Task 6 - Saketh & Aman
Datalog

Problem and Motivation

What is it and why do we use it?.

Modern LLM's are very powerful for what they're able to do, however they lack a form of precision and verifiability.

Datalog helps us quantify and measure the accuracy of responses. Thus helping models:

- Answer with more accuracy to the topic
- Be better with knowledge retention and usage
- Helping with Safety and Security concerns

Syntax and Use

Define Facts

Syntax: parent(Alice, Bob)

Apply Rules and Conditions

Syntax: grandparent(X, Y) :- parent(X, Z), parent(Z, Y).

Test adherence and get results

Syntax: ?- grandparent(alice, Y).

In this small simulation we've defined what it means to be a parent (and therefore grandparent) and are trying to determine whether a model can accurately get to that logical conclusion

How does it evaluate?

Uses bottom-up evaluation ("forward chaining") that starts with facts, repeatedly applies rules until no new facts appear. This means it always terminates because Datalog has no function symbols

- Example — Recursive ancestor rule:
  —————————————————————
- ancestor(X, Y) :- parent(X, Y).
- ancestor(X, Y) :- parent(X, Z), ancestor(Z, Y).
  —————————————————————
- Query: ancestor(alice, Y)
- → evaluates all ancestor relationships exhaustively.

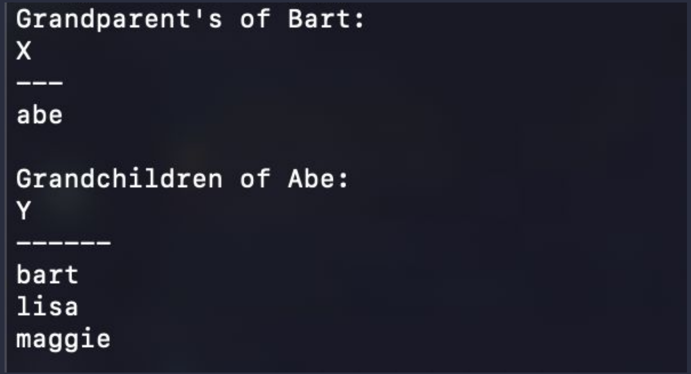
Use Cases

Datalog is used primarily as said before in model evaluation capabilities, offering LLM's essentially something to check their own knowledge and reasoning systems. In the real world it can be used for:

- RBAC and Security system control
- Graph analysis of performance
- Query Optimization
- Prototyping newer LLM models

Link to Repo

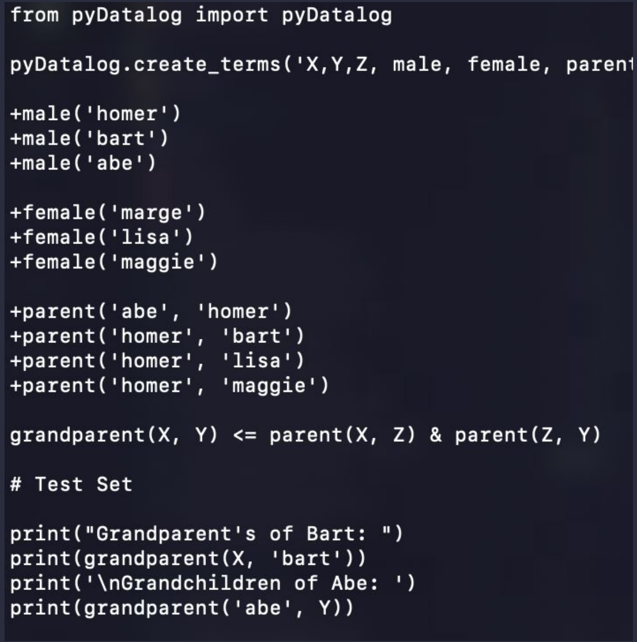
Result



```
Grandparent's of Bart:
X
---
abe

Grandchildren of Abe:
Y
------
bart
lisa
maggie
```

Demo Code



```
from pyDatalog import pyDatalog

pyDatalog.create_terms('X,Y,Z, male, female, parent

+male('homer')
+male('bart')
+male('abe')

+female('marge')
+female('lisa')
+female('maggie')

+parent('abe', 'homer')
+parent('homer', 'bart')
+parent('homer', 'lisa')
+parent('homer', 'maggie')

grandparent(X, Y) <= parent(X, Z) & parent(Z, Y)

# Test Set

print("Grandparent's of Bart: ")
print(grandparent(X, 'bart'))
print('\nGrandchildren of Abe: ')
print(grandparent('abe', Y))
```