Факультет Физико-математических и естественных наук
Кафедра Информационных технологий
Направления Прикладная математика и информатика
Математика. Компьютерные науки

ТЕОРИЯ БАЗ ДАННЫХ

Обязательная дисциплина, привязанная к 5 семестру.
Трудоемкость – 3 кредита, 2 часа лекций и 2 часа лабораторных в неделю.

Занятие 1. Основные элементы языка Клипер.

1. Алфавит языка. Латинские буквы, цифры, специальные символы (по мере необходимости).
2. Константы.
   Числа (целые и с точкой – тип N)
   Символы (*любые* знаки, заключенные в ' или " – тип C)
   Логические (.t., .f. – тип L)
   Даты (задаются только через функции – тип D)
3. Переменные.
   Последовательность букв и цифр, начиная буквы, но не более 8 знаков. Нет предварительного назначения типов. Тип определяется в момент присвоения значения.
   Элементы массива. Массив задается либо объявлением размерности, либо набором выражений, заключенных в фигурные скобки.
4. Функции.
   Обычное представление – имя(фактические параметры). Функций очень много, но для нас интересны следующие:

	N	C	L	D
N	Int(N)	Str(N[,l[,d]]) Space(N)	N сравнение N	
C	Val(C) Len(C)	Substr(C,i,l) Ltrim(C)	C сравнение C	CtoD(C)
L	If(L,N1,N2)	If(L,C1,C2)		
D	Day(D) Month(D) Year(D)	DtoC(D) DtoS(D)	D сравнение D	D + N D - N

   Date()
5. Выражения.
   Последовательность констант, переменных и указателей функций, объединенных знаками соответствующих операций с использованием круглых скобок.
   Операции: арифметические (+ - * / %), символьные (+), логические (! или .not. .and. .or.), сравнения (< <= = # >= >)
6. Операторы.
   Присваивания: переменная = выражение
   Последовательность операторов: каждый оператор с новой строки. Знак ; в конце строки означает, что следующая строка является продолжением оператора.
   Выбор: **if** условие
   операторы
   elseif условие-I //может повторяться 0 и более раз
   операторы
   else //может отсутствовать
   операторы
   endif
   Повторение: **for** переменная = начало **to** конец [**step** шаг] или **do while**
   операторы операторы
   next **enddo**
   Комментарии:

- * комментарий
- Текст // комментарий
- /*
  комментарий
  */

Вывод: ? список полей или ?? список полей

7. Программа (исходный текст). В любом редакторе в ASCII кодах с именем ИМЯ.prg (имя не более 8 знаков). Первым оператором желательно писать set date Italian, для вывода даты в виде день-месяц-год

8. Help – www.itk.ru

9. Трансляция hbmk2 ИМЯ.prg
   В результате exe код под тем же именем

10. Исполнение ./ИМЯ

11. Решение задач. (Объясняются условия, дается время на их решение. Для большинства они переходят в домашнее задание)
    - Простейшая задача для навыка запуска программ
      A=1
      B=15
      C=a + 2*b
      ? 'a=',a,'b=',b,'c=',c
      (вариации по красивой печати)
    - Написать программу печати таблицы квадратов чисел от M до N. Например:
      M=3
      N=24
      3 9
      4 16
      5 25
      …
      (определить к-во знаков максимального числа, к-во чисел в строке и организовать циклы для формирования таблицы)
    - Программа печати треугольника Паскаля

Занятие 2. Отношение - основной объект баз данных.

Данные можно представить в виде таблицы, в которой есть строки и столбцы и можно интерпретировать как совокупность данных об экземплярах объекта-отношения.

Строка – набор характеристик конкретного экземпляра объект, а *столбец* – отдельная характеристика (свойство): совокупность данных конкретного свойства всех экземпляров объекта-отношения. Все допустимые значения отдельного свойства называются *атрибутом*, а данные одной строки – *кортежом*.

При работе с отношением можно выделить три рода информации: структура, данные и служебная, причем первые два хранятся во внешней памяти под выбранным именем. В структуре отношения собрано описание атрибутов: название, тип, способ хранения, ограничения и т.п. При работе с отношением выделяется память под хранение служебной информации: общее количество кортежей, номер текущего кортежа, значения некоторых функций и т.д. В данных представлены значения каждого из атрибутов всех кортежей. Полезно представление о данных как о списке, в котором каждый кортеж – это узел, а указатель списка выделяет среди всех кортежей текущий. Мы можем говорить о предыдущем или следующем кортеже.

В рамках баз данных над объектом-отношением можно выполнить следующие операции: Создать, Открыть, Закрыть, Модифицировать.

Создание. Организация во внешней памяти нового файла под указанным именем с расширением .dbf. Изначально в этом файле хранится структура отношения и пустой список кортежей. Структура (схема) отношения описывается двумерным массивом, строка которого – данные об отдельном атрибуте. В клипере структура записывается в виде

{{<имя атрибута_1>,<тип>,<длина>,<дробная часть>},{<имя атрибута_2>,…,…,…}…}

где *тип* – один из символов: "N", "C", "L", "D", *длина* – количество символов в представлении числа или строки (1 для типа "L" и 8 для типа "D"), *дробная часть* – количество знаков дробной части в представлении числа и 0 в остальных случаях. Создание нового отношения осуществляется оператором-функцией dbcreate(<имя отношения>,<структура>).

Открытие. Оператор **use** <имя> **new**. Так как одновременно может быть открыто несколько отношений, то параметр new как раз используется для этого. При открытии отношения в оперативной памяти выделяется место для служебной информации и структуры, а данные хранятся по принципу страничного замещения групп кортежей. Одно из открытых отношений (последнее открытое или выделенное командой **select** <имя>) считается активным в данный момент исполнения программы. Имена атрибутов активного отношения становятся переменными программы.

Закрытие. Команда есть, но так как по окончании работы программы все открытые отношения автоматически закрываются, то мы ей не пользуемся.

Модификация. Предполагается, что схема отношения изменяется крайне редко. Т.о. речь идет только о кортежах. Над ними можно выполнять следующие действия: Добавить, Удалить, Изменить значения текущего кортежа, Перейти к другому кортежу. Все эти операции можно выполнить над активным отношением.

Добавить кортеж. Команда **append blank**. Новый кортеж всегда добавляется в коней списка кортежей, заполняется значениями по умолчанию и становится текущим кортежом.

Удалить кортеж. Команда **delete**. На самом деле информация кортежа не удаляется физически, а помечается и, следовательно, исключается из просмотра. Как следствие, после этой команды ни количество кортежей, ни их нумерация не изменяется.

Изменить кортеж. Команда **replace** <имя атрибута> **with** <новое значение>{, < имя атрибута i> **with** <новое значение>}. Напомним, что изменения выполняются только в текущем кортеже.

Перейти к кортежу. Две команды **goto** N и **skip** [±N]. После исполнения первой команды текущим кортежом становится кортеж с номером N, а исполнение второй указатель текущего кортежа перемещает на N кортежей вверх (+) или вниз (-) (по умолчанию +1). При этом в подсчет учитываются только не удаленные кортежи. Если при исполнении команды достигнут хвост или голова списка, то исполнение команды прекращается, а значением функции eof() при достижении хвоста списка (bof() для головы) становится ИСТИНА.

Задача 1. Создать и наполнить (15-20 кортежей) отношение

set date Italian

```
st={{'fio','C',15,0},{'sex','L',1,0},{'dr','D',8,0},{'stip','N',8,2}}
dbcreate('STUD',st)
use STUD new
append blank
repl fio with "Ivanov',…
```
Последние две команды повторяются 15-20 раз с разными данными.

Задача 2. Напечатать данные созданного отношения (самостоятельно).

```
set date Italian
use STUD new
do while !eof()
? fio, sex, dr, stip
skip
enddo
```

Занятие 3. Индексация отношений.

Невозможность физической сортировки из-за:

- большие объемы
- при исполнении программы просматривать кортежи в различном порядке

Идея индексации. Создание нового отношения со схемой:

1. ссылка на кортеж базового отношения (через № кортежа или каким либо другим способом),
2. значение выражения построенного на атрибутах базового отношения.

Это отношение существенно меньше базового по объему и именно оно перестраивается по возрастанию значения выражения. Теперь команды перемещения по кортежам выполняются по этому отношению, а через ссылку автоматически перемещается указатель текущего кортежа в базовом отношении. Этим достигается нужная последовательность просмотра кортежей.

Естественно, при внесении изменений в базовое отношение автоматически перестраивается индексное отношение.

Для одного базового отношения можно построить несколько индексных отношений (способов реализации этого несколько), чем достигается достаточно простой метод замены одного порядка просмотра кортежей на другой.

Команда клипера на построение индексного отношения

index on <выражение> **to** <имя инд.отношения>

Индексное отношение принимает расширение ntx.

Для получения убывающего порядка следует воспользоваться функцией descend(<выражение>), которая значение вычисленного выражения заменяет на его побитовое дополнение.

Задача. Вывести списки студентов:

1. в алфавитном порядке
2. в обратном алфавитном порядке
3. в алфавитном порядке сначала мужчины а потом женщины
4. мужчины в алфавитном порядке, а потом женщины в обратном алфавитном порядке
5. по месяцу рождения
6. по месяцу рождения, а внутри по возрастанию стипендии
7. по месяцу рождения, а внутри по убыванию стипендии

Занятие 4. Фильтрация в отношениях.

Проблема в том, чтобы просмотру кортежей подлежали не все кортежи, а только удовлетворяющие каким либо требованиям (условиям).

Занятие разбивается на 2 части: фильтры и команда seek.

1. Команда **set filter to** <условие>

По этой команде формируется процедура, которая при перемещении указателя текущего кортежа видит только те кортежи, которые удовлетворяют указанному условию.

В стандартную программу просмотра кортежей вставить команду

set filter to FIO='B'

Контрольные вопросы:

- будут ли выведены все студенты на букву В
- будут ли выведены только студенты на букву В

Объяснить, что происходит с указателем кортежей.

Команды **go top** – переход в начало списка**, go bottom** – переход в конец списка.

Решить задачи: выдать всех мужчин
выдать всех мужчин старше 20 лет
выдать всех родившихся в октябре
выдать всех, у кого стипендия меньше 1000 р.

2. Для большого количества записей этот процесс протекает медленно. Поставим задачу так, чтобы просмотр шел только тех кортежей, которые нас интересуют.

Это возможно, если мы можем достаточно быстро выйти на нужный нам кортеж и во время прекратить просмотр кортежей.

Это возможно, если имеется "правильное" индексное отношение.

Имеется команда seek <выражение поиска>

По этой команде в индексном отношении (а это списки, организованные по принципу B^+ деревьев) ищется первый кортеж, удовлетворяющий условию

<индексное выражение> >= <выражение поиска>.

Таким образом, быстро находится 1-й интересующий нас кортеж. Просматриваем все кортежи, что нам нужны и теперь следует во время остановиться. Упражнение: понять, каким образом.

Решить теперь верхние задачи и дополнительно:

выдать всех родившихся с мая по октябрь
выдать всех родившихся либо в мае, либо в октябре

Занятие 5, 6. Связи между отношениями.

Для того, чтобы увидеть как работают связи между отношениями, необходимо иметь как минимум два отношения. Для этого возвращаемся к первой задаче (создание STUD.dbf) и добавляем в него:

- расширение схемы студента новым атрибутом Ngr (тип и длина на усмотрение);
- добавляем для каждого студента значение группы;
- создаем новое отношение GR (Ngr, Spec).

Транслируем и пропускаем эту программу.

Следующая программа выводит для всех студентов: ФИО, пол, дату рождения, стипендию, номер группы и специальность группы:

```
set date Italian
set softseek on
use GR new
index on Ngr to GR
use STUD new
set relation to Ngr into GR
do while !eof()
? fio, sex, dr, stip,Ngr, GR->Spec
skip
enddo
```

set softseek on – включение режима «мягкого» поиска, которая при отсутствии искомого элемента устанавливает текущий указатель кортежей на тот кортеж, который бы стоял следующим после искомого элеиента, в случае присутствия искомого элемента.

Имеется команда **select** <имя>, которая переключает рабочую область на ранее открытое отношение с указанным именем.

Имеется функция **found()**, которая выдает истину, если **seek** нашла кортеж и ложь в противном случае.

Задание на конец и все следующее занятие:

1. ФИО №гр Спец
2. №гр
   ФИО Спец
3. №гр Спец
   ФИО
4. Спец №гр
   ФИО
5. Спец
   №гр
   ФИО
6. Спец
   ФИО №гр