# AI LAB – 8

**AIM**: **Database Handling in Prolog**

Two types of databases:-static and dynamic.

- Static database is a part of the program that is complied along with it. It does not change during execution of the program.
- Dynamic database can change dynamically at execution time and are of two types.

  Type1: created at each execution. It grows, shrinks and is deleted at the end of program.

  - This type of database is no longer available after program finishes its execution and is called working memory.

  Type2: Other types of dynamic databases are those which are stored in files and called database files.

  - These are consulted in any program whenever required.
  - These types of databases are not part of any particular program and are available for use in future by different programs using system defined predicates called save and consult.
  - While executing a Prolog program one can load database file(s) using 'consult' predicate.
  - These files can be updated dynamically at run time and saved using 'save' predicate.

  The format of predicates 'save' and 'consult' are as follows:

  - save(filename) - succeeds after saving the contents of a file named 'filename'.
  - consult(filename) - succeeds after loading or adding all the clauses from a file stored in 'filename' in the current program being executed.

Clauses can be added to a database at run time using following predicates.

asserta(X) & assertz(X) - succeed by adding fact X in the beginning & at the end of database of facts respectively.

Similarly obsolete clauses can be deleted by using system defined predicate called retract from dynamic database at the run time.

Sample Code I: Dynamic Database Type1 Example

```
domains

 name,sub=symbol

 marks=integer

database

 result(name,sub,marks)

predicates

 write1

 read1

clauses

 write1:- readln(Name),readln(Sub),readint(Marks),

                   asserta(result(Name,Sub,Marks)).

 read1:-retract(result(Sname,Ssubj,Smarks)),

      write(Sname),nl, write(Ssubj),nl,write(Smarks),nl,fail.
```

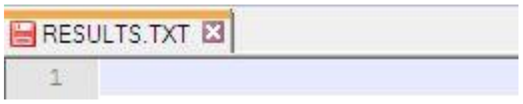
**Output trace for Sample Code I**:

```
Goal: read1
No
Goal: write1
apurva
dm
21
Goal: write1
jatayu
dm
25
Goal: read1
jatayu
dm
25
apurva
dm
21
Goal: read1
No
```

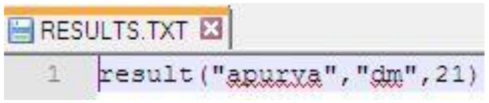
Sample Code II: Dynamic Database Type2 Example

```
domains

 name,sub=symbol

 marks=integer

database

 result(name,sub,marks)

predicates

 write1

 read1

 open1

 delete1(name)

 update1

 update2(name)

clauses

 open1:-consult("results.txt").

 write1:- readln(Name),readln(Sub),readint(Marks),
          asserta(result(Name,Sub,Marks)),save("results.txt").

 read1:-retract(result(Sname,Ssubj,Smarks)),

          write(Sname),nl, write(Ssubj),nl,write(Smarks),nl,fail.

 delete1(X):-retract(result(X,_,_)),

                  save("results.txt"),nl.

 update1:-readln(X),retract(result(X,_,_)),

             readln(Y),readint(Z), asserta(result(X,Y,Z)), save("results.txt").

 update2(X):-retract(result(X,_,_)),readln(Y),readint(Z),

               asserta(result(X,Y,Z)), save("results.txt").
```

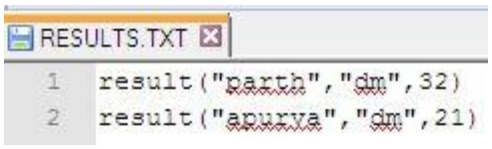
**Output trace for Sample Code II**:


RESULTS.TXT

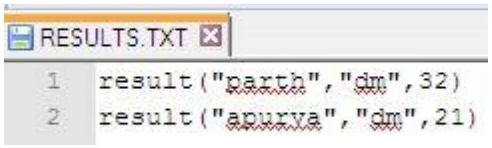
Goal: open1
Goal: write1
apurva
dm
21


RESULTS.TXT
```
result("apurva","dm",21)
```

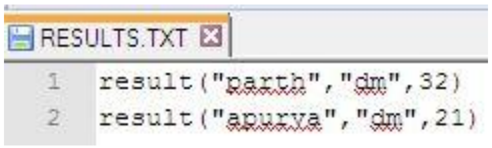
Goal: write1
parth
dm
32


RESULTS.TXT
```
result("parth","dm",32)
result("apurva","dm",21)
```

Goal: read1
parth
dm
32
apurva
dm
21


RESULTS.TXT
```
result("parth","dm",32)
result("apurva","dm",21)
```

Goal: read1
No


RESULTS.TXT
```
result("parth","dm",32)
result("apurva","dm",21)
```

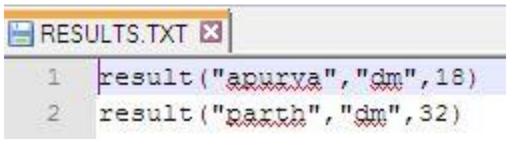
Goal: open1
Goal: read1
parth
dm
32
apurva
dm
21
[*Got the point? File data are fetched into dynamic database and once read, vanished from database, so need to load file again*☺]

Goal: open1

Goal:update2(apurva)

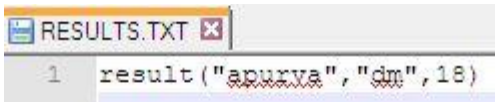
dm

18



```
result("apurva","dm",18)
result("parth","dm",32)
```

Goal: delete1(parth)



```
result("apurva","dm",18)
```

Sample Code III: Code for ProLog Program of searching a students data when Name or a phone no is input in Artificial Intelligence

```
domains
   name,address = symbol
   phone = string
   l = integer*
predicates
   start
   repeat
   selectItem(integer)
   studentData
   subjectL(l)
   searchByName(name)
   searchByPhone(phone)
database
   studentDB(name,address,phone,l)
goal
   clearwindow,
   makewindow(1,7,7,"Search Student Detail",0,0,25,80),
   start.
```

```
clauses
  repeat.
  repeat:-
      repeat.
  start:-
    repeat,
    write("\n0.Exit"),
    write("\n1.Enter student data"),
    write("\n2.Search by Name"),
    write("\n3.Search by Phone number"),
    write("\n4.Show all Student Data"),
    write("\nEnter your choice::"),
    readint(Choice),
    selectItem(Choice),
    Choice=0.
  selectItem(0).
  selectItem(1):-
    studentData,
    fail.
  selectItem(2):-
    write("\nEnter your name::"),
    readln(Name),
    searchByName(Name),
    fail.
  selectItem(3):-
    write("\nEnter the phone no::"),
    readln(Phone),
    searchByPhone(Phone),

    fail.

selectItem(4):-

    studentDB(Name,Address,Phone,Marks),

    write(Name,"  ",Address,"  ",Phone,"  ",Marks),nl,

    fail.

studentData:-

    write("\nEnter the name of the student::"),

    readln(Name),

    write("\nEnter the address of the student::"),

    readln(Address),

    write("\nEnter the phone number of the student::"),

    readln(Phone),

    write("\nEnter the five subject marks of the student"),

    subjectL(Marks),

    assert(studentDB(Name,Address,Phone,Marks)).

subjectL(Marks):-

    write("\nC ::"),

    readint(C),

    write("\nC++ ::"),

    readint(CC),

    write("\nVB ::"),

    readint(VB),

    write("\nJAVA ::"),

    readint(Java),

    write("\nPROLOG ::"),

    readint(Prolog),
    Marks=[C,CC,VB,Java,Prolog].

searchByName(Name1):-

    studentDB(Name1,Address,Phone,Marks),

    write("\nName::",Name1),

    write("\nAddress::",Address),

    write("\nPhone::",Phone),

    write("\nMarks[C,C++,VB,Java,Prolog]::",Marks).

searchByPhone(Phone1):-

    studentDB(Name,Address,Phone1,Marks),

    write("\nName::",Name),

    write("\nAddress::",Address),

    write("\nPhone::",Phone1),

    write("\nMarks[C,C++,VB,Java,Prolog]::",Marks).
```

## Exercises

1. Write a prolog program to create a game like “KBC” using dynamic database and compound objects, use file to store data.
2. Write a prolog program to create application like “marriage bureau” using dynamic database and compound objects, use file to store data.