Just as for propositional logic, for predicate logic, a *definite* Horn clause is one that contains exactly one positive atom, i.e., a clause of the form

\[
\forall X_1 \ldots \forall X_n (B_1 \land \ldots \land B_m \rightarrow H)
\]

or

\[
\forall X_1 \ldots \forall X_n (\top \rightarrow H)
\]

where the \(B_i\) and \(H\) are atoms.

A *logic program* is a set of definite Horn Clauses.

A *goal* or *query* is a formula of the form

\[
\exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_m)
\]

where the \(G_i\) are atoms.
Example: the following set of clauses is a logic program:

1: $\forall X \forall Y (\text{edge}(X,Y) \to \text{path}(X,Y))$
2: $\forall X \forall Y \forall Z (\text{edge}(X,Y) \land \text{path}(Y,Z) \to \text{path}(X,Z))$
3: $\top \to \text{edge}(a,b)$
4: $\top \to \text{edge}(b,c)$

The following are queries:

$$\exists X \exists Y \exists Z (\text{edge}(X,Y) \land \text{path}(X,Y))$$
$$\exists X \exists Y (\text{path}(X,Y))$$

The following logical equivalences hold:

$$\neg \forall X (\phi) \equiv \exists x (\neg \phi)$$
$$\neg \exists X (\phi) \equiv \forall x (\neg \phi)$$

Note that

$$\neg \exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_n) \equiv \forall X_1 \neg \exists X_2 \ldots \exists X_n (G_1 \land \ldots \land G_n)$$
$$\equiv \forall X_1 \ldots \forall X_n (\neg (G_1 \land \ldots \land G_n))$$
$$\equiv \forall X_1 \ldots \forall X_n ((G_1 \land \ldots \land G_n) \to \bot)$$

Note that if $\Gamma$ is a logic program, then

$$\Gamma \cup \{\forall X_1 \ldots \forall X_n ((G_1 \land \ldots \land G_n) \to \bot)\}$$

is a set of Horn clauses.
Thus, if $\Gamma$ is a logic program and $\exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_n)$ is a goal, then the following are equivalent:

- $\Gamma \models \exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_m)$
- $\Gamma \cup \{\forall X_1 \ldots \forall X_n ((G_1 \land \ldots \land G_m) \rightarrow \bot)\}$ is unsatisfiable.
- The set of ground instances of $\Gamma \cup \{\forall X_1 \ldots \forall X_n ((G_1 \land \ldots \land G_m) \rightarrow \bot)\}$ is unsatisfiable.
- There exists a ground instance $g$ of $\forall X_1 \ldots \forall X_n ((G_1 \land \ldots \land G_m) \rightarrow \bot)$ such that some branch of the SLD resolution tree for $\Gamma \cup \{g\}$ has the leaf $\bot$.

**Substitutions**

A substitution $\theta : V \rightarrow \text{Terms}$ is a mapping from some (possibly empty) set of variables $V$ to the set of terms.

We may represent a substitution by a list of pairs of the form $X \rightarrow t$ indicating that variable $X$ maps to term $t$.

A substitution $\theta$ is ground if for every variable $X$ in the domain $V$ of $\theta$, its value $\theta(X)$ is a ground term.

Examples:
- $[X \rightarrow f(Y,Z,a), \ Y \rightarrow b]$ is a (non-ground) substitution
- $[X \rightarrow a, \ Y \rightarrow b, \ Z \rightarrow f(a)]$ is a ground substitution
- $[]$ or $\epsilon$ is the null substitution (this is ground)
If \( \phi \) is a formula and \( \theta \) is a substitution, we may apply \( \theta \) to \( \phi \) by replacing each free occurrence of a variable \( X \) in the domain of \( \theta \) by \( \theta(X) \).

We denote the resulting formula by \( \phi\theta \).

Examples:

\[
p(X,Y,Z)[X \to f(Y), Y \to a] = p(f(Y),a,Z)
\]

\[
(p(X) \land (r(X,Z) \to \forall X(q(X,Y,Z))))[X \to a, Y \to b] = p(a) \land (r(a,Z) \to \forall X(q(X,b,Z)))
\]

Note:

- We do the substitution for all variables at the same time, not one after the other. (If we did \( X \) before \( Y \) in the first example we would get \( p(f(a),a,Z) \))

- Variables not in the domain of the substitution (e.g. \( Z \) in the examples above) are left unchanged.
**Important Fact:** If $\Gamma$ is a logic program and 
$\exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_n)$ is a goal, then 
$$\Gamma \models \exists X_1 \ldots \exists X_n (G_1 \land \ldots \land G_n)$$
if and only if there exists a ground substitution 
$[X_1 \rightarrow t_1, \ldots, X_n \rightarrow t_n]$ such that 
$$\Gamma \models (G_1 \land \ldots \land G_n)[X_1 \rightarrow t_1, \ldots, X_n \rightarrow t_n].$$
A substitution $[X_1 \rightarrow t_1, \ldots, X_n \rightarrow t_n]$ satisfying the condition above is called an *answer substitution* for the query.

Thus, not only can we ask “does the goal follow from $\Gamma$?” we can also ask “if the goal follows from $\Gamma$, what values of $X_1 \ldots X_n$ make this true?”

Note: answer substitutions are not guaranteed to exist when $\Gamma$ is not a logic program:
$$P(a) \lor P(b) \models \exists X (P(X))$$
but neither $P(a) \lor P(b) \models P(a)$ nor $P(a) \lor P(b) \models P(b)$. 
Prolog Syntactic Conventions

- Variables are in upper case.
- Constants and predicates are in lower case.
- Each formula is terminated by a period: “.”
- The clause $\forall X_1, \ldots, \forall X_n (B_1 \land \ldots \land B_n \landarrow H)$ is written
  
  \[ H : \neg B_1, \ldots, B_n. \]
- The clause $\forall X_1, \ldots, \forall X_n (\top \landarrow H)$ is written
  
  \[ H. \]
- A query $\exists X_1, \ldots, \exists X_n (G_1 \land \ldots \land G_n)$ is written
  
  \[ G_1, \ldots, G_n? \]

The example as a Prolog program:

```prolog
edge(a, b).
edge(b, c).

path(X, Y) : - edge(X, Y).
path(X, Z) : - edge(X, Y), path(Y, Z).
```

6
Running iprolog

- Create a file “filename.pl” containing the logic program. Don’t forget the dots at the end of each clause!
- Start the prolog interpreter running by “prolog” or “iprolog”. The prompt is “:”. (If you see “>” it means the command being input is not yet complete.)
- Load the program using the query “consult(‘filename.pl’)!”.
  Use “listing!” to see the logic program currently loaded.
- Enter a query (e.g. “path(X,Y)?”). The system returns a set of answer substitutions.
- Exit the system using Control-D.