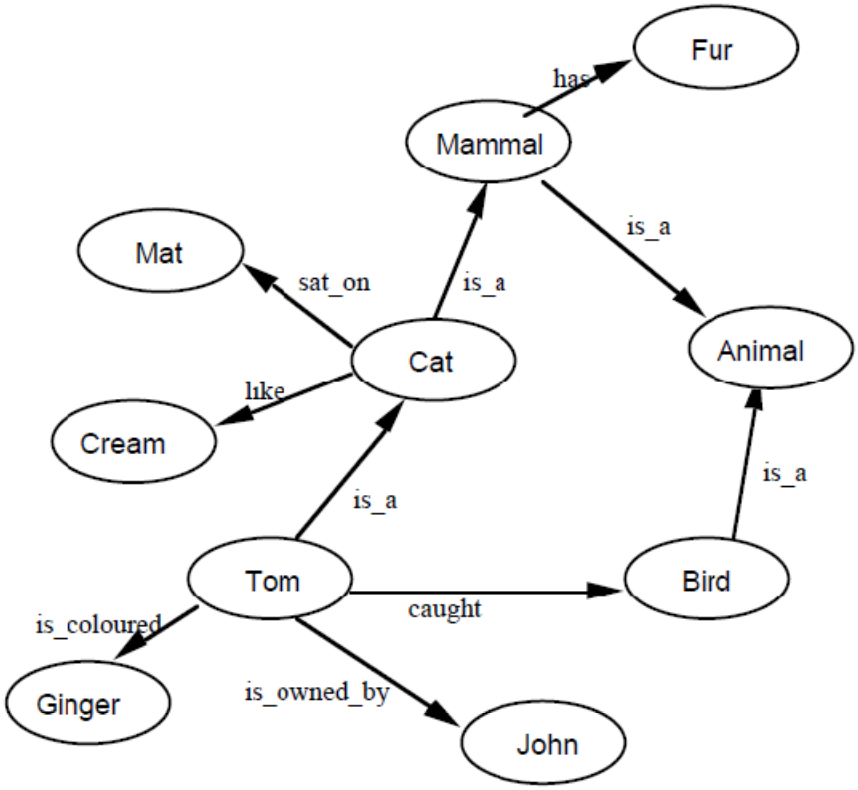
Praktikum Representasi Pengetahuan

Semantic Net dan Frame

Semantic Network

Semantic Network merupakan cara lain untuk merepresentasikan pengetahuan. Ide dasarnya adalah bagaimana dapat menyimpan pengetahuan dalam bentuk grafik, dengan node yang mewakili objek-objek di dunia nyata dan link yang mewakili hubungan antara objek. Sebagai contoh adalah sebagai berikut :



Semantic Network diatas mewakili data :

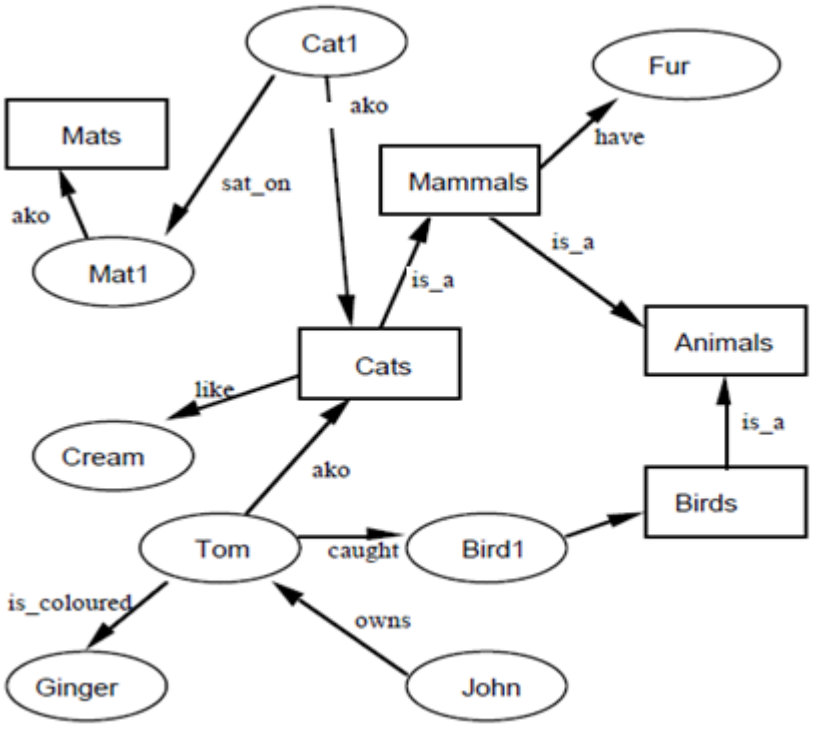
Tom is a cat. *Tom caught a bird.* *Tom is owned by John.* *Tom is ginger in colour.* *Cats like cream.* *The cat sat on the mat.* *A cat is a mammal.* *A bird is an animal.* *All mammals are animals.* *Mammals have fur.*

Node **Cat** diatas menyatakan "cat who sat on the mat" dan semua class cat adalah mamalia dan suka cream(which are mammals and which like cream). Relasi **is_a** bisa mempunyai dua makna yang berbeda yaitu sebuah individu/objek merupakan anggota dari class tersebut contoh Tom merupakan anggota dari class kucing atau sebuah class merupakan subset dari class lainnya, contoh class kucing merupakan subset dari class mamalia. Kebingungan ini tidak terjadi di logika, karena penggunaan quantifiers, nama dan predikat membuat jelas apa yang dimaksud sehingga:

Tom is a cat is represented by `Cat(Tom)` *The cat sat on the mat* is represented by $\exists x \exists y (Cat(x) \wedge Mat(y) \wedge SatOn(x,y))$ *A cat is a mammal* is represented by $\forall x (Cat(X) \rightarrow Mammal(x))$

Untuk membedakan dua maksud dari relasi is_a yaitu node yang menyatakan individu/objek atau node yang menyatakan class, maka relasi :
is_a : sebuah class merupakan subset dari class lainnya (inheritance)
a_kind_of (ako) / instance : untuk menyatakan individu/objek

Relasi is_owned_by hindari penggunaan relasi pasif gunakan relasi aktif, sehingga "Tom is owned by John" menjadi john owns Tom, jangan lupa untuk mengubah arah dari link.



Praktikum 1 : Bentuk representasi predicate dari semantic network

```
cat(tom).
cat(cat1).
mat(mat1).
sat_on(cat1,mat1).
bird(bird1).
```

```
caught(tom,bird1).
like(X,cream) :- cat(X).
mammal(X) :- cat(X).
has(X,fur) :- mammal(X).
animal(X) :- mammal(X).
animal(X) :- bird(X).
owns(john,tom).
is_coloured(tom,ginger).
```

Relasi `a_kind_of` antara class c dan individu/objek m dari class tersebut dinyatakan dengan fakta c(m). Relasi `is_a` antara subclass c dan superclass s dinyatakan dengan s(X) :- c(X)

Cobalah dengan pertanyaan di bawah ini !

Siapa yang menyukai cream ?

```
?- like(X,cream).
```

Siapa saja yang merupakan mamalia ?

```
?- mammal(X).
```

Siapa saja yang mempunyai bulu ?

```
?- has(X,fur).
```

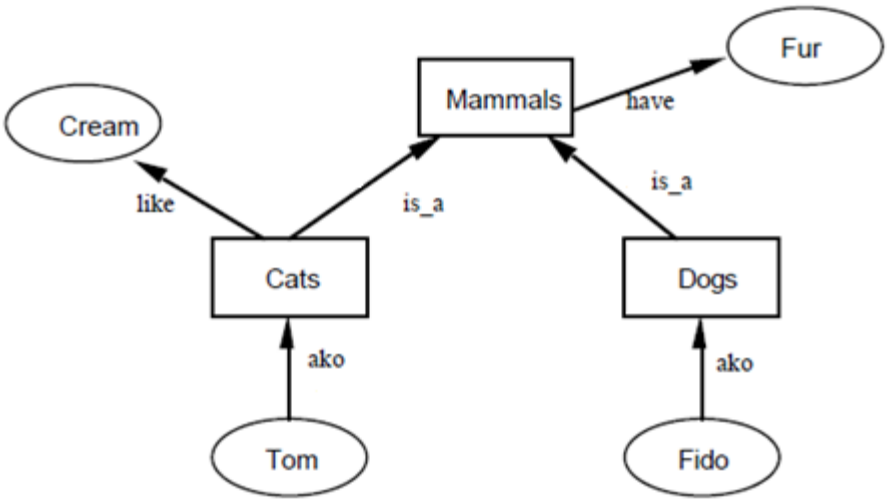
Siapa saja yang merupakan animal?

```
?- animal(X).
```

Inheritance

Konsep Inheritance (relasi a_kind_of) adalah semua objek yang berasal dari class tersebut, maka mewarisi semua property yang dimiliki oleh class tersebut. Sebagai contoh relasi `likes` antara cats dan cream berarti “semua cat suka cream”, termasuk Tom dan Cat1 (relasi `a_kind_of`). Namun relasi `is_coloured` antara Tom dan ginger (bukan cats dan ginger) menunjukkan bahwa `ginger` adalah property Tom sebagai individu bukan berlaku untuk semua cat.

Inheritance juga berlaku untuk relasi `is_a`. Contoh semua property mammal atau animal secara otomatis menjadi property cats. Dari semantic networks diatas menunjukkan bahwa Tom mempunyai fur karena Tom adalah cat, cat merupakan mamal dan mamal mempunyai fur. Sedangkan subclass mammals adalah Dog dan Fido merupakan dog, Fido mewarisi property mempunyai fur dari mamal, tapi tidak memiliki property likes cream, property ini hanya untuk cat.



Reification

Kita dapat merepresentasikan semantic network menggunakan Prolog. Kita dapat menyatakan setiap edge pada semantic network menjadi fakta dengan nama predicate sama seperti label pada edge. Node-node pada graph (menyatakan objek/individu atau class) menjadi argument fakta. Bentuk representasi graph diatas dalam bahasa Prolog :

Praktikum 2

```
a_kind_of(mat1,mats).
a_kind_of(cat1,cats).
a_kind_of(tom,cats).
a_kind_of(bird1,birds).
caught(tom,bird1).
is_a(cats,mammals).
is_a(mammals,animals).
is_a(birds,animals).
like(cats,cream).
owns(john,tom).
sat_on(cat1,mat1).
is_coloured(tom,ginger).
have(mammals,fur).

subclass(Class1,Class2) :- is_a(Class1,Class2).
subclass(Class1,Class2) :- is_a(Class1,Class3), subclass(Class3,Class2).

aninstance(Obj,Class) :- a_kind_of(Obj,Class).
aninstance(Obj,Class) :- a_kind_of(Obj,Class1), subclass(Class1,Class).

attribute(Obj,X) :- aninstance(Obj,Class),have(Class,X).
attribute(Obj,X) :- aninstance(Obj,Class),like(Class,X).
```

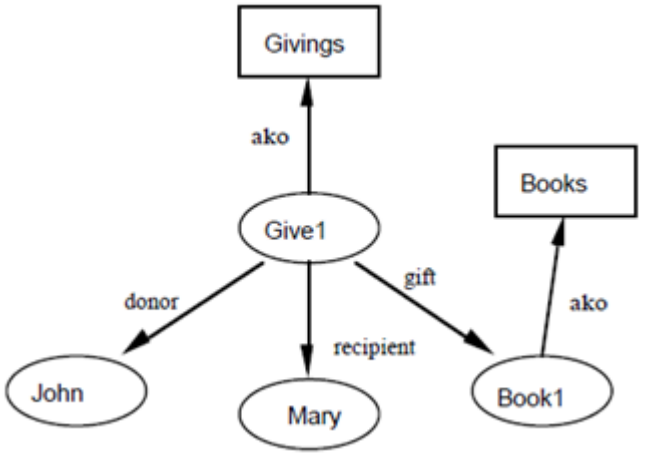
Proses mengubah predikat/relasi menjadi objek dalam sistem representasi pengetahuan disebut **reification.** Symbol cats menyatakan himpunan semua kucing, yang dianggap sebagai objek lain.

Berikan pertanyaan atau query ke system yang telah dibuat !

```
?- attribute(X,Y).
```

Bagaimana outputnya ? Apakah telah menerapkan konsep inheritance ?

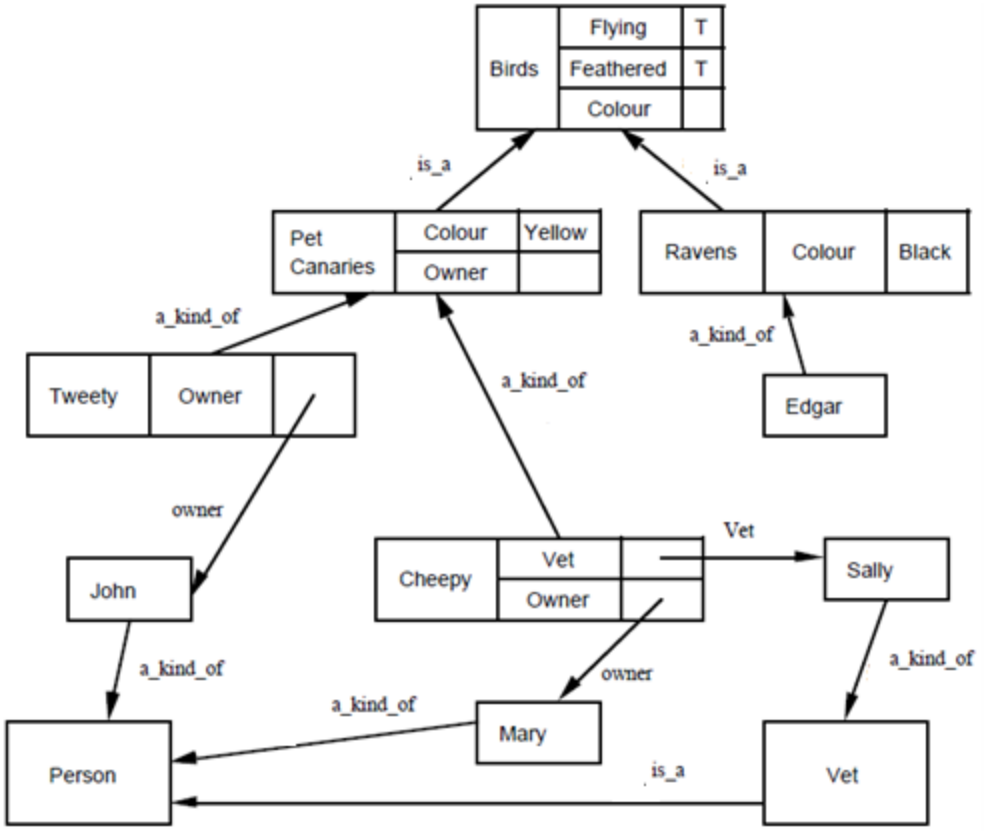
Pada contoh sebelumnya, telah direpresentasikan relasi binary, bagaimana dengan relasi lebih dari dua argument ? Contoh bagaimana merepresentasikan "John gave the book to Mary" ? di predicate logic, kita dapat membuat 3-ary yaitu gave(John, Book1, Mary), argument 1 adalah pemilik, argument 2 adalah objek yang diberikan dan argument 3 adalah orang yang diberi. Pada contoh ini terdapat proses reification yaitu mengubah relasi gave menjadi objek give1.



Frames, Slots and Fillers

Dalam contoh ini terdapat general class `birds` dan semua bird mempunyai atribut `flying`, `feathered` dan `colour`. Atribut `flying` dan `feathered` mempunyai nilai Boolean dan diset dengan true, yang berarti semua bird mempunyai atribut `flying` true dan atribut `feathered` true. Attribut `colour` dalam hal ini tidak diisi yang berarti semua bird mempunyai warna sendiri, warnanya dapat bervariasi. Dua subclass bird `pet_canaries` mempunyai `colour` yellow dan `ravens` mempunyai `colour` black. Class `pet_canaries` mempunyai slot tambahan `owner` yang berarti semua `pet canary` mempunyai owner/pemilik. Dengan kata lain semua instance dari class `pet_canary` mempunyai atribut `colour` yellow, `feathered` true, `flying` true dan `owner`nya bervariasi untuk setiap instance. Setiap instance/objek dari class `raven` mempunyai `colour` black, `feathered` true, `flying` true tetapi tidak mempunyai atribut `owner`. Dua instance dari `pet_canary` yaitu `Tweety` dan `Cheepy` mempunyai `owner` John dan Mary yang merupakan instance dari class person. Instance `pet_canary` Cheepy mempunyai atribut dibatasi oleh class `vet`(dokter hewan), yang merupakan instance dari class `Person`.

Representasi framenya sebagai berikut :



Dari frame diatas, representasi bahasa Prolog adalah sebagai berikut:

Praktikum 3 :

```
attribute(birds,flying,true).
attribute(birds,feathered,true).
attribute(pet_canaries,colour,yellow).
attribute(ravens,colour,black).
attribute(tweety,owner,john).
attribute(cheepy,owner,mary).
attribute(cheepy,vet,sally).
is_a(pet_canaries,birds).
is_a(ravens,birds).
is_a(vet,person).

a_kind_of(edgar,ravens).
a_kind_of(tweety,pet_canaries).
a_kind_of(cheepy,pet_canaries).
a_kind_of(sally,vet).
a_kind_of(john,person).
a_kind_of(mary,person).
```

Kita definisikan aturan umum system :

```
subclass(Class1,Class2) :- is_a (Class1,Class2).
subclass(Class1,Class2) :- is_a (Class1,Class3), subclass(Class3,Class2).
```

```
aninstance(Obj,Class) :- a_kind_of(Obj,Class).
aninstance(Obj,Class) :- a_kind_of(Obj,Class1), subclass(Class1,Class).

value(Obj,Property,Value) :- attribute(Obj,Property,Value).
value(Obj,Property,Value):- aninstance(Obj,Class),
attribute(Class,Property,Value).
```

Selanjutnya lakukan pertanyaan seperti di bawah ini :

Sebutkan semua instance X adalah Y !
```
?- aninstance(X,Y)
```

Sebutkan semua subclass X adalah Y !
```
?- subclass(X,Y)
```

Sebutkan value Z dari objek X dan Properti Y !
```
?- value(X,Y,Z)
```

Sebutkan value dari colour tweety !
```
| ?- value(tweety,colour,V).
```

Sebutkan value dari feather john ?
```
| ?- value(john,feathered,V).
```

Apakah atribut feathered dari person adalah true?
```
attribute(person,feathered,true).
```

Demons and Object-Oriented Programming

Pada representasi frame, nilai pada slot/atribut selain dapat diisi dengan nilai tertentu, dapat diisi dengan sebuah procedure untuk menghitung nilai dari beberapa atribut. Prosedur ini disebut dengan demon. Sebagai contoh, kita ingin menambahkan atribut maintenance yang merupakan harga maintenance yang digunakan untuk subclass `pet_canaries`, akan mengembalikan nilai 5 untuk pet canary sendiri tanpa biaya dokter hewan(vet), sedangkan untuk pet_canary yang memiliki dokter hewan terkena biaya 5 + biaya dokter hewan yaitu 25.

Pengenalan variable "anonymous" ditulis dengan _

Missal terdapat aturan

```
hasachild(X) :- parent(X,Y).
```

maksud dari aturan diatas adalah X mempunyai anak (child) jika X adalah orangtua (parent) dari Y. Properti hasachild tidak tergantung dengan nama child (ditulis dengan Y), sehingga kita dapat menggunakan variable "anonymous". Bentuk lain dari aturan diatas adalah :

```
hasachild(X) :- parent(X,_).
```

Contoh lain

Seseorang yang mempunyai child jika terdapat dua object dimana satu orang sebagai parent yang lain sebagai anak.

`somebody_has_child :- parent(_,_).` Sama dengan

```
somebody_has_child :- parent(X,Y).
```

Tambahkan fakta prolog sebagai berikut :

Praktikum 4 :

```
attribute(_,birds,flying,true).
attribute(_,birds,feathered,true).
attribute(_,pet_canaries,colour,yellow).
attribute(_,ravens,colour,black).
attribute(_,tweety,owner,john).
attribute(_,cheepy,owner,mary).
attribute(_,cheepy,vet,sally).
attribute(_,sally,fees,20).

is_a(pet_canaries,birds).
is_a(ravens,birds).
is_a(vet,person).

a_kind_of(edgar,ravens).
a_kind_of(tweety,pet_canaries).
a_kind_of(cheepy,pet_canaries).
a_kind_of(sally,vet).
a_kind_of(john,person).
a_kind_of(mary,person).

aninstance(Obj,Class) :- a_kind_of(Obj,Class).
aninstance(Obj,Class) :- a_kind_of(Obj,Class1), subclass(Class1,Class).

subclass(Class1,Class2) :- is_a(Class1,Class2).
subclass(Class1,Class2) :- is_a(Class1,Class3), subclass(Class3,Class2).

value(Obj,Property,Value) :- attribute(Obj,Obj,Property,Value).
value(Obj,Property,Value) :- aninstance(Obj,Class),

attribute(Obj,Class,Property,Value).

attribute(Self,pet_canaries,maintenance,Costs) :-
eval_maintenance(Self,Costs).

eval_maintenance(Self,Costs) :- value(Self,vet,SelfsVet),
!,value(SelfsVet,fees,VetFees),Costs is VetFees+5.
eval_maintenance(Self,5).
```

Lakukan pertanyaan sebagai berikut ?

```
?- value(sally,fees,X).
?- value(pet_canaries, maintenance,X).
```

```
?- value(tweety, maintenance,X).
?- value(cheepy, maintenance, X).
```

Bagaimana outputnya ? Jelaskan

Defaults and Overrides
(Praktikum 5)

Contoh bird dapat terbang, dalam bahasa Prolog

```
Flies(X) :- bird(X).
```

Bagaimana untuk kasus tertentu dimana bird tidak dapat terbang, contoh kiwi dan penguin, termasuk juga bird apapun yang sayapnya rusak juga tidak dapat terbang. Dalam bahasa prolognya adalah sebagai berikut:

```
flies(X) :– bird(X), \+kiwi(X), \+penguin(X), \+broken_wing(X).
```

dapat juga dinyatakan dengan:

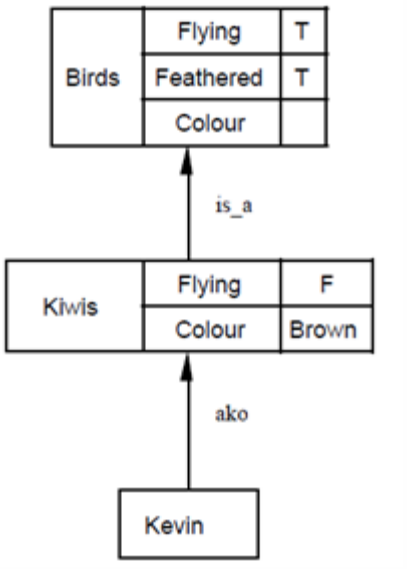
```
flies(X) :– bird(X), \+ab(X).
ab(X) :– kiwi(X).
ab(X) :– penguin(X).
ab(X) :– broken_wing(X).
```

Dimana ab(X), X adalah bird yang abnormal

```
attribute(pet_canaries,colour,yellow).
attribute(ravens,colour,black).
attribute(tweety,owner,john).
attribute(cheepy,owner,mary).
attribute(cheepy,vet,sally).
is_a(pet_canaries,birds).
is_a(ravens,birds).
is_a(vet,person).
is_a(kiwis,birds).
a_kind_of(edgar,ravens).
a_kind_of(tweety,pet_canaries).
a_kind_of(cheepy,pet_canaries).
a_kind_of(sally,vet).
a_kind_of(john,person).
a_kind_of(mary,person).
a_kind_of(kevin,kiwis).
```

Pada umumnya hampir semua burung dapat terbang, hanya burung tertentu saja yang tidak dapat terbang. Untuk implementasi contoh ini dapat menggunakan konsep overridden. Atribut pada superclass akan diwarisi oleh subclass, tapi jika nilai atribut tidak sesuai lagi maka bisa diganti (dioverride) dengan nilai yang baru pada subclass.

Kita telah menambahkan atribut colour untuk kiwi



Lakukan query sebagai berikut ganti X =`tweety`, `cheepy` or `edgar`,
`| ?- value(X,flying,V).`
Akan memberikan respon
`V = true ?`
tetapi
`| ?- value(kevin,flying,V).`
Akan memberikan respon
`V = false ?`
`V = true ?`

Untuk mencegah terjadinya hal ini, perlu melakukan cut di aturan(rule), sehingga mencari nilai pada hirarki tree yang paling tinggi dengan property yang sama.

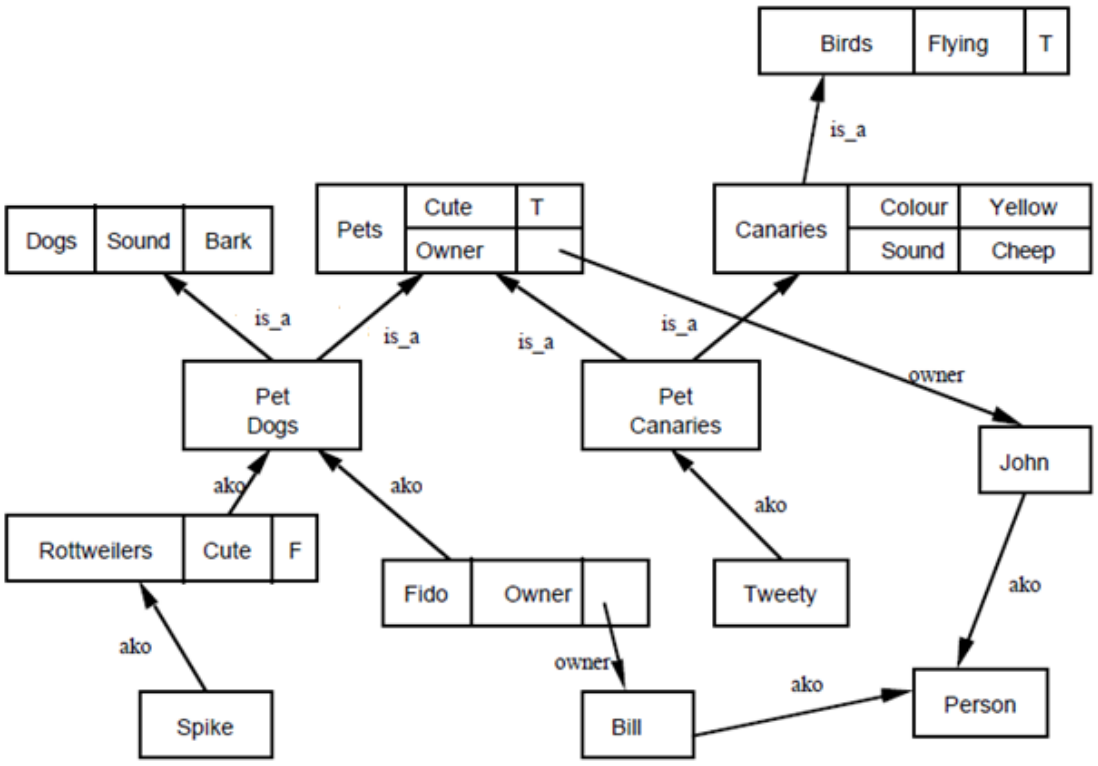
Tambahkan

```
value(Obj,Property,Value) :- attribute(Obj,Property,Value), !.
value(Obj,Property,Value):-
aninstance(Obj,Class), attribute(Class,Property,Value), !.
```

Multiple Inheritance

(Praktikum 6)

Terdapat class pets dan canaries. Class pet canaries mewarisi property pets dan canaries. Pets mempunyai property cute, birds mempunyai property flying dan canaries mempunyai property default coloured yellow, sound cheep. Terdapat class pet dogs, semua dogs mempunyai default property sound bark. Untuk mengilustrasikan overridden, class Rottweilers adalah subclass dari pet_dogs yang memiliki property cute true, yang dioveriden sehingga menjadi cute false. John adalah default pemilik/owner dari pet, sehingga yang tidak memiliki pemilik, maka dianggap pemiliknya adalah John.



Representasi dari diagram diatas dalam bahasa Prolog adalah :

```
attribute(birds,flying,true).
attribute(dogs,sound,bark).
attribute(pets,cute,true).
attribute(pets,owner,john).
attribute(canaries,colour,yellow).
attribute(canaries,sound,cheep).
attribute(rottweilers,cute,false).
attribute(fido,owner,bill).
is_a(canaries,birds).
is_a(pet_canaries,canaries).
is_a(pet_canaries,pets).
is_a(pet_dogs,dogs).
is_a(pet_dogs,pets).
is_a(rottweilers,pet_dogs).
a_kind_of(tweety,pet_canaries).
a_kind_of(spike,rottweilers).
a_kind_of(fido,pet_dogs).
a_kind_of(john,person).
a_kind_of(bill,person).

aninstance(Obj,Class) :- a_kind_of(Obj,Class).
aninstance(Obj,Class) :- a_kind_of(Obj,Class1), subclass(Class1,Class).
subclass(Class1,Class2) :- is_a(Class1,Class2).
subclass(Class1,Class2) :- is_a(Class1,Class3), subclass(Class3,Class2).
value(Obj,Property,Value) :- attribute(Obj,Property,Value),!.
value(Obj,Property,Value) :- aninstance(Obj,Class),
attribute(Class,Property,Value),!.
```

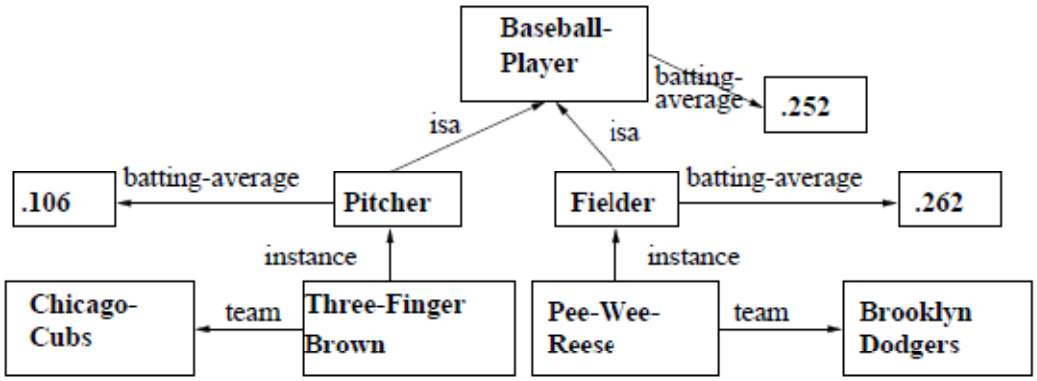
Jalankan program diatas, lakukan query sebagai berikut :

```
| ?- value(fido,sound,S).
S = bark ?
```
Menunjukkan fido mewarisi properti sound bark dari dogs,

```
| ?- value(fido,cute,V).
V = true?
```
menunjukkan fido mewarisi cute : true dari pets,

```
| ?- value(spike,cute,V).
V = false?
```
Menunjukkan property cute : true dari pet_dogs di overridde oleh rottweiler spike.

Mengubah bentuk Semantic Network menjadi Frame



Baseball Player
is-a: Adult Male
batting average: .252
bats: equal to handed
team:
⋮

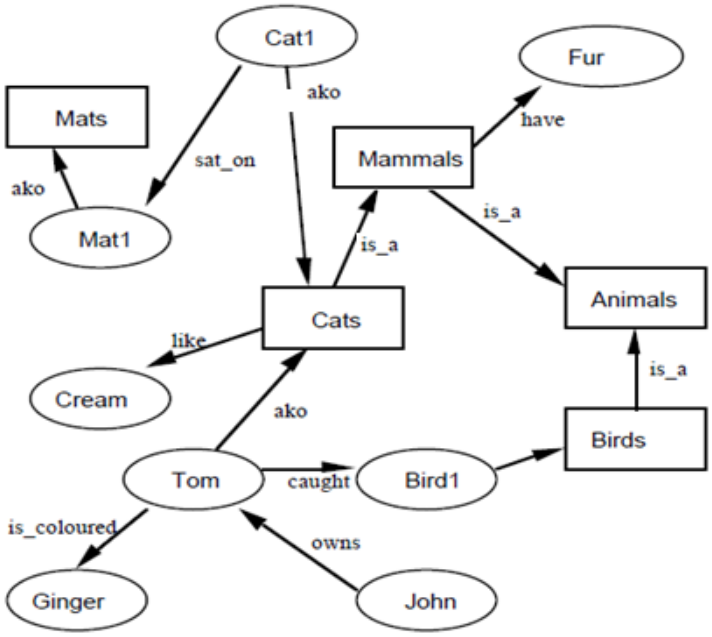
Fielder
is-a: Baseball player
batting average: .262

• • •

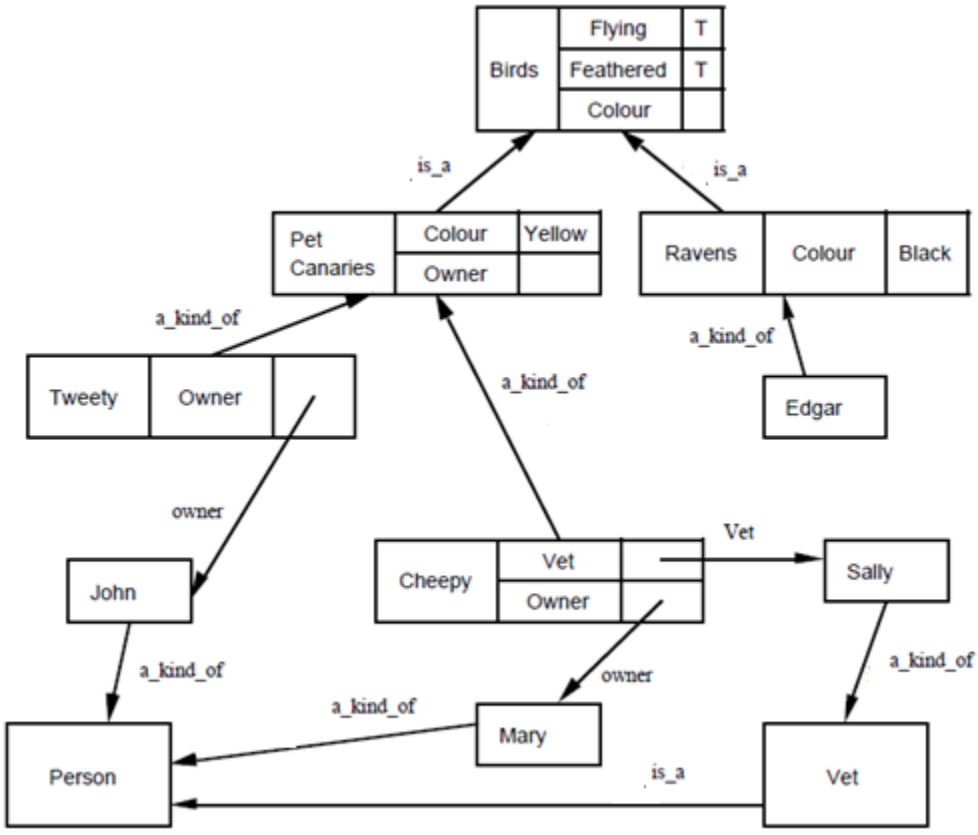
Pee-Wee-Reese
instance: Baseball player
team: Brooklyn Dodgers

Tugas

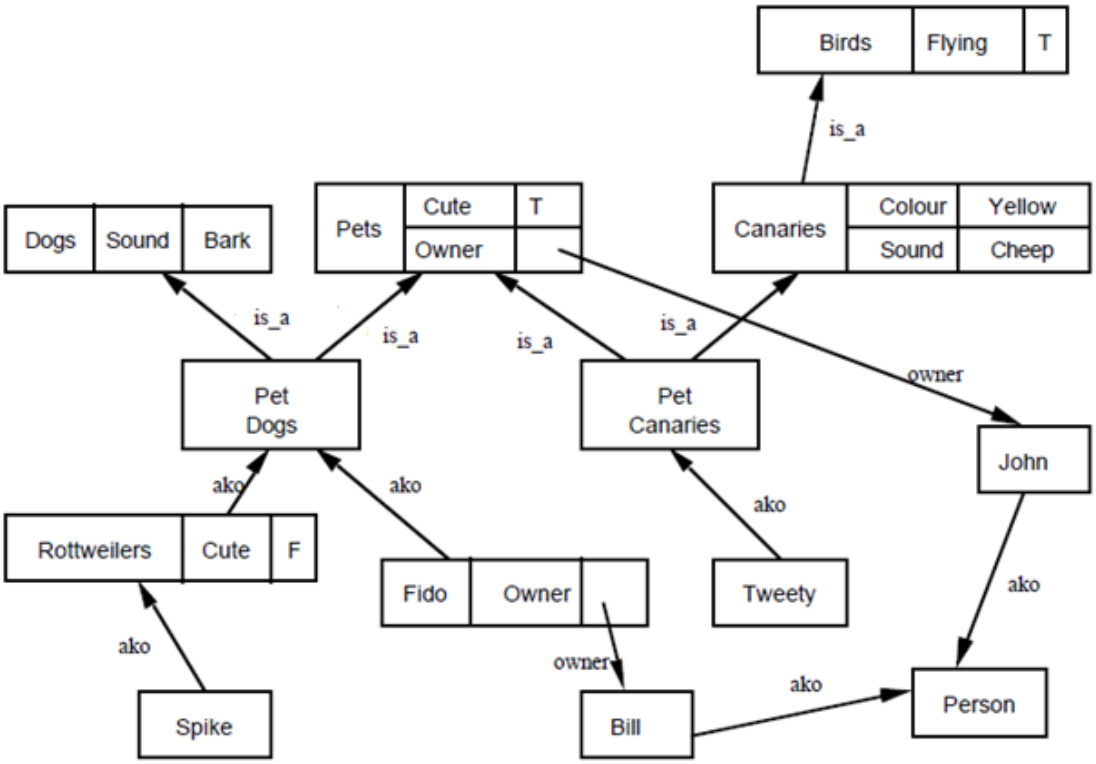
1. Ubahlah studi kasus dibawah ini dari bentuk Semantic Network menjadi Frame



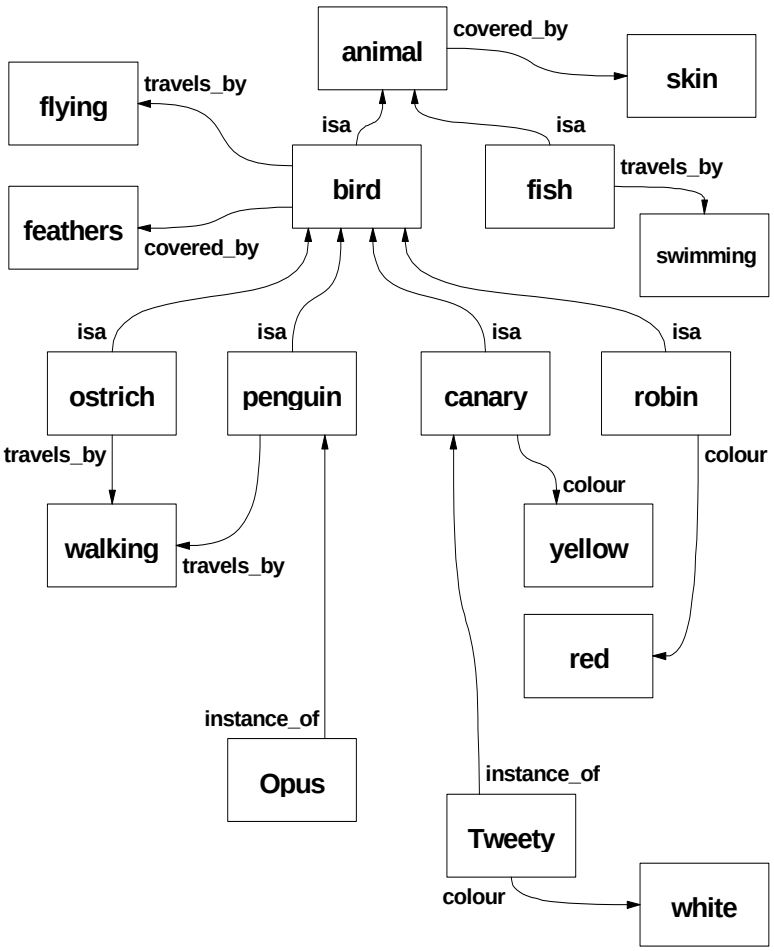
2. Ubahlah studi kasus dibawah ini dari bentuk Frame menjadi Semantic Network



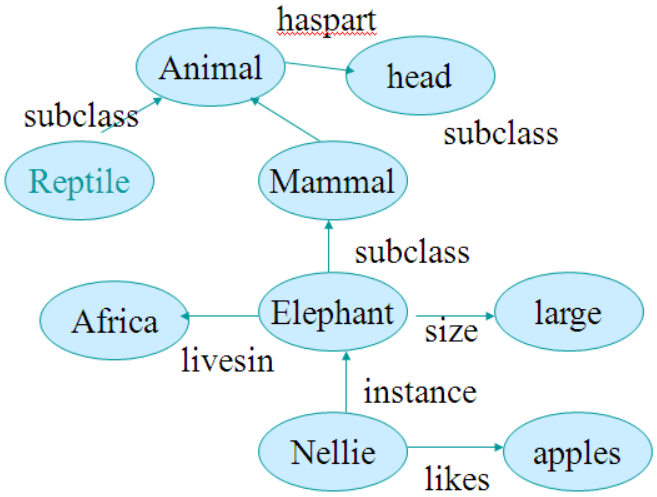
3. Ubahlah studi kasus dibawah ini dari bentuk Frame menjadi Semantic Network



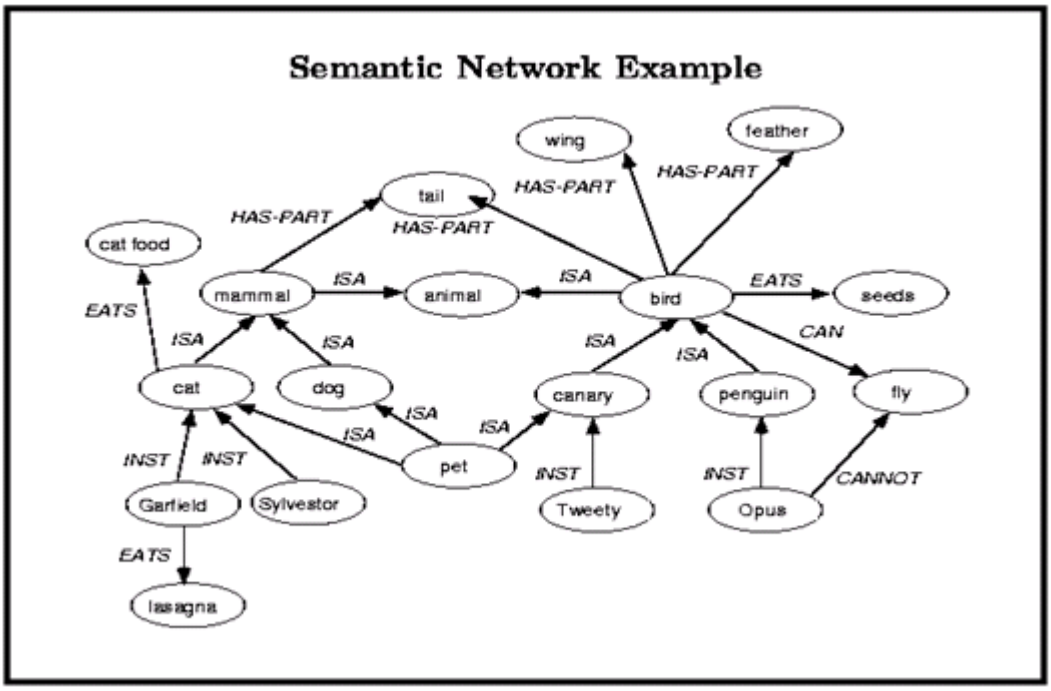
4. Ubahlah studi kasus dibawah ini dari bentuk Semantic Network menjadi Frame



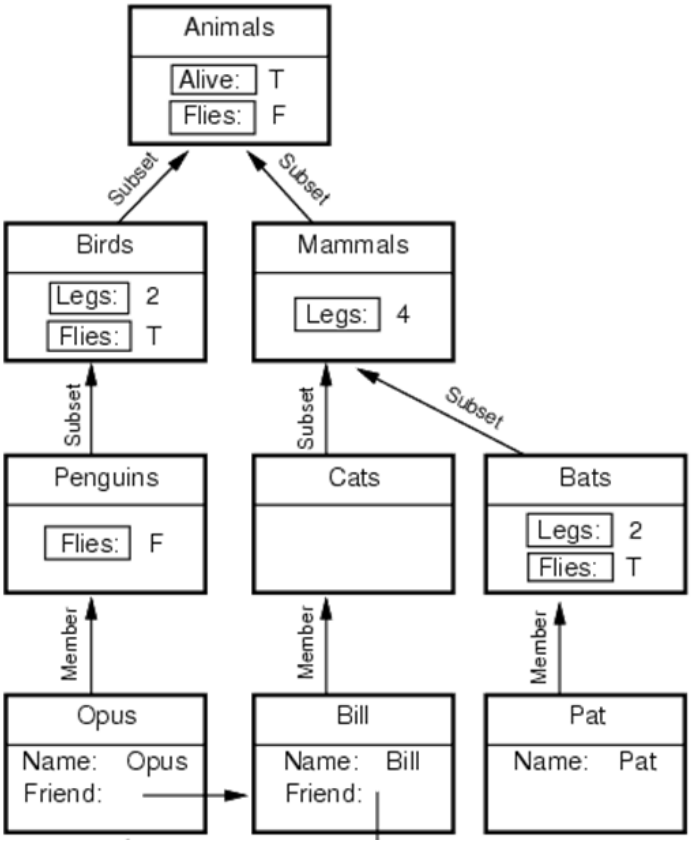
5. Ubahlah studi kasus dibawah ini dari bentuk Semantic Network menjadi Frame



6. Ubahlah studi kasus dibawah ini dari bentuk Semantic Network menjadi Frame



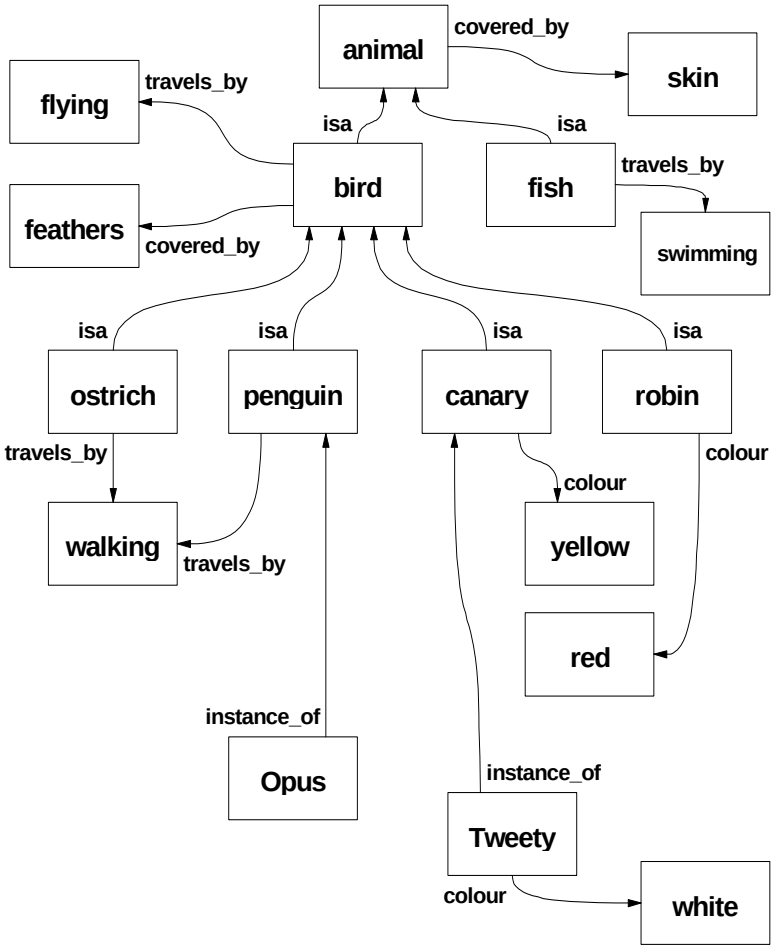
7. Ubahlah studi kasus dibawah ini dari bentuk Frame menjadi Semantic Network



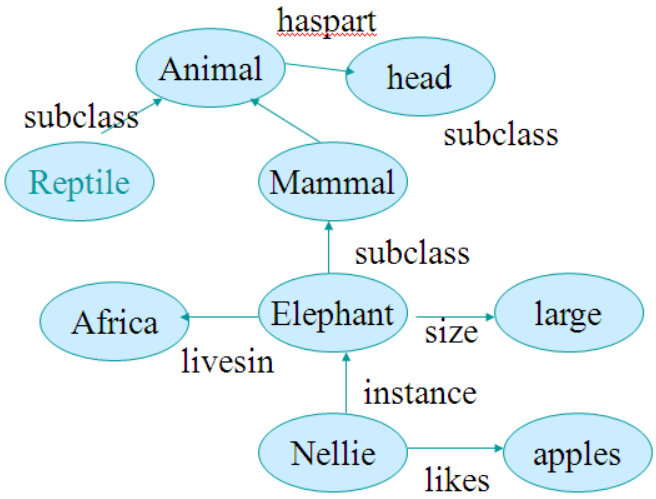
Soal Praktikum

1. Kerjakan praktikum 1 sampai dengan praktikum 6. Lakukan hal-hal di bawah ini :
   a. Berikan penjelasan konsep apa yang dimaksud dalam praktikum tersebut! (dalam menjelaskan gunakan kalimat kalian sendiri).
   b. Lakukan query untuk praktikum 1 – 6 dan jelaskan !

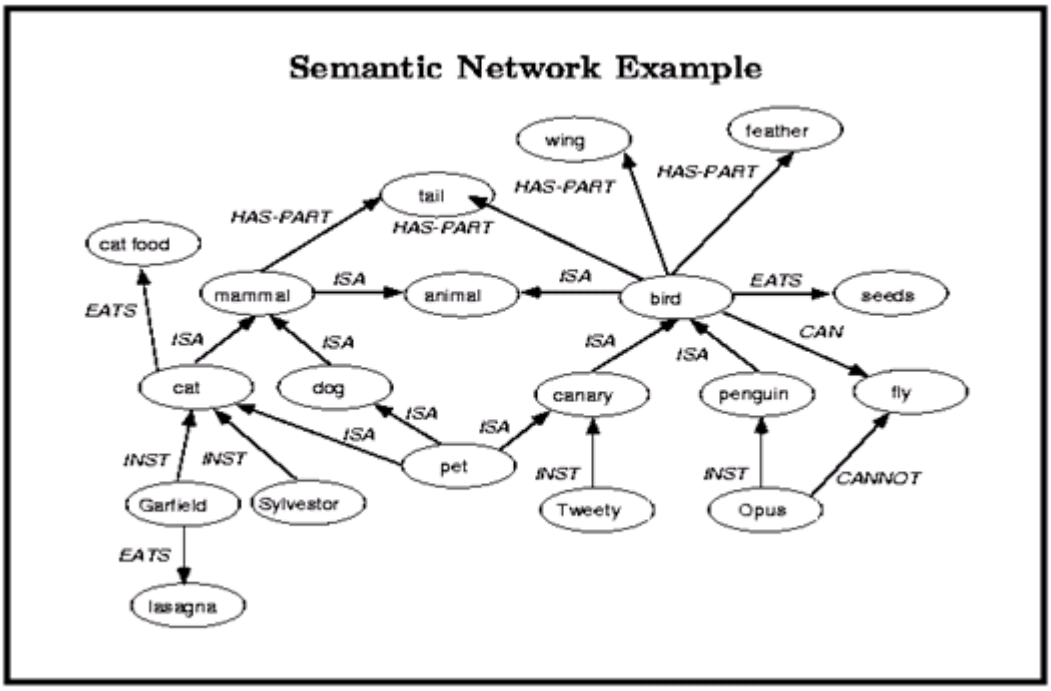
2. Dari diagram di bawah ini,
   a. buatlah dalam bahasa Prolog.
   b. Selanjutnya lakukan query !
   c. Ubahlah menjadi bentuk frame !



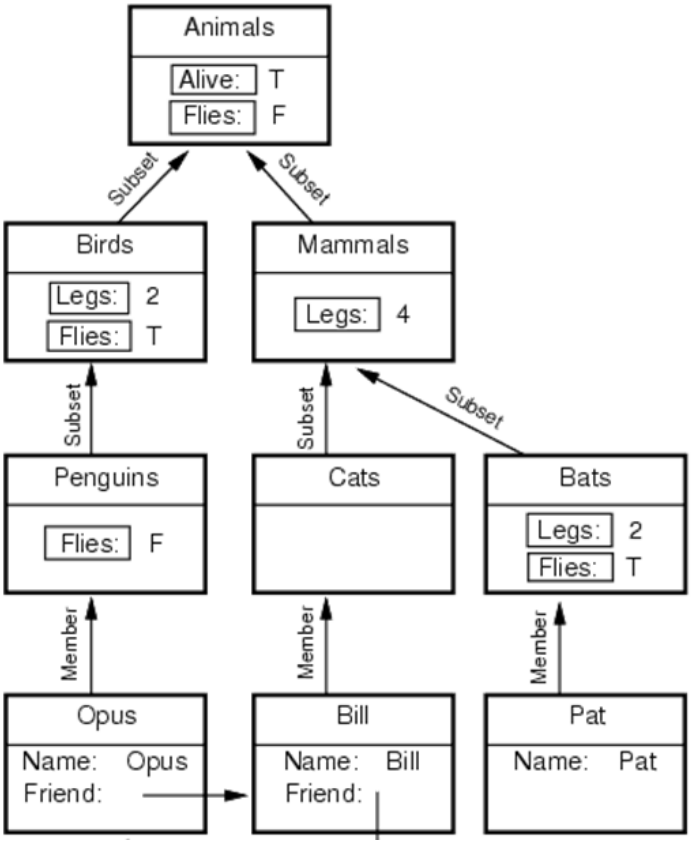
3. Dari diagram di bawah ini
   a. buatlah dalam bahasa Prolog.
   b. Selanjutnya lakukan query !
   c. Ubahlah menjadi bentuk frame !



4. Dari diagram di bawah ini
   a. buatlah dalam bahasa Prolog.
   b. Selanjutnya lakukan query !
   c. Ubahlah menjadi bentuk frame !



5. a. Representasikan dalam bahasa Prolog !
   b. Buatlah pertanyaan ke system yang telah di buat !
   c. Ubahlah menjadi bentuk Semantic Network !


Animals
Alive: T
Flies: F
Subset
Subset
Birds
Legs: 2
Flies: T
Mammals
Legs: 4
Subset
Subset
Subset
Penguins
Flies: F
Cats
Bats
Legs: 2
Flies: T
Member
Member
Member
Opus
Name: Opus
Friend:
Bill
Name: Bill
Friend:
Pat
Name: Pat