Question Answering Tool for Shopping-at-a-Supermarket Knowledge Base

Shivam Gupta (SXG190040)
Department of Computer Science
The University of Texas at Dallas
Email: *shivam.gupta@utdallas.edu*

Introduction:

Question Answering tool is a very effective tool which helps in answering the questions based on some specific domain sentences. In this project, a proving technique called, Resolution Refutation has been utilized to output Yes/No for a given set of user-asked questions based on the knowledge-base in first-order logic format constructed for shopping in a supermarket. First-order logic, also known as first-order predicate calculus is the form in which each sentence or statement, is broken down into predicates. New predicates were defined and used according to the requirements. Operators, like as, "&" for AND, "-" for NOT, "|" for OR and "->" for implies are utilized. All 15 questions taken from the book question number 12.5 and 12.6 were tested.

Implementation

A tool, Prover9, has been utilized for the proof of these questions. It consists of two sections namely, the **Assumption** section which consists all the knowledge base predicates converted into first-order logic and the Goal section which consists the clauses to be proved.

Below you can find knowledge base in plain English which is used to implement first order logic.

Some predicates used:

Person(x), Item(x), FoodItem(x), FoodItem(x), Veggie(x) , Supermarket(x) ,Locale(x,y), Customer(x), MeetItem(x), DairyItem (x), FurnitureItem(x), Card(x), Supermarket(x), HasBankAccount(x,z) , FashionItem(x) ,Company (x), Purchases(x,y,z,q,t), Owns(x,y,q), Quantity(q), Time(t) , Volume(x), Pound(x), Ounces(x), etc

Results:

Below are 15 examples of some Questions that were taken in the problems 12.5 and 12.6 from textbook:

##Q1: Is John a child or an adult? [Adult]:

`FOL: Adult(John).`

Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 37548 was started by sadam on LAPTOP-2AE3ESDG,
Sat May  1 22:13:43 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.03) seconds.
% Length of proof is 11.
% Level of proof is 5.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x all y (Person(x) & HasPaymentCard(x,y) & Card(y) -> Adult(x))) # label(non_clause).  [assumption].
4 Adult(John) # label(non_clause) # label(goal).  [goal].
5 Person(John).  [assumption].
6 -Person(x) | -HasPaymentCard(x,y) | -Card(y) | Adult(x).  [clausify(1)].
8 -HasPaymentCard(John,x) | -Card(x) | Adult(John).  [resolve(5,a,6,a)].
9 HasPaymentCard(x,y).  [assumption].
11 -Card(x) | Adult(John).  [resolve(8,a,9,a)].
12 Card(Amex).  [assumption].
13 Adult(John).  [resolve(11,a,12,a)].
14 -Adult(John).  [deny(4)].
15 $F.  [resolve(13,a,14,a)].

============================== end of proof ==========================
```

##Q2: Does John now have at least two tomatoes? [Yes]

FOL: `Owns(John, Tomato,two) -> Volume(two).`

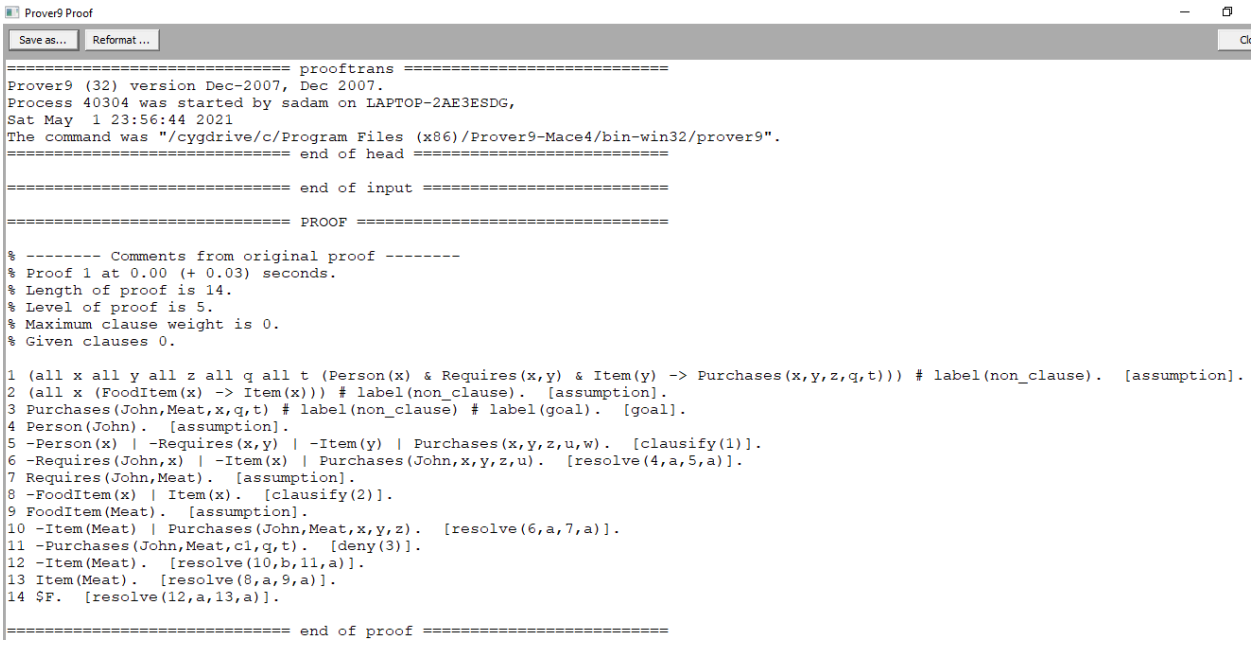
Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 33548 was started by sadam on LAPTOP-2AE3ESDG,
Sat May  1 23:08:13 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.03 (+ 0.00) seconds.
% Length of proof is 11.
% Level of proof is 3.
% Maximum clause weight is 4.
% Given clauses 3.

3 Pounds(two) -> Volume(ten) # label(non_clause).  [assumption].
5 Volume(ten) -> Volume(two) # label(non_clause).  [assumption].
6 Purchases(John,Tomato,Safeway,two,t) & Pounds(two) # label(non_clause).  [assumption].
8 Owns(John,Tomato,two) -> Volume(two) # label(non_clause) # label(goal).  [goal].
14 Pounds(two).  [clausify(6)].
15 -Pounds(two) | Volume(ten).  [clausify(3)].
19 -Volume(ten) | Volume(two).  [clausify(5)].
20 -Volume(two).  [deny(8)].
21 Volume(ten).  [resolve(14,a,15,a)].
22 -Volume(ten).  [resolve(20,a,19,b)].
23 $F.  [resolve(22,a,21,a)].

============================== end of proof ==========================
```

##Q3: Did John buy any meat? [Yes]

FOL: `Purchases(John, Meet, z, q, t).`

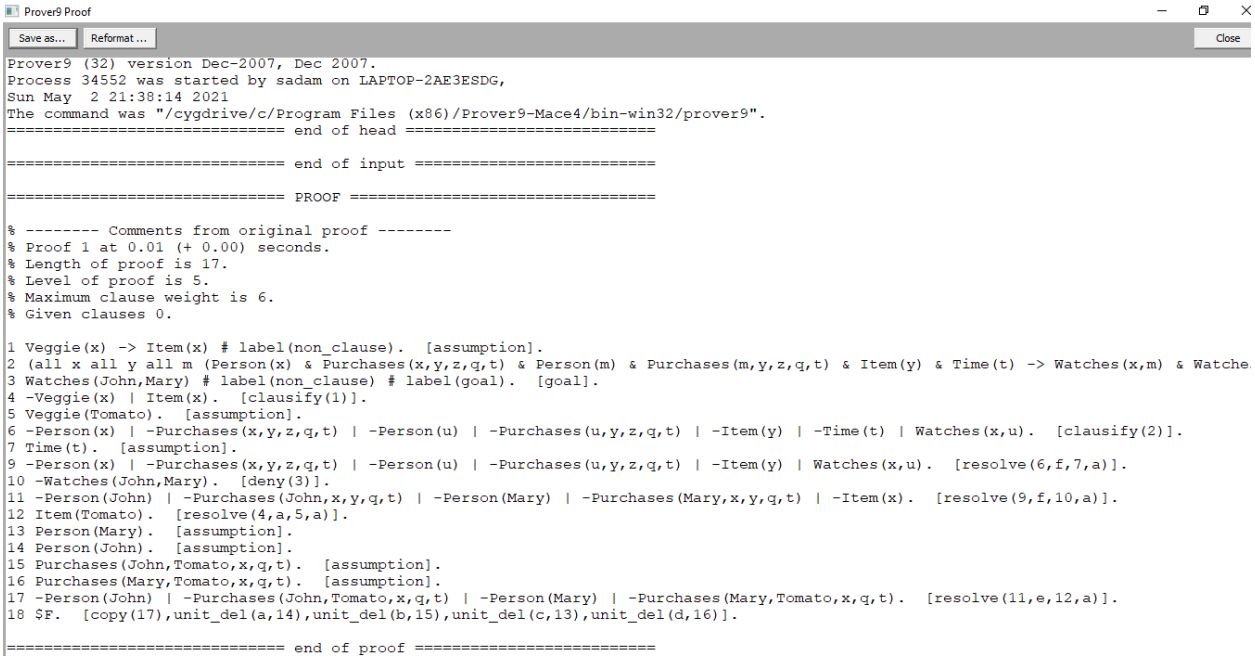


Prover9 Proof

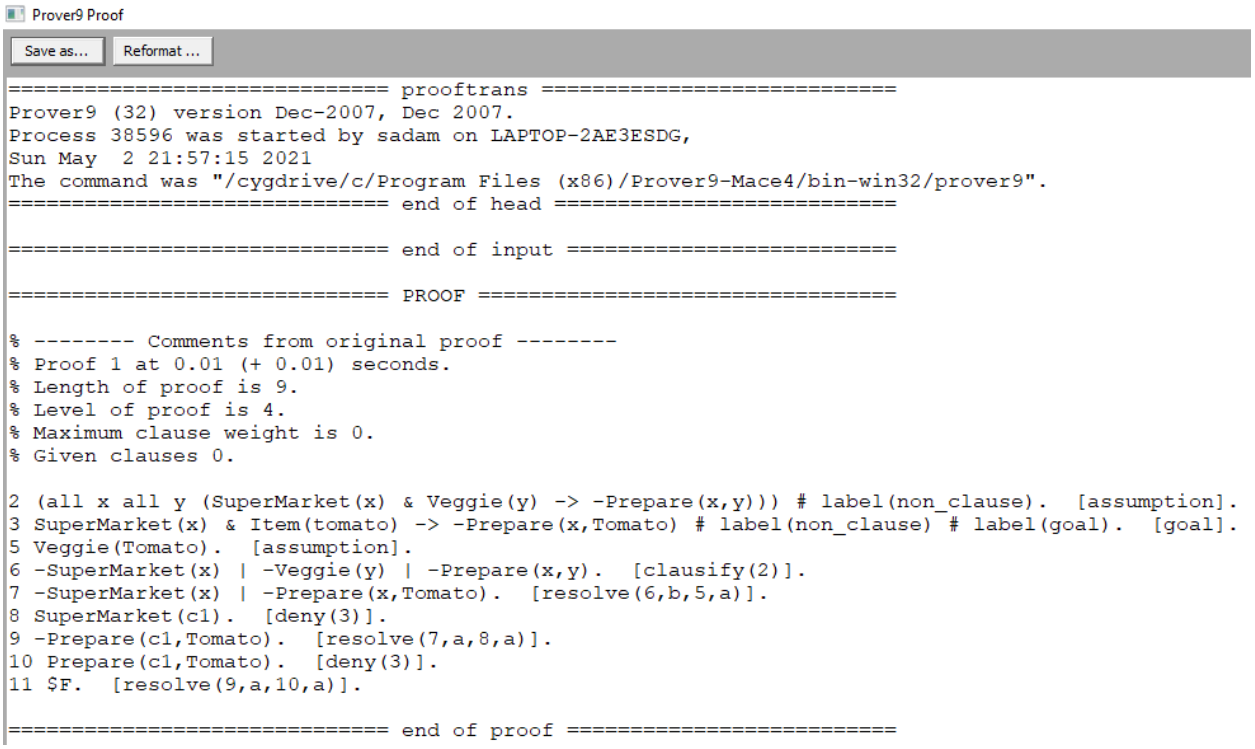
Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 40304 was started by sadam on LAPTOP-2AE3ESDG,
Sat May  1 23:56:44 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.03) seconds.
% Length of proof is 14.
% Level of proof is 5.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x all y all z all q all t (Person(x) & Requires(x,y) & Item(y) -> Purchases(x,y,z,q,t))) # label(non_clause).  [assumption].
2 (all x (FoodItem(x) -> Item(x))) # label(non_clause).  [assumption].
3 Purchases(John,Meat,x,q,t) # label(non_clause) # label(goal).  [goal].
4 Person(John).  [assumption].
5 -Person(x) | -Requires(x,y) | -Item(y) | Purchases(x,y,z,u,w).  [clausify(1)].
6 -Requires(John,x) | -Item(x) | Purchases(John,x,y,z,u).  [resolve(4,a,5,a)].
7 Requires(John,Meat).  [assumption].
8 -FoodItem(x) | Item(x).  [clausify(2)].
9 FoodItem(Meat).  [assumption].
10 -Item(Meat) | Purchases(John,Meat,x,y,z).  [resolve(6,a,7,a)].
11 -Purchases(John,Meat,c1,q,t).  [deny(3)].
12 -Item(Meat).  [resolve(10,b,11,a)].
13 Item(Meat).  [resolve(8,a,9,a)].
14 $F.  [resolve(12,a,13,a)].

============================== end of proof ==========================
```

##Q4: If Mary was buying tomatoes at the same time as John, did he see her? [Yes]
FOL: Watches(John, Mary).

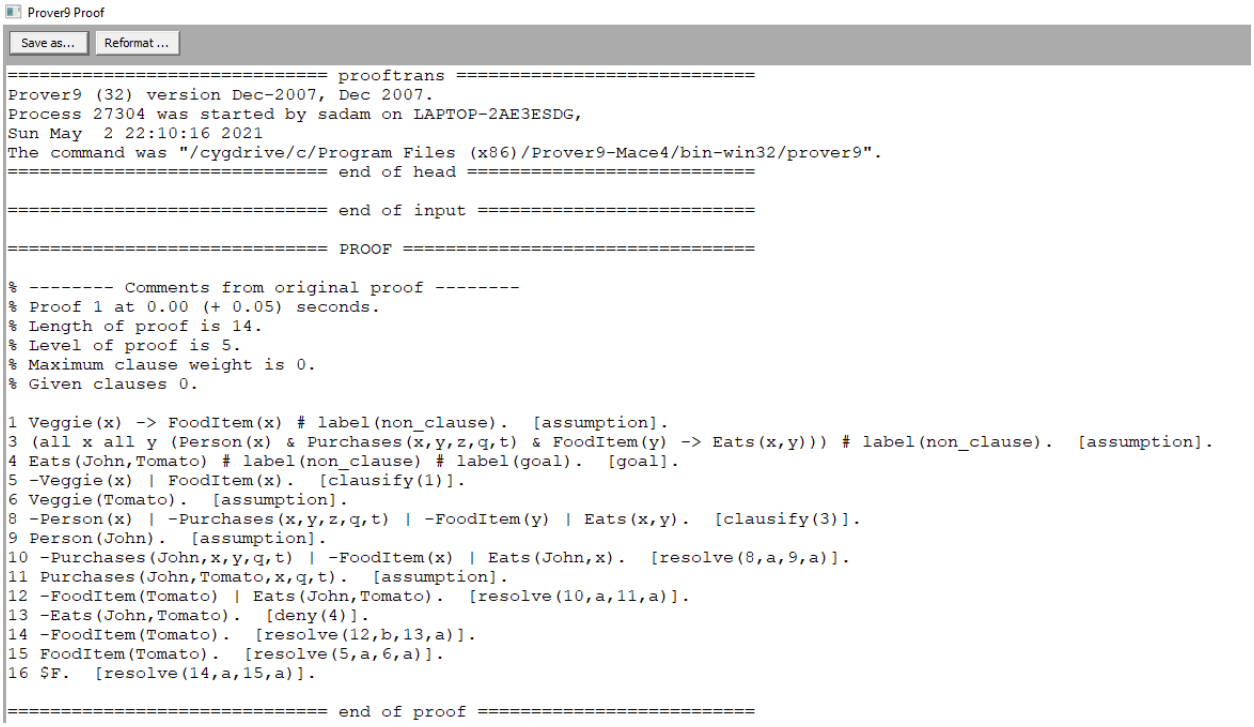


##Q5: Are the tomatoes made in the supermarket? [No]
FOL: SuperMarket(x) & Item(tomato) -> -Prepare(x, Tomato).



##Q6: What is John going to do with the tomatoes? Eat Them

FOL: Eats(John, Tomato).



##Q7: Does Safeway sell deodorant? [Yes]

FOL: Sells(Safeway,Deodorant).

Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 10644 was started by sadam on LAPTOP-2AE3ESDG,
Sun May  2 23:00:08 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.03) seconds.
% Length of proof is 14.
% Level of proof is 5.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x (FashionItem(x) -> Item(x))) # label(non_clause).  [assumption].
2 (all x all y (Supermarket(x) & Has(x,y) & Item(y) -> Sells(x,y))) # label(non_clause).  [assumption].
3 Sells(Safeway,Deodorant) # label(non_clause) # label(goal).  [goal].
4 -FashionItem(x) | Item(x).  [clausify(1)].
5 FashionItem(Deodorant).  [assumption].
6 -Supermarket(x) | -Has(x,y) | -Item(y) | Sells(x,y).  [clausify(2)].
7 Has(Safeway,Deodorant).  [assumption].
8 -Supermarket(Safeway) | -Item(Deodorant) | Sells(Safeway,Deodorant).  [resolve(6,b,7,a)].
9 Supermarket(Safeway).  [assumption].
10 -Item(Deodorant) | Sells(Safeway,Deodorant).  [resolve(8,a,9,a)].
11 -Sells(Safeway,Deodorant).  [deny(3)].
12 -Item(Deodorant).  [resolve(10,b,11,a)].
13 Item(Deodorant).  [resolve(4,a,5,a)].
14 $F.  [resolve(12,a,13,a)].

============================== end of proof ==========================
```

##Q8: Did John bring some money or a credit card to the supermarket? [Yes]

FOL: bringsMoney(John).

Prover9 Proof

Save as... Reformat ... Close

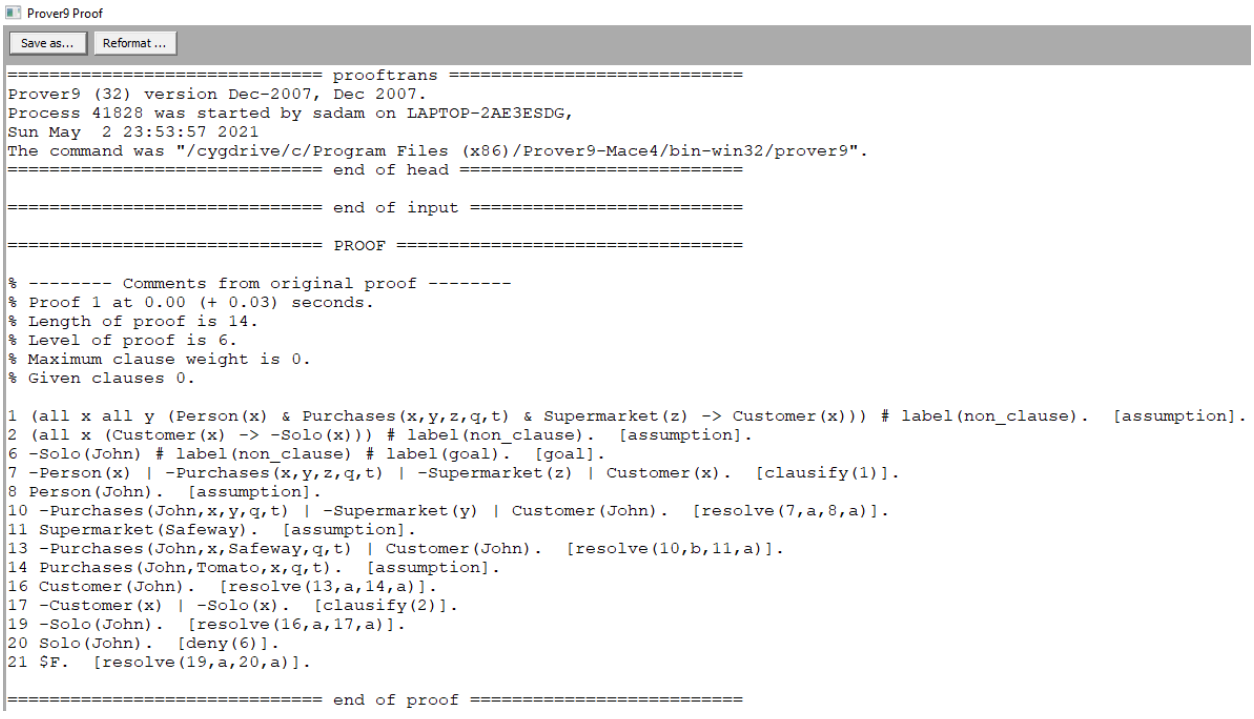
```
Process 37368 was started by sadam on LAPTOP-2AE3ESDG,
Sun May  2 23:24:06 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.03 (+ 0.00) seconds.
% Length of proof is 16.
% Level of proof is 6.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x (HasBankAccount(x,y) -> HasCreditCard(x))) # label(non_clause).  [assumption].
2 (all x all y all z all k (Person(x) & Needs(x,y) & HasCreditCard(x) & Supermarket(k) -> bringsMoney(x))) # label(non_clause).  [assur
3 bringsMoney(John) # label(non_clause) # label(goal).  [goal].
4 -Person(x) | -Needs(x,y) | -HasCreditCard(x) | -Supermarket(z) | bringsMoney(x).  [clausify(2)].
5 Person(John).  [assumption].
6 -Needs(John,x) | -HasCreditCard(John) | -Supermarket(y) | bringsMoney(John).  [resolve(4,a,5,a)].
7 Supermarket(Safeway).  [assumption].
8 -HasBankAccount(x,y) | HasCreditCard(x).  [clausify(1)].
9 HasBankAccount(John,AmericanExpress).  [assumption].
10 -Needs(John,x) | -HasCreditCard(John) | bringsMoney(John).  [resolve(6,c,7,a)].
11 Needs(John,x).  [assumption].
12 -HasCreditCard(John) | bringsMoney(John).  [resolve(10,a,11,a)].
13 -bringsMoney(John).  [deny(3)].
14 -HasCreditCard(John).  [resolve(12,b,13,a)].
15 HasCreditCard(John).  [resolve(8,a,9,a)].
16 $F.  [resolve(14,a,15,a)].

============================== end of proof ==========================
```

##Q9: Does John have less money after going to the supermarket? [Yes] :

FOL: Visited(John,x) & Supermarket(x) & HasMoney(John,g) ->LessThan(h,g)& HasMoney(John,h).

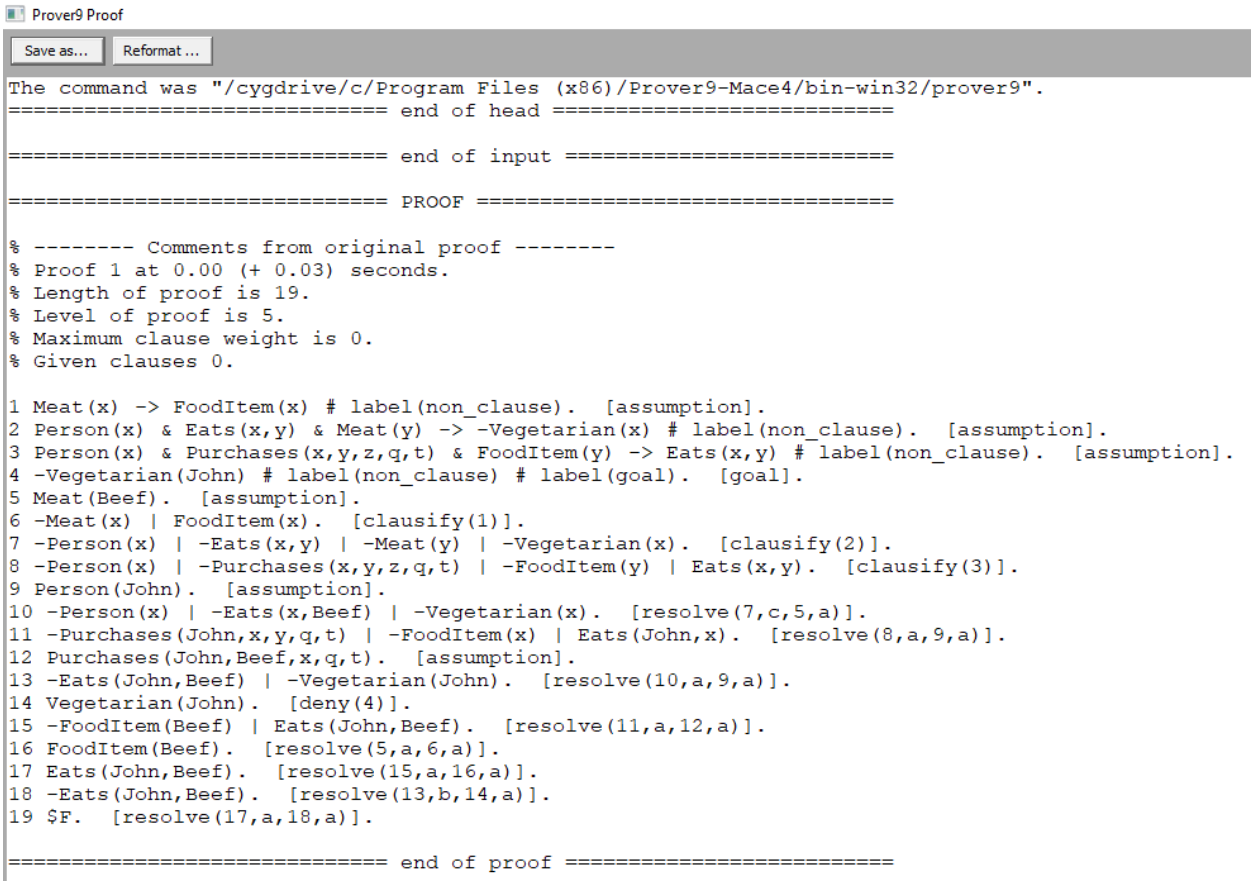
##Q10: Are there other people in Safeway while John is there?

FOL: -Solo(John).



##Q11: Is John a vegetarian? [No]

FOL: -Vegetarian(John).



##Q12: Who owns the deodorant in Safeway? [Safeway Corporation]:

FOL: Owns(SafewayCorporation, Deodorant,q).

Prover9 Proof

Save as... Reformat ...

```
Prover9 (32) version Dec-2007, Dec 2007.
Process 10616 was started by sadam on LAPTOP-2AE3ESDG,
Mon May  3 12:09:43 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.03 (+ 0.01) seconds.
% Length of proof is 16.
% Level of proof is 6.
% Maximum clause weight is 0.
% Given clauses 0.

1 Supermarket(x) & Item(y) & Purchases(x,y,z,q,t) & Company(z) -> Owns(z,y,q) # label(non_clause).  [assumption].
2 (all x (FashionItem(x) -> Item(x))) # label(non_clause).  [assumption].
3 Owns(SafewayCorporation,Deodorant,q) # label(non_clause) # label(goal).  [goal].
4 Supermarket(Safeway).  [assumption].
5 -Supermarket(x) | -Item(y) | -Purchases(x,y,z,q,t) | -Company(z) | Owns(z,y,q).  [clausify(1)].
6 -Item(x) | -Purchases(Safeway,x,y,q,t) | -Company(y) | Owns(y,x,q).  [resolve(4,a,5,a)].
7 Company(SafewayCorporation).  [assumption].
8 -FashionItem(x) | Item(x).  [clausify(2)].
9 FashionItem(Deodorant).  [assumption].
10 -Item(x) | -Purchases(Safeway,x,SafewayCorporation,q,t) | Owns(SafewayCorporation,x,q).  [resolve(6,c,7,a)].
11 Purchases(Safeway,Deodorant,SafewayCorporation,q,t).  [assumption].
12 -Item(Deodorant) | Owns(SafewayCorporation,Deodorant,q).  [resolve(10,b,11,a)].
13 -Owns(SafewayCorporation,Deodorant,q).  [deny(3)].
14 -Item(Deodorant).  [resolve(12,b,13,a)].
15 Item(Deodorant).  [resolve(8,a,9,a)].
16 $F.  [resolve(14,a,15,a)].

============================== end of proof ==========================
```

##Q13: Did John have an ounce of ground beef? [Yes]:

FOL: Purchases(John,GroundBeef,z,one,t) & Ounces(one).

Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 40940 was started by sadam on LAPTOP-2AE3ESDG,
Mon May  3 14:43:31 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 13.
% Level of proof is 4.
% Maximum clause weight is 4.
% Given clauses 2.

1 (all x all y (Person(x) & Purchases(x,y,z,one,t) & Pound(one) & Item(y))) # label(non_clause).  [assumption].
2 Pound(one) -> Ounces(sixteen) # label(non_clause).  [assumption].
3 Ounces(sixteen) -> Ounces(one) # label(non_clause).  [assumption].
6 Purchases(John,GroundBeef,x,one,t) & Ounces(one) # label(non_clause) # label(goal).  [goal].
7 -Purchases(John,GroundBeef,c1,one,t) | -Ounces(one).  [deny(6)].
8 Purchases(x,y,z,one,t).  [clausify(1)].
10 -Pound(one) | Ounces(sixteen).  [clausify(2)].
11 Pound(one).  [clausify(1)].
13 -Ounces(sixteen) | Ounces(one).  [clausify(3)].
14 -Ounces(one).  [resolve(7,a,8,a)].
15 Ounces(sixteen).  [resolve(10,a,11,a)].
16 -Ounces(sixteen).  [resolve(14,a,13,b)].
17 $F.  [resolve(16,a,15,a)].

============================== end of proof ==========================
```

##Q14: Does the Shell station next door have any gas? [Yes]:

FOL: NearTo(Shell,Safeway) & Owns(Shell,x,q) & Gas(x).

Prover9 Proof

Save as... Reformat ...

```
Process 9592 was started by sadam on LAPTOP-2AE3ESDG,
Mon May  3 15:00:14 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.00) seconds.
% Length of proof is 17.
% Level of proof is 5.
% Maximum clause weight is 6.
% Given clauses 3.

1 (all x all y (GasStation(x) & Owns(x,y,q) & Gas(y))) # label(non_clause).  [assumption].
3 Supermarket(x) & Locale(x,y) & GasStation(z) & Locale(z,y) -> NearTo(x,z) & NearTo(z,x) # label(non_clause).  [assumption].
4 NearTo(Shell,Safeway) & Owns(Shell,x,q) & Gas(x) # label(non_clause) # label(goal).  [goal].
6 GasStation(Shell).  [assumption].
9 -Supermarket(x) | -Locale(x,y) | -GasStation(z) | -Locale(z,y) | NearTo(z,x).  [clausify(3)].
10 -NearTo(Shell,Safeway) | -Owns(Shell,c1,q) | -Gas(c1).  [deny(4)].
11 Owns(x,y,q).  [clausify(1)].
12 -NearTo(Shell,Safeway) | -Gas(c1).  [resolve(10,b,11,a)].
13 Gas(x).  [clausify(1)].
15 Supermarket(Safeway).  [assumption].
17 -Supermarket(x) | -Locale(x,y) | -Locale(Shell,y) | NearTo(Shell,x).  [resolve(9,c,6,a)].
20 -NearTo(Shell,Safeway).  [resolve(12,b,13,a)].
22 -Locale(Safeway,x) | -Locale(Shell,x) | NearTo(Shell,Safeway).  [resolve(17,a,15,a)].
24 Locale(Safeway,NorthBerkeley).  [assumption].
25 Locale(Shell,NorthBerkeley).  [assumption].
26 -Locale(Safeway,x) | -Locale(Shell,x).  [resolve(22,c,20,a)].
27 $F.  [resolve(26,a,24,a),unit_del(a,25)].

============================== end of proof ==========================
```

##Q15: Do the tomatoes fit in John's car trunk? [Yes]:

FOL: Adjusts(Tomato,z) & CarTrunk(z).

Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 35340 was started by sadam on LAPTOP-2AE3ESDG,
Mon May  3 15:11:01 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 7.
% Level of proof is 3.
% Maximum clause weight is 0.
% Given clauses 0.

2 FoodItems(x) & Adjusts(x,y) & CarTrunk(y) # label(non_clause).  [assumption].
3 Adjusts(Tomato,x) & CarTrunk(x) # label(non_clause) # label(goal).  [goal].
6 -Adjusts(Tomato,c1) | -CarTrunk(c1).  [deny(3)].
7 Adjusts(x,y).  [clausify(2)].
8 -CarTrunk(c1).  [resolve(6,a,7,a)].
9 CarTrunk(x).  [clausify(2)].
10 $F.  [resolve(8,a,9,a)].

============================== end of proof ==========================
```

Additional 2 Questions apart from 12.5 and 12.6

Q16: Who owns the Milk in Safeway? [Safeway Corporation]:

FOL: Owns(SafewayCorporation, Milk,q).

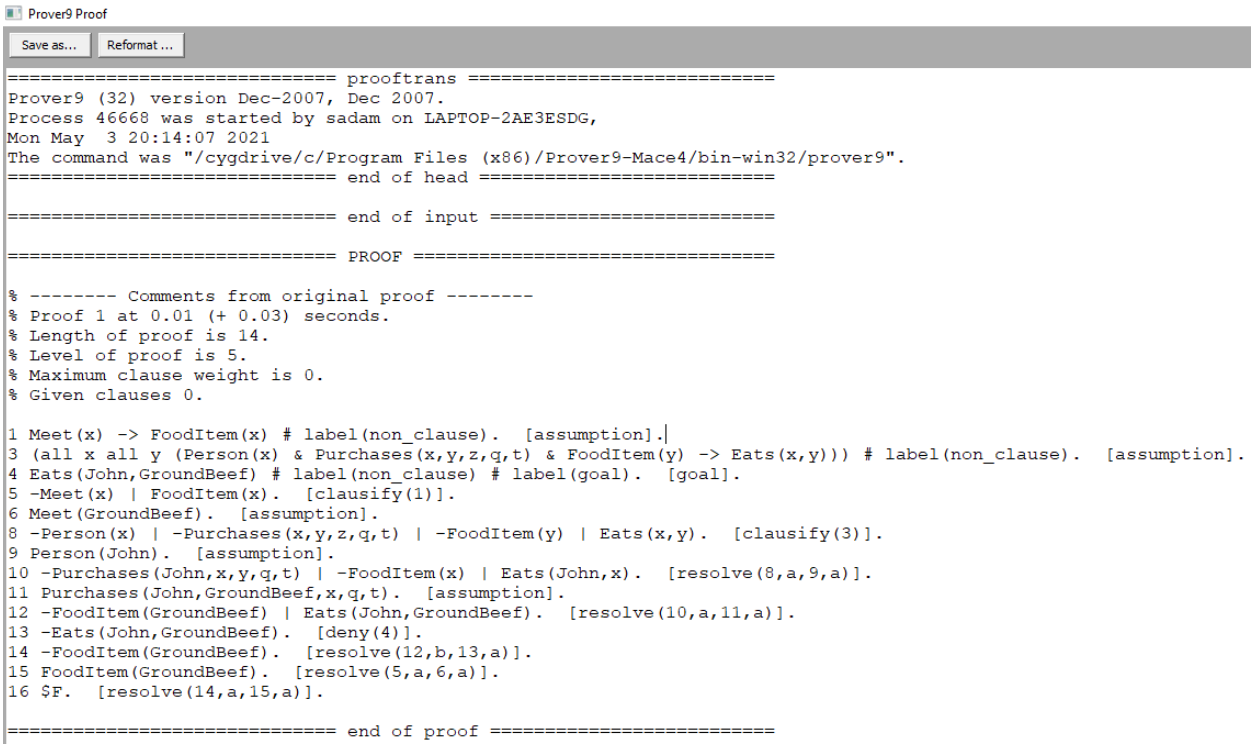
Prover9 Proof

Save as... Reformat ...

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 43936 was started by sadam on LAPTOP-2AE3ESDG,
Mon May  3 20:08:39 2021
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.00) seconds.
% Length of proof is 16.
% Level of proof is 6.
% Maximum clause weight is 0.
% Given clauses 0.

1 Supermarket(x) & Item(y) & Purchases(x,y,z,q,t) & Company(z) -> Owns(z,y,q) # label(non_clause).  [assumption]
2 (all x (DairyItem(x) -> Item(x))) # label(non_clause).  [assumption].
3 Owns(SafewayCorporation,Milk,q) # label(non_clause) # label(goal).  [goal].
4 Supermarket(Safeway).  [assumption].
5 -Supermarket(x) | -Item(y) | -Purchases(x,y,z,q,t) | -Company(z) | Owns(z,y,q).  [clausify(1)].
6 -Item(x) | -Purchases(Safeway,x,y,q,t) | -Company(y) | Owns(y,x,q).  [resolve(4,a,5,a)].
7 Company(SafewayCorporation).  [assumption].
8 -DairyItem(x) | Item(x).  [clausify(2)].
9 DairyItem(Milk).  [assumption].
10 -Item(x) | -Purchases(Safeway,x,SafewayCorporation,q,t) | Owns(SafewayCorporation,x,q).  [resolve(6,c,7,a)].
11 Purchases(Safeway,Milk,SafewayCorporation,q,t).  [assumption].
12 -Item(Milk) | Owns(SafewayCorporation,Milk,q).  [resolve(10,b,11,a)].
13 -Owns(SafewayCorporation,Milk,q).  [deny(3)].
14 -Item(Milk).  [resolve(12,b,13,a)].
15 Item(Milk).  [resolve(8,a,9,a)].
16 $F.  [resolve(14,a,15,a)].

============================== end of proof ==========================
```

Q17: What is John going to do with the GroundBeef?

FOL: Eats(John, GroundBeef).



Conclusion and Future Work:

In the end I would like to conclude that this project has helped me in building the question answering tool based on a given specific context and how Knowledge Base is constructed in the field of Artificial Intelligence. This project also helped me in constructing the first order logic form for Knowledge base in any system. The performance of Prover 9 tool came out to be quite decent as I tested on many different questions, it gave me relevant answers. This project could be expanded to a bigger context (1000's of Pages) and this tool could be tested on varied data questions in order to check how indirect questions are being answered through this tool.

References

[1] Stuart Russell and Peter Norvig. Arti_cial Intelligence: A Modern Approach. 3rd Edition, Prentice Hall Press, 2009.

[2] Christiane Fellbaum. WordNet: An Electronic Lexical Database. Bradford Books, 1998.

[3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pages 1247{1250, 2008.