A Horn clause is a clause containing at most one positive literal.

A definite clause contains exactly one positive literal.

Examples of a Horn Clause

\[ \neg\text{Child}, \neg\text{Mail}, \text{Boy} \]

Not a Horn Clause

\[ \text{Rain}, \text{Sleet}, \text{Snow} \]
Horn Clauses (Cont)

\[ p_1 \land \ldots \land p_n \rightarrow q \]

\[ [\neg p_1, \ldots, \neg p_n, q] \]

*Positive Horn Clause*

*Negative Horn Clause*
Some Observations

There is a derivation of a negative clause (including the empty clause) from a set of Horn clauses $S$ iff there is one where each new clause in the derivation is a negative resolvent of the previous clause in the derivation and some element of $S$. 
SLD Resolution Pattern
SLD Resolution

For any set $S$ of clauses, an SLD derivation of a clause $c$ from $S$ is a sequence of clauses $c_1, c_2, \ldots, c_n$ such that $c_n = c, c_1 \in S$ and $c_{i+1}$ is a resolvent of $c_i$ and some clause of $S$. We write $S \vdash_{SLD} c$ if there is an SLD derivation of $c$ from $S$.

$$\text{if } S \vdash_{SLD} \square \text{ then } S \vdash \square$$

For Horn clauses:

$$\text{if } S \vdash_{SLD} \square \text{ iff } S \vdash \square$$
Example

Toddler

Toddler $\rightarrow$ Child

$(\text{Child} \land \text{Male}) \rightarrow \text{Boy}$

Infant $\rightarrow$ Child

$(\text{Child} \land \text{Female}) \rightarrow \text{Girl}$

Female

Query

$KB \models \text{Girl}$
Another Example: Lists

Constant NIL Binary Function CONS, e.g.,
CONS(t₁, t₂)

Definition of APPEND(X,Y,Z)

APPEND(NIL, y, y)

APPEND(x, y, z) →

APPEND(CONS(w, x), y, CONS(w, z))

We wish to show that this entails the following:

APPEND(CONS(a, CONS(b, NIL)), CONS(c, NIL), CONS(a, CONS(b, CONS(c, NIL))))
Back-Chaining procedure

**Input:** a finite list of atomic sentences, \( q_1, \ldots, q_n \)

**Output:** yes or no depending on whether a given \( KB \) entails all of the \( q_i \)

\[
\text{SOLVE}[q_1, \ldots, q_n] = \\
\text{If } n = 0 \text{ then return } \text{yes} \\
\text{For each clause } c \in KB, \text{ do} \\
\quad \text{If } c = [q_1, \neg p_1, \ldots, \neg p_m] \text{ and} \\
\quad \text{SOLVE}[p_1, \ldots, p_m, q_2, \ldots, q_n] \\
\quad \text{then return } \text{yes} \\
\text{end for} \\
\text{Return } \text{no}
\]
Forward-Chaining procedure

**Input:** a finite list of atomic sentences, $q_1, \ldots, q_n$

**Output:** yes or no depending on whether a given KB entails all of the $q_i$

1. if all of the goals $q_i$ are marked as solved, then return yes

2. check if there is a clause $[q_1, \neg p_1, \ldots, \neg p_n]$ in the KB, such that all of its negative atoms $\neg p_1, \ldots, \neg p_n$ are marked as solved, and such that the positive atom $p$ is not marked as solved

3. if there is such a clause, mark $p$ as solved and go to step 1

4. otherwise, return no