**Everything**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **English Axiom** | **First Order Logic** | **CNF** |
| 1 | Person in the supermarket are either staff or customer. | ∀x Person(x) ∧ ∀y Supermarket(y) ∧ In(x,y) => Staff(x) v Customer(x) | ~Person (x) v ~Supermarket(x) v ~In(x,y) v Staff(x) v Customer(x) |
| 2 | If the supermarket is open, then there are people in the supermarket.(#) | ∀x Supermarket(x) ∧ IsOpen(x) => ∃y Person(y) ∧ In(y,x) | ~Supermarket(x) v ~IsOpen(x) v Person(f(x))  ~Supermarket(x) v ~IsOpen(x) v In(f(x),x) |
| 5 | If a person buys products then they eat them. | ∀x ∀y Person(x) ∧ Products(y) ∧ Buys(x,y) => Eats(x,y) | ~Person(x) v ~Product(y) v ~Buys(x,y) v Eats(x,y) |
| 6 | If a person buys products then they are adults. | ∀x ∀y Person(x) ∧ Products(y) ∧ Buys(x,y) => Adult(x) | ~Person(x) v ~Product(y) v ~Buys(x,y) v Adult(x) |
| 7 | Adults cannot be kids. | ∀x Adult(x) => ~Kids(x) | ~Adult(x) v ~Kids(x) |
| 8 | Kids cannot be adults. | ∀x Kids(x) => ~Adult(x) | ~Kids (x) v ~Adults(x) |
| 9 | A person can be either an Adult or a kid | ∀x Person(x) => Adult(x) v Kids(x) | ~Person(x) v Adult(x) v Kids(x) |
| 10 | Acceptable Payment Method include Credit, Debit or Cash. | ∀x IsAcceptablePayment(x) => Cash(x) v Debit(x) v Credit(x) | ~ IsAcceptablePayment(x) v Cash(x) v Debit(x) v Credit(x) |
| 11 | Items are products. | ∀x Items(x) => Products(x) | ~Items(x) v Products(x) |
| 14 | If a person returns a product, then they must have a receipt. (#) | ∀x ∀y Person(x) ∧ Product(y) ∧ Returns(x,y) => ∃z Receipt(z) ∧ Has(x,z) | ~Person(x) v ~Product(y) v ~Returns(x,y) v Receipt(f(x))  ~Person(x) v ~Product(y) v ~Returns(x,y) v Has(x,f(x)) |
| 15 | If a person returns a product then their money increases. (#) | ∀x ∀y Person(x) ∧ Product(y) ∧ Returns(x,y) => ∃z Money(z) ∧ Has(x,z) ∧ IsIncreased(z) | ~Person(x) v ~Product(y) v ~Returns(x,y) v Money(f(x))  ~Person(x) v ~Product(y) v ~Returns(x,y) v Has(x,f(x))  ~Person(x) v ~Product(y) v ~Returns(x,y) v IsIncreased(f(x)) |
| 17 | If a person buys a product, then their money decreases (#) | ∀x ∀y Person(x) ∧ Product(y) ∧ Buys(x,y) => ∃z Money(z) ∧ Has(x,z) ∧ IsDecreased(z) | ~Person(x) v ~Product(y) v ~Buys(x,y) v Money(f(x))  ~Person(x) v ~Product(y) v ~Buys (x,y) v Has(x,f(x))  ~Person(x) v ~Product(y) v ~ Buys (x,y) v IsDecreased(f(x)) |
| 18 | Supermarket contains a parking lot. | ∀x Supermarket(x) => ∃y Parking\_Lot(y) ∧ Contains(x,y) | ~Supermarket(x) v Parking\_Lot(f(x))  ~Supermarket(x) v Contains(x,f(x)) |
| 20 | If a person buys meat, then they are non-vegetarians. | ∀x ∀y Person(x) ∧ Meat(y) ∧ Buys(x,y) => Non\_Vegetarian(x) | ~Person(x) v ~Meat(y) v ~Buys(x,y) v Non\_Vegetarian (x) |
| 21 | If a person does not buy meat, then they are vegetarians. | ∀x ∀y Person(x) ∧ Meat(y) ∧ ~Buys(x,y) => Vegetarian(x) | ~Person(x) v ~Meat(y) v Buys(x,y) v Vegetarian (x) |
| 22 | A vegetarian cannot be a Non-vegetarian | ∀x Vegetarian(x) => ~ Non\_Vegetarian(x) | ~ Vegetarian(x) v ~ Non\_Vegetarian(x) |
| 23 | A person can either be a vegetarian or a non-vegetarian. | ∀x Person(x) => Vegetarian(x) v Non\_Vegetarian(x) | ~Person(x) v Vegetarian(x) v Non\_Vegetarian(x) |
| 24 | Any product can either be liquid or solid. | ∀x Product(x) => Liquid(x) v Solid(x) | ~Product(x) v Liquid(x) v Solid(x) |
| 25 | Liquid products are either milk or water or energy\_drink or oil. | ∀x Milk(x) => Liquid(x)  ∀x Oil(x) => Liquid(x)  ∀x Water(x) => Liquid(x)  ∀x Energy\_Drink (x) => Liquid(x) | ~Milk(x) v Liquid(x)  ~Oil(x) v Liquid(x)  ~Water(x) v Liquid(x)  ~Energy\_Drink(x) v Liquid(x) |
| 26 | Liquid products are not solid. | ∀x Liquid(x) => ~Solid(x) | ~Liquid(x) v ~Solid(x) |
| 27 | Solid products are any product that is not liquid. | ∀x Solid(x) => ~Liquid(x) | ~Solid(x) V ~Liquid(x) |
| 28 | Solids are measured in pounds. | ∀x ∀y Solid(x) ∧ Pounds(y) => MeasuredIn(x,y) | ~Solid(x) v ~Pounds(y) V MeasuredIn(x,y) |
| 29 | Liquids are measured in ounces. | ∀x ∀y Liquid(x) ∧ Ounces(y) => MeasuredIn(x,y) | ~Liquid(x) v ~Ounces(y) V MeasuredIn(x,y) |
| 30 | Groceries are available in the supermarket.(#) | ∀x ∀y Groceries(x) ∧ Supermarket(y) => IsAvailable(x,y) | ~Groceries(x) v ~Supermarket(y) v IsAvailable(x,y) |
| 31 | Meat is available in the supermarket. (#) | ∀x ∀y Meat(x) ∧ Supermarket(y) => IsAvailable(x,y) | ~Meat(x) v ~Supermarket(y) v IsAvailable(x,y) |
| 32 | None of the products sold in the supermarket are made in the supermarket. | ∀x ∀y Supermarket(x) ∧ Products(y) ∧ Sells(x,y) => ~Made(y,x) | ~Supermarket(x) v ~Products(y) v ~Sells(x,y) v ~Made(y,x) |
| 33 | Color are Red, Blue, Green, Black, Yellow, Pink, White, Orange, Purple. | ∀x Red(x) => Color(x)  ∀x Blue(x) => Color(x)  ∀x Green(x) => Color(x)  ∀x Black(x) => Color(x)  ∀x Yellow(x) => Color(x)  ∀x Pink(x) => Color(x)  ∀x White(x) => Color(x)  ∀x Orange(x) => Color(x)  ∀x Purple(x) => Color(x) | ~Red(x) v Color(x)  ~Blue(x) v Color(x)  ~Green(x) v Color(x)  ~Black(x) v Color(x)  ~Yellow(x) v Color(x)  ~Pink(x) v Color(x)  ~White(x) v Color(x)  ~Orange(x) v Color(x)  ~Purple(x) v Color(x) |
| 34 | Product attributes are color, taste, shape.(#) |  |  |
| 35 | Butter is a Dairy Product  DairyProducts are Products. | ∀x Butter(x) => DairyProduct(x)  ∀x DairyProduct (x) => Product(x) | ~ Butter(x) v DairyProduct(x)  ~ DairyProduct (x) v Product(x) |
| 36 | Milk is a Dairy Product | ∀x Milk (x) => DairyProduct(x) | ~ Milk (x) v DairyProduct(x) |
| 37 | Cheese is a Dairy Product | ∀x Cheese(x) => DairyProduct(x) | ~ Cheese(x) v DairyProduct(x) |
| 38 | Yogurt is a Dairy Product | ∀x Yogurt(x) => DairyProduct(x) | ~ Yogurt(x) v DairyProduct(x) |
| 39 | If a product contains milk, then it is a Dairy Product.  Dairy product, contains milk. | ∀x ∀y Product(x) ∧ Milk(y) ∧ Contains(x,y) => DairyProduct(x)  ∀x DairyProduct(x) => ∃y Milk(y) ∧ Contains(x,y) | ~Product(x) v ~Milk(y) v ~Contains(x,y) v DairyProduct(x)  ~ DairyProduct(x) v Milk(f(x))  ~ DairyProduct(x) v Contains(x, f(x)) |
| 40 | Butter contains milk. | ∀x ∀y Butter(x) ∧ Milk (y) => Contains(x,y) | ~Butter(x) v ~Milk (y) v Contains(x,y) |
| 41 | Cheese contains milk. | ∀x ∀y Cheese(x) ∧ Milk (y) => Contains(x,y) | ~Cheese(x) v ~Milk (y) v Contains(x,y) |
| 42 | Yogurt contains milk. | ∀x ∀y Yogurt(x) ∧ Milk (y) => Contains(x,y) | ~Yogurt(x) v ~Milk (y) v Contains(x,y) |
| 43 | Yogurt is sweetened. | ∀x Yogurt(x) => IsSweet(x) | ~ Yogurt(x) v IsSweet(x) |
| 44 | Butter can be salted or unsalted. [#] | ∀x Butter(x) => IsSalted(x) v ~IsSalted(x) | ~ Butter(x) v IsSalted(x) v ~IsSalted(x) |
| 45 | Cheese is salted. | ∀x Cheese(x) => IsSalted(x) | ~ Cheese(x) v IsSalted(x) |
| 46 | Milk can be sweetened or unsweetened [#] | ∀x Milk(x) => IsSweet(x) v ~IsSweet(x) | ~ Milk(x) v IsSweet(x) v ~IsSweet(x) |
| 48 | If the product is milk, then it is white in color and liquid. | ∀x ∀y Product(x) ∧ Milk(y) ∧ Is(x,y)=> White(x)  ∀x ∀y Product(x) ∧ Milk(y) ∧ Is(x,y)=> Liquid(x) | ~Product(x) v ~Milk(y) v ~Is(x,y) v White(x)  ~Product(x) v ~Milk(y) v ~Is(x,y) v Liquid(x) |
| 49 | If the product is butter, then it is yellow in color and solid. | ∀x ∀y Product(x) ∧ Butter(y) ∧ Is(x,y)=> Yellow(x)  ∀x ∀y Product(x) ∧ Butter(y) ∧ Is(x,y)=> Solid(x) | ~Product(x) v ~Butter(y) v ~Is(x,y) v Yellow(x)  ~Product(x) v ~Butter(y) v ~Is(x,y) v Solid(x) |
| 50 | If the product is yogurt then it is white in color and solid. | ∀x ∀y Product(x) ∧ Yogurt(y) ∧ Is(x,y)=> White(x)  ∀x ∀y Product(x) ∧ Yogurt(y) ∧ Is(x,y)=> Solid(x) | ~Product(x) v ~Yogurt(y) v ~Is(x,y) v White(x)  ~Product(x) v ~Yogurt(y) v ~Is(x,y) v Solid(x) |
| 51 | If the product is cheese, then it is yellow or white in color and solid. | ∀x ∀y Product(x) ∧ Cheese(y) ∧ Is(x,y)=> Yellow(x) V White(x)  ∀x ∀y Product(x) ∧ Cheese(y) ∧ Is(x,y)=> Solid(x) | ~Product(x) v ~Cheese(y) v ~Is(x,y) v Yellow(x) v White(x)  ~Product(x) v ~Cheese(y) v ~Is(x,y) v Solid(x) |
| 52 | Dairy Products are part of Groceries | ∀x DairyProduct(x) => Groceries(x) | ~ DairyProduct(x) v Groceries(x) |
| 53 | Vegetables can be Frozen Vegetables or Canned Vegetables or Fresh Vegetables | ∀x Vegetables(x) => Frozen(x) v Canned (x) v Fresh (x) | ~Vegetables(x) v Frozen (x) v Canned(x) v Fresh (x) |
| 54 | Fruits can be Frozen or Canned or Fresh Fruits. | ∀x Fruits(x) => Frozen(x) v Canned (x) v Fresh (x) | ~Fruits(x) v Frozen (x) v Canned(x) v Fresh (x) |
| 55 | Frozen items cannot be Fresh Items or Canned Items. | ∀x Frozen(x) => ~ Canned (x)  ∀x Frozen(x) => ~Fresh(x) | ~Frozen(x) v ~ Canned (x)  ~Frozen(x) v ~Fresh(x) |
| 56 | Fresh Items cannot be Canned Items or Frozen Items. | ∀x Fresh (x) => ~ Canned (x)  ∀x Fresh (x) => ~ Frozen (x) | ~ Fresh (x) v ~ Canned (x)  ~Fresh (x) v ~ Frozen (x) |
| 57 | Canned items cannot be Frozen items or Fresh Items. | ∀x Canned (x) => ~ Fresh (x)  ∀x Canned (x) => ~ Frozen (x) | ~Canned (x) v ~ Fresh (x)  ~Canned (x) v ~ Frozen (x) |
| 58 | Vegetables are part of Groceries. | ∀x Vegetables(x) => Groceries(x) | ~ Vegetables(x) v Groceries(x) |
| 59 | Fruits are part of Groceries. | ∀x Fruits(x) => Groceries(x) | ~ Fruits(x) v Groceries(x) |
| 60 | Potatoes or Onions or Jalapenos or Bell Pepper or Cabbage or Okra or Green Onions or Ginger or Tomatoes or Broccoli or Mushrooms or Lemons or Spinach or Kale or Fenugreek is vegetables | ∀x Potatoes(x) => Vegetables(x)  ∀x Onions(x) => Vegetables(x)  ∀x Jalapenos(x) => Vegetables(x)  ∀x Bell\_peppers(x) => Vegetables(x)  ∀x Cabbages(x) => Vegetables(x)  ∀x Okra(x) => Vegetables(x)  ∀x Green\_Onions(x) => Vegetables(x)  ∀x Ginger(x) => Vegetables(x)  ∀x Tomatoes (x) => Vegetables(x)  ∀x Broccoli (x) => Vegetables(x)  ∀x Mushrooms (x) => Vegetables(x)  ∀x Lemons(x) => Vegetables(x)  ∀x Spinach (x) => Vegetables(x)  ∀x Kale (x) => Vegetables(x)  ∀x Fenugreek (x) => Vegetables(x) | ~Potatoes(x) v Vegetables(x)  ~Onions(x) v Vegetables(x)  ~Jalapenos(x) v Vegetables(x)  ~Bell\_pepper(x) v Vegetables(x)  ~Cabbages(x) v Vegetables(x)  ~Okra(x) v Vegetables(x)  ~Green\_Onions(x) v Vegetables(x)  ~Ginger(x) v Vegetables(x)  ~Tomatoes (x) v Vegetables(x)  ~Broccoli (x) v Vegetables(x)  ~Mushrooms (x) v Vegetables(x)  ~Lemons(x) v Vegetables(x)  ~Spinach (x) v Vegetables(x)  ~Kale (x) v Vegetables(x)  ~Fenugreek (x) v Vegetables(x) |
| 61 | Potatoes are oval or round in shape. | ∀x Potatoes(x) => Oval(x) v Round(x) | ~Potatoes(x) v Oval(x) v Round(x) |
| 62 | Potatoes can be red, brown, yellow or white. | ∀x Potatoes(x) => Red(x) v Brown(x) v Yellow(x) v White(x) | ~Potatoes(x) v Red(x) v Brown(x) v Yellow(x) v White(x) |
| 63 | Tomatoes are round in shape. | ∀x Tomatoes(x) => Round(x) | ~Tomatoes(x) v Round(x) |
| 64 | Tomatoes are red or green in color | ∀x Tomatoes(x) => Red(x) v Green(x) | ~Tomatoes(x) v Red(x) v Green(x) |
| 65 | Jalapenos are green, red, yellow or orange in color. | ∀x Jalapenos(x) => Green(x) v Red(x) v Yellow(x) V Orange(x) | ~Jalapenos(x) v Green(x) v Red(x) v Yellow(x) V Orange(x) |
| 66 | Bell Pepper are green, red, yellow or orange in color. | ∀x Bell\_peppers(x) => Green(x) v Red(x) v Yellow(x) V Orange(x) | ~Bell\_peppers(x) v Green(x) v Red(x) v Yellow(x) V Orange(x) |
| 67 | Cabbages are green or purple in color. | ∀x Cabbages(x) => Green(x) v Purple(x) | ~Cabbages(x) v Green(x) v Purple(x) |
| 68 | Green Onions are Onions | ∀x Green\_Onions(x) => Onions(x) | ~Green\_Onions(x) v Onions(x) |
| 69 | Onions can be white or purple. | ∀x Onions(x) => White(x) v Purple(x) | ~ Onions(x) v White(x) v Purple(x) |
| 70 | Broccoli is green in color. | ∀x Broccoli (x)=> Green(x) | ~ Broccoli (x) v Green(x) |
| 72 | Lemon is yellow or green in color and round in shape. | ∀x Lemon(x) => Yellow(x) v Green(x)  ∀x Lemon(x) => Round(x) | ~Lemon(x) v Yellow(x) v Green(x)  ~Lemon(x) v Round(x) |
| 73 | Leafy\_Vegetables are also vegetables.  Leafy Vegetables include Kale, Spinach and fenugreek. | ∀x Leafy\_Vegetables(x) => Vegetables(x)  ∀x Spinach (x) => Leafy\_Vegetables(x)  ∀x Kale (x) => Leafy\_Vegetables(x)  ∀x Fenugreek (x) => Leafy\_Vegetables(x) | ~Leafy\_Vegetables(x) v Vegetables(x)  ~Spinach (x) v Leafy\_Vegetables(x)  ~Kale (x) v Leafy\_Vegetables(x)  ~Fenugreek (x) v Leafy\_Vegetables(x) |
| 74 | Kale, Spinach and fenugreek are green in color. | ∀x Spinach (x) => Green(x)  ∀x Kale (x) => Green(x)  ∀x Fenugreek (x) => Green(x) | ~Spinach (x) v Green(x)  ~Kale (x) v Green(x)  ~Fenugreek (x) v Green(x) |
| 75 | Apples or Oranges or Grapes or Strawberries or Blackberry or Blueberry or Raspberry or Kiwi or Watermelon or Cherry or Banana or Mangoes or Pear are Fruits. | ∀x Apples(x) => Fruits(x)  ∀x Oranges(x) => Fruits(x)  ∀x Grapes(x) => Fruits(x)  ∀x Strawberries(x) => Fruits(x)  ∀x Blackberries(x) => Fruits(x)  ∀x Blueberries(x) => Fruits(x)  ∀x Raspberry(x) => Fruits(x)  ∀x Kiwi(x) => Fruits(x)  ∀x Watermelon (x) => Fruits(x)  ∀x Cherry(x) => Fruits(x)  ∀x Banana(x) => Fruits(x)  ∀x Mangoes(x) => Fruits(x)  ∀x Pear(x) => Fruits(x) | ~ Apples(x) v Fruits(x)  ~Oranges(x) v Fruits(x)  ~Grapes(x) v Fruits(x)  ~Strawberries(x) v Fruits(x)  ~Blackberries(x) v Fruits(x)  ~Blueberries(x) v Fruits(x)  ~Raspberry(x) v Fruits(x)  ~Kiwi(x) v Fruits(x)  ~Watermelon (x) v Fruits(x)  ~Cherry(x) v Fruits(x)  ~Banana(x) v Fruits(x)  ~Mangoes(x) v Fruits(x)  ~Pear(x) v Fruits(x) |
| 76 | Apples are red in color and sweet and sour in taste.  Strawberries and Raspberry are red in color and sweet in taste. | ∀x Apples(x) => Red(x)  ∀x Apples(x) => IsSweet(x) v IsSour(x)  ∀x Strawberries(x) => Red(x)  ∀x Strawberries(x) => IsSweet(x)  ∀x Raspberry(x) => Red(x)  ∀x Raspberry(x) => IsSweet(x) | ~Apples(x) v Red(x)  ~Apples(x) v Sweet(x) v Sour(x)  ~ Strawberries(x) v Red(x)  ~Strawberries(x) v Sweet(x)  ~ Raspberry(x) v Red(x)  ~Raspberry(x) v Sweet(x) |
| 77 | Grapes are round in shape and come in black, red and green colors and sweet in taste. | ∀x Grapes(x) => Round(x)  ∀x Grapes(x) => Red(x) v Green(x) v Black(x)  ∀x Grapes(x) => IsSweet(x) | ~ Grapes(x) v Round(x)  ~ Grapes(x) v Red(x) v Green(x) v Black(x)  ~ Grapes(x) v Sweet(x) |
| 78 | Oranges, Pears and Mangoes are yellow or orange in color and sweet or tangy in taste. | ∀x Oranges(x) => Yellow(x) v Orange(x)  ∀x Oranges(x) => IsSweet(x) v IsTangy(x)  ∀x Mangoes(x) => Yellow(x) v Orange(x)  ∀x Mangoes(x) => IsSweet(x) v IsTangy(x)  ∀x Pear(x) => Yellow(x) v Orange(x)  ∀x Pear(x) => IsSweet(x) v IsTangy(x) | ~ Oranges(x) v Yellow(x) v Orange(x)  ~Oranges(x) v IsSweet(x) v IsTangy(x)  ~Mangoes(x) v Yellow(x) v Orange(x)  ~Mangoes(x) v IsSweet(x) v IsTangy(x)  ~Pear(x) v Yellow(x) v Orange(x)  ~Pear(x) => IsSweet(x) v IsTangy(x) |
| 79 | Banana is yellow in color and sweet in taste. | ∀x Banana(x) => Yellow(x)  ∀x Banana(x) =>IsSweet(x) | ~Banana(x) v Yellow(x)  ~Banana(x) v Sweet(x) |
| 80 | Blueberry and black berry are black in color and sweet in taste. | ∀x Blackberries(x) => Black(x)  ∀x Blackberries(x) => IsSweet(x)  ∀x Blueberries(x) => Black(x)  ∀x Blueberries(x) => IsSweet(x) | ~Blackberries(x) v Black(x)  ~ Blackberries(x) v IsSweet(x)  ~Blueberries(x) v Black(x)  ~Blueberries(x) v IsSweet(x) |
| 81 | Kiwi is green in color and tangy in taste. | ∀x Kiwi(x) => Green(x)  ∀x Kiwi(x) => IsTangy(x) | ~Kiwi(x) v Green(x)  ~ Kiwi(x) v IsTangy(x) |
| 82 | Cherry is red in color and sweet in taste. | ∀x Cherry(x) => Red(x)  ∀x Cherry(x) =>IsSweet(x) | ~ Cherry(x) v Red(x)  ~ Cherry(x) v IsSweet(x) |
| 83 | Blueberry, strawberry, raspberry, blackberry are berries. | ∀x Blueberries(x) => Berries(x)  ∀x Raspberry(x) => Berries(x)  ∀x Strawberries(x) => Berries(x) | ~Blueberries(x) v Berries(x)  ~ Raspberry(x) v Berries(x)  ~Strawberries(x) v Berries(x) |
| 84 | Berries are sweet in taste. | ∀x Berries(x) => IsSweet(x) | ~Berries(x) v IsSweet(x) |
| 85 | Watermelon are green in color and sweet in taste and round in shape. | ∀x Watermelon (x) => Green(x)  ∀x Watermelon (x) => IsSweet(x)  ∀x Watermelon (x)=> Round(x) | ~Watermelon (x) v Green(x)  ~Watermelon (x) v IsSweet(x)  ~Watermelon (x) v Round(x) |
| 88 | Fruits cannot be Vegetables. | ∀x Fruits(x) => ~Vegetables(x) | ~ Fruits(x) v ~Vegetables(x) |
| 89 | Fruits cannot be Dairy Products. | ∀x Fruits(x) => ~DairyProduct(x) | ~ Fruits(x) v ~DairyProduct(x) |
| 90 | Vegetable cannot be Dairy Products | ∀x Vegetables(x) => ~DairyProduct(x) | ~Vegetables(x) v ~DairyProduct(x) |
| 91 | Eggs or Chicken or Beef or Turkey or Sausage or Pork or Bacon or Fish is Meat. | ∀x Eggs(x) => Meat(x)  ∀x Chicken(x) => Meat(x)  ∀x Beef(x) => Meat(x)  ∀x Turkey => Meat(x)  ∀x Sausage(x) => Meat(x)  ∀x Pork(x) => Meat(x)  ∀x Bacon(x)=> Meat(x)  ∀x Fish(x) => Meat(x) | ~Eggs(x) v Meat(x)  ~Chicken(x) v Meat(x)  ~Beef(x) v Meat(x)  ~Turkey v Meat(x)  ~Sausage(x) v Meat(x)  ~Pork(x) v Meat(x)  ~Bacon(x) v Meat(x)  ~Fish(x) v Meat(x) |
| 92 | Meat can be Frozen, Canned or Fresh. | ∀x Meat(x) => Frozen(x) v Canned(x) v Fresh(x) | ~Meat(x) v Frozen(x) v Canned(x) v Fresh(x) |
| 93 | Meat cannot Dairy Products. | ∀x Meat(x) => ~DairyProduct(x) | ~ Meat(x) v ~DairyProduct(x) |
| 94 | Meat cannot be Fruits. | ∀x Meat(x) => ~Fruits(x) | ~ Meat(x) v ~Fruits(x) |
| 95 | Eggs are white or brown in color and oval in shape. | ∀x Eggs(x) => Brown(x) v White(x)  ∀x Eggs(x) => Oval(x) | ~Eggs(x) v Brown(x) v White(x)  ~Eggs(x) v Oval(x) |
| 96 | Meat cannot be Vegetables. | ∀x Meat(x) => ~Vegetables(x) | ~ Meat(x) v ~Vegetables(x) |
| 97 | Green Cardamom or Cloves or Pepper or Salt or Garlic or Italian Seasoning or Cumin Seeds or Mustard Seeds or Paprika or Smoked Paprika or Basil or Oregano are Spices | ∀x Green\_Cardamom(x) => Spices(x)  ∀x Cloves(x) => Spices(x)  ∀x Pepper(x) => Spices(x)  ∀x Salt(x) => Spices(x)  ∀x Garlic(x) => Spices(x)  ∀x Italian\_Seasoning (x)=> Spices(x)  ∀x Cumin\_Seeds (x) => Spices(x)  ∀x Mustard\_Seeds (x)=> Spices(x)  ∀x Paprika(x) => Spices(x)  ∀x Smoked\_Paprika(x)=> Spices(x)  ∀x Basil(x) => Spices(x)  ∀x Oregano(x) => Spices(x) | ~Green\_Cardamom(x) v Spices(x)  ~Cloves(x) v Spices(x)  ~Pepper(x) v Spices(x)  ~Salt(x) v Spices(x)  ~Garlic(x) v Spices(x)  ~Italian\_Seasoning (x) v Spices(x)  ~Cumin\_Seeds (x) v Spices(x)  ~Mustard\_Seeds (x) v Spices(x)  ~Paprika(x) v Spices(x)  ~Smoked\_Paprika(x) v Spices(x)  ~Basil(x) v Spices(x)  ~Oregano(x) v Spices(x) |
| 98 | Spices are groceries. | ∀x Spices(x) => Groceries(x) | ~Spices(x) v Groceries(x) |
| 99 | Spices cannot be fruits, vegetables, meat or dairy products. | ∀x Spices(x)=> ~DairyProduct(x)  ∀x Spices(x)=> ~Fruits(x)  ∀x Spices(x)=> ~Vegetables(x)  ∀x Spices(x)=> ~ Meat(x) | ~ Spices(x) v ~DairyProduct(x)  ~Spices(x) v ~Fruits(x)  ~Spices(x) v ~Vegetables(x)  ~Spices(x) v ~ Meat(x) |
| 100 | If a product is sweet then it has sugar in it. (#) | ∀x∀ y Product(x) ∧ IsSweet(x) ∧ Sugar(y) => Contains(x,y) | ~Product(x) v ~IsSweet(x) v ~Sugar(y) v Contains(x,y) |
| 101 | If a product is salty then it has salt in it. (#) | ∀x∀ y Product(x) ∧ IsSalty(x) ∧ Salt(y) => Contains(x,y) | ~Product(x) v ~IsSalty(x) v ~Salt(y) v Contains(x,y) |
| 102 | Icecream and Popsicle are frozen foods. | ∀x Icecream(x) => Frozen(x)  ∀x Popsicle(x) => Frozen(x) | ~Icecream(x) v Frozen(x)  ~ Popsicle(x) v Frozen(x) |
| 103 | Other items in supermarket include Cereal, Bread, rice, Oil, Sugar, chocolate, energy drink, bottled water.  Other items are groceries. | ∀x Cereal(x) => Others(x)  ∀x Bread(x) => Others(x)  ∀x Rice(x) => Others(x)  ∀x Oil(x) => Others(x)  ∀x Sugar(x) => Others(x)  ∀x Chocolate(x) => Others(x)  ∀x Energy\_Drink (x) => Others(x)  ∀x Water(x) => Others(x)  ∀x Others(x) => Groceries(x) | ~ Cereal(x) v Others(x)  ~ Bread(x) v Others(x)  ~Rice(x) v Others(x)  ~ Oil(x) v Others(x)  ~Sugar(x) v Others(x)  ~Chocolate(x) v Others(x)  ~Energy\_Drink (x) v Others(x)  ~Water(x) v Others(x)  ~ Others(x) v Groceries(x) |
| 104 | Milk and cereal are eaten together. | ∀x ∀y Cereal(x) ∧ Milk(y) => EatenWith(x,y) | ~Cereal(x) v ~Milk(y) v EatenWith(x,y) |
| 105 | Bread and butter are eaten together | ∀x ∀y Bread(x) ∧ Butter(y) => EatenWith(x,y) | ~Bread(x) v ~Butter(y) v EatenWith(x,y) |
| 108 | Sugar can be both white and brown in color. | ∀x Sugar(x) => White(x) v Brown(x) | ~ Sugar(x) v White(x) v Brown(x) |
| 110 | Sugar is sweet in taste. | ∀x Sugar(x) => IsSweet(x) | ~ Sugar(x) v IsSweet(x) |
| 111 | Chocolate contains sugar. | ∀x ∀y Chocolate(x) ∧ Sugar(y) => Contains(x,y) | ~Chocolate(x) v ~Sugar(y) v Contains(x,y) |
| 112 | Energy Drink is a liquid and it is sweet in taste. | ∀x Energy\_Drink (x) => Liquid(x)  ∀x Energy\_Drink (x) => IsSweet(x) | ~ Energy\_Drink (x) v Liquid(x)  ~ Energy\_Drink (x) v IsSweet(x) |
| 115 | Sweet, sour, tangy, salty are tastes. | ∀x IsSweet(x)=> Tastes(x)  ∀x IsSour(x)=> Tastes(x)  ∀x IsTangy(x)=> Tastes(x)  ∀x IsSalty(x)=> Tastes(x) | ~ IsSweet(x) v Tastes(x)  ~IsSour(x) v Tastes(x)  ~IsTangy(x) v Tastes(x)  ~ IsSalty(x) v Tastes(x) |
| 116 | Round, Oval, Elliptical are Shapes. | ∀x Round(x)=> Shapes(x)  ∀x Oval(x)=> Shapes(x)  ∀x Elliptical(x)=> Shapes(x) | ~ Round(x) v Shapes(x)  ~Oval(x) v Shapes(x)  ~Elliptical(x) v Shapes(x) |
| **117** | If it’s a vegetable then it can’t be eaten raw. | ∀x Vegetables(x) => EatenAfterCooking(x)  ∀x Vegetables(x) => ~EatenRaw(x) | ~ Vegetables(x) v EatenAfterCooking(x)  ~Vegetables(x) v ~EatenRaw(x) |
| 118 | If it’s a fruit, then it can be eaten raw or after cooking. | ∀x Fruits(x) => EatenRaw(x) v EatenAfterCooking(x) | ~Fruits(x) v EatenRaw(x) v EatenAfterCooking(x) |
| 119 | Dairy Products can be eaten raw or cooked. | ∀x DairyProducts(x) => EatenRaw(x) v EatenAfterCooking(x) | ~DairyProducts(x) v EatenRaw(x) v EatenAfterCooking(x) |
| 120 | Meat can only be eaten raw not cooked. | ∀x Meat(x) => EatenAfterCooking(x)  ∀x Meat(x) => ~EatenRaw(x) | ~ Meat(x) v EatenAfterCooking(x)  ~Meat(x) v ~EatenRaw(x) |
| 121 | Other items may or may not be eaten raw or after cooking. | ∀x Cereal(x) => EatenRaw(x) v ~EatenAfterCooking(x)    ∀x Bread(x) => EatenRaw(x) v EatenAfterCooking(x)  ∀x Rice(x) => ~EatenRaw(x) v EatenAfterCooking(x)  ∀x Oil(x) => ~EatenRaw(x) v EatenAfterCooking(x)  ∀x Sugar(x) => EatenRaw(x) v EatenAfterCooking(x)  ∀x Chocolate(x) => EatenRaw(x) v EatenAfterCooking(x)  ∀x Energy\_Drink (x) => EatenRaw(x) v ~EatenAfterCooking(x)  ∀x Water(x) => EatenRaw(x) v EatenAfterCooking(x) | ~Cereal(x) v EatenRaw(x) v ~EatenAfterCooking(x)    ~Bread(x) v EatenRaw(x) v EatenAfterCooking(x)  ~Rice(x) v ~EatenRaw(x) v EatenAfterCooking(x)  ~Oil(x) v ~EatenRaw(x) v EatenAfterCooking(x)  ~Sugar(x) v EatenRaw(x) v EatenAfterCooking(x)  ~Chocolate(x) v EatenRaw(x) v EatenAfterCooking(x)  ~Energy\_Drink (x) v EatenRaw(x) v ~EatenAfterCooking(x)  ~Water(x) v EatenRaw(x) v EatenAfterCooking(x) |
| 122 | No two products are the same. | ∀x ∀y Product(x) ∧ Product(y) => (x != y) | -Product(x) | -Product(y) | (x != y). |
| 123 | Salmon and Tuna are fishes. | ∀x Salmon(x) => Fish(x)  ∀x Tuna(x) => Fish(x) | -Salmon(x) | Fish(x).  -Tuna(x) | Fish(x). |
| 124 | No two colors are the same. | ∀x ∀y Color(x) ∧ Color(y) => (x != y) | -Color(x) | -Color(y) | (x != y). |
| 125 | No two taste are the same. | ∀x ∀y Tastes (x) ∧ Tastes (y) => (x != y) | -Tastes(x) | -Tastes (y) | (x != y). |
| 126 | No staff can be the same as customer. | ∀x Staff(x) => -Customer(x)  ∀x Customer(x) => -Staff(x) | -Staff(x) | -Customer(x).  -Customer(x) | -Staff(x) |
| 127 | No two shapes are the same. | ∀x ∀y Shapes(x) ∧ Shapes(y) => (x != y) | -Shapes(x) | -Shapes (y) | (x != y). |
| 128 | Groceries are products. | ∀x Groceries(x) => Product(x) | -Groceries(x) | Product(x) |
| 129 | Ice cream and Popsicle are others. | ∀x Icecream(x) => Others(x)  ∀x Popsicle(x) => Others(x) | -Icecream(x)| Others(x).  -Popsicle(x)| Others(x). |