Содержание

Введение...........................................................................................................3

Актуальность....................................................................................................4

1. База данных..................................................................................................5

* 1. Реляционная модель данных......................................................................5
  2. Общая характеристика...............................................................................6
  3. Целостность сущности и ссылок................................................................7

2. Создание базы данных ...............................................................................8

* 1. Определение ключевых полей ..................................................................9

2.2 Связывание таблиц на схеме данных.......................................................9

1. Создание вычисляемых полей в запросах..............................................21
2. Автоматическое создание формы на основе таблицы или запроса.............................................................................................................24
3. Что такое SQL............................................................................................26
   1. Основы SQL Server....................................................................................32
   2. Реляционное ядро БД SQL Server............................................................33
   3. Редакции SQL Server. ..............................................................................33
4. Введение в реляционную базу данных....................................................35
   1. Пользовательские базы данных...............................................................36
   2. Создание пользовательской базы данных ...............................................37
5. Проектирование базы данных на SQL.....................................................38

Заключение.......................................................................................................41

Список используемой литературы.................................................................42

Введение

На сегодняшний день применение баз данных приобрело весьма важное значение для многих организаций, которые для упрощения своей работы применяют компьютерные технологии.

Базы данных стали основой информационных систем и в корне изменили методы работы многих организаций. В частности, в последние годы развитие технологии баз данных привело к созданию весьма мощных и удобных в эксплуатации систем. Благодаря этому системы баз данных стали доступными широкому кругу пользователей.

Большинство, если не все бизнес-приложения предназначены для обработки бизнес-данных. Самые первые бизнес-решения так и назывались — программы обработки данных. Эффективное хранение, обработка и взаимодействие с данными - только важная составляющая управления предприятием, что компании инвестируют значительные средства в разработку компьютеризированных системы для эффективного решения этих задач. Один из способов повышения эффективности обработки данных — организовать их эффективное хранение и получение. Самый распространенный подход к хранению данных на сегодня — использовать реляционную базу данных

В любом случае БД — это просто средство хранения данных;

БД – это, прежде всего, хранилище объектов данных, т.е. набор возможных понятий или событий, описываемых базой данных, с возможностью поиска этих объектов по признакам. Базой данных можно считать не только таблицы, индексирующие файлы со знаниями разных форматов, но и сами эти файлы, потому, что они являются не типизированными хранилищами знаний в такой базе данных. БД могут применяться как вспомогательное средство, позволяющее реализовать какую-то полезную функцию. Например, хранение настроек программы, Internet - адресов для рассылки рекламы и т.д.

Сервер приложений – это мостик между программами – клиентами и одним или несколькими серверами баз данных.

SQL Server 2000 представляет собой мощный полнофункциональный сервер баз данных, отличающийся высокой производительностью быстротой освоения и удобным интерфейсом администрирования. Под его управлением могут работать базы данных в широком диапазоне от уровня среднего звена предприятия до распределённых баз масштаба корпорации.

Актуальность

В настоящее время, несмотря на повышение компьютеризации общества, в сфере бизнеса и торговли до сих пор нет средств, позволяющих в достаточной мере автоматизировать процесс ведения документации и отчетности.

Одной из основных задач можно рассматривать проблему ведения отчетности, а так же оперативную корректировку данных при возникновении необходимости в этом.

О своевременности и актуальности рассматриваемой проблемы говорит тот факт, что большую часть своего времени администрация магазина тратит на оформление различной документации и отчетов. Огромное количество магазинов и отсутствие предложений в данной сфере гарантирует высокую потребность в данном продукте.

Объектом для создания базы данных являлся книжный магазин. Данная база данных предлагает введение отчетности, хранения данных, ввод и корректировку данных.

Базу данных могут использовать не только администрация , но также и рабочий персонал магазина.

Так как существуют большие книжные магазины им необходимо иметь свою базу данных, чтобы контролировать информационный поток данных. Целью разработки проектирования базы данных является создание базы данных научной конференции , которая может хранить данные и предоставлять пользователю удобную работу с данными.

#### 1. База данных

База данных - набор всех данных, определенных <схемами> в среде. Понятие среды является определяемым реализацией.

Базы данных — это совокупность сведений (о реальных объектах, процессах, событиях или явлениях), относящихся к определенной теме или задаче, организованная таким образом, чтобы обеспечить удобное представление этой совокупности, как в целой, так и любой ее части. Реляционная база данных представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного типа. Каждая строка таблицы включает данные об одном объекте, а столбцы таблицы содержат различные характеристики этих объектов — атрибуты. Строки таблицы называются записями; все записи имеют одинаковую структуру — они состоят из полей, в которых хранятся атрибуты объекта. Каждое поле записи содержит одну характеристику объекта и имеет строго определенный тип данных (например, текстовая строка, число, дата). Все записи имеют одни и те же поля, только в них содержатся разные значения атрибутов.

Для работы с данными используются системы управления базами данных (СУБД). Основные функции СУБД — это определение данных (описание структуры баз данных), обработка данных и управление данными.

### 1.1 Реляционная модель данных

### Реляционная база данных – это совокупность отношений, содержащих всю информацию, которая должна храниться в БД. Однако пользователи могут воспринимать такую базу данных как совокупность таблиц.

Хотя понятие модели данных является общим, и можно говорить об иерархической, сетевой, семантической и других моделях данных, нужно отметить, что в области баз данных это понятие было введено Эдгаром Коддом применительно к реляционным системам и наиболее эффективно используется именно в данном контексте. Попытки прямолинейного применения аналогичных моделей к дореляционным организациям показывают, что реляционная модель слишком «велика», а для постреляционных организаций она оказывается «мала».

#### 1.2 Общая характеристика

Хотя понятие реляционной модели данных первым ввел основоположник реляционного подхода Эдгар Кодд, наиболее распространенная трактовка реляционной модели данных, по-видимому, принадлежит известному популяризатору идей Кодда Кристоферу Дейту, который воспроизводит ее (с различными уточнениями) практически во всех своих книгах (см., например, К. Дейт. Введение в системы баз данных. 6-е изд., М.; СПб.: Вильямс.– 2000). Согласно трактовке Дейта, реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода: структурной части, манипуляционной части и целостной части.

В структурной части модели фиксируется, что единственной родовой структурой1) данных, используемой в реляционных БД, является нормализованное n-арное отношение. Определяются понятия доменов, атрибутов, кортежей, заголовка, тела и переменной отношения. По сути дела, в двух предыдущих разделах этой лекции мы рассматривали именно понятия и свойства структурной составляющей реляционной модели.

В манипуляционной части модели определяются два фундаментальных механизма манипулирования реляционными БД – реляционная алгебра и реляционное исчисление. Первый механизм базируется в основном на классической теории множеств (с некоторыми уточнениями и добавлениями), а второй – на классическом логическом аппарате исчисления предикатов первого порядка. Мы рассмотрим эти механизмы более подробно в следующих лекциях, а пока лишь заметим, что основной функцией манипуляционной части реляционной модели является обеспечение меры реляционности любого конкретного языка реляционных БД: язык называется реляционным, если он обладает не меньшей выразительностью и мощностью, чем реляционная алгебра или реляционное исчисление.

#### 1.3 Целостность сущности и ссылок

Наконец, в целостной части реляционной модели данных фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД. Первое требование называется требованием целостности сущности (entity integrity). Объекту или сущности реального мира в реляционных БД соответствуют кортежи отношений. Конкретно требование состоит в том, что любой кортеж любого значения-отношения любой переменной отношения должен быть отличим от любого другого кортежа этого значения отношения по составным значениям заранее определенного множества атрибутов переменной отношения, т. е., другими словами, любая переменная отношения должно обладать первичным ключом. Как мы видели в предыдущем разделе, это требование автоматически удовлетворяется, если в системе не нарушаются базовые свойства отношений.

На самом деле, требование целостности сущности полностью звучит следующим образом: у любой переменной отношения должен существовать первичный ключ, и никакое значение первичного ключа в кортежах значения-отношения переменной отношения не должно содержать неопределенных значений. Чтобы эта формулировка была полностью понятна, мы должны хотя бы кратко обсудить понятие неопределенного значения (NULL).

2. Создание базы данных

В Microsoft Access поддерживаются два способа создания базы данных. Имеется возможность создать пустую базу данных, а затем добавить в нее таблицы, формы, отчеты и другие объекты. Такой способ является наиболее гибким, но требует отдельного определения каждого элемента базы данных. Имеется также возможность сразу создать с помощью мастера базу данных определенного типа со всеми необходимыми таблицами, формами и отчетами. Это простейший способ начального создания базы данных. В любом случае на экране появится окно с содержимым таблицы

Открыть таблицу в режиме Конструктора можно одним из двух следующих способов:

* выделить таблицу в списке таблиц в окне базы данных и нажать кнопку Конструктор в верхней части окна базы данных;
* щелкнуть правой кнопкой мыши на имени таблицы и из контекстного меню выбрать команду Конструктор.

После выполнения указанных действий на экране появится окно таблицы в режиме Конструктора, а также панель инструментов Конструктор таблиц Окно таблицы в режиме Конструктора состоит из двух панелей.

* Верхняя панель содержит таблицу из трех столбцов: Имя поля, Тип данных и Описание. В столбце Имя поля содержатся имена полей таблицы, в столбце Тип данных указывается тип данных для каждого поля. Заполнение двух первых столбцов является обязательным.
* На нижней панели располагаются две вкладки свойств полей: Общие и Подстановка, а также область контекстной справки.

Ключевое поле таблицы помечается специальным значком — ключик в поле выделения в левой части окна. Чтобы выделить поле, достаточно просто щелкнуть мышью по строке, в которой описывается данное поле. При этом в нижней части окна будут показаны параметры именно для этого поля. Перемещаться между столбцами в верхней панели окна можно с помощью клавиши <Таb>, а для перемещения по списку полей вверх и вниз пользуйтесь клавишами со стрелками. С помощью клавиши <F6> можно перемещаться между панелями.

2.1 Определение ключевых полей

Ключевое поле — это одно или несколько полей, комбинация значений которых однозначно определяет каждую запись в таблице. Если для таблицы определены ключевые поля, то Microsoft Access предотвращает дублирование или ввод пустых значений в ключевое поле. Ключевые поля используются для быстрого поиска и связи данных из разных таблиц при помощи запросов, форм и отчетов.

В Microsoft Access можно выделить три типа ключевых полей: счетчик, простой ключ и составной ключ. Рассмотрим каждый из этих типов.

Для создания ключевого поля типа Счетчик необходимо в режиме Конструктора таблиц:

1. Включить в таблицу поле счетчика.
2. Задать для него автоматическое увеличение на 1.
3. Указать это поле в качестве ключевого путем нажатия на кнопку Ключевое поле на панели инструментов Конструктор таблиц .
4. Если до сохранения созданной таблицы ключевые поля не были определены, то при сохранении будет выдано сообщение о создании ключевого поля. При нажатии кнопки. Да будет создано ключевое поле счетчика с именем Код (ID) и типом данных Счетчик

2.2 Связывание таблиц на схеме данных

После создания различных таблиц Books,Janre, Oblojka, Izdatelstvo, содержащих данные, относящиеся к различным аспектам базы данных, следующим шагом будет объединение этих данных при их извлечении из базы данных. Вторым шагом при этом является определение связей между таблицами. После этого становится возможным создание запросов, форм и отчетов, в которых выводятся данные из нескольких таблиц сразу.

Как работает связь между таблицами?

Связь между таблицами устанавливает отношения между совпадающими значениями в ключевых полях, обычно между полями разных таблиц, имеющими одинаковые имена. В большинстве случаев с ключевым полем одной таблицы, являющимся уникальным идентификатором каждой записи, связывается внешний ключ другой таблицы. Например, для сопоставления сведений о сотрудниках и о принятых ими заказах, следует определить связь по полям «КодКниги» в двух таблицах.

Отношение «один-ко-многим»

В нашей базе данных использовался тип связи «один-ко-многим»

Связь с отношением «один-ко-многим» является наиболее часто используемым типом связи между таблицами. В такой связи каждой записи в таблице A могут соответствовать несколько записей в таблице B, а запись в таблице B не может иметь более одной соответствующей ей записи в таблице A.

Определение связей между таблицами

Для того чтобы определить связь между таблицами, следует добавить таблицы в окно Схема данных и перенести с помощью мыши ключевое поле одной таблицы в другую таблицу. Для того чтобы было удобно просматривать, создавать, удалять и модифицировать связи между таблицами, в Microsoft Access используется схема данных.

Описание ниже приведенной таблицы.

В эту таблицу занесены все данные по книжному магазину: Автор, название книг, жанр, издательство, кол-во стр., дата издания, обложка, цена, описание, поставщик, дата поставки.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Id\_tovar | autor | nazvanie | janr | izdatelstvo | k\_vo\_str | data\_izd | oblogka | cena | postavshic | data\_post | prnjal |
| 1 | Э.Радзинский | Нерон | Автобиография | «Энергия»Москва | 350 | 03.09.2001 | Мягкая | 150,00р. | Ростов | 25.10.2004 | Морозова |
| 2 | Мэри Спенсер | «Поверь в любовь» | Роман | «АСТ» Москва | 349 | 01.12.2000 | Твердая | 90,00р. | Москва | 28.04.2001 | Морозова |
| 3 | Фиби Конн | «Ураган страсти» | Роман | «АСТ» Москва | 429 | 04.07.2001 | Твердая | 126,55р. | Оренбург | 04.08.2002 | Морозова |
| 4 | Н.Корнилова | Воровки | Детектив | «Экспресс»Москва | 508 | 01.06.2002 | Мягкая | 120,00р. | Ростов | 05.04.2004 | Морозова |
| 5 | Джейн Фрейзер | Возлюбленный враг | Детектив | «АСТ» Москва | 461 | 20.12.2002 | Мягкая | 366,00р. | Ростов | 07.11.2003 | Кравцов |
| 6 | Дарья Истомина | «Торговка» | Роман | «Олимп» Москва | 300 | 01.01.2001 | Твердая | 196,00р. | Воркута | 23.06.2002 | Морозова |
| 7 | Екатерина Маркова | «Блудница» | Роман | «Астрель» Москва | 348 | 20.02.2001 | Мягкая | 88,00р. | Львов | 30.01.2004 | Лебедев |
| 8 | Аманда Квик | «Искушение» | Роман | «Квест» Москва | 316 | 29.09.2003 | Мягкая | 129,00р. | Калининград | 23.09.2003 | Кузьменко |
| 9 | Джеки Алессандро | «Свадебный водоворот» | Роман | «Аист» Москва | 317 | 14.03.2002 | Мягкая | 56,00р. | Москва | 27.03.2003 | Красников |
| 10 | Джудит Ливори | «Право первой ночи» | Роман | «Квест» Москва | 316 | 02.05.2001 | Твердая | 156,00р. | Москва | 24.05.2002 | Тевиков |
| 11 | Лора Эштон | «Тайна прекрасной Марии « | Роман | «Диалог» Москва | 421 | 09.08.2002 | Твердая | 55,00р. | Ростов | 03.04.2003 | Тевиков |
| 12 | \_лизабет Филлипс | «Блестящая девочка» | Роман | «АСТ» Москва | 444 | 01.02.2000 | Твердая | 63,00р. | Орёл | 02.02.2002 | Лебедев |
| 13 | М.Серёгин | Разговорчики в строю | Детектив | «ЭКСМО»Москва | 347 | 03.09.2001 | Мягкая | 57,00р. | Москва | 19.01.2003 | Кравцов |
| 14 | Н.Кровцова | Группа особого назначения | Детектив | «Нева»С.-Петербург | 349 | 12.05.2004 | Твердая | 88,00р. | Москва | 01.02.2004 | Сривцунов |
| 15 | С.Маршак | Машина для пахана | Детектив | «Олма»Москва | 382 | 30.03.2001 | Мягкая | 96,00р. | Донецк | 23.03.2003 | Соколенко |
| 16 | А.Каменская | Антикиллер 5 | Детектив | «АСТ»Москва | 289 | 12.06.2003 | Твердая | 153,00р. | Каменск | 05.09.2004 | Макаренко |
| 17 | С.Луганова | Сталин | Автобиография | «Арго»Орёл | 293 | 04.05.2003 | Твердая | 171,00р. | Москва | 30.12.2004 | Морозова |

Общая таблица

Далее делаем нормализацию таблицы (вторая нормальная форма). Находим те поля, которые будут повторяться, и выносим их в отдельные таблицы:

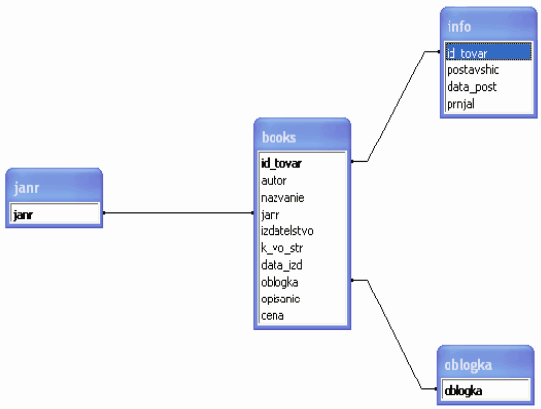
|  |  |  |  |
| --- | --- | --- | --- |
| id\_tovar | postavshic | data\_post | prnjal |
| 1 | Ростов | 25.10.2004 | Морозова |
| 2 | Москва | 28.04.2001 | Морозова |
| 3 | Оренбург | 04.08.2002 | Морозова |
| 4 | Ростов | 05.04.2004 | Морозова |
| 5 | Ростов | 07.11.2003 | Кравцов |
| 6 | Воркута | 23.06.2002 | Морозова |
| 7 | Львов | 30.01.2004 | Лебедев |
| 8 | Калининград | 23.09.2003 | Кузьменко |
| 9 | Москва | 27.03.2003 | Красников |
| 10 | Москва | 24.05.2002 | Тевиков |
| 11 | Ростов | 03.04.2003 | Тевиков |
| 12 | Орёл | 02.02.2002 | Лебедев |
| 13 | Москва | 19.01.2003 | Кравцов |
| 14 | Москва | 01.02.2004 | Сривцунов |
| 15 | Донецк | 23.03.2003 | Соколенко |
| 16 | Каменск | 05.09.2004 | Макаренко |
| 17 | Москва | 30.12.2004 | Морозова |

|  |
| --- |
| janr |
| Автобиография |
| Детектив |
| Детская |
| Журналы |
| Исторический |
| Роман |
| Словари |
| Справочники |
| Учебный |
| Фантастика |
| Энциклопедии |

| oblogka |
| --- |
| Мягкая |
| Твердая |

Создаем связь между таблицами:

Внутреннее соединение двух таблиц по одному полю в реляционной базе данных строится на основе отношения «один-ко-многим». Примером может служить демонстрационная база данных “books” ,в которой все соединения являются внутренними соединениями по одному полю на основе указанного отношения. Каждая таблица должна иметь первичный ключ с уникальными значениями. Отсутствие повторений значений поля или полей первичного ключа в таблице Access устанавливает автоматически.



И получаем вторую нормальную форму:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id\_tovar | autor | nazvanie | janr | izdatelstvo | k\_vo\_str | data\_izd | oblogka | opisanie | Cena |
| 1 | Э.Радзинский | Нерон | Автобиография | «Энергия»Москва | 350 | 03.09.2001 | Мягкая |  | 150,00р. |
| 2 | Мэри Спенсер | «Поверь в любовь» | Роман | «АСТ» Москва | 349 | 01.12.2000 | Твердая |  | 90,00р. |
| 3 | Фиби Конн | «Ураган страсти» | Роман | «АСТ» Москва | 429 | 04.07.2001 | Твердая |  | 126,55р. |
| 4 | Н.Корнилова | Воровки | Детектив | «Экспресс»Москва | 508 | 01.06.2002 | Мягкая |  | 120,00р. |
| 5 | Джейн Фрейзер | Возлюбленный враг | Детектив | «АСТ» Москва | 461 | 20.12.2002 | Мягкая |  | 366,00р. |
| 6 | Дарья Истомина | «Торговка» | Роман | «Олимп» Москва | 300 | 01.01.2001 | Твердая |  | 196,00р. |
| 7 | Екатерина Маркова | «Блудница» | Роман | «Астрель» Москва | 348 | 20.02.2001 | Мягкая |  | 88,00р. |
| 8 | Аманда Квик | «Искушение» | Роман | «Квест» Москва | 316 | 29.09.2003 | Мягкая |  | 129,00р. |
| 9 | Джеки Алессандро | «Свадебный водоворот» | Роман | «Аист» Москва | 317 | 14.03.2002 | Мягкая |  | 56,00р. |
| 10 | Джудит Ливори | «Право первой ночи» | Роман | «Квест» Москва | 316 | 02.05.2001 | Твердая |  | 156,00р. |
| 11 | Лора Эштон | Тайна прекрасной Марии | Роман | «Диалог» Москва | 421 | 09.08.2002 | Твердая |  | 55,00р. |
| 12 | Э\_лизабет Филлипс | «Блестящая девочка» | Роман | «АСТ» Москва | 444 | 01.02.2000 | Твердая |  | 63,00р. |
| 13 | М.Серёгин | Разговорчики в строю | Детектив | «ЭКСМО»Москва | 347 | 03.09.2001 | Мягкая |  | 57,00р. |
| 14 | Н.Кровцова | Группа особого назначения | Детектив | «Нева»С.-Петербург | 349 | 12.05.2004 | Твердая |  | 88,00р. |
| 15 | С.Маршак | Машина для пахана | Детектив | «Олма»Москва | 382 | 30.03.2001 | Мягкая |  | 96,00р. |
| 16 | А.Каменская | Антикиллер 5 | Детектив | «АСТ»Москва | 289 | 12.06.2003 | Твердая |  | 153,00р. |
| 17 | С.Луганова | Сталин | Автобиография | «Арго»Орёл | 293 | 04.05.2003 | Твердая |  | 171,00р. |

#### Далее мы заносим эти данные в Microsoft Access 2003.

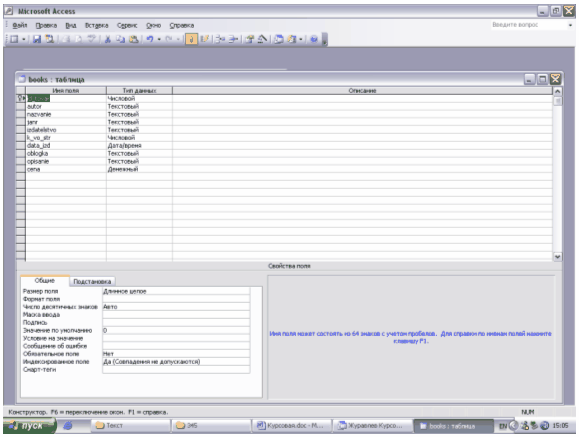
#### Главная таблица



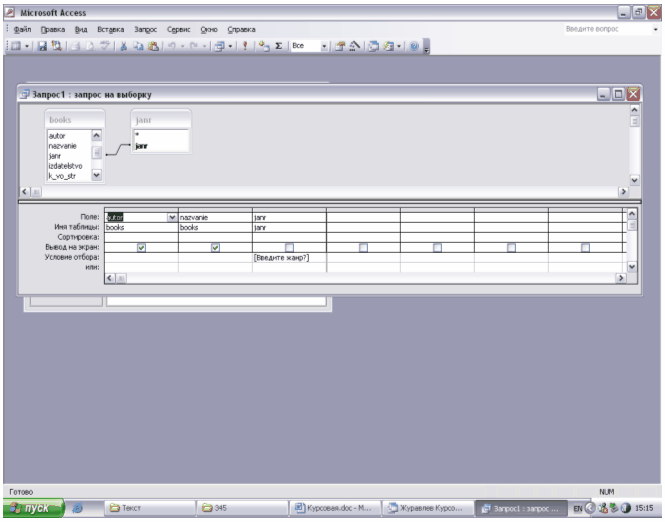
Мы использовали в своей работе следующие типы данных:

* Текстовый— символьные или числовые данные, не требующие вычислений. Поле данного типа может содержать до 255 символов. Размер текстового поля задается с помощью свойства Размер поля, в котором указывается максимальное количество символов, которые могут быть введены в данное поле.
* Поле MEMO— поле MEMO предназначено для ввода текстовой информации, по объему превышающей 255 символов. Такое поле может содержать до 65 535 символов. Этот тип данных отличается от типа Текстовый тем, что в таблице хранятся не сами данные, а ссылки на блоки данных, хранящиеся отдельно.
* Числовой— числовой тип применяется для хранения числовых данных, используемых в математических расчетах.
* Дата/Время— тип для представления даты и времени. Позволяет вводить даты с 100 по 9999 год.
* Денежный— тип данных, предназначенный для хранения данных, точность представления которых колеблется от 1 до 4 десятичных знаков.

#### Счетчик— поле содержит 4-байтный уникальный номер, определяемый Microsoft Access автоматически для каждой новой записи либо случайным образом, либо путем увеличения предыдущего значения на 1.



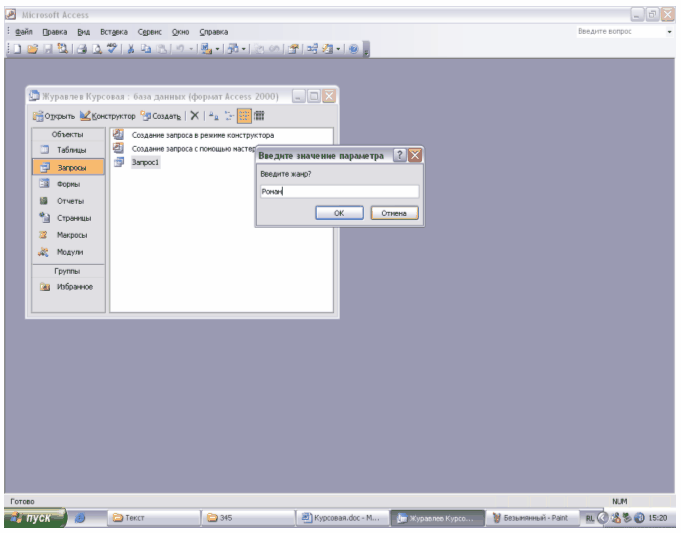
3. Создание вычисляемых полей в запросах



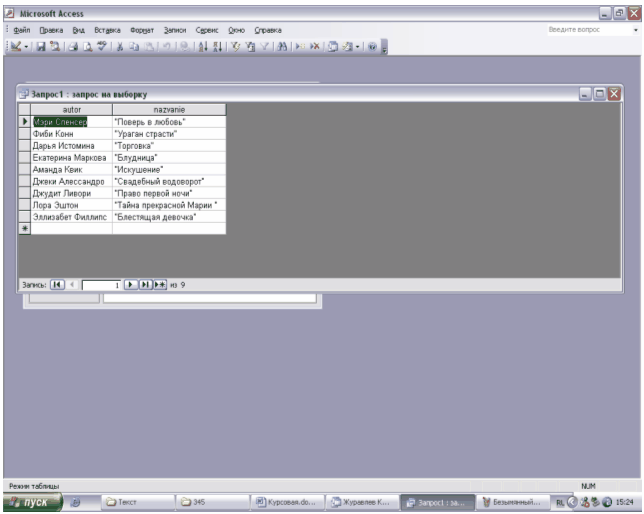
Создание запросов интересно не только тем, что вы можете в виде одной таблицы представить данные из нескольких связанных таблиц и отобрать нужные записи из этих таблиц. Вы можете создавать столбцы в запросе, которые являются результатом вычислений над значениями других столбцов. Такие столбцы называются вычисляемыми. Это существенно расширяет возможности запросов. Чтобы создать вычисляемое поле, нужно ввести выражение, которое вычисляет требуемое значение, в строку Поле свободного столбца бланка запроса. В данном примере это выражение представляет собой конкатенацию полей, содержащих имя и фамилию сотрудника, с пробелом между ними.

В этом выражении мы используем ссылки на поля таблицы, которые в выражении заключаются в квадратные скобки.

Запрос в режиме Конструктора



Запрос по выбору жанра.



После того как выбрали жанр у нас появится таблица с данными, поэтому жанра.

4. Автоматическое создание формы на основе таблицы или запроса

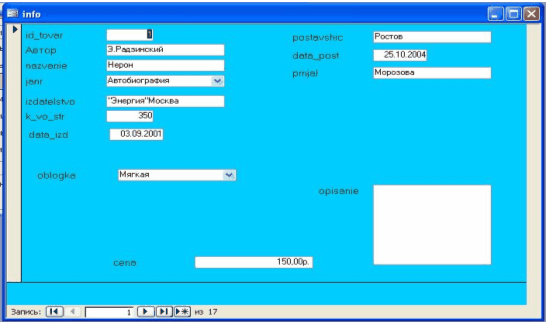
Access 2003 предлагает несколько способов создания форм. Самым простым из них является использование средств автоматического создания форм на основе таблицы или запроса. Автоматически создаваемые формы бывают нескольких видов, каждый из которых отличается способом отображения данных.

* Автоформа, организованная "в столбец" .В такой форме поля каждой записи отображаются в виде набора элементов управления, расположенных в один или несколько столбцов. Это компактное и, пожалуй, самое удачное представление для быстрого создания формы.

Автоматически созданная форма включает все поля выбранного источника данных. Чтобы создать форму с помощью средства автоматического создания форм:

1. Щелкните по ярлыку Формы в окне База данных и нажмите кнопку Создать. Появится диалоговое окно Новая форма, представленное на рис ….
2. В списке диалогового окна Новая форма выделите один из вариантов автоформы, например: Автоформа: в столбец
3. В поле со списком, находящимся в нижней части диалогового окна Новая форма, содержатся имена всех таблиц и запросов базы данных, которые могут быть использованы в качестве источника данных для формы. Щелкните левой кнопкой мыши по кнопке со стрелкой, чтобы раскрыть список, и выберите в нем нужный элемент.
4. Нажмите кнопку ОК.

В результате будет автоматически создана и открыта форма выбранного вида. Чтобы созданную форму можно было использовать в дальнейшем, ее необходимо сохранить.



Для сохранения формы выберите команду Файл, Сохранить. В поле Имя формы появившегося диалогового окна Сохранение введите нужное название и нажмите кнопку ОК.

После разработки базы данных средствами Microsoft Access мы решили разработать эту базу данных при помощи SQL. Создание запросов является его главным направлением, а также при помощи SQL можно пересылать данные по локальной и другим видам сетей при помощи сервера.

5. Что такое SQL?

Все языки манипулирования данными (ЯМД), созданные до появления реляционных баз данных и разработанные для многих систем управления базами данных (СУБД) персональных компьютеров, были ориентированы на операции с данными, представленными в виде логических записей файлов. Это требовало от пользователей детального знания организации хранения данных и достаточных усилий для указания не только того, какие данные нужны, но и того, где они размещены и как шаг за шагом получить их.

Рассматриваемый же ниже непроцедурный язык SQL (Structured Query Language - структуризованный язык запросов) ориентирован на операции с данными, представленными в виде логически взаимосвязанных совокупностей таблиц. Особенность предложений этого языка состоит в том, что они ориентированы в большей степени на конечный результат обработки данных, чем на процедуру этой обработки. SQL сам определяет, где находятся данные, какие индексы и даже наиболее эффективные последовательности операций следует использовать для их получения: не надо указывать эти детали в запросе к базе данных.

Для иллюстрации различий между ЯМД рассмотрим следующую ситуацию. Пусть, например, вы собираетесь посмотреть кинофильм и хотите воспользоваться для поездки в кинотеатр услугами такси. Одному шоферу такси достаточно сказать название фильма - и он сам найдет вам кинотеатр, в котором показывают нужный фильм. (Подобным же образом, самостоятельно, отыскивает запрошенные данные SQL.)

Для другого шофера такси вам, возможно, потребуется самому узнать, где демонстрируется нужный фильм и назвать кинотеатр. Тогда водитель должен найти адрес этого кинотеатра. Может случиться и так, что вам придется самому узнать адрес кинотеатра и предложить водителю проехать к нему по таким-то и таким-то улицам. В самом худшем случае вам, может быть, даже придется по дороге давать указания: "Повернуть налево... проехать пять кварталов... повернуть направо...". (Аналогично больший или меньший уровень детализации запроса приходится создавать пользователю в разных СУБД, не имеющих языка SQL.)

Появление теории реляционных баз данных и предложенного Коддом языка запросов "alpha", основанного на реляционном исчислении [2, 3], инициировало разработку ряда языков запросов, которые можно отнести к двум классам:

1. Алгебраические языки, позволяющие выражать запросы средствами специализированных операторов, применяемых к отношениям (JOIN - соединить, INTERSECT - пересечь, SUBTRACT - вычесть и т.д.).
2. Языки исчисления предикатов представляют собой набор правил для записи выражения, определяющего новое отношение из заданной совокупности существующих отношений. Другими словами исчисление предикатов есть метод определения того отношения, которое нам желательно получить (как ответ на запроc) из отношений, уже имеющихся в базе данных.

Разработка, в основном, шла в отделениях фирмы IBM (языки ISBL, SQL, QBE) и университетах США (PIQUE, QUEL) [3]. Последний создавался для СУБД INGRES (Interactive Graphics and Retrieval System), которая была разработана в начале 70-х годов в Университете шт. Калифорния и сегодня входит в пятерку лучших профессиональных СУБД. Сегодня из всех этих языков полностью сохранились и развиваются QBE (Query-By-Example - запрос по образцу) и SQL, а из остальных взяты в расширение внутренних языков СУБД только наиболее интересные конструкции.

В начале 80-х годов SQL "победил" другие языки запросов и стал фактическим стандартом таких языков для профессиональных реляционных СУБД. В 1987 году он стал международным стандартом языка баз данных и начал внедряться во все распро-страненные СУБД персональных компьютеров. Почему же это произошло?

Непрерывный рост быстродействия, а также снижение энергопотребления, размеров и стоимости компьютеров привели к резкому расширению возможных рынков их сбыта, круга пользователей, разнообразия типов и цен. Естественно, что расширился спрос на разнообразное программное обеспечение.

Борясь за покупателя, фирмы, производящие программное обеспечение, стали выпускать на рынок все более и более интеллектуальные и, следовательно, объемные программные комплексы. Приобретая (желая приобрести) такие комплексы, многие организации и отдельные пользователи часто не могли разместить их на собственных ЭВМ, однако не хотели и отказываться от нового сервиса. Для обмена информацией и ее обобществления были созданы сети ЭВМ, где обобществляемые программы и данные стали размещать на специальных обслуживающих устройствах - файловых серверах.

СУБД, работающие с файловыми серверами, позволяют множеству пользователей разных ЭВМ (иногда расположенных достаточно далеко друг от друга) получать доступ к одним и тем же базам данных. При этом упрощается разработка различных автоматизированных систем управления организациями, учебных комплексов, информационных и других систем, где множество сотрудников (учащихся) должны использовать общие данные и обмениваться создаваемыми в процессе работы (обучения). Однако при такой идеологии вся обработка запросов из программ или с терминалов пользовательских ЭВМ выполняется на этих же ЭВМ. Поэтому для реализации даже простого запроса ЭВМ часто должна считывать из файлового сервера и (или) записывать на сервер целые файлы, что ведет к конфликтным ситуациям и перегрузке сети.

Для исключения указанных и некоторых других недостатков была предложена технология "Клиент-Сервер", по которой запросы пользовательских ЭВМ (Клиент) обрабатываются на специальных серверах баз данных (Сервер), а на ЭВМ возвращаются лишь результаты обработки запроса. При этом, естественно, нужен единый язык общения с Сервером и в качестве такого языка выбран SQL. Поэтому все современные версии профессиональных реляционных СУБД (DB2, Oracle, Ingres, Informix, Sybase, Progress, Rdb) и даже нереляционных СУБД (например, Adabas) используют технологию "Клиент-Сервер" и язык SQL. К тому же приходят разработчики СУБД персональных ЭВМ, многие из которых уже сегодня снабжены языком SQL.

Бытует мнение: Поскольку большая часть запросов формулируется на SQL, практически безразлично, что это за СУБД - был бы SQL.

Реализация в SQL концепции операций, ориентированных на табличное представление данных, позволило создать компактный язык с небольшим (менее 30) набором предложений. SQL может использоваться как интерактивный (для выполнения запросов) и как встроенный (для построения прикладных программ). В нем существуют:

* предложения определения данных (определение баз данных, а также определение и уничтожение таблиц и индексов);
* запросы на выбор данных (предложение SELECT);
* предложения модификации данных (добавление, удаление и изменение данных);
* предложения управления данными (предоставление и отмена привилегий на доступ к данным, управление транзакциями и другие). Кроме того, он предоставляет возможность выполнять в этих предложениях:
* арифметические вычисления (включая разнообразные функциональные преобразования), обработку текстовых строк и выполнение операций сравнения значений арифметических выражений и текстов;
* упорядочение строк и (или) столбцов при выводе содержимого таблиц на печать или экран дисплея;
* создание представлений (виртуальных таблиц), позволяющих пользователям иметь свой взгляд на данные без увеличения их объема в базе данных;
* запоминание выводимого по запросу содержимого таблицы, нескольких таблиц или представления в другой таблице (реляционная операция присваивания).
* агрегатирование данных: группирование данных и применение к этим группам таких операций, как среднее, сумма, максимум, минимум, число элементов и т.п.

В SQL используются следующие основные типы данных, форматы которых могут несколько различаться для разных СУБД:

INTEGER

- целое число (обычно до 10 значащих цифр и знак);

SMALLINT

- "короткое целое" (обычно до 5 значащих цифр и знак);

DECIMAL(p,q)

- десятичное число, имеющее p цифр (0 < p < 16) и знак; с помощью q задается число цифр справа от десятичной точки (q < p, если q = 0, оно может быть опущено);

FLOAT

- вещественное число с 15 значащими цифрами и целочисленным порядком, определяемым типом СУБД;

CHAR(n)

- символьная строка фиксированной длины из n символов (0 < n < 256);

VARCHAR(n)

- символьная строка переменной длины, не превышающей n символов (n > 0 и разное в разных СУБД, но не меньше 4096);

DATE

- дата в формате, определяемом специальной командой (по умолчанию mm/dd/yy); поля даты могут содержать только реальные даты, начинающиеся за несколько тысячелетий до н.э. и ограниченные пятым-десятым тысячелетием н.э.;

TIME

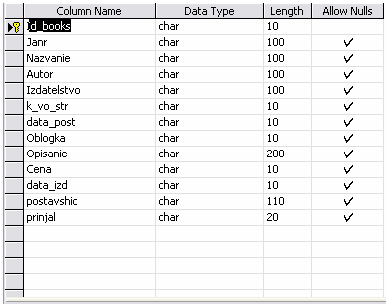
- время в формате, определяемом специальной командой, (по умолчанию hh.mm.ss);

DATETIME

- комбинация даты и времени;

MONEY

- деньги в формате, определяющем символ денежной единицы ($, руб, ...) и его расположение (суффикс или префикс), точность дробной части и условие для показа денежного значения.



В некоторых СУБД еще существует тип данных LOGICAL, DOUBLE и ряд других. СУБД INGRES предоставляет пользователю возможность самостоятельного определения новых типов данных, например, плоскостные или пространственные координаты, единицы различных метрик, пяти- или шестидневные недели (рабочая неделя, где сразу после пятницы или субботы следует понедельник), дроби, графика, большие целые числа (что стало очень актуальным для российских банков) и т.п.

Ориентированный на работу с таблицами SQL не имеет достаточных средств для создания сложных прикладных программ. Поэтому в разных СУБД он либо используется вместе с языками программирования высокого уровня (например, такими как Си или Паскаль), либо включен в состав команд специально разработанного языка СУБД (язык систем dBASE, R:BASE и т.п.). Унификация полных языков современных профессиональных СУБД достигается за счет внедрения объектно-ориентированного языка четвертого поколения 4GL. Последний позволяет организовывать циклы, условные предложения, меню, экранные формы, сложные запросы к базам данных с интерфейсом, ориентированным как на алфавитно-цифровые терминалы, так и на оконный графический интерфейс (X-Windows, MS-Windows).

5.1 Основы SQL Server

SQL Server — семейство продуктов, разработанных для хранения данных в больших системах, осуществляющих обработку информации, и обслуживания коммерческих Web-узлов. SQL Server прост и удобен в использовании, он широко применяется как в сложных системах, с которыми работают сотни пользователей, так и в

малом бизнесе. Он популярен также у отдельных пользователей, которым нужен на-

дежный и удобный сервер БД. В состав SQL Server входят две основные службы,

предназначенные для новой платформы Microsoft .NET и систем с традиционной

двухуровневой клиент-серверной архитектурой

Традиционная клиент-серверная архитектура БД состоит из двух компонентов: клиентского приложения и системы управления реляционными базами данных которая обеспечивает хранение и управление

данными.

5.2 Реляционное ядро БД SQL Server

Реляционное ядро БД SQL Server — это реляционная СУБД, хранящая и осуществляющая управление данными в реляционных таблицах. Каждая таблица представляет отдельный объект, например клиентов, сотрудников или товары, которыми торгует фирма. Столбцы таблиц представляют' атрибуты, а ряды — экземпляры соответствующих объектов. По запросу приложения реляционное ядро БД связывает таблицы друг с другом. Реляционное ядро БД хранит подробные записи о транзакциях, генерируемых системами оперативной обработки транзакций (OLTP) по запросу специализированных хранилищ данных. Реляционное ядро БД обеспечивает достоверность и защиту хранимых данных, отказоустойчивость, динамически оптимизирует производительность, а также налагает блокировки для реализации параллелизма.

5.3 Редакции SQL Server.

Существует шесть различных редакций SQL Server . Четыре из них предназначены для производственных систем, одна — для разработки приложений и еще одна —ознакомительная редакция. Кроме того, компонент SQL Server Desktop Engine можно распространять вместе с использующими его приложениями.

Компоненты SQL Server.

В SQL Server имеется несколько различных видов компонентов. Основу SQL Server составляют серверные компоненты, в большинстве своем реализованные в виде 32-разрядных служб Windows. В состав SQL Server входят также различные средства администрирования сервера — графические клиентские приложения и утилиты командной строки. Они, как и другие клиентские приложения, используют средства обмена данными между клиентом и сервером, предоставляемые SQL Server. Компоненты SQL Server, обеспечивающие обмен данными между клиентом и сервером, предоставляют клиентским приложениям множество способов доступа к данным сервера. Эти компоненты реализованы в виде поставщиков, драйверов, интерфейсов БД и сетевых библиотек Net-Libraries. Помимо них, в число компонентов SQL Server входит интерактивная справочная система SQL Server Books Online в HTML-формате.

## 6. Введение в реляционную базу данных

SQL (ОБЫЧНО ПРОИЗНОСИМАЯ КАК "СИКВЭЛ") символизирует собой Структурированный Язык Запросов. Это - язык который дает вам возможность создавать и работать в реляционных базах данных, которые являются наборами связанной информации сохраняемой в таблицах.

Мир баз данных становится все более и более единым, что привело к необходимости создания стандартного языка, который мог бы использоваться для функционирования в большом количестве различных видов компьютерных сред. Стандартный язык позволит пользователям, знающим один набор команд, использовать их, чтобы создавать, отыскивать, изменять, и передавать информацию, независимо от того, работают ли они на персональном компьютере, сетевой рабочей станции, или на универсальной ЭВМ.

В нашем все более и более взаимосвязанном компьютерном мире, пользователь снабженный таким языком, имеет огромное преимущество в использовании и обобщении информации из ряда источников с помощью большого количества способов.

Элегантность и независимость от специфики компьютерных технологий, а также его поддержка лидерами промышленности в области технологии реляционных баз данных, сделало SQL, и вероятно в течение обозримого будущего оставит его, основным стандартным языком. По этой причине, любой кто хочет работать с базами данных 90-х годов должен знать SQL.

Стандарт SQL определяется ANSI (Американским Национальным Институтом Стандартов) и в данное время также принимается ISO (Международной организацией по стандартизации). Однако, большинство коммерческих программ баз данных расширяют SQL без уведомления ANSI, добавляя разные другие особенности в этот язык, которые, как они считают, будут весьма полезны. Иногда они несколько нарушают стандарт языка, хотя хорошие идеи имеют тенденцию развиваться и вскоре становиться стандартами "рынка" сами по себе в силу полезности своих качеств.

6.1 Пользовательские базы данных

При создании новой пользовательской БД следует помнить о том, что имена БД уникальны для каждого экземпляра SQL Server. Для создания пользовательской БД

не нужно явно определять никакие другие атрибуты. При выборе имени рекомендуется следовать правилам, установленным для имен идентификаторов SQL Server,

хотя это и не обязательно:

• первым символом в имени должна быть буква, символ подчеркивания (\_), знак @,

который обозначает локальную переменную или параметр, или знак #, который

обозначает временную таблицу или процедуру;

• все символы в имени идентификатора после первого могут быть цифрами или знаком доллара ($);

• имя идентификатора не должно содержать пробелов и специальных символов;

• запрещается использовать в качестве имен идентификаторов зарезервированные

слова SQL Server, набранные в верхнем или нижнем регистре.

Примечание Если при выборе имени идентификатора не соблюдались перечисленные выше правила именования, то при использовании имени идентификатора в операторах необходимо заключать это имя в двойные кавычки или квадратные скобки.

Существуют также и другие параметры БД, которые вам необходимо определить.

Как правило, при создании БД указываются ее размер, физическое и логическое имя

файла, а также размещение основного файла данных и первого файла журнала транзакций на диске. При создании БД можно добавить несколько файлов данных и файлов журнала транзакций, разместив их на разных дисках, или сделать это уже после

6.2 Создание пользовательской базы данных

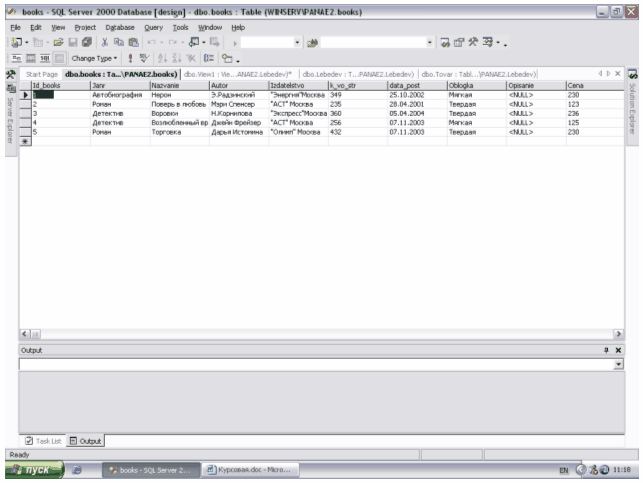
Создания пользовательской БД. Кроме того, можно объединить файлы данных в группы файлов и изменить группу файлов, выбранную по умолчанию. На занятии 4 этой главы вы узнаете, как разместить файлы БД на нескольких дисках, как использовать множество файлов данных при создании пользовательской БД и как создать пользовательские группы файлов для БД. Для каждого созданного вами файла данных и файла журнала транзакций вы можете определить следующие параметры: включить или отключить автоматическое увеличение размера файла при его заполнении, определить величину автоматического приращения размера файла, а также установить максимальный размер файла БД. На занятии 3 этой главы вы узнаете, каким образом вы можете контролировать увеличение размера БД.

Если при создании пользовательской БД не определены перечисленные выше дополнительные параметры, SQL Server использует значения по умолчанию. В табл.

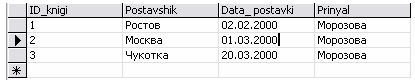
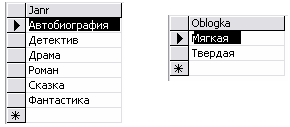
6-1 перечислены значения параметров по умолчанию для БД с именем SelfPaced, созданной на экземпляре SQL Server по умолчанию.

7. Проектирование базы данных на SQL

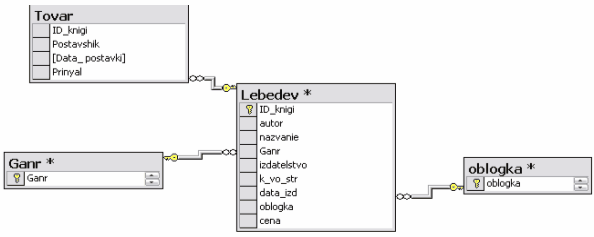
Проектирование базы данных также проводилось при помощи SQL. Данная таблица имеет такие же поля, что и таблица в ACCESS. Ниже приведена главная таблица в SQL:



Аналогично ACCESS выносим данные, которые повторяются в отдельные таблицы.

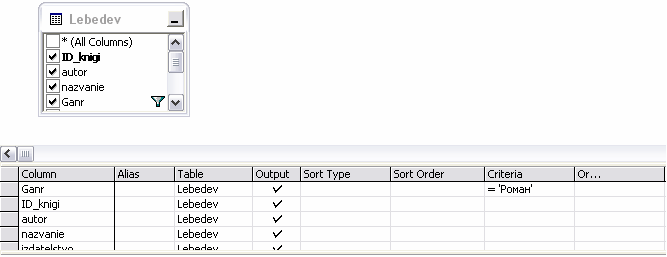
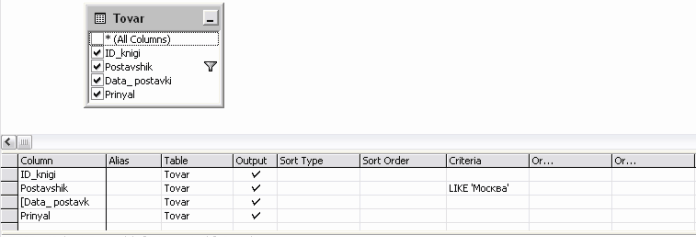


Данные таблицы связанны между собой. Связи между данными таблицами необходимы для объединения данных в одну таблицу, которая может содержать все данные в одной таблице



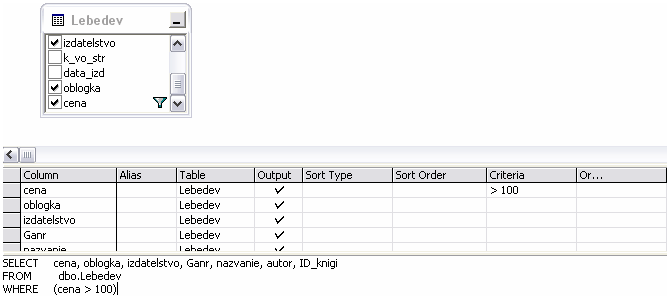
Это запрос на выбор данных о поставщике, дате поставки и кто принимал товар из Москвы.

Этот запрос по выбору жанра “Роман”.



А этот запрос на выбор книг, у которых цена больше 100 р.

Заключение



Использование баз данных и информационных систем – это составная часть функционирования различных преуспевающих организаций и деятельности современного человека. В связи с этим большую актуальность приобретает освоение принципа построения и эффективного применения соответствующих технологий и программных продуктов.

В настоящее время базы данных почти во все сферы человеческой деятельности. Так как с помощью электронных баз данных работа персонала организации, будь то хоть книжный магазин, становится быстрой, качественной и удобной.

В результате нашей работы, объектом которой являлся книжный магазин, была создана база данных «Books».

Проектирование базы данных «Books», проводилась помощью СУБД Microsoft Access и языка запросов SQL.Данная база данных, содержит основные характеристики : книжный жанр, название книги, автора книги, издательство, количество страниц, твердый или мягкий переплет, количество страниц, цену, поставщика, дату поставки. Данный продукт существенно облегчает работу пользователей.

В данной курсовой работе изложено об основных понятиях, функциях баз данных, использованию и созданию в Microsoft Access , а также в SQL.

Также в ней описано об основных моделях баз данных, которые известны в настоящее время. Но и проектирование базы данных «Books»,является неотъемлемой частью курсовой работы.

После описания проектирования баз данных в СУБД Microsoft Access далее информация о языке запросов SQL.При помощи которого была разработана база данных .

В итоге данная база данных «Books», будет, эффективно применятся в книжных магазинах, обеспечивая отличный результативность работы.

Список используемой литературы

1. Информатика. Базовый курс /Симонович С.В. и др. – СПб: Издательство «Питер», 2000. – 640 с.
2. Системы управления базами данных. Учебное пособие /Ломтадзе В.В., Шишкина Л.П. – Иркутск: ИрГТУ, 1999. – 116 с.
3. Теория реляционных баз данных. Учебное пособие / Мейер М.М. – Москва: Мир, 1999. -610 с.
4. Access для профессионалов. Учебное пособие / Верман А.Я. – СПб: Издательство «Питер, 1998. – 760 с.