Data Structure

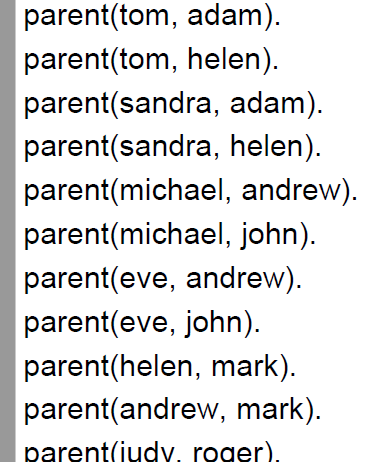
1. Values of data 数据的值
2. Operations that we can have for data 我们可以对数据进行的操作
3. How we represent data 我们如何表现数据

在PROLOG中 DATA STRUCTURE:TERM

有三个要素组成

1. Atom:任何identifier以小写字母开头
2. Number:整数
3. Variable变量：任何Identifier以大写字母开头

Compound term：有一个atom 叫做functor，以及一系列的语句每一个都是一个term



parent(tom, adam ). 记住要小写，加句号，意思是tom是adam的爹

arity:在一个parent(x,x)中参数的数量，parent/2 代表parent这个functor有两个argument factor 参数

单个的语句叫做clause,statement

一共有两种clause：

1. fact()既定事实，我们已经设置了tom是adaM的爹

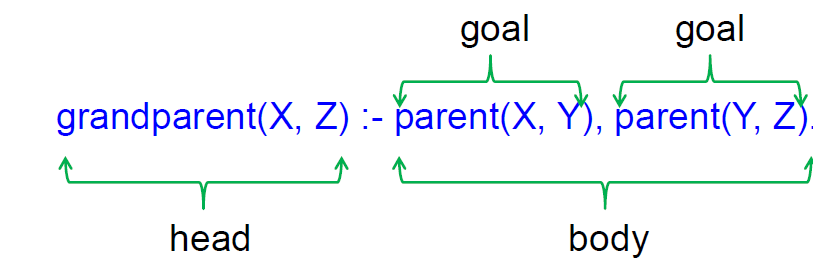
所有的fact()组成database

1. Rule:加于DATABASE上的规则，基于FACT

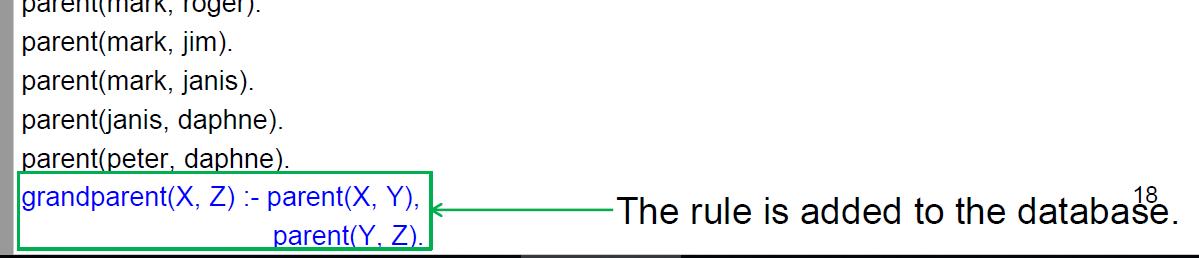
Head：-body.

Conclusion:-condition.

意思是，如果BODY部分成立，那么HEAD的逻辑关系就成立



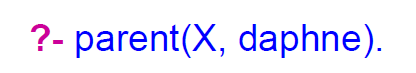
逗号表示并列关系，分好表示或。



如果X是Y的爹，Y是Z，的爹，那么X就是Z的爷

Query质疑

一共有两种query

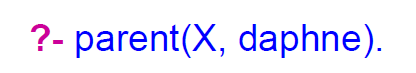
怎么query?-

Ground query



会弹出true of false（PETER是不是DAPHNE的爹）

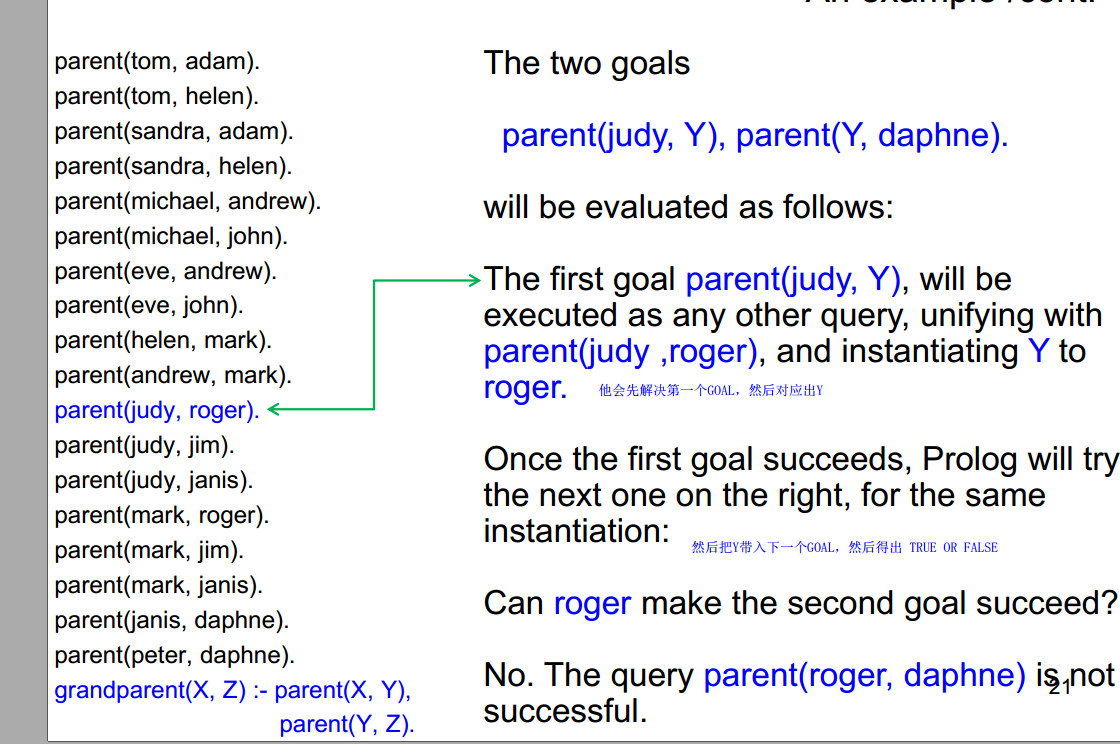
Non-Ground query

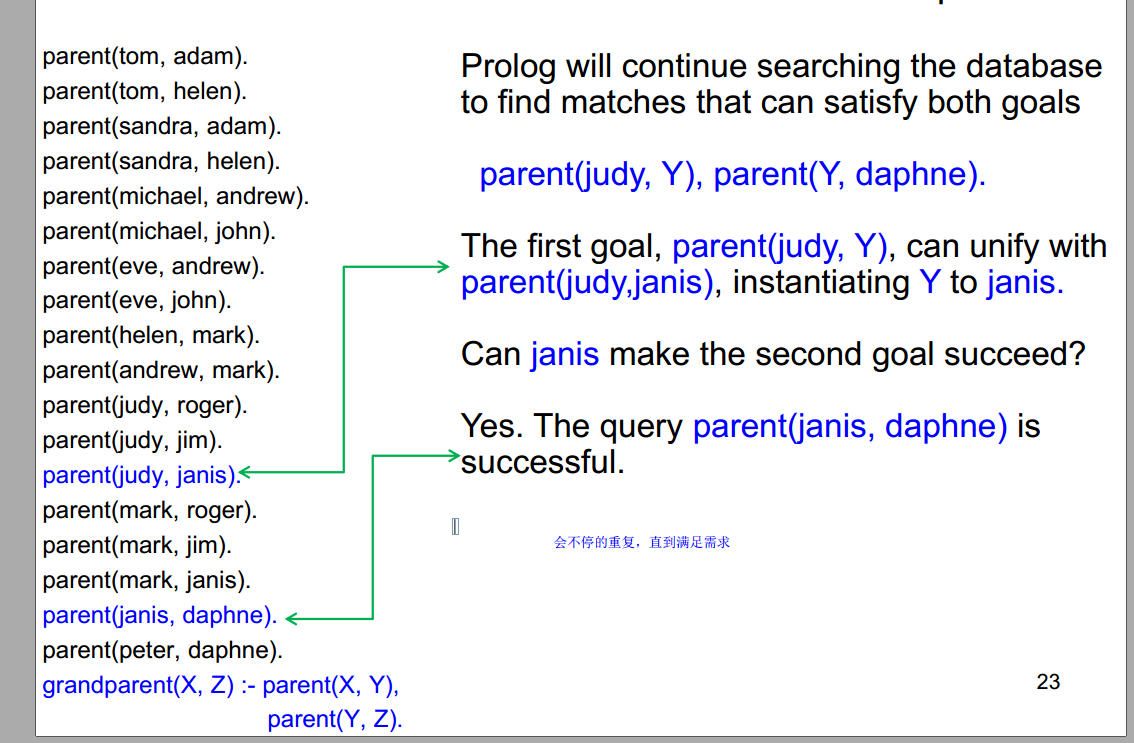


有一个X,输入这个会弹出X的值（daphne的父母）

对于RULE怎么具体验证：

1. ground query
2. 找到对应规则，通过第一个GOAL给出Y，然后把Y带入第二个goal得到YES OR TRUE。
3. 不停的重复，直到得到YES。OR遍历完以后得到NO

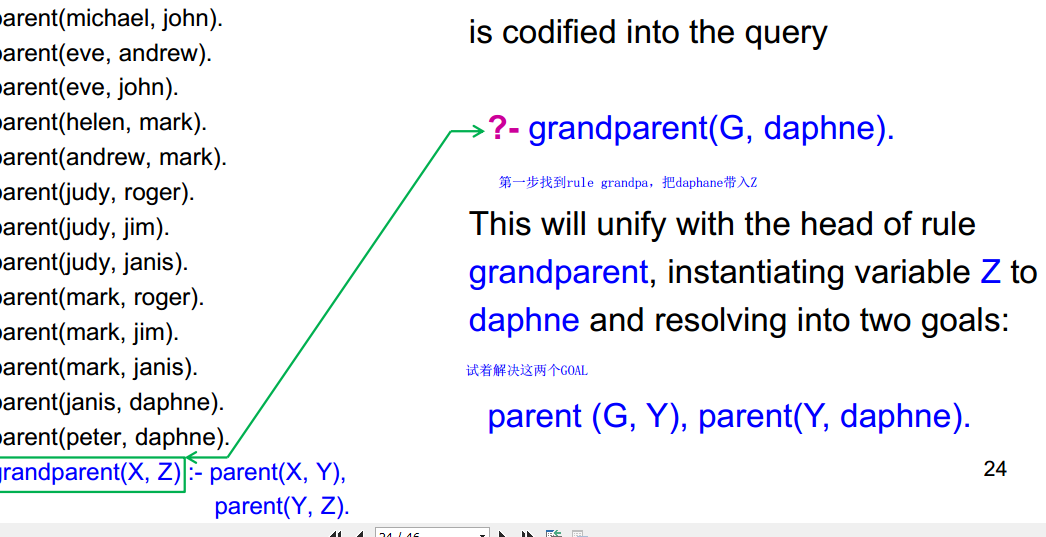


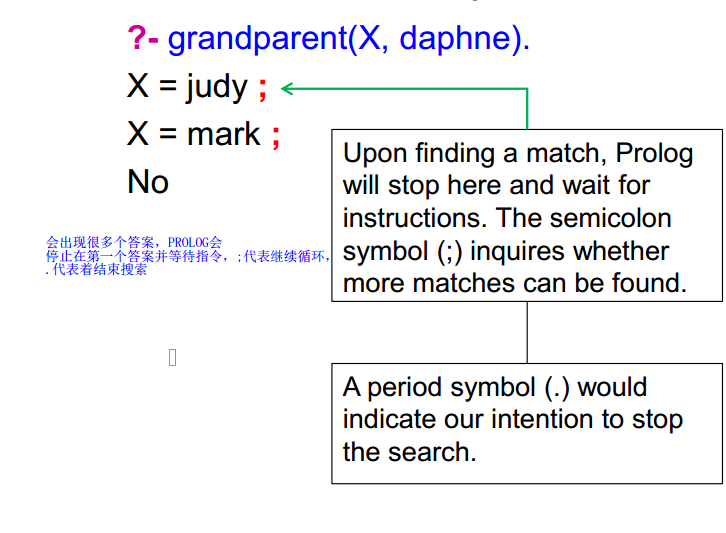


1. Non-ground

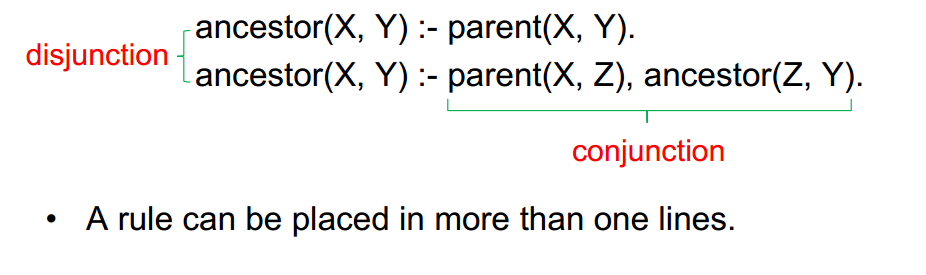
第一步找到对应法则，带入空缺的那位，然后找到两个GOAL（）两个都是NON-GROUND，

每一次循环只会出来一个答案并等待指令，按;才会继续 按.就结束



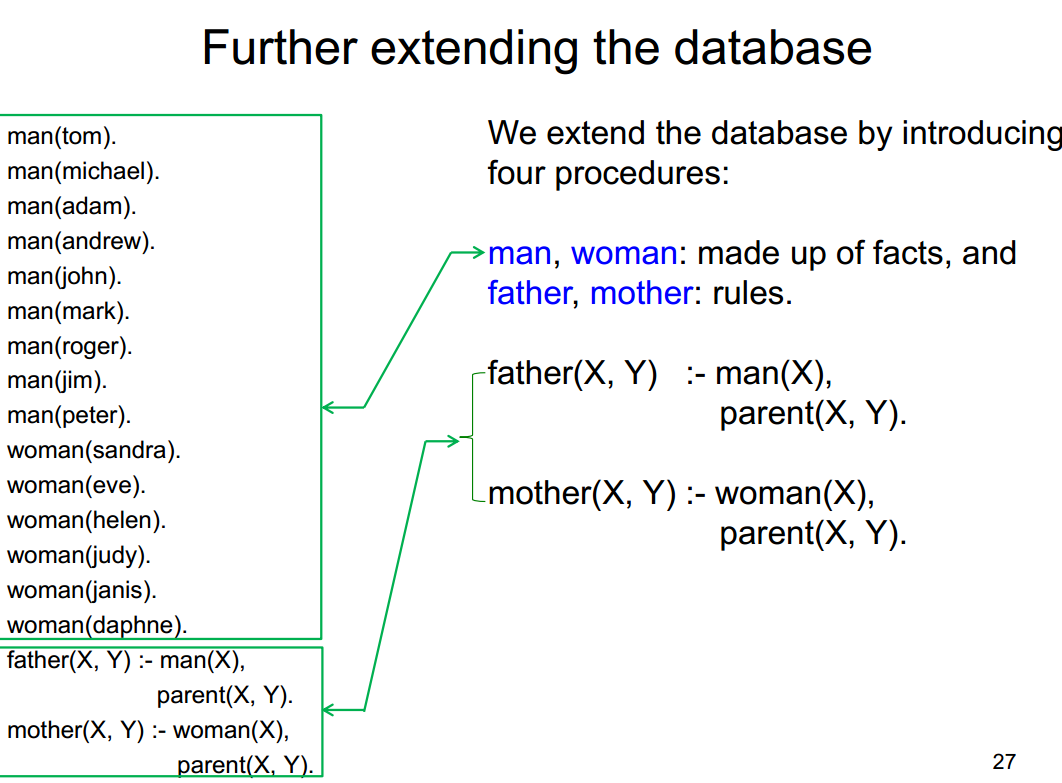


Disjunction

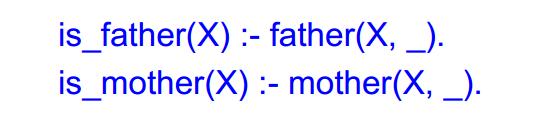


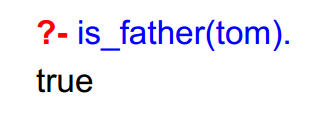
一条rule可以被多种方式描述，叫做disjuntion

一条rule有用逗号隔开的不同GOAL，两个GOAL之间有CONJUNCTION

更进一步拓展DATABASE：为个人添加属性  


如果一个关系里面有一个参数不重要，我们可以用 \_ 下划线来代替 anonymous variable匿名的变量





怎么样写ANCESTOR这种一连串的关系：  
错误ancestor(X,Y):-parent(x,y);parent(x,z),parent(z,y)

第一个是父母关系，第二个是爷孙关系

正确ancestor(x,y):-parent(x,y);parent(x,z),ancestor(z,y)

比如说X,Z1,Z2,Y

X是Y的的爹吗，不是。X是Z1的爹，并且Z1是Y的爷爷吗？针对第二句，无限循环

最后得到第一个GOAL，Z2是Y的爹

路径图· path(x,y):-edge(x,y);edge(X,Z),PATH(Z,Y)

总而言之，第一个是基本条件，第二个GOAL里面前半句是引出她的下一个，后半句是recursion