# DVA-265 Artificial Intelligence 2 LAB - 02

### Khalid Hasan Ador and Pauline Laval

**Q1:** Add one more bird category at the level of 'laysun\_albatross', 'black\_footed\_albatross', and 'rumpeter\_swan'. You can choose a Swedish/European bird, or any other bird from the American continent.

Remember that your task is to implement an expert system, think about the current expert system, and decide if you can add a Swedish or European bird easily, or if the current implementation only allows adding American birds.

=> The current expert system uses logical rules to infer a bird species based on its family, order, and traits (like color, voice, feet, etc.). It is structured hierarchically as:

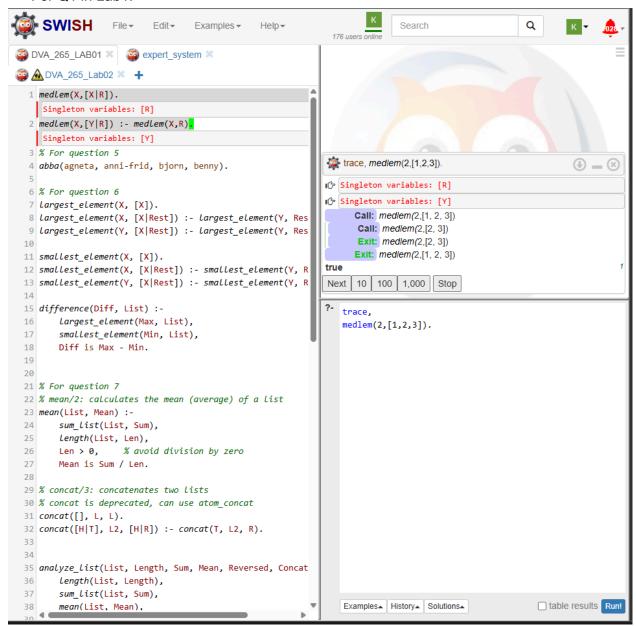
- bird/1 (species-level)
- family/1
- order/1
- traits (attributes used in rules and user queries)

Here, I can add Swedish/European birds, but I'll need to describe them using the traits my system currently uses (color, bill, wings, feet, voice, etc.).

So I added the new bird category (mute\_swan) -> same Swan family, but the voice trait is different.

# Q2: trace

#### => For Q1 in Lab1:



## Q3:

- 1) findall(Template, Goal, Bag):findall(Template, Goal, Bag, [])
  - Template: what to collect (e.g., a variable)
  - Goal: the predicate to solve
  - Bag: resulting list of all matching Template values
  - Succeeds with an empty list if the Goal has no solutions.
- 2) bagof(Template, Goal, List).
  - Like findall, but groups answers by free variables
  - Fails if the goal has no solution
- 3) setof(Template, Goal, List).
  - Like bagof/3, but results are sorted and duplicate-free

# Q4:

Unification is the fundamental operation in Prolog that attempts to make two terms equal by finding a suitable substitution (i.e., binding variables so they match).

The screenshot below shows a trace session in **SWI-Prolog**, where we are calling:

```
?- trace, bird(trumpeter_swan).
```

#### 1) Unifying the Query with a Rule Head

```
We have:

bird(trumpeter_swan):-

family(swan),

voice(loud_trumpeting).
```

Prolog matches the query bird(trumpeter\_swan) with this rule head. This is unification—Prolog sees that the query exactly matches the head of a rule, so it proceeds to evaluate the rule's body.

### 2) Resolving Subgoal: family(swan)

```
Prolog tries to unify the family(swan) with this rule:
family(swan):-
order(waterfowl),
neck(long),
color(white),
flight(ponderous).
```

Since the goal and rule head match (family(swan)), unification succeeds again, and Prolog moves into resolving its body.

## 3) Resolving Sub-subgoal: order(waterfowl)

```
This unifies with:

order(waterfowl):-

feet(webbed),

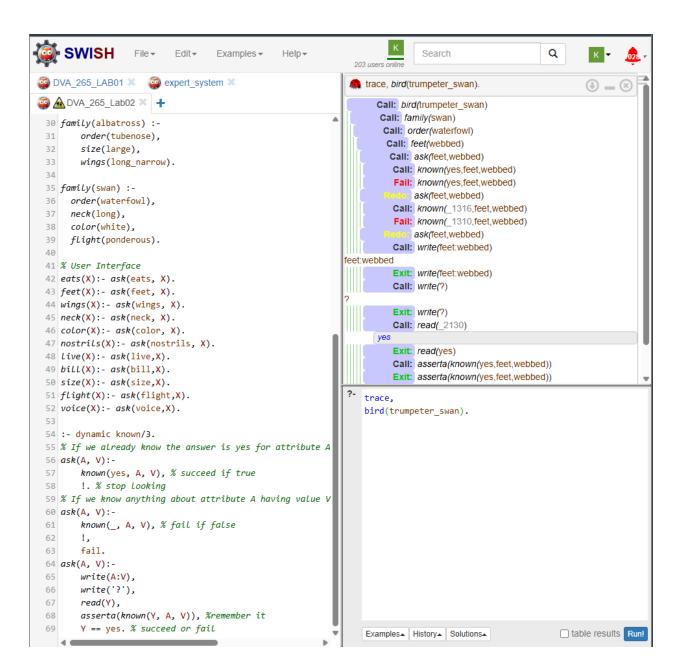
bill(flat).
```

Again, unification matches the head, and Prolog checks each trait.

At every step, Prolog tries to match:

- A goal (like feet (webbed))
- With a rule head or known fact

If it finds a match (or can bind variables to make a match), **unification succeeds**, and execution continues. If not, Prolog **backtracks** to try other options.



```
К
SWISH
                                                                                                               Q
                              Edit≠
                                       Examples -
                                                                                   Search
                                                                     198 users online
                                                                             EXIL. redu(yes)
 🔯 DVA_265_LAB01 🗱 🛛 expert_system 💥
                                                                             Call: asserta(known(yes,feet,webbed))
                                                                             Exit: asserta(known(yes,feet,webbed))
 Call: yes==yes
   1 bird(laysan albatross):-
                                                                             Exit: yes==yes
          family(albatross),
                                                                            Exit: ask(feet, webbed)
          color(white).
                                                                            Exit: feet(webbed)
                                                                           Call: bill(flat)
   5 bird(black footed albatross):-
                                                                  bill:flat?
   6
         family(albatross),
          color(dark).
                                                                           Exit: bill(flat)
   8
                                                                           Exit: order(waterfowl)
   9 bird(whistling_swan):-
                                                                           Call: neck(long)
  10
          family(swan),
                                                                  neck:long?
  11
          voice(muffled musical whistle).
  12
                                                                          Exit: neck(long)
  13 bird(trumpeter_swan):-
                                                                          Call: color(white)
          family(swan).
  14
                                                                   color:white?
  15
          voice(loud trumpeting).
  16
                                                                           Exit: color(white)
  17 bird(mute_swan):-
                                                                          Call: flight(ponderous)
         family(swan),
                                                                  flight:ponderous?
  19
          voice(quite).
                                                                         yes
  20
                                                                          Exit: flight(ponderous)
  21 order(tubenose):-
                                                                          Exit: family(swan)
          nostrils(external_tubular),
                                                                          Call: voice(loud_trumpeting)
  23
          live(at_sea),
                                                                           Call: ask(voice,loud_trumpeting)
         bill(hooked).
  24
                                                                           Call: known(yes,voice,loud trumpeting)
  25
                                                                            Fail: known(yes,voice,loud_trumpeting)
  26 order(waterfowl) :-
                                                                         Redo: ask(voice,loud trumpeting)
          feet(webbed),
  27
                                                                           Call: known(_1312,voice,loud_trumpeting)
  28
          bill(flat).
                                                                            Fail: known(_1318,voice,loud_trumpeting)
  29
                                                                            ask(voice,loud_trumpeting)
  30 family(albatross) :-
                                                                           Call: write(voice:loud trumpeting)
  31
         order(tubenose),
                                                                   voice:loud_trumpeting
  32
          size(large),
                                                                           Exit: write(voice:loud_trumpeting)
  33
         wings(long_narrow).
                                                                           Call: write(?)
  34
  35 family(swan) :-
                                                                            Exit: write(?)
  36 order(waterfowl),
                                                                           Call: read(_1172)
  37
       neck(long),
  38
       color(white),
                                                                           Exit: read(yes)
  39
       flight(ponderous).
                                                                   true
```