Introducción a Prolog

lunes, 6 de diciembre de 2021 22:43

- Su nombre viene de ***PRO***gramación ***LÓgica***
- **Características**
  - Se basa en lógica simbólica y ***programación declarativa***
  - No se especifica ***cómo*** debe hacerse, sino ***qué*** debe lograrse
  - Es un estilo de programación orientado a ***metas***

- **Aplicaciones**
  - Pruebas matemáticas
    - Demostraciones
  - Inteligencia Artificial
    - Sistemas Expertos
  - Consultas de Base de Datos
    - Inferir relaciones no especificadas a priori

Hechos y Consultas

- 

- Preguntar por el abuelo de **aldo**:

$$\exists X, Y : (X \text{ es padre de } Y) \cap (Y \text{ es padre de aldo})$$

- que se expresa en Prolog como:

```
?- padre(X, Y), padre(Y, aldo).
=> X = pedro
   Y = paty
```

- Preguntar por los nietos de **juan**:

$$\exists X, Y : (\text{juan es padre de } X) \cap (X \text{ es padre de } Y)$$

que se expresa en Prolog como:

```
?- padre(juan, X), padre(X, Y).
=> X = pedro
   Y = ana ;

=> X = pedro
   Y = paty
```

- Preguntar si **ana** y **paty** tienen un padre en común:

$$\exists X : (X \text{ es padre de ana}) \cap (X \text{ es padre de paty})$$

que se expresa en Prolog como:

```
?- padre(X, ana), padre(X, paty).
=> X = pedro
```

- Reglas
  - La relación $a \subset b$ se expresa, en Prolog, como ***a :- b***
  - Una cláusula de este tipo se denomina ***regla***, que tiene la siguiente estructura
    - La ***cabeza*** es la ***conclusión*** (parte izquierda de :-)
    - El ***cuerpo*** es la proposición definida)parte derecha de :-)
  - Resolución simple
    - La relación ***hijo de*** corresponde a:

$$\forall X, Y : (Y \text{ es hijo de } X) \subset (X \text{ es padre de } Y)$$

    - que se expresa en Prolog, usando lo definido anteriormente, como:

```
hijo(X, Y) :- padre(Y, X).
```

    - **Ejemplo:** la meta siguiente es evaluada como:

La meta:	hijo(paty, pedro)
se convierte en submeta:	padre(pedro, paty)
Se busca este hecho:	yes

Reglas: ejemplo

- Agregar cláusulas sobre el género de las personas: alternativas.

- 

  - Reglas Recursivas
    - La relación ***antepasado*** se define sobre la base de una regla de descendencia directa y otra regla de descendencia indirecta:

$$\forall X, Z : (X \text{ es un antepasado de } Z), \text{ si}$$
$$[X \text{ es padre de } Z] \cup$$
$$[\exists Y : (X \text{ es padre de } Y) \cap (Y \text{ es antepasado de } Z)]$$

lo que en Prolog se expresa como :

```
antepasado(X, Z) :- padre(X, Z).
antepasado(X, Z) :- padre(X, Y), antepasado(Y, Z).
```

Reglas Recursivas

- 

- Se puede definir ahora varias nuevas reglas como:

```
papa(X, Y)      :- padre(X, Y), masculino(X).
mama(X, Y)      :- padre(X, Y), femenino(X).
abuelo(X, Y)    :- padre(X, Z), padre(Z, Y).
hermana(X, Y)   :- padre(Z, X), padre(Z, Y), femenino(X).
```

- Se puede definir ahora varias nuevas reglas como:

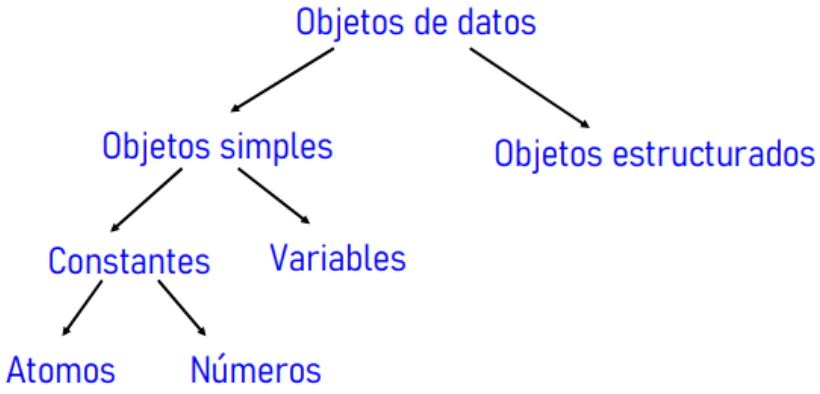
```
?- hermana(ana, paty).
=> yes

?- hermana(X, paty).
   => X = ana ;
   => X = paty        oops ... paty es hermana de ella misma

   ¡Falta excluir este caso:
   hermana(X, Y) :- padre(Z, X), padre(Z, Y),
                    diferente(X, Y), femenino(X).
```

Tipos de Datos en Prolog

jueves, 16 de diciembre de 2021 18:58



- **Objetos Simples**
  - **Reconocimiento de Tipos**
    - Se reconoce el tipo de un dato por su forma sintáctica, es decir no hay que declarar el tipo de dato
    - Ejemplo
      - Las ***variables*** empiezan con una letra mayúscula
      - Los átomos empiezan con una letra minúscula
  - **Formación de Variables y Átomos**
    - Strings formados con una combinación de los siguientes caracteres
      - Letras mayúsculas o minúsculas
      - Dígitos
      - Caracteres especiales
  - **Números**
    - Números enteros
    - Reales, como decimales
  - **Variables**
    - Strings de letras, dígitos y guiones bajos, comenzando con mayúscula o guion bajo
      - X, Res, _X1
    - Si una variable aparece solo una vez en una cláusula se puede reemplazar por una variable anónima *guion bajo*