## § 6. БАЗЫ ДАННЫХ И СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

### 6.1. ПОНЯТИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Базы данных - важнейшая составная часть **информационных систем.** Здесь мы ограничимся лишь кратчайшими общими сведениями об информационных системах, сосредоточив внимание на базах данных как таковых.

Информационные системы предназначены для хранения и обработки больших объемов информации. Изначально такие системы существовали в письменном виде. Для этого использовались различные картотеки, папки, журналы, библиотечные каталоги и т.д. Любая информационная система должна выполнять три основные функции: ввод данных, запросы по данным, составление отчетов.

**Ввод данных.** Система должна предоставлять возможность накапливания и упорядочивания данных. Необходимо обеспечить просмотр этих данных, внесение в них изменений и дополнений с тем, чтобы поддерживать актуальность информации.

**Запросы по данным.** В системе должна существовать возможность находить и просматривать отдельные части накопленной информации.

**Составление отчетов**. Время от времени возникает необходимость обобщать и анализировать большую группу данных (или даже все данные) информационной системы, представляя ее в виде документа.

Обслуживание информационных систем, реализованных в письменном (бумажном) виде, сопряжено со многими трудностями: чем больше информационная система, тем больше бумаги (карточек) и места требуется для их хранения (в этом можно убедиться на примере библиотеки); много времени тратится на поиск нужной информации. Сложности возникают при обновлении, анализе и обработке данных.

Предположим мы хотим собрать информацию про альбомы музыкальных групп. Пусть имеется информация о некоторых альбомах: 1965, Led Zeppelin 4, Lp, Help!, Atlantic, 1971. Lp(England), EMI. 1970, Flash Gordon, Parlophone, 1980, Led Zeppelin 3, Soundtrack, Lp, Atlantic. Этот список мало о чем говорит. Извлечь какую-либо информацию из этого набора данных практически невозможно.

Представим данные в виде табл. 2.2.

**Таблица 2.2 Информация об альбомах музыкальных групп**

|  |  |  |  |
| --- | --- | --- | --- |
| Название альбома | Год выпуска | Тип | Фирма альбома |
| Help! | 1965 | Lp (England) | Parlophone |
| Led Zeppelin 4 | 1971 | Lp | Atlantic |
| Led Zeppelin 3 | 1970 | Lp | Atlantic |
| Flash Gordon | 1980 | Soundtrack | EMI |

Теперь воспринимать и использовать информацию стало гораздо удобнее. Представленная таблица является **информационной моделью.** Объектами, отраженными в этой модели, являются музыкальные альбомы (групп), причем все данныевзаимосвязаны.

### 6.2. ВИДЫ СТРУКТУР ДАННЫХ

В информатике совокупность взаимосвязанных данных называется **информационной** **структурой,** или **структурой данных**. В нашем примере объектами модели являются музыкальные альбомы. Свойства же этих объектов находятся в столбцах таблицы («Название альбома», «Год выпуска», «Тип альбома», «Фирма»), их называют **атрибутами** объектов. Таким образом, каждая строка таблицы - есть совокупность атрибутов объекта. Такую строку называют **записью,** а столбец - **полем записи**.

Помимо сведений, указанных в атрибутах, табличная организация данных позволяет получить дополнительную информацию. К примеру, нетрудно узнать (в предположении, что наша табл. 2.2 заполнена данными):

• какая группа выпустила больше альбомов за определенный период;

• число альбомов данной группы;

• сколько имеется альбомов типа Soundtrack (музыка к фильму);

• какая фирма выпустила наибольшее число альбомов данной группы.

Табличная организация данных называется также **реляционной.** Кроме табличной структуры данных существуют другие виды структурной организации данных.

Для **иерархических структур** (рис.2.22) характерна подчиненность объектов нижнего уровня объектам верхнего уровня. Важно отметить, что в дереве, между верхними и нижними объектами, задано отношение «один ко многим» (т.е. одной группе соответствует много альбомов, одному альбому соответствует много песен).



*Рис. 2.22.* Пример иерархической организации данных

Несмотря на то, что в атрибутах, описывающих песню, нет названия альбома, глядя на дерево по линиям связи можно сказать, какая песня принадлежит альбому. Благодаря линиям связи можно определить принадлежность альбома группе. Из данной иерархической структуры можно узнать:

• в каком альбоме больше песен;

• число альбомов выпущенных группой;

• есть ли в альбомах одинаковые песни и т.д.

**Сетевую структуру** данных можно представить в виде схемы, рис. 2.23.



*Рис. 2.23.* Пример сетевой организации данных

Глядя на рис. 2.23, можно определить, какими инструментами владеет музыкант, является ли он вокалистом. В этом случае есть два уровня взаимосвязанных объектов, но отношение между ними «многие ко многим».

Пусть в этой сетевой структуре данные о музыкантах и «инструментах» состоят из следующих атрибутов: музыкант - ФИО, рост, цвет волос, время рождения; инструмент- название инструмента, какой фирмой изготовлен инструмент.

Тогда схема позволяет ответить на следующие вопросы:

• гитары какой фирмы предпочитает большинство музыкантов;

• какой музыкант владеет наибольшим количеством инструментов и др.

Построение структуры данных происходит в следующем порядке:

• определяются объекты описания;

• определяются структуры этих объектов;

• выбирается тип структуры, отображающий отношения между объектами (табличная, иерархическая, сети);

• строится конкретная информационная структура.

### 6.3. ВИДЫ БАЗ ДАННЫХ

Дадим основное определение. База данных - это реализованная с помощью компьютера информационная структура (модель), отражающая состояние объектов и их отношения.

Следует учесть, что это определение не является единственно возможным. Информатика в отношении определений чаще всего не похожа на математику с ее полной однозначностью. Если подойти к понятию «база данных» с чисто пользовательской точки зрения, то возникает другое определение: база данных - совокупность хранимых операционных данных некоторого предприятия. Все дело в том, какой аспект доминирует в рассмотрении; в данной главе первое из определений более уместно.

Поскольку основу любой базы данных составляет информационная структура, базы данных делят на три рассмотренные выше типа: табличные (реляционные), сетевые, иерархические.

Опыт использования баз данных позволяет выделить общий набор их рабочих характеристик:

• **полнота** - чем полнее база данных, тем вероятнее, что она содержит нужную информацию (однако, не должно быть избыточной информации);

• **правильная организация** - чем лучше структурирована база данных, тем легче в ней найти необходимые сведения;

**• актуальность -** любая база данных может быть точной и полной, если она постоянно обновляется, т.е. необходимо, чтобы база данных в каждый момент времени полностью соответствовала состоянию отображаемого ею объекта;

• **удобство для использования** - база данных должна быть проста и удобна в использовании и иметь развитые методы доступа к любой части информации.

Соответственно возможностям организации реляционных, иерархических и сетевых информационых структур, существуют и аналогичные виды баз данных. В них данные представлены в формах, адекватных соответствующим структурам. Однако иерархические и сетевые базы данных являются гораздо менее распространенными, чем реляционные и не могут быть реализованы с помощью наиболее популярных СУБД, входящих в состав программного обеспечения ЭВМ, поэтому на них далее останавливаться не будем.

**Реляционные базы данных**

Наиболее распространенными в практике являются реляционные базы данных. Название «реляционная» (в переводе с английского relation - отношение) связано с тем, что каждая запись в таблице содержит информацию, относящуюся только к одному конкретному объекту.

Всякое отношение должно иметь свое имя. Пусть есть отношение с названием «Альбомы группы». В этом случае структура базы данных, состоящая из одной таблицы, запишется так: *Альбомы группы (название альбома, год выпуска, тип* *альбома, фирма).*Однако чаще база данных строится на основе нескольких таблиц, связанных между собой через общие атрибуты. Пусть, например, в базе данных «Рок-энциклопедия» содержатся две таблицы - 2.3, *а* и 2.3, *б.*

**Таблица 2.3, *а* Музыкальные альбомы групп**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Код альбома | Код группы | Название альбома | Год выпуска | Тип альбома | Фирма |
| 25 | 1 | Help! | 1965 | Lp (English) | Pariophone |
| 36 | 2 | Led Zeppelin 4 | l97l | Lp | Atlantic |
| 35 | 2 | Led Zeppelin 4 - | 1970 | Lp | Atlantic |
| 34 | 3 | Flash Gordon | 1980 | Soundtrack | EMI |

**Таблица 2.3, *б Рок* группы**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код группы | Название группы | Страна | Дата создания | Дата распада |
| 1 | The Bealles | Англия | 1963 | I970 |
| 2 3 | Led Zeppelin 4 Flash Gordon | Англия Англия | 1989 199I | -- |

Эти две таблицы связаны между собой общим полем «Код группы». Поле «Код альбома» в таблице 2.3, *а* создается для того, чтобы отличать альбомы друг от друга. Это очень важно, так как в таблице могут находиться альбомы с одинаковыми названиями.

Необходимость использования больше одной таблицы станет заметной, если объединить эти таблицы в одну (табл. 2.4).

**Таблица 2.4 Объединение таблиц 23**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название группы | Страна | Дата создания | Дата распада | Название альбома | Год выпуска | Тип альбома | Фирма |
|  |  |  |  |  |  |  |  |
| The Beatles | Англия | 1963 | I970 | With the Beatles | 1963 | Lp | Pariophone |
| The Beatles | Англия | 1963 | I970 | Please, please me | 1963 | Lp | Pariophone |
| The Beatles | Англия | 1963 | I970 | Rubber soul | 1963 | Lp | Pariophone |

Из таблицы 2.4 видно, что при внесении в нее данных об альбомах определенной группы каждый раз приходится дублировать информацию первых четырех полей таблицы. Многократное сохранение в БД одних и тех же данных (название группы, страна, дата создания, дата распада) приведет к неэффективному использованию памяти, к тому же существенно возрастет вероятность ошибок при вводе данных. Разбив же данные по таблицам, можно в значительной степени избежать этих трудностей.

Через связь, определенную между этими таблицами, можно узнать

• сколько альбомов выпустила группа;

• выпускались ли альбомы у фирмыEMI;

• в каком году было выпущено максимальное количество альбомов и т.п.

Реляционные базы данных удобны еще и тем, что для получения ответов на различные запросы существует разработанный математический аппарат, который называется исчислением отношений или реляционной алгеброй. Ответы на запросы получаются путем «разрезания» и «склеивания» таблиц по строкам и столбцам. При этом ясно, что ответы также будут иметь форму таблиц.

Надо отметить, что база данных - это, собственно, хранилище информации и не более того. Однако, работа с базами данных трудоемкая и утомительная. Для создания, ведения и осуществления возможности коллективного пользования базами данных используются программные средства, называемые системами управления базами данных (СУБД).

### 6.4. СОСТАВ И ФУНКЦИИ СИСТЕМ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

База данных предполагает наличие комплекса программных средств, обслуживающих эту базу данных и позволяющих использовать содержащуюся в ней информацию. Такие комплексы программ называют СУБД. СУБД - это программная система, поддерживающая наполнение и манипулирование данными, представляющими интерес для пользователей при решении прикладных задач. Иными словами, СУБД является интерфейсом между базой данных и прикладными задачами.

Ниже перечислены основные функции СУБД.

1. **Определение данных** - определить, какая именно информация будет храниться в базе данных, задать свойства данных, их тип (например, число цифр или символов), а также указать, как эти данные связаны между собой. В некоторых случаях есть возможность задавать форматы и критерии проверки данных.

2. **Обработка данных** - данные могут обрабатываться самыми различными способами. Можно выбирать любые поля, фильтровать и сортировать данные. Можно объединять данные с другой, связанной с ними, информацией и вычислять итоговые значения.

3. **Управление данными** - можно указать, кому разрешено знакомиться с данными, корректировать их или добавлять новую информацию. Можно также определять правила коллективного доступа.

Входящие в состав современных СУБД средства совместно выполняют следующие функции:

• *описание* данных, их структуры (обычно описание данных и их структуры происходит при инициировании новой базы данных или добавлении к существующей базе новых разделов (отношений); описание данных необходимо для контроля корректности использования данных, для поддержания целостности базы данных);

• *первичный ввод, пополнение* информации в базе данных;

• *удаление* устаревшей информации из базы данных;

• *корректировку* данных для поддержания их актуальности;

• *упорядочение (сортировку)* данных по некоторым признакам;

• *поиск информации* по некоторым признакам (для описания запросов имеется специальный язык запросов, он обеспечивает также интерфейс между базой данных и прикладными программами пользователей, позволяет этим программам использовать базы данных);

• *подготовку и генерацию отчетов* (средства подготовки отчетов позволяют создавать и распечатывать сводки по заданным формам на основе информации базы данных);

• *защиту информации и разграничение доступа* пользователей к ней (некоторые разделы базы данных могут быть закрыты для пользователя совсем, открыты только для чтения или открыты для изменения; кроме того, при многопользовательском режиме работы с базой данных необходимо, чтобы изменения вносились корректно; для сохранения целостности данных служит механизм трансакций при манипулировании данными - выполнение манипуляций небольшими пакетами, результаты каждого из которых в случае возникновения некорректности операций «откатываются» и данные возвращаются к исходному состоянию);

*• резервное сохранение и восстановление* базы данных, которое позволяет восстановить утраченную при сбоях и авариях аппаратуры информацию базы данных, а также накопить статистику работы пользователей с базой данных;

• *поддержку интерфейса с пользователями,* который обеспечивается средствами ведения диалога (по мере развития и совершенствования СУБД этот интерфейс становится все более дружественным; дружественность существующих средств интерфейса предполагает

• наличие развитой системы помощи (подсказки), к которой в любой момент может обратиться пользователь, не прерывая сеанса работы с компьютером и базой данных;

• защиту от необдуманных действий, предупреждающую пользователя и предотвращающую потерю информации в случае поспешных или ошибочных команд;

• наличие нескольких вариантов выполнения одних и тех же действий, из которых пользователь может выбрать наиболее удобные для себя, соответствующие его подготовке, квалификации, привычкам;

• тщательно продуманную систему ведения человеко-машинного диалога, отображение информации на дисплее, использование клавиш клавиатуры). В настоящее время выделяют пять уровней проблематики систем управления базами данных:

• реляционные базы данных, 1970 - 90 гг.;

• объектно-ориентированные базы данных, 1980 - 90 гг.;

• интеллектуальные базы данных, 1985 - 90 гг.;

• распределенные базы данных, начало 1990 гг.;

• базы данных мультимедиа и виртуальной реальности настоящего времени.

Архитектурно СУБД состоит из двух основных компонентов; **языка описания данных** (ЯОД), позволяющего создать схему описания данных в базе, и **языка манипулирования данными** (ЯМД), выполняющего операции с базой данных (наполнение, обновление, удаление, выборку информации). Данные языки могут быть реализованы в виде тренажеров или интерпретаторов. Помимо ЯОД и ЯМД к СУБД следует отнести **средства** (или языки) **подготовки отчетов** (СПО), позволяющие подготовить сводки (отчеты) на основе информации, найденной в базе данных, по заданным формам.

Язык описания данных (ЯОД) - это язык высокого уровня *декларативного* (непроцедурного) типа, предназначенный для формализованного описания типов данных, их структур и взаимосвязей. Исходные тексты описания данных на этом языке после трансляции отображаются в управляющие таблицы, задающие размещение в памяти ЭВМ и связи между собой рассматриваемых данных. В соответствии с этими описаниями СУБД находит в базе требуемые данные, правильно преобразует их и передает, например, в прикладную программу пользователя, которой они потребовались. При записи данных в базу СУБД по этим описаниям определяет место в памяти ЭВМ, куда их требуется поместить, преобразует к заданному виду и устанавливает необходимые связи.

Язык манипулирования данными (или язык запросов) представляет собой систему команд, например, следующего типа:

• произвести выборку данного, значение которого удовлетворяет заданным условиям;

• произвести выборку всех данных определенного типа, значения которых удовлетворяют заданным условиям;

• найти в базе позицию данного и поместить туда новое значение (или удалить данное) и т.д.

Широкое распространение имеют СУБД для персональных компьютеров типа DBASE (DBASE III, IV, FoxPro, Paradox), Clipper, Clarion. Эти СУБД ориентированы на однопользовательский режим работы с базой данных и имеют очень ограниченные возможности. Языки подобных СУБД представляют собой сочетание команд выборки, организации диалога, генерации отчетов. В связи с развитием компьютерных сетей, в которых персональные компьютеры выступают в качестве развитых (интеллектуальных) терминалов, новые версии СУБД все в большей степени включают в себя возможности описанного ниже языка манипулирования данными SQL.

В последнее время стали среди СУБД популярными ACCESS (входит в состав MS Office), Lotus, Oracle.

**Язык манипулирования данными SQL**

Рассмотрим в качестве примера языка манипулирования данными некоторые команды языка SQL (от английских слов Structured Query Language), ставшего классическим языком реляционных баз данных.

Простейшая операция выборки представляется командой SELECT - FROM -WHERE (выбрать **-** из - где):

select <список атрибутов>

 from <отношение>

**where <условие>.**

Например, если необходимоиз отношения «Успеваемость», имеющего схему:

Успеваемость (ФПО\_студента, Дисциплина, Оценка, Дата, Преподаватель)

произвести выборку данных о том, какие оценки студент Иванов И.И. получил и по каким предметам, надо задать команду:

select Дисциплина, Оценка

from Успеваемость

where ФИО\_студента = «Иванов И. И.».

Часть команды «where» не является обязательной. Например, можно получить список всех студентов из отношения «Успеваемость» с помощью следующей команды:

select unique ФИО\_студента

from Успеваемость.

Ключевое слово unique позволяет исключить из результата дубликаты значений атрибута. Выбрать полностью информацию из таблицы можно с помощью команды

select \*

from Успеваемость.

Условие, следующее за «where», может включать операторы сравнения =, **<>,** >=, **<, <=,** булевы операторы AND, OR, NOT, а также скобки для указания желаемого порядка операции. Например, выбрать из таблицы «Успеваемость» фамилии студентов, сдавших на "5" экзамен по информатике, можно с помощью команды

select ФИО\_студента

from Успеваемость

where Дисциплина = «Информатика» AND Оценка=5.

Выборка может быть и вложенной, когда необходимо использовать в условии результаты-другой выборки. Например, если надо из отношения «Успеваемость» выбрать только студентов физико-математического факультета, пользуясь отношением «Студент», то команда select может выглядеть так:

select ФИО\_студента

from Успеваемость

where ФИО\_студента is in

(select Фамилия

from Студент

where Ф\_т = «физмат»).

Здесь «is in» является представлением оператора принадлежности элемента множеству. Можно также использовать операторы «is not in» («не принадлежит множеству»), «contains» - содержит, «does not contains» - не содержит. Смысл выражения «A contains В» (А содержит В) тот же, что и выражения «В is in А» (В принадлежит множеству А). Помимо слов select, from, where в команде выборки можно использовать и другие служебные слова, например:

order by **<**атрибут**>** asc - определяет сортировку результата выборки
 в порядке возрастания (asc) или убывания (desc)
 значения атрибута;

group by <атрибут1> - группирует данные по значениям атрибута;

having set <атрибут2>

minus - операция вычитания множеств (данных выборок).

Помимо команды выборки select, язык SQL имеет команды, позволяющие обновлять данные (update), вставлять (insert) и удалять (delete). Например, если студенты переводятся со 2-го курса на третий, информацию можно обновить командой

update Студент

set Kypc=3

where Kypc=2.

Если атрибут «Семенов С.С.» сдал экзамен по информатике на «5» 15 января 1996 г. преподавателю Петрову П.П., то информация об этом может быть добавлена в таблицу «Успеваемость» командой

insert inio Успеваемость:

<«Семенов С.С.», «Информатика», 5,15/01/96, Петров П.П.>.

Оператор insert может быть использован для включения одной строки (как в этом примере) или произвольного числа строк, определенных списком кортежей, заключенных в скобки, или операций выборки select из какой-либо другой таблицы. Команда delete используется для удаления информации из таблицы. Например,

delete Успеваемость

where Оценка=2

позволяет удалить информацию о студентах, получивших 2 (в случае их отчисления).

Существенно расширяют возможности языка библиотечные функции, такие как count (подсчет), sum (суммирование), avg (среднее), max и min.

Например, подсчитать число студентов в таблице «Студент»: select count (\*) from Студент.

### *6.5.* ПРИМЕРЫ СИСТЕМ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

**СУБД DBASE**

СУБД типа DBASE позволяют работать с реляционными базами данных (БД), структура которых состоит из трех элементов:

• число полей БД;

• характеристикакаждогополя**;**

• число записей в БД.

Каждое поле имеет следующие характеристики:

Field name Type Width Dec

(имя поля) (тип) (ширина) (дес.знаки).

**Field name**  - может состоять из набора символов, но без пробелов.

**Type** - в системах типа DBASE имеется 5 типов полей:

 С (Character) - символьный (текстовый) тип;

 N (Numerical) - числовой тип;

 L (Logical) - логический тип;

 D (Date) - поле дат, содержит даты в виде dd/mm/yy;

 М (Memo) - поле памяти, содержит большой текст (файл).

**Width** - обозначает допустимую ширину поля.

**Dec**  - используется для числовых полей и определяет точность
 задаваемых чисел.

DBASE создает следующие типы файлов:

.dbf - файлы с записями БД;

.prg - файлы с текстами программ;

.frm - файлы структуры форматных отчетов;

.ndx - индексные файлы, сортирующие записи по определенному ключу;

.mem - файлы данных переменной Mem.

Запуск СУБД осуществляется **из** операционной системы ехе-файлом (db.exe, foxdb и т.п.), выход-командой **.Quit.**

Теперь опишем кратко основные команды СУБД.

Создание БД осуществляется командой **Create.**

Create (например, «Абитуриент»)

После ввода этой команды на экране появится форма:

|  |  |  |  |
| --- | --- | --- | --- |
| Field name (имя поля) .001 | Type (тип) | Width(ширина) | Dec(дес.знаки). |

 .001

В соответствии с этой формой создадим структуру таблицы:

001 ФИО, С, 18

002 год\_рожд, С, 7

003 район. С, 13

004 адрес. С, 100

005 группа, С, 3

006 оценка 1,N,3

007 оценка2,N, 3

008 оценкаЗ,N, 3

Теперь можно начать заполнение таблицы записями.

В случае заполнения записями уже существующей базы данных, необходимо предварительно эту базу командой Use сделать активной:

.Use Абитуриент (use - использовать),

.Append (добавить)

Данные вводят в карточки, имеющие следующую форму:

Запись #00001

ФИО:

год\_рожд:

район :

адрес:

группа:

оценка1:

оценка2:

оценка3:

Например,

Запись #00005

ФИО: Семенов Сергей Викторович

год\_рожд: 1980

район: Туруханский

адрес: ул. Декабристов, д. 12, кв.23

группа: И2

оценка!: 5

оценка2: 4

оценкаЗ: 4

Запись можно ввести в определенное место БД, введя одну из команд:

.**Insert** (вставить)

или

.**Insert** **before** .

Перемещение по таблице и просмотр записей БД осуществляется командами:

**Go** **top** - (идти наверх) установка указателя на первую запись,

**Go** **bottom** - (идти вниз) установка указателя на последнюю запись;

**List** - (список) просмотр всех записей БД;

**Display** - (отобразить) просмотр записи, на которой находится указатель,

**Browse** - (просмотреть) помимо просмотра позволяет редактировать
 записи БД.

Редактирование записей позволяют проводить следующие команды:

**Edit** **N** - редактирование записи с номером *N;*

**Change** - (поменять) изменения только в некоторых полях или записях,
 удовлетворяющих заданным условиям;

**Delete** - (удалить) стирание ненужных записей;

**Copy** -(копировать) копирование записей.

Изменить структуру БД можно командой **Modify**. Ниже предложен перечень команд, осуществляющих обработку данных:

**Report** **form** - (отчет, форма) создание отчетов;

**Sort** - (сортировка) упорядочение БД по какому-либо ключу;

**Index** - (индекс) индексирование БД;

**Find** - (найти) поиск в БД.

Работу с несколькими БД помогают вести команды:

**Select** - (выбор) сделать активной какую-либо БД;

**Update** - (расширить) передача данных из одной БД в другую;

**Join** **to** - (присоединить) соединение целых БД.

Для осуществления интерактивности БД используют команды ввода и вывода:

**Wait** - (ожидание) пауза, приостановка;

**Input** - (вход) ввод данных;

**Say** - (сказать) вывод информации;

**Read** - (читать) ввод данных.

**СУБД Microsoft Access**

Access - в переводе с английского означает «доступ». MS Access - это функционально полная реляционная СУБД. Кроме того, MS Access одна из самых мощных, гибких и простых в использовании СУБД. В ней можно создавать большинство приложений, не написав ни единой строки программы, но если нужно создать нечто очень сложное, то на этот случай MS Access предоставляет мощный язык программирования - Visual Basic Aplication.

Популярность СУБД Microsoft Access обусловлена следующими причинами:

• Access является одной из самых легкодоступных и понятных систем как для профессионалов, так и для начинающих пользователей, позволяющая быстро освоить основные принципы работы с базами данных;

• система имеет полностью русифицированную версию;

• полная интегрированность с пакетами Microsoft Office: Word, Excel, Power Point, Mail;

• идеология Windows позволяет представлять информацию красочно и наглядно;

• возможность использования OLE технологии, что позволяет установить связь с объектами другого приложения или внедрить какие-либо объекты в базу данных Access;

• технология WYSIWIG позволяет пользователю постоянно видеть все результаты своих действий;

• широко и наглядно представлена справочная система;

• существует набор «мастеров» по разработке объектов, облегчающий создание таблиц, форм и отчетов.

Запустить систему Access можно несколькими способами:

• запуск с помощью главного меню в WINDOWS 95;

• запуск с помощью ярлыка на панели инструментов.

После запуска системы появится главное окно Access, рис. 2.24. Здесь можно открывать другие окна, каждое из которых по-своему представляет обрабатываемые данные. Ниже приведены основные элементы главного окна Access, о которых необходимо иметь представление.



*Рис.2.24.* Экран СУБД Access

В строке заголовка отображается имя активной в данный момент программы. Строка заголовка главного окна Access всегда отображает имя программы MICROSOFT Access.

**Пиктограмма системного меню** - условная кнопка в верхнем левом углу главного окна практически любого приложения. После щелчка на этой пиктограмме появляется меню, ко.торое позволяет перемещать, разворачивать, сворачивать или закрывать окно текущего приложения и изменять его размеры. При двойном щелчке на пиктограмме системного меню работа приложения завершается.

**Полоса меню** содержит названия нескольких подменю. Когда активизируется любое из этих названий, на экране появляется соответствующее подменю. Перечень подменю на полосе Access и их содержание изменяются в зависимости от режима работы системы.

**Панель инструментов** - это группа пиктограмм, расположенных непосредственно под полосой меню. Главное ее назначение - ускоренный вызов команд меню. Кнопки панели инструментов тоже могут изменяться в зависимости от выполняемых операций. Можно изменять размер панели инструментов и передвигать ее по экрану. Также можно отобразить, спрятать, создать новую панель инструментов или настроить любую панель инструментов.

В левой части **строки состояния** отображается информация о том, что вы делаете в настоящее время.

**Окно базы данных** появляется при открытой базе данных. В нем сосредоточены все «рычат управления» базой данных. Окно базы данных используется для открытия объектов, содержащихся в базе данных, таких как таблицы, запросы, отчеты, формы, макросы и модули. Кроме того, в строке заголовка окна базы данных всегда отображается имя открытой базы данных.

С помощью **вкладки** **объектов** можно выбрать тип нужного объекта (таблицу, запрос, отчет, форму, макрос, модуль). Необходимо сказать, что при открытии окна базы данных всегда активизируется вкладка-таблица и выводится список доступных таблиц базы данных. Для выбора вкладки других объектов базы данныхнужнощелкнуть по ней мышью.

Условные кнопки, расположенные вдоль правого края окна базы данных, используются для работы с текущим объектом базы данных. Они позволяют создавать, открывать или изменять объекты базы данных.

К основным объектам Access относятся таблицы, запросы, формы, отчеты, макросы и модули.

**Таблица** - это объект, который определяется и используется для хранения данных. Каждая таблица включает информацию об объекте определенного типа. Как уже известно, таблица содержит поля (столбцы) и записи (строки). Работать с таблицей можно в двух основных режимах: в режиме конструктора и в режиме таблицы.

В режиме конструктора задается структура таблицы, т.е. определяются типы, свойства полей, их число и названия (заголовки столбцов). Он используется, если нужно изменить структуру таблицы, а не хранящиеся в ней данные. В этом режиме каждая строка верхней панели окна соответствует одному из полей определяемой таблицы.

Режим таблицы используется для просмотра, добавления, изменения, простейшей сортировки или удаления данных. Чтобы перейти в режим таблицы, надо дважды щелкнуть мышью по имени нужной таблицы в окне базы данных (или, выделив в окне БД имя нужной таблицы, воспользоваться кнопкой открытого окна БД).

Из режима конструктора перейти в режим таблицы можно, щелкнувпо кнопкетаблицы на панели инструментов.

В режиме конструктора и в режиме таблицы перемещение между полями осуществляется с помощью клавиши ТАВ, а также вверх или вниз по записям с помощью клавиш, но в большинстве случаев пользоваться мышью гораздо удобнее.

Вследствие того, что в таблицах, как правило, содержится большое количество записей, размещение всех их на экране невозможно. Поэтом) для перемещения по таблице используют полосы прокрутки, расположенные в нижней и правой части окна. Левее нижней полосы прокрутки выводится номер текущей записи и общее число записей таблицы. Для перехода к записям с нужным номером необходимо активизировать поле *Номера записи,* щелкнув по нему, или нажать клавишу F5, после чего набрать на клавиатуре новый номер записи и затем нажать клавишу <Enter>.

**Запрос** - это объект, который позволяет пользователю получить нужные данные из одной или нескольких таблиц. Можно создать запросы на выбор, обновление, удаление или на добавление данных. С помощью запросов можно создавать новые таблицы, используя данные уже существующих одной или нескольких таблиц.

По сути дела, запрос - это вопрос, который пользователь задает Access о хранящейся в базе данных информации. Работать с запросами можно в двухосновныхрежимах: в режиме конструктора и в режиме таблицы.

Здесь надо вспомнить о том, что ответы на запросы получаются путем «разрезания» и «склеивания» таблиц по строкам и столбцам, и что ответы будут также иметь форму таблиц. В режиме конструктора формируется вопрос к базе данных.

**Форма** - это объект, в основном, предназначенный для удобного ввода отображения данных. Надо отметить, что в отличие от таблиц, з формах не содержится информации баз данных (как это может показаться на первый взгляд). Форма - это всего лишь формат (бланк) показа данных на экране компьютера. Формы могут строиться только на основе таблиц или запросов. Построение форм на основе запросов позволяет представлять в них информацию из нескольких таблиц.

В форму могут быть внедрены рисунки, диаграммы, аудио (звук) и видео (изображение).

Режимы работы с формой:

*•режим формы* используется для просмотра и редактирования данных; предоставляет дружественную среду для работы с данными и удобный дизайн их представления на экране;

*•режим конструктора форм* необходим, если необходимо изменить определение

формы (структуру или шаблон формы, а не представленные в ней данные), надо открыть форму в режиме конструктора;

*•режим таблицы* позволяет увидеть таблицу, включающую все поля формы; чтобы переключиться в этот режим при работе с формой, надо нажать кнопку таблицы на панели инструментов.

**Отчет** - это объект, предназначенный для создания документа, который впоследствии может быть распечатан или включен в документ другого приложения. Отчеты, как и формы, могут создаваться на основе запросов и таблиц, но не позволяют вводить данные.

Режимы работы с отчетом:

*Режим предварительного просмотра* позволяет увидеть отчет таким, каким он будет воплощен при печати. Для того чтобы открыть отчет в режиме предварительного просмотра,надо

• щелкнуть по вкладке *Отчеты,*

*•* кнопкой выбрать необходимый отчет в окне базы данных;

• щелкнуть по кнопке *Просмотра.*

*Режим конструктора* предназначен для изменения шаблона (структуры отчета).

**Макрос** - это объект, представляющий собой структурированное описание одного или нескольких действий, которые должен выполнить Access в ответ на определенное событие. Например, можно определить макрос, который в ответ на выбор некоторого элемента в основной форме открывает другую форму. С помощью другого макроса можно осуществлять проверку значения некоторого поля при изменении его содержания. В макрос можно включить дополнительные условия для выполнения или невыполнения тех или иных включенных в него действии. Возможно также из одного макроса запустить другой макрос или функцию модуля.

Работа с формами и отчетами существенно облегчается за счет использования *макрокоманд.* В MS Access имеется свыше 40 макрокоманд, которые можно включать в макросы. Макрокоманды выполняют такие действия, как открытие таблиц и форм, выполнение запросов, запуск других макросов, выбор опций из меню, изменение размеров открытых окон и т.п. Макрокоманды позволяют нажатием одной (или нескольких одновременно) кнопки выполнять комплекс действий, который часто приходится выполнять в течение работы. С их помощью можно даже осуществлять запуск приложений, поддерживающих динамический обмен данных (DDE), например MS Excel, и производить обмен данными между вашей базой данных и этими приложениями. Один макрос может содержать несколько макрокоманд. Можно также задать условия выполнения отдельных макрокоманд или их набора.

**Модуль** - объект, содержащий программы на MS Access Basic, которые позволяют разбить процесс на более мелкие действия и обнаружить те ошибки, которые невозможно было бы найти с использованием макросов.

Завершив работу с Access (или с ее приложением), надо корректно закончить сеанс. Простое выключение компьютера - плохой метод, который может привести к возникновению проблем. При работе WINDOWS приложения используют множество файлов, о существовании которых пользователь может даже не подозревать. После выключения машины эти файлы останутся открытыми, что в будущем может сказаться на надежности файловой системы жесткого диска.

Безопасно выйти из Access можно несколькими способами:

• двойным щелчком мыши на пиктограмме системного меню в строке заголовка главного окна Access;

• из меню Access выбором пункта *Файл Выход,*

• нажатием комбинации клавиш Alt + F4.