Работа с динамическими базами данных.

Лекция 6.

Специальности : 230105, 010501

Основные сведения о базах данных.

Под *Базой Данных (БД)* понимается совокупность совместно используемых логически связанных данных в виде упорядоченных совокупностей наборов фактов и цифр. *Логические связи данных* отображают типичные для конкретной предметной области отношения.

Совокупность БД и функциональных программ образуют Систему Управления Базой Данных (*СУБД*).

Среди решаемых *функциональными программами* задач следует выделить манипулирование данными и вычислительные задачи. Для манипулирования базой данных используются специализированные *языки запросов*, примером может служить язык SQL. При решении задач вычислительного характера используются *включающие языки* как подмножества общепринятых языков программирования (пример - Паскаль в среде Delphi).

Объекты и атрибуты.

Объект **следует рассматривать как уникально идентифицируемую сущность - объект материального мира. Свойство объекта описывается с помощью *атрибутов*.**

Атрибут **ассоциирует некоторое значение из своего *домена значений*, описываемого в Пролог-программе в разделе domains, с каждым объектом в *наборе* объектов. Под *набором* понимается совокупность объектов, с которыми может иметь дело конкретная БД. Атрибут или множество атрибутов, значения которых уникально идентифицирует каждый объект в наборе объектов, называется *ключом* этого набора объектов.**

Связь **между наборами объектов представляет собой упорядоченный список наборов объектов. Конкретный набор объектов может появиться в этом списке не более одного раза.**

В соответствии с количеством объектов из одного набора, который ассоциируется с некоторым количеством объектов из другого набора, различают *связи* : “*один к одному*” (функциональная зависимость), “*многие к одному*” (частичная функция), “*многие ко многим*”.

Уровни абстракции в СУБД.

В оперативной памяти и на дисках хранится только *физическая БД* как совокупность файлов и несложных структур данных. *Концептуальная БД* есть абстрактное отображение *физической*.

Совокупность хранящихся в базе актуальных данных, представляющих ее текущее содержимое, называется *экземпляром БД*.

***План БД* представляет собой перечень типов относящихся к БД объектов, связей между ними и способов выражения объектов и связей некоторого уровня абстракции на более низком (более конкретном) уровне. Для обозначения планов используется термин схема : *концептуальная схема* как план *концептуальной БД* и *физическая схема* как план *физической БД*.**

***Представление* или *подсхема* - это абстрактная модель некоторой части концептуальной БД или концептуальной схемы. План представления часто для простоты называется *подсхемой*.**

Концептуальная схема и три модели данных.

В целях спецификации концептуальной схемы и некоторых деталей ее реализации физической схемой СУБД предоставляет так называемый *язык определения данных*. Подобный язык позволяет записывать концептуальную схему в терминах некоторой "*модели данных*".

Существуют *три* основные *модели* организации БД :

1. *Иерархическая модель*. Для нее характерно представление данных в виде деревьев, в которых совокупность дочерних узлов, представляющая набор объектов, ассоциируется с родительской вершиной посредством связи типа "многие к одному".

2. *Сетевая модель*. Здесь данные представляются ориентированным графом, в котором каждой вершине сопоставляется тип логической записи, между логическими записями допускаются бинарные связи "многие к одному", для двух записей различают запись-владелец и запись-подчиненный, причем у подчиненной записи всегда один владелец.

3.*Реляционная модель* - основана на представлении данных в виде таблиц.

Реляционные базы данных.

В основе реляционной модели лежит математическое понятие теоретико-множественного *отношения*, которое представляет собой подмножество декартова произведения списка доменов. Декартовым произведением доменов $D_1 \times D_2 \times \ldots \times D_k$ называется множество всех кортежей $(v_1, v_2, \ldots, v_k)$ длины k : $v_1 \in D_1$, $v_2 \in D_2$, … , $v_k \in D_k$.

Кортежами **называются элементы отношения. О каждом отношении - подмножестве декартова произведения $D_1 \times D_2 \times \ldots \times D_k$ можно сказать, что оно имеет *арность* k. Кортеж $(v_1, v_2, \ldots, v_k)$ имеет k компонентов. В реляционной модели отношение представляется как таблица, где каждая строка есть кортеж и каждый столбец соответствует одному компоненту. Имя столбца задает *имя атрибута*. *Элементом отношения* в реляционной базе данных обычно называют строку таблицы. По аналогии с мощностью множества, число элементов таблицы называется *мощностью отношения*.**

Список имен атрибутов называется *схемой отношения*. Совокупность схем отношений, используемых для представления информации, называется схемой (реляционной) базы данных, а текущие значения соответствующих отношений - *(реляционной) базой данных)*.

Базы данных в Турбо-Прологе.

Средства Турбо-Пролога позволяют работать как со статическими, так и с динамическими БД. Код динамической БД не зависит от программного кода в отличие от статической БД, утверждения которой (элементы отношения в терминологии реляционных БД) являются частью кода программы и не могут быть изменены во время ее выполнения.

Благодаря указанному свойству динамическая БД может храниться на диске в отдельном файле и при выполнении программы считываться с диска в оперативную память. При этом файл БД представляет собой набор связанных между собой записей, каждая запись соответствует кортежу. БД, располагающаяся в оперативной памяти компьютера, называется *резидентной БД*.

Можно указать следующее соответствие понятий БД Турбо-Пролога и реляционной БД :

БД Турбо-Пролога	*Реляционная БД*
предикат БД	отношение
объект	атрибут
отдельное утверждение	элемент отношения
количество утверждений	мощность

Описание динамической БД в Турбо-Прологе.

Для описания предикатов динамической БД в программу на Турбо-Прологе вводится раздел database. В качестве примера рассмотрим описание производителя, включающее название и полный юридический адрес :

```
domains
   address=address(zip,country,city,street,house)
   name,zip,country,city,street,house,phone,fax=symbol
database
   dmanufacturer(name,address,phone,fax).
```

Предикаты статической БД имеют ту же форму представления, но описываются в разделе predicates. Так, предикат статической БД, соответствующий предикату dmanufacturer описанной выше динамической БД есть

```
predicates
   manufacturer(name,address,phone,fax).
```

В целях различения предикатов динамической и статической БД в обозначение предиката динамической БД обычно добавляется латинская буква d.

Предикаты динамической базы данных.

Наряду со стандартными средствами работы с предикатами и утверждениями, в Турбо-Прологе имеются и специальные встроенные предикаты для работы с динамическими базами данных. Основным отличием Турбо-Пролога от других реализаций этого языка является то, что в динамической БД могут содержаться только факты (но не правила !).

К встроенным предикатам для работы с динамической БД относятся : asserta, assertz, retract, save, consult, readterm и findall.

Предикаты asserta и assertz заносят новые факты в резидентную БД. При этом при использовании asserta добавляемый факт помещается перед всеми уже внесенными утверждениями данного предиката, а при использовании assertz - после. Синтаксис : asserta(Clause) и assertz(Clause).

Предикат retract удаляет имеющееся утверждение из динамической БД. Его синтаксис : retract(Existing_clause).

Для модификации БД рекомендуется использовать комбинацию выражений с предикатами asserta, assertz и retract.

Создание и модификация БД.

В качестве примера рассмотрим БД ведущих предприятий Новгородской области. В результате согласования конъюнкции ЦУ :

```
asserta(dmanufacturer("НПО Планета",
          address("173004","Россия","Новгородская область",
                "Великий Новгород","ул.Федоровский Ручей","2/13"),
         "(81622) 31736", "(81622) 33286")),
asserta(dmanufacturer("ФГУП ПО Квант",
         address("173000","Россия","Новгородская область",
              "Великий Новгород","ул.Большая Санкт-Петербургская",
              "73/1"),
         "(81622) 27117", "(81622) 24333")),
assertz(dmanufacturer("ОАО Завод Старорусприбор",
         address("175200","Россия","Новгородская область",
               "Старая Русса","ул.Минеральная",
               "24"),
         "(81652) 27460", "(81652) 35682")).
```

Создание и модификация БД (продолжение).

в оперативной памяти будет создана БД из трех утверждений :

```
dmanufacturer("ФГУП ПО Квант",
        address("173000","Россия","Новгородская область",
                "Великий Новгород",
                "ул.Большая Санкт-Петербургская",
                 "73/1"),
        "(81622) 27117","(81622) 24333")
dmanufacturer("НПО Планета",
        address("173004","Россия","Новгородская область",
                "Великий Новгород","ул.Федоровский Ручей",
                 "2/13"),
        "(81622) 31736","(81622) 33286")
dmanufacturer("ОАО Завод Старорусприбор",
        address("175200","Россия","Новгородская область",
                "Старая Русса","ул.Минеральная","24"),
        "(81652) 27460","(81652) 35682")
```

Удаление утверждения из динамической БД.

Добавим с помощью предиката assertz к имеющимся в нашей БД утверждениям информацию о ЗАО Новтрак :

```
dmanufacturer("ЗАО Новтрак",
        address("173008","Россия","Новгородская область",
            "Великий Новгород","ул.Магистральная",
            "15"),"(8162) 640951","(8162) 643049").
```

Рассмотрим варианты удаления введенного утверждения с использованием встроенного предиката *retract*.

Для удаления из БД информации о некотором предприятии можно указать все его данные в соответствии с содержащимся в БД утвержеднии, например :

```
retract(dmanufacturer("ЗАО Новтрак",
        address("173008","Россия","Новгородская область",
            "Великий Новгород","ул.Магистральная",
            "15"),"(8162) 640951","(8162) 643049")).
```

Но поскольку в нашей БД название уникально идентифицирует каждое предприятие и может быть рассмотрено как ключ, то для удаления утверждения, касающегося некоторого предприятия, предикату *retract* достаточно указать название :

```
retract(dmanufacturer("ЗАО Новтрак",_,_,_).
```

Сбор данных БД в список.

Для сбора данных из БД с целью их последующей обработки используется встроенный предикат findall. Синтаксис предиката следующий : findall(Var_name,Dbase_pred_name,List_name), где

Dbase_pred_name - имя предиката БД, Var_name соответствует имени атрибута, List_name - имя переменной выходного списка, причем элементы списка принадлежат к тому же домену, что и Var_name. При этом домен списка должен быть объявлен в разделе domains. Пример для рассматриваемой БД предприятий (база загружается с диска в оперативную память при помощи описываемого далее встроенного предиката *consult*):

```
domains
  . . .
    name_list=name*

goal

    consult("manufact.dba"),

    findall(Name,dmanufacturer(Name,_,_,_),Name_list).
```

Согласование данной конъюнкции ЦУ дает :

Name_list=[“НПО Планета”,”ФГУП ПО Квант”,”ОАО Завод Старорусприбор”,”ЗАО Новтрак”]

Предикаты для работы с БД в целом.

Предикат *save* сохраняет находящуюся в оперативной памяти БД в текстовом файле. Синтаксис этого предиката :

***save(DOS_file_name)*, причем DOS_file_name есть допустимое в MS DOS или PC DOS имя файла. Если файл с таким именем уже имеется на диске, то он будет перезаписан !**

Файл БД может быть считан в память (загружен) с помощью предиката consult : *consult(DOS_file_name).* Предикат consult неуспешен, если файл с указанным именем отсутствует на диске, содержит ошибки, как, например, несоответствие синтаксиса предиката из файла описаниям из раздела программы database, или в случае отсутствия места в памяти для размещения содержимого файла.

Для считывания с диска данных, записанных в форме утверждений, используется встроенный предикат Турбо-Пролога *readterm*. Его назначение - чтение из файла объектов, относящихся к определенному в программе домену и записанных туда при помощи предиката write. Синтаксис этого предиката :

***readterm(Domain, Term)*, где *Domain* задает имя домена, а *Term* - различные наборы объектов этого домена.**

Чтение объекта из файла БД.

Рассмотрим чтение информации об отдельном предприятии из созданной нами БД. Будем считать, что созданная нами БД сохранена в файл “manufact.dba” на диск в текущий каталог с использованием предиката save : *save("manufact.dba")*. Для обеспечения возможности чтения с помощью предиката readterm записанной в файл информации зададим в разделе domains тип домена хранящихся в файле значений объектов : *manuf_record=dmanufacturer(name,address,phone,fax)*.

Для получения доступа к файлу его необходимо открыть с помощью предиката openread : openread(data_file,”manufact.dba”), отождествив файл на диске с логическим файлом *data_file*. Файловый домен с именем *data_file* должен быть предварительно описан в разделе *domains*.

```
domains
   address=address(zip,country,region,city,street,house)
   name,zip,country,region,city,street,house,phone,fax=symbol
   file=data_file
   manuf_record=dmanufacturer(name,address,phone,fax)
database
   dmanufacturer(name,address,phone,fax).
goal
   openread(data_file,"manufact.dba"),readdevice(data_file),
   readterm(manuf_record,dmanufacturer(Name,Address,Phone,Fax)),
   closefile(data_file).
```

Создание БД на диске.

Для работы с БД на диске используются предикаты для чтения и записи внешних файлов : *existfile, writedevice, openappend, openwrite, closefile*.

В целях обеспечения эффективности доступа программы к информации нерезидентной БД, в частности, для организации доступа по ключу, применяются индексные файлы. Работа с индексными файлами в Турбо-Прологе может быть организована с применением встроенных предикатов форматированного вывода информации и чтения/записи со смещением в файл :

writef("%<кол_поз>\n",Pos) - предикат форматированного вывода значения индекса. Индекс задается переменной Pos. Для записи индекса отводится поле из позиций в количестве <кол_поз>. Присвоенное переменной Pos значение определяет положение записи в БД.

filepos(Logical_file_name,Pos,Mode) - предикат задания значения смещения положения указателя (Pos) файла с логическим именем Logical_file_name. Mode=0 - смещение отсчитывается от начала файла, Mode=1 - от текущей позиции, Mode=2 - от конца файла. Предикат может быть использован для получения значения смещения текущего положения указателя файла. При этом смещение берется относительно начала файла.

Пример использования индексного файла при создании БД предприятий.

```
domains
   address=address(zip,country,region,city,street,house)
   name,zip,country,region,city,street,house,phone,fax=symbol
   file=data_file;index_file
   manuf_record=dmanufacturer(name,address,phone,fax)
   filename=string
database
   dmanufacturer(name,address,phone,fax).
predicates
   add_info(manuf_record,filename,filename).
clauses
   add_info(Term,Indexfile,Datafile):- existfile(Indexfile), existfile(Datafile),
         openappend(data_file,Datafile), writedevice(data_file),
         filepos(data_file,Pos,0), write(Term),nl, closefile(data_file),
         openappend(index_file,Indexfile), writedevice(index_file),
         writef("%7.0\n",Pos), closefile(index_file).
   add_info(Term,Indexfile,Datafile):- openwrite(data_file,Datafile),
         writedevice(data_file), filepos(data_file,Pos,0), write(Term),nl,
         closefile(data_file), openwrite(index_file,Indexfile),
         writedevice(index_file), writef("%7.0\n",Pos), closefile(index_file).
```

Пример использования индексного файла при создании БД предприятий (продолжение).

В результате согласования конъюнкции ЦУ :
goal

```
add_info(dmanufacturer("НПО Планета",
        address("173004","Россия","Новгородская область",
          "Великий Новгород","ул.Федоровский Ручей","2/13"),
        "(81622) 31736","(81622) 33286"),
      "manufact.ind","manufact.dba"),
add_info(dmanufacturer("ФГУП ПО Квант",
        address("173000","Россия","Новгородская область",
          "Великий Новгород","ул.Большая Санкт-Петербургская",
          "73/1"),"(81622) 27117","(81622) 24333"),
      "manufact.ind","manufact.dba"),
add_info(dmanufacturer("ОАО Завод Старорусприбор",
        address("175200","Россия","Новгородская область",
          "Старая Русса","ул.Минеральная","24"),
        "(81652) 27460","(81652) 35682"),
      "manufact.ind","manufact.dba").
```

помимо файла БД "manufact.dba" на диске будет создан файл "manufact.ind", содержащий информацию о позициях начала записи порций информации отдельных предприятий.

Выводы.

Программы БД на Турбо-Прологе есть частный случай СУБД. Основанные на правилах средства Турбо-Пролога для работы с БД позволяют использовать результаты запросов к БД в качестве новых данных, которые можно поместить в БД в новых записях.Турбо-Пролог в наибольщей степени адаптирован для написания диалоговых систем именно для реляционной БД : внутренние унификационные процедуры языка осуществляют автоматическую выборку фактов с нужными значениями известных параметров и присваивают новые значения еще не определенным. Механизм отката позволяет находить все имеющиеся ответы на сделанный запрос.

Литература.

Ин Ц., Соломон Д. Использование Турбо-Пролога : Пер. с англ. - М.: Мир, 1993. С. 230-287,360-405

Ульман Дж. Основы систем баз данных : Пер. с англ. - М.: Финансы и статистика, 1983. С. 10-25,72-97