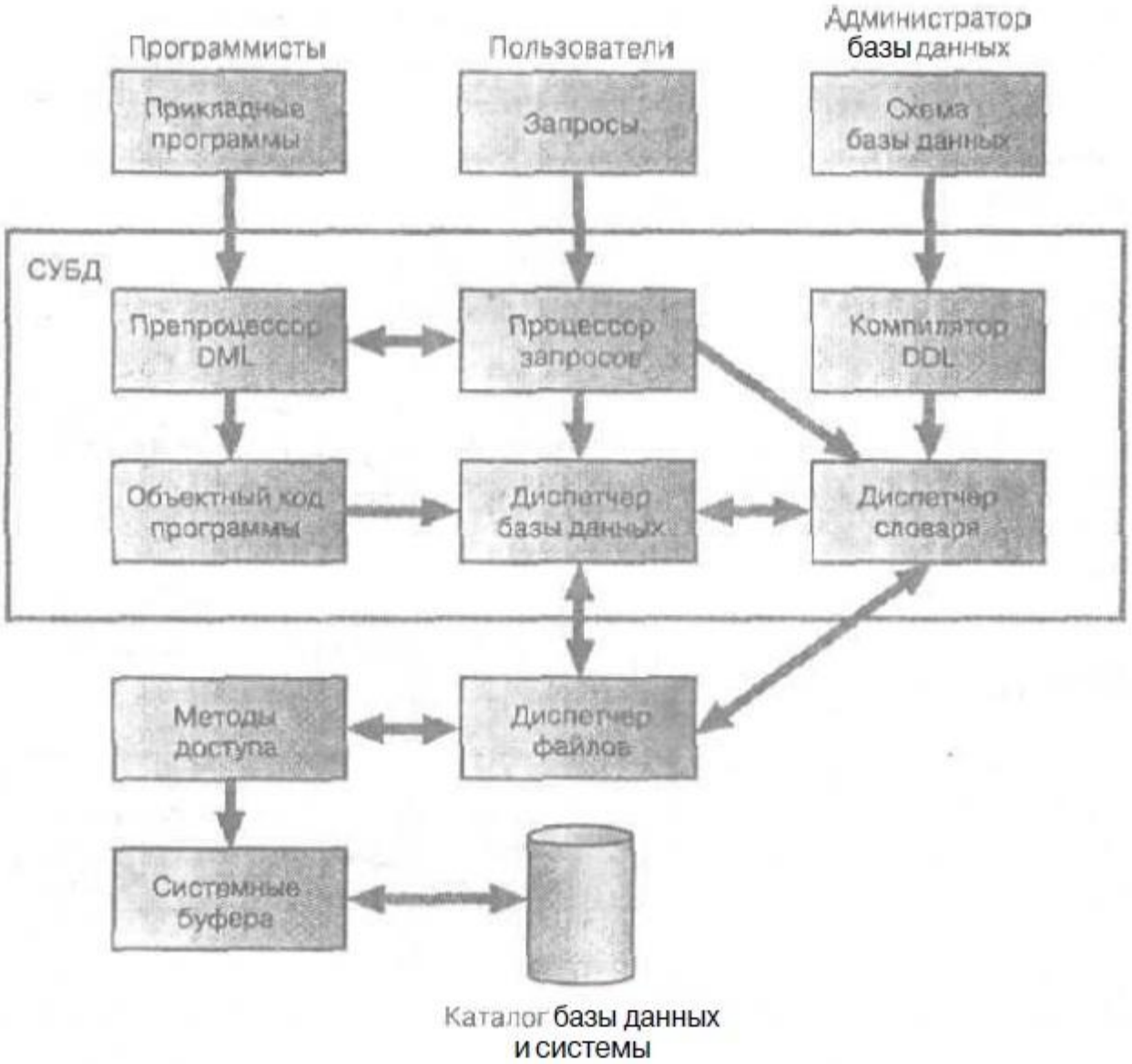
Обычно предназначены для оказания помощи АБД в эффективном администрировании базы данных.

Некоторые примеры подобных утилит.

- Утилиты импортирования.
- Средства мониторинга.
- Программы статистического анализа.
- Инструменты реорганизации индексов.
- Инструменты сборки мусора и перераспределения памяти для физического устранения удаленных записей с запоминающих устройств, объединения освобожденного пространства и перераспределения памяти по мере необходимости.

Преимущества наличия ССД

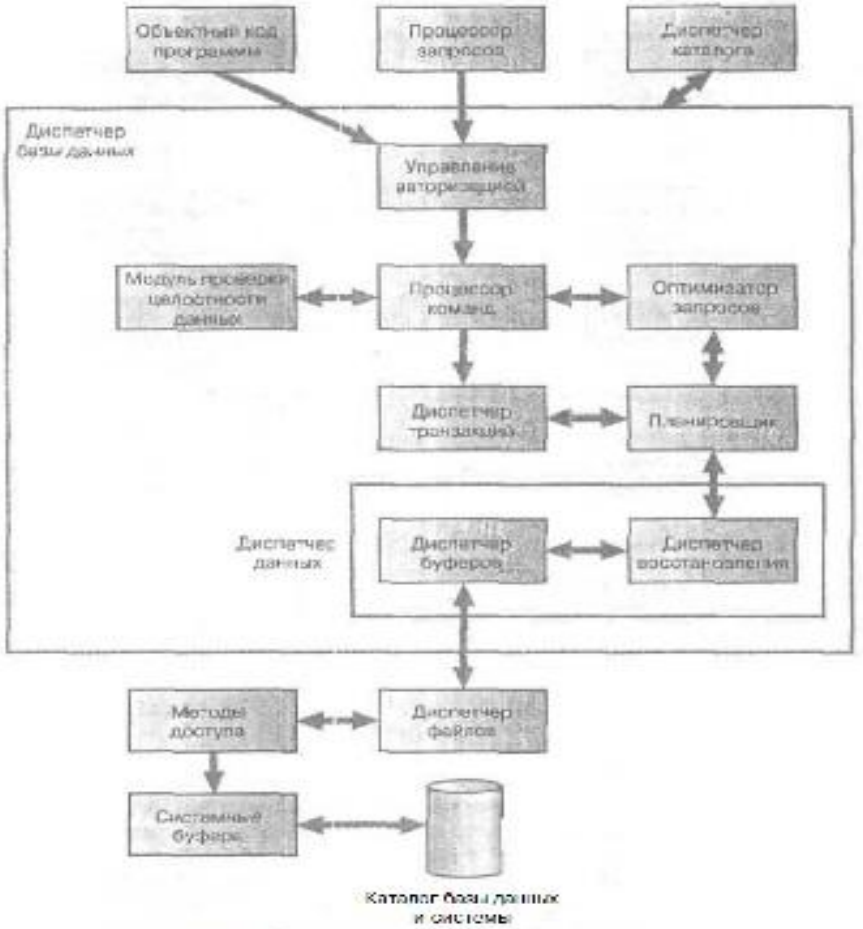
- Информация о данных может быть централизованно собрана и сохранена.
- Можно определить смысл данных.
- Упрощается общение.
- В системном каталоге также могут быть указаны один или несколько пользователей.
- Благодаря централизованному хранению избыточность и противоречивость описания отдельных элементов данных могут быть легко обнаружены.
- Внесенные в базу данных изменения могут быть запротоколированы.
- Последствия любых изменений могут быть определены еще до их внесения.
- Меры обеспечения безопасности могут быть дополнительно усилены.
- Появляются новые возможности организации поддержки целостности данных.
- Может выполняться аудит хранимой информации.



Основные компоненты типичной системы управления базами данных

Основные программные компоненты СУБД

- **Процессор запросов.**
- **Диспетчер базы данных**.
- **Диспетчер файлов**.



Компоненты диспетчера базы данных

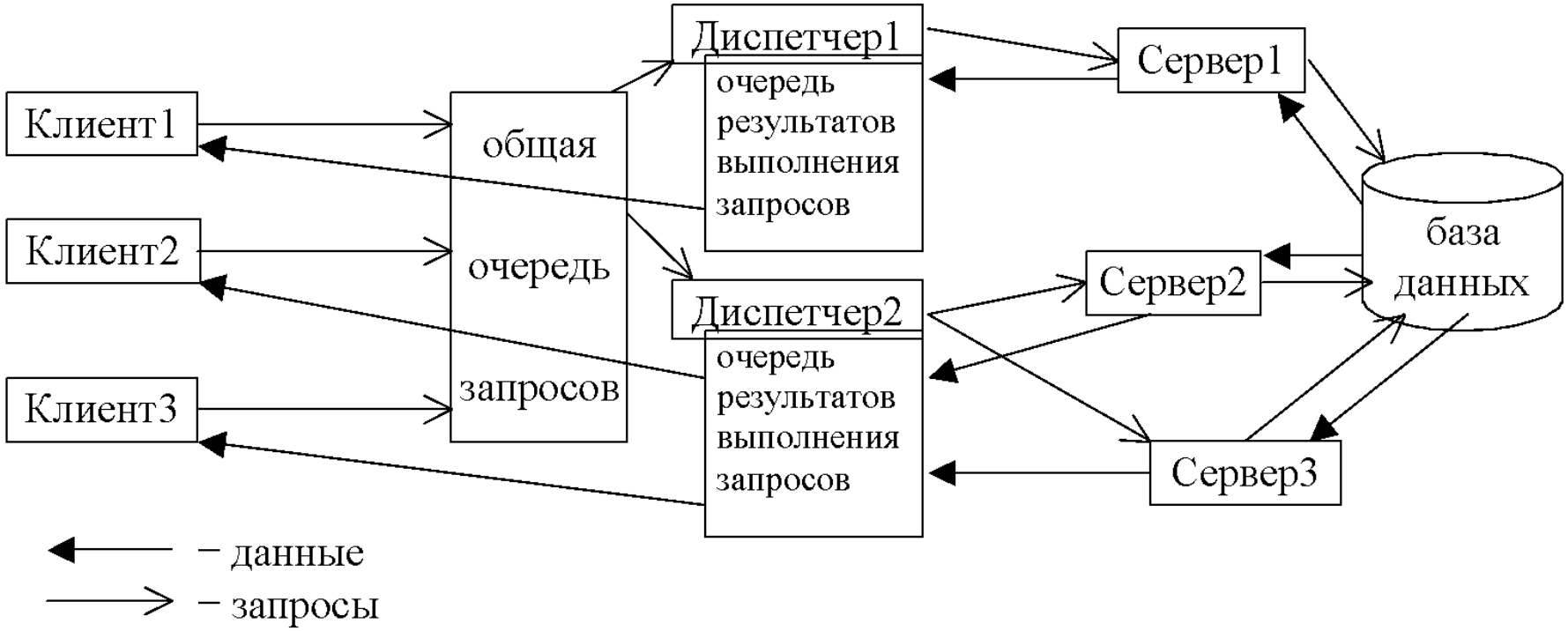
Основные программные компоненты СУБД

- **Препроцессор языка DML**.
- **Компилятор языка DDL**.
- **Диспетчер словаря**.
- **Модуль контроля прав доступа**.
- **Процессор команд**.
- **Средства контроля целостности**.
- **Оптимизатор запросов**.

Основные программные компоненты СУБД

- **Диспетчер транзакций**.
- **Планировщик**.
- **Диспетчер восстановления**.
- **Диспетчер буферов**.

- Архитектура MTS



Лекция.

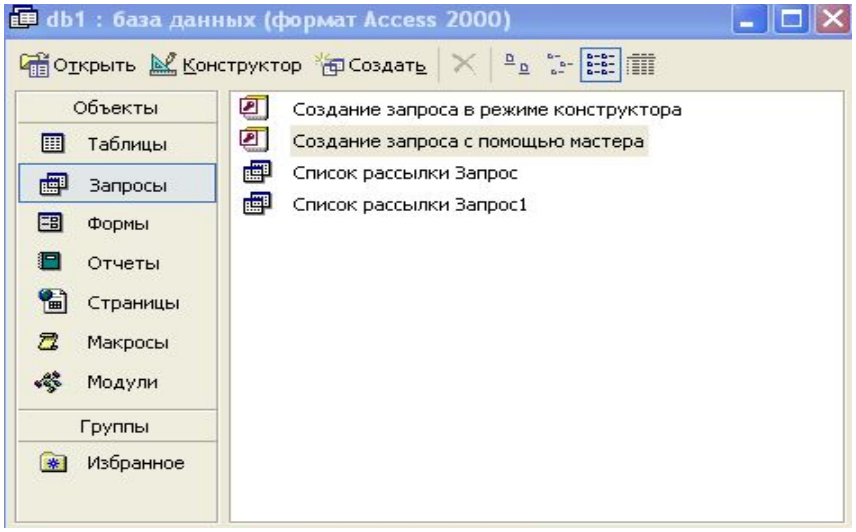
MS Access: запросы, формы, отчеты, макросы.

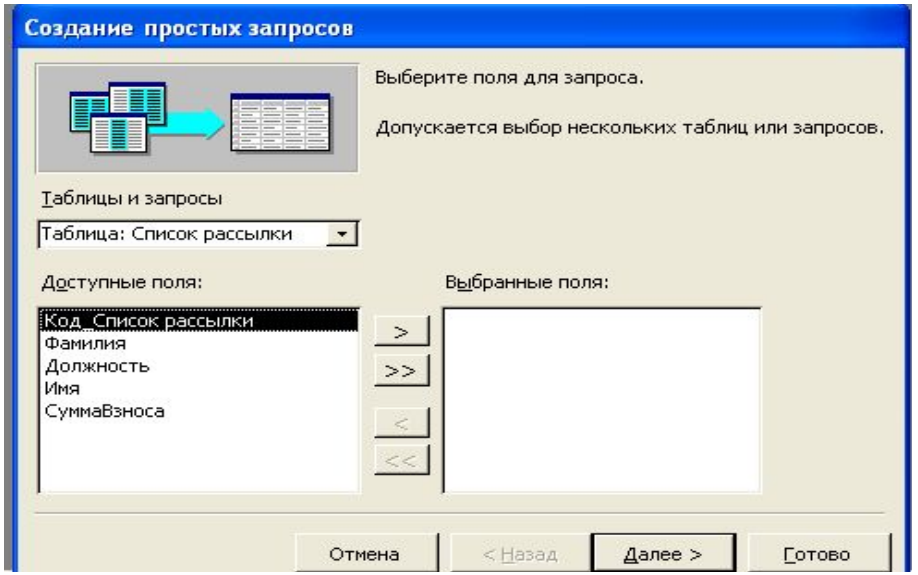
Запрос — это объект базы данных, являющийся основным инструментом **выборки, обновления и обработки данных** в таблицах базы данных.

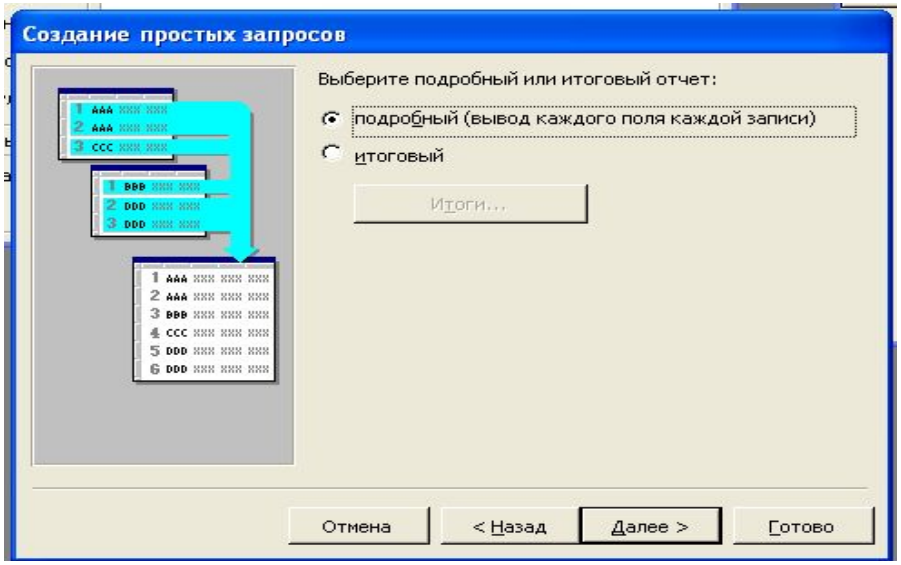
Запрос на выборку позволяет сформировать пользовательское представление о данных. **Результат выполнения запроса на выборку — это новая, чаще всего временная, таблица, которая существует до закрытия запроса. Записи формируются путем объединения записей таблиц, участвующих в запросе. Условия отбора, сформулированные в запросе, позволяют фильтровать записи, составляющие результат объединения таблиц.**

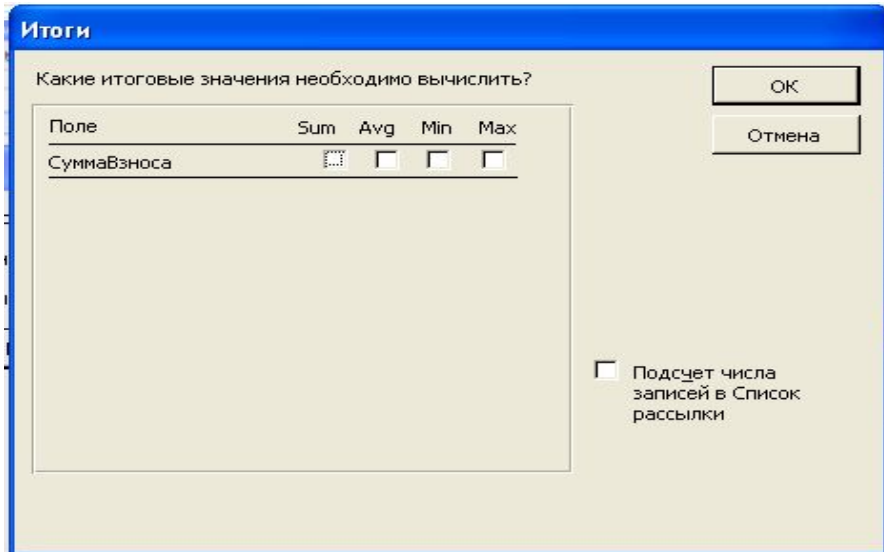
Простейшие запросы могут быть созданы с помощью мастера. **Любой запрос** можно создать в режиме конструктора.

Мастер запросов

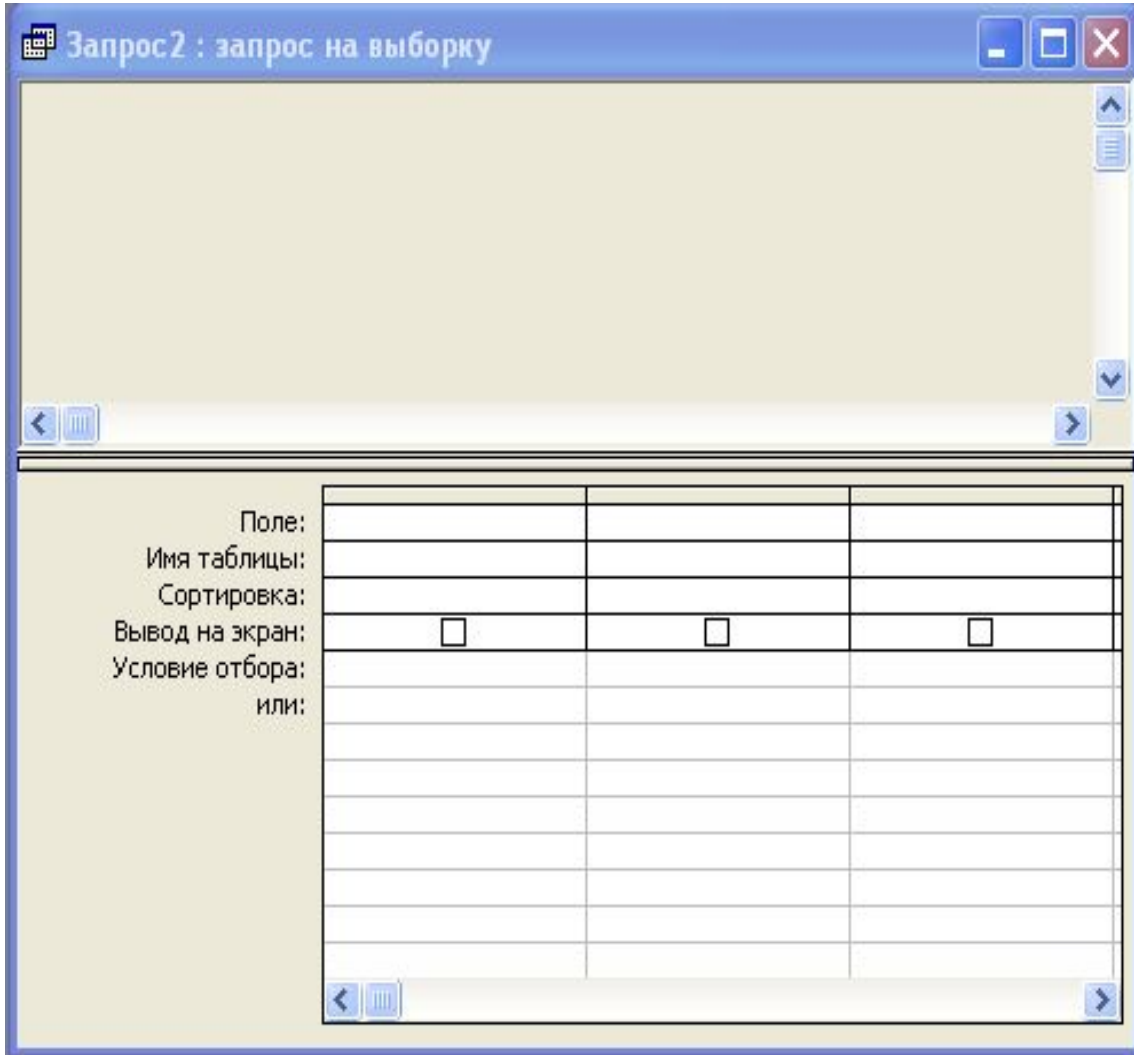






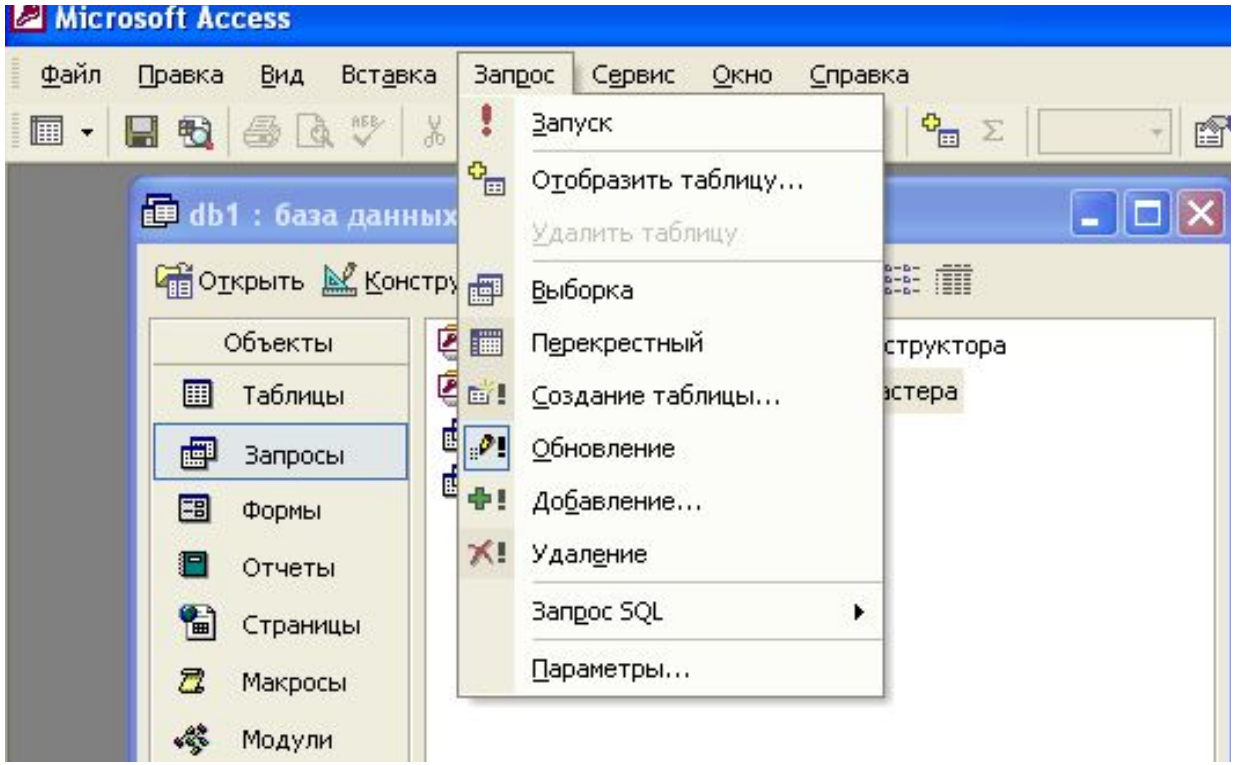


Конструктор запросов



Назначение и виды запросов

Запрос позволяет выбрать необходимые данные из одной или нескольких взаимосвязанных таблиц, произвести вычисления и получить результат в виде виртуальной таблицы.



С помощью запроса можно выполнить следующие **виды обработки данных**:

-сформировать на основе объединения записей взаимосвязанных таблиц **новую виртуальную** таблицу;
-**включить** в результирующую таблицу запроса **заданные пользователем поля**;
-выбрать записи, удовлетворяющие **условиям отбора**;
-произвести **вычисления** в каждой из полученных записей;
-**сгруппировать записи**, которые имеют одинаковые значения в одном или нескольких полях, в одну запись с одновременным выполнением над другими полями статистических функций;
-добавить в результирующую таблицу запроса **строку итогов**;
-произвести **обновление полей** в выбранном подмножестве записей;
-создать **новую таблицу** базы данных, используя данные из существующих таблиц;

В Access может быть создано несколько **видов запроса**:

-запрос на выборку — выбирает данные из взаимосвязанных таблиц базы данных и таблиц запросов. Результатом является таблица, которая существует до закрытия запроса. На основе такого запроса могут строиться запросы других видов;

-запрос на создание таблицы — также выбирает данные из взаимосвязанных таблиц и других запросов, но в отличие от запроса на выборку результат сохраняется в новой постоянной таблице базы данных;

-запросы на обновление, добавление, удаление — являются запросами, в результате выполнения которых изменяются данные в таблицах.

Для создания запроса может быть использован либо **режим конструктора, либо мастер**. Если пользователь знаком с созданием инструкций SQL можно создать запрос в **режиме SQL**.

Вычисляемые поля

В запросе для каждой записи могут производиться вычисления с числовыми, строковыми значениями или значениями дат с использованием данных из одного или нескольких полей.

Результат вычисления образует в таблице запроса новое вычисляемое поле.
В выражениях вычисляемых полей помимо имен полей могут использоваться константы и функции.

В результате обработки выражения может получаться только одно значение.

Групповые операции в запросах. Назначение групповых операций

Групповые операции позволяют выделить группы записей с одинаковыми значениями в указанных полях и использовать для этих групп одну из **статистических функций**.

В Access предусматривается **девять статистических функций:**
Sum — сумма значений некоторого поля для группы;
Avg — среднее от всех значений поля в группе;
Max, Min — максимальное, минимальное значение поля в группе;
Count — число значений поля в группе без учета пустых значений;
StDev – среднеквадратичное отклонение от среднего значения поля в группе.
Var -дисперсия значений поля в группе;
First, Last — значение поля из первой или последней записи в группе.
Результат запроса с использованием групповых операций содержит по одной записи для каждой группы.

Многотабличные запросы

Многотабличный запрос часто осуществляет объединение данных, которые на этапе проектирования были разделены им множество объектов в соответствии с требованиями нормализации.

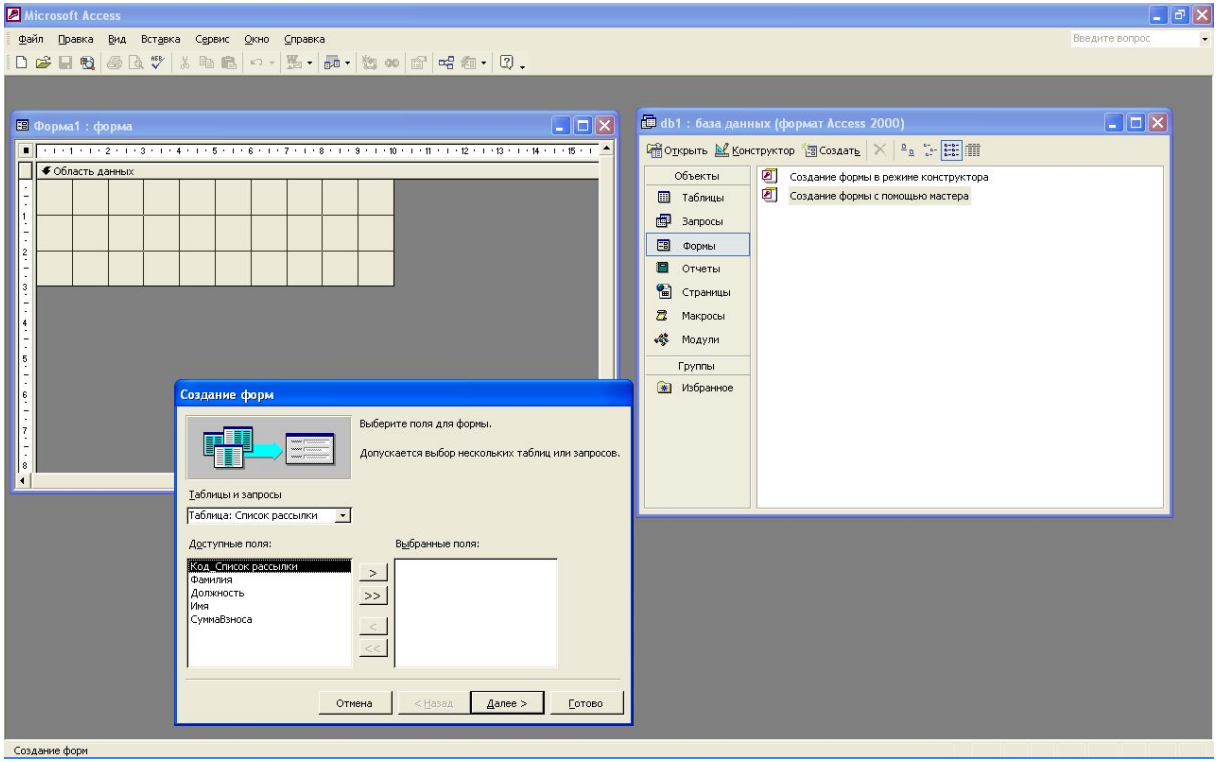
Разделение на объекты обеспечивает, прежде всего, отсутствие дублируемости данных и базе, повторяются только значения ключевых полей. **В результате выполнения запроса формируется таблица с повторяющимися данными**, в которой каждая запись собирает необходимые данные из разных объектов — таблиц.

При проектировании и конструировании запроса важнейшим условием является правильное представление о том, как идет объединение записей таблиц при формировании результата.

Формы

Формы являются основой разработки **диалоговых приложений пользователя для работы с базой данных**.

Работая с формой, пользователь может **добавлять, удалять и изменять записи таблиц, получать расчетные данные**. В процессе работы может осуществляться **контроль вводимых данных, могут проверяться ограничения на доступ к данным, выводиться необходимые дополнительные сведения.**



Форма состоит из **элементов управления**, которые отображают поля таблиц, и графические элементы, не связанные с полями таблиц. Графические элементы управления предназначены, прежде всего, для разработки **макета формы**: **надписей, внедряемых объектов** (рисунков, диаграмм), **вычисляемых полей**, **кнопок**, выполняющих печать, открывающих другие объекты или задачи.

Как форма в целом, так и каждый из ее элементов обладает множеством свойств. Посредством их изменения можно настроить внешний вид, размер, местоположение элементов в форме, определить источник данных формы, режим ввода/вывода, привязать к элементу выражение, макрос или программу. Набор свойств доступен в соответствующем окне, где они разбиты на категории, каждая из которых представлена на своей вкладке. Основными вкладками в окне свойств являются:

Макет — представляет свойства, ориентированные на определение внешнего вида формы или ее элементов;

Данные— представляет свойства для определения источника данных формы или ее элементов, режима использования формы (только разрешение на изменение, добавление, удаление и т. п.);

События — событиями называют определенные действия, возникающие при работе с конкретным объектом или элементом: нажатие кнопки мыши, изменение данных, до обновления, после обновления, открытие или закрытие формы и т. д. Они могут быть инициированы пользователем или системой

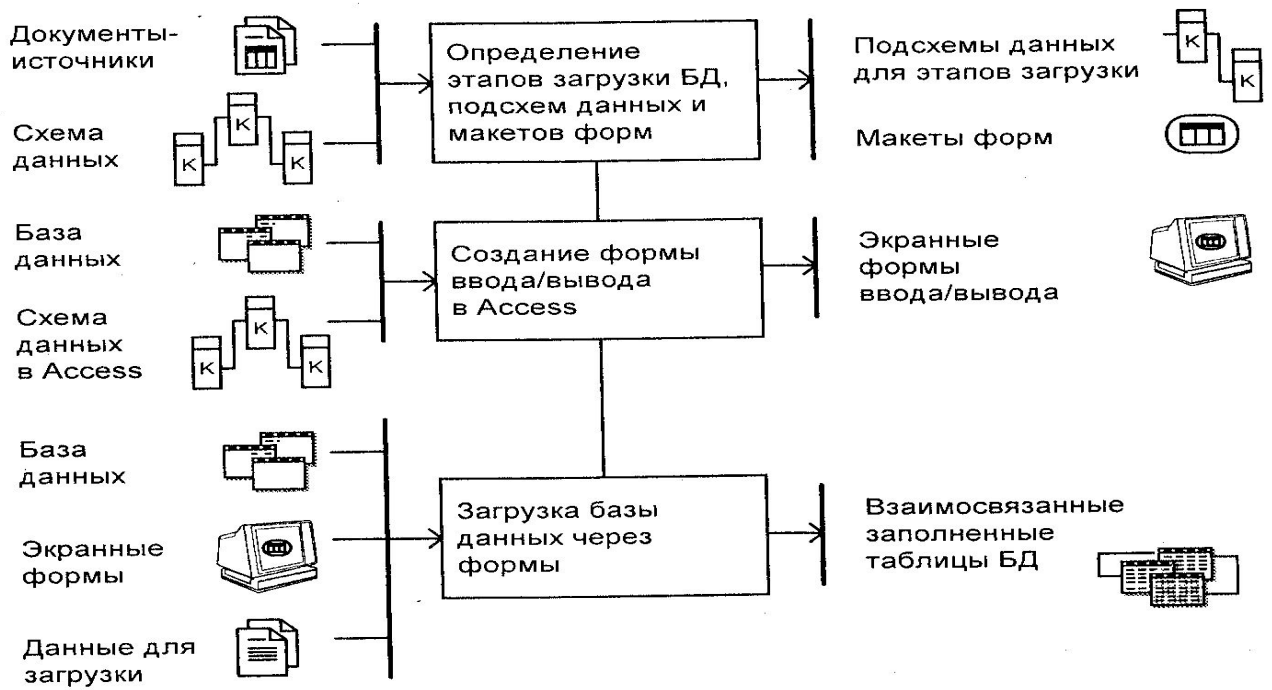
Формы в Access могут быть представлены в трех режимах:

-Режим формы предназначен для ввода, просмотра и корректировки данных таблиц, на которых основана форма.

-Режим макета обеспечивает просмотр данных почти в таком виде, в каком они отображаются в режиме формы, и в то же время позволяет изменять форму.

-Режим конструктора предназначен для разработки формы с помощью полного набора инструментов, обеспечивающего более детальную проработку структуры формы, использование всех элементов управления.

-Этапы создания интерфейса в Access



Отчеты

Новые средства Access 2007 позволяют создать профессионально оформленные **отчеты** не только с **помощью мастера или конструктора, но и в режиме макета.** При этом простыми средствами перетаскивания в отчет нужных полей из таблиц базы данных строится запрос — источник записей.
Чтобы правильно создавать отчеты, необходимо понимать назначение каждого его раздела.

Назначение каждого из разделов:

-Заголовок отчета обычно включает эмблему компании, название отчета, дату. Заголовок отображается перед верхним колонтитулом только один раз в начале отчета;
-Верхний колонтитул отображается вверху каждой страницы и используется в случае, когда нужно, чтобы название отчета и другая общая информация повторялись на каждой странице;
-Заголовок группы (Report Headler) используется при группировке записей отчета для вывода названия группы и однократного отображения полей, по которым производится группировка;

Назначение каждого из разделов:

-Область данных (Detail) отображает записи из источника данных, составляющие основное содержание отчета;

-Примечание группы (Footer) используется для отображения итогов и другой сводной информации по группе в конце каждой группы записей.;

-Нижний колонтитул применяется для нумерации страниц и отображения другой информации внизу каждой страницы;

-Примечание отчета служит для отображения итогов и другой сводной информации по всему отчету один раз в конце отчета.

В Access существуют **два представления, в которых можно вносить изменения в отчет: режим макета и режим конструктора.**

Режим макета является наиболее удобным для внесения изменений в отчет, поскольку пользователь **сразу видит данные отчета.** В этом режиме предусмотрено большинство инструментов, необходимых для его настройки.

В режиме конструктора отображаются разделы отчета и предусмотрены дополнительные инструменты и возможности разработки.

Просматривать отчет можно в режимах **Представление отчета** (Report View), **Предварительный просмотр** (Print Preview) или **Макет** (Layout Preview).

В режиме **Представление отчета** можно отфильтровать данные для отображения только заданных строк**. Режим предварительного просмотра** предназначен для просмотра отчета перед печатью. **Режим макета** позволяет, просматривая данные отчета, изменять его макет.

Макросы

Макрос – программа, состоящая из последовательности макрокоманд (макрос от слова "макрокоманда").
Макрокоманда — это инструкция, ориентированная на выполнение определенного действия над объектами Access и их элементами.
Макрокоманда **ВыполнитьКоманду** (RunCommand) позволяет выполнить любую встроенную команду Access, которые выводятся на вкладках ленты или в контекстном меню. **Имеющийся в Access набор макрокоманд** (более 50) реализует практически любые действия, которые необходимы для решения задачи.

Макрос может быть наряду с другими объектами представлен как **отдельный объект** (изолированный макрос), который отображается в области переходов в группе **Макросы** (Macros). **В Access 2007 макрос может быть внедрен в любые события** в форме, отчете или элементе управления (внедренный макрос). **При этом он не отображается как объект в группе Макросы (Macros), а становится компонентом формы, отчета.**

Макросы могут запускаться на выполнение прямо из области переходов..

Макросы

Внедренные макросы выполняются **в ответ на многочисленные виды события** возникающих в формах, отчетах и их элементах управления. Они наступают при выполнении определенных действий с объектами, к которым относятся прежде всего, действия пользователя. **Связь макросов с событиями позволяет автоматизировать приложения**, используя макросы для открытия форм, печати отчетов, выполнения последовательности запросов и многого другого. Сохранение внедренных макросов вместе с формами и отчетами упрощает управление объектами приложения.

Программы на языке макросов реализуют алгоритмы решения отдельных задач приложения.

Механизм связывания макросов с событиями в объектах позволяет объединить разрозненные задачи приложения в единый комплекс, управляемый пользователем.

Пользователь, выполняя различные действия в формах, инициирует выполнение макросов, автоматизирующих решение связанных с действиями пользователя задач.

Конструирование макроса

Создание макросов осуществляется в диалоговом режиме и сводится к **записи в окне макроса последовательности макрокоманд**, для которых задаются аргументы.

Каждому макросу присваивается имя. При выполнении макроса **макрокоманды выполняются последовательно** в порядке их расположения.

Имеется возможность **изменить порядок выполнения макрокоманд**, определяя условия их выполнения.

Выполнение макросов инициируется простой операцией и может сводиться к его открытию, как это делается и для других объектов базы данных. Помимо этого, Access предоставляет возможность автоматически инициировать выполнение макроса при наступлении некоторого события.